QUICKSILVER CONTROLS, INC

Command Reference

SilverMax™ X-Series
SilverNugget™ X-Series
SilverDust™
SilverSterling™

Revision 6.18

6/22/2016

For QuickControl 6.18

DISCLAIMER

ALL PRODUCTS, PRODUCT SPECIFICATIONS AND DATA ARE SUBJECT TO CHANGE WITHOUT NOTICE.
The information contained in this datasheet is based on data we believe to be reliable and is given for information only and without guarantee and does not constitute a warranty. We disclaim any and all liability for any errors, inaccuracies or incompleteness contained herein or in any other material relating to the product. We are not able to anticipate every set of conditions, so always suggest that users should also satisfy themselves as to the suitability of our products for their particular environment and application and not make any assumptions based on information in this data sheet that is included or omitted.
No license, express, implied, or otherwise, to any intellectual property rights is granted by this document or by any conduct of QuickSilver Controls, Inc., nor its representatives (collectively, "QCI"). QCI makes no warranty, representation or guarantee other than as set forth in the terms and conditions of purchase. To the maximum extent permitted by applicable law, QCI disclaims (i) any and all liability arising out of the application or use of any product, (ii) any and all liability, including without limitation special, consequential or incidental damages, and (iii) any and all implied warranties, including warranties of fitness for particular purpose, non-infringement and merchantability.
The products shown herein are not designed for use in life-saving or life-sustaining applications. Customers using or selling QCI products in such applications do so entirely at their own risk and agree to fully indemnify QCI for any damages arising or resulting from such use or sale.
This datasheet supersedes any previous information/datasheet released and is subject to change without notice.

How to Use This Manual

The Command Reference contains a detailed description of the commands for the SilverMax™ X, SilverDust™, SilverSterling™, and SilverNugget™ X Product Families. It should be used as a reference not as a tutorial. For general information on QuickControl, please refer to the User Manual.

This manual is broken up into several chapters with each chapter detailing a category of commands. For example, there are chapters for Initialization, Mode and Motion commands. Within these chapters, each command is described in one or more pages.

Warnings

The QuickSilver Controls, Inc (QCI) SilverMax/SilverSterling/SilverDust/SilverNugget servos are high performance motion system. As with any motion system, it is capable of producing sufficient mechanical output to cause bodily injury and/or equipment damage if it is improperly operated or if it malfunctions. The user shall not attach a QCI product to any mechanism until its operation is fully understood. Furthermore, the user shall provide sufficient safety means and measures to protect any operator from misuse or malfunction of the motion system. The user assumes all liability for its use.

User must remove motor from load before configuring the servo or aligning motor index pulse to prevent potential injury or damage. This applies to non-integrated and non I-Grade (motor + controller) systems.

After replacing a motor, encoder, or driver, the motor must be removed from the load prior to energizing the system, and the user must re-run the Configuration Wizard and Initialization Wizard in QuickControl to prevent potential injury or damage. This step is not required if attaching I-Grade motors to I-Grade SilverSterling Controllers, or using a SilverMax Integrated system.

Units shall not be used in life critical applications without the signed authorization of the President of QuickSilver Controls.

User is responsible to provide safety interlocks for any application that may cause injury or damage in either normal or abnormal operation of the unit.

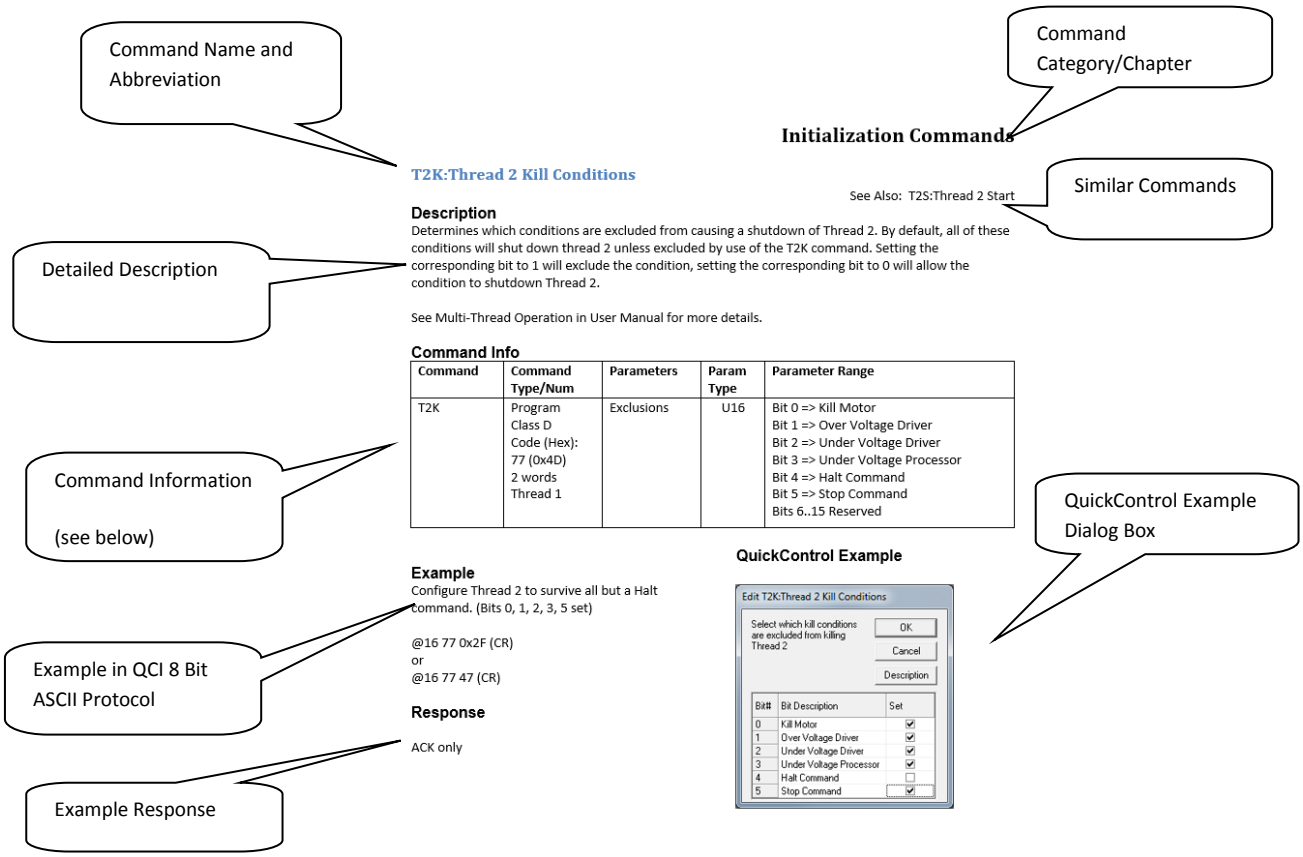
Do not mechanically back drive the motor of a SilverLode servo without a voltage clamp present. The voltage generated may damage the electronics.

User shall limit input current to the controllers to not more than 6A for S2, N2, and D2 controllers, and 17 and 23 frame SilverMax systems, and not more than 25A for S3 and N3 controllers and for 34 frame SilverMax systems.

Trademarks

Copyright

Page Anatomy



Command Information

Command Name

- Name of command and its three-letter acronym.
- First firmware revision this command appeared. Blank implies the command is available on all units.
    - SilverMax X-series (SX)
    - SilverSterling(SS)
    - SilverDust (SD)
    - SilverNugget X-Series (NX)
    - all: available in all revisions of this product
        - Example SS 03
          Available on SilverSterling Rev 03 and newer

Command Type/Number

- Command Type (Program or Immediate). See below for details.
- Command Class (A through F). See below for details.
- Command numbers range from 0 to 255.
- Commands with numbers less than 64 are Host level Immediate Mode only commands (See Command Types below for more details).
- Command numbers 64 or greater are commands that can be contained in a program.
- Commands with numbers 64 or greater will generate a “Busy” “NAK” code if sent to the motor while it is executing a command or a program. There are certain exceptions – see Enable MultiTasking (EMT) for more details.
- Commands numbers are given in decimal and hexadecimal format. In the above example the command number is 77 (0x4D). 77 decimal and 4D hex.
- Thread Execution. Class D and E commands can execute in either Thread 1, Thread 2 or Thread 1&2.

Parameters

- List of parameters for this command
- Parameters must always be included in the command even if the value is “0”.

Parameter Type

- S32 indicates a signed 32-bit parameter, which can range in value from -2147483648 to +2147483647.
- U32 indicates an unsigned 32-bit parameter, which can range in value from 0 to 4294967295.
- S16 indicates a signed 16-bit parameter, which can range in value from -32768 to +32767.
- U16 indicates an unsigned 16-bit parameter, which can range in value from 0 to 65535.

Parameter Range

- Typical parameter range

Command Types

The command structure is divided into two major classifications: Immediate Mode Commands and Program Mode Commands. The Immediate Mode Commands may only be executed via the serial link, while Program Mode Commands may be executed via the serial link or from the non-volatile memory. Program Mode Commands are temporarily stored in the Program Buffer prior to execution. Before executing a Program, the Program Buffer is filled with the given Program from either the serial communications or the non-volatile memory.

Immediate Mode

Immediate Mode Commands typically give an immediate result or return data when executed. Most of these commands can be executed at any time even during operation. Some Immediate Mode Commands – those which affect the command buffer - cannot be executed simultaneous to Program Buffer operations. These commands and the conditions for execution are noted in the command description. If command execution is attempted when not appropriate, the device will produce a "NAK Device Busy" response.

Immediate Mode commands do not use the Program Buffer. They are executed as soon as they are received. (Exception- Stop and change velocity immediate; these overwrite the buffer and take over motion and command processing, and may be used to stop or close down the program in progress.)

Immediate Mode commands can only be used via the serial communications interface; they cannot be used within a Program that is downloaded to the device for Program execution. A “Host” controller may use Immediate Mode Commands to set up, control, or determine status of a device.

Program Mode

Program Mode Commands can be executed either from the serial communications interface or from non-volatile memory. Program Mode Commands, as the name implies, can be part of a Program. When these commands are sent, they are first loaded into the Program Buffer, and then executed. This requires that the buffer not be in use at the time the command is sent. For example, they cannot be executed while the Load Program or Store Program commands are active. If a Program Mode Command is sent while the motor is active, a "NAK Device Busy" response is returned.

Program Mode Commands can also be downloaded to the Program Buffer without being executed. Once a Program has been assembled, it can either be executed immediately or it can be written to the non-volatile Memory. Programs can also be loaded from the Non-volatile Memory and executed.

Combo-Commands

Combo-Commands provide a macro like program construct in which user selections cause the parameters of multiple native commands to be simultaneously edited. These are particularly useful where the information must be bit or byte packed for the implementation, and save manual calculations of these fields. All native commands have three letter acronyms, whereas the Combo-Commands have four letter acronyms, to allow for easy recognition. The Combo-Commands may be ***expanded*** to see the underlying commands by right clicking on the Combo-Command and selecting Expand from the pop up menu. They may be restored to a single line by the same process. The individual commands are “greyed out” as they may not be edited individually. However, they may be copied and pasted in to a program by selecting only the individual commands (and not the Combo-Command) and performing a copy and then a paste operation. At this point, they are no longer associated with the Combo-Command and may be individually edited.

Command Classifications

The command set has been broken into the following classifications. Each class of command has a set of rules that define how or when a command can be used.
NOTE: “executed” for this section means to “Send a command real-time from a Host controller to the device using the serial communications interface”

Class A Commands

These are serial communications interface only. They may not be contained within a Program and their execution does not incidentally affect the Program Buffer contents. They may be executed at any time.

Class B Commands

These are serial communications interface only. They may not be contained within a Program, but their execution affects the Program Buffer. They may be executed only while the motor is idle (No Motion or Program is running- No EEPROM operation active). Multi-Tasking – Allows these commands to be executed when a Motion is running but not when a Program is running nor when an EEPROM operation is active.

Class C Commands

These are serial communications interface only. They may not be contained within a Program, but their execution affects the Program Buffer. They may be executed only while the motor is idle (No Motion or Program is running). The Program Buffer must also be loaded prior to execution.

Class D Commands

These commands can be executed from the serial communications interface or as part of a Program. Their execution from the "Host" affects the Program Buffer. They may only be executed when the motor is idle. They are then stored to the buffer when in download (Program Download) mode. All of these commands have a command code of 64 (hex 0x40) or higher. Multi-Tasking – Allows these commands to be executed when a Motion is running but not when a Program is running. Most commands will execute immediately while the "Motion" or "Profile Move" commands will be buffered until the current Motion is complete. Commands involving branching in or modifying the command buffer contents should only be downloaded such that the full context is present (not trying to branch to non-existent code).

Class E Commands

These commands are executed as part of a Program. They may be executed from the serial communications interface but should only be used within a Program or the motor operation may not be what is expected. They rely on what has been previously loaded to the Program buffer for operation. They may only be sent when the motor is idle. They will be stored to the buffer when in download (Program Download) mode. All of these commands have a command code of 64 (hex 0x40) or higher. Multi-Tasking – Allows these commands to be executed while a Motion is running, but care must be taken to avoid unexpected results.

Class F Commands

These are serial communications interface only. They may not be contained within a Program, but their execution affects the Program Buffer. They may be executed while the motor is running or idle.

CII:Configure I/O, Immediate Mode

See Also: CIO:Configure I/O, COB:Clear Output Bit, SOB:Set Output Bit

Description

The is an Immediate Mode version of CIO. Using the Serial Interface this command can be used at any time, even during program execution.

See Input and Output Functions in User Manual for more details.

** Pull-up/Pull-down only available on X-series: SilverNugget and SilverMax
The configurable pull up/down resistors are 2.2k, and pull to 0 or 3.3v or are tri-stated according to configuration. By default, the configurable pull resistors are floating (tri-state) with only the weak internal pull-ups active.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CII	Immediate Class A 31 (0x1F) 3 words Thread 1&2	I/O Line #	U16	I/O Line #
		Mode	S16	-1 = Input mode 0 = Clear (Low) 1 = Set (High) * -4 = No pull ** -3 = Pull-down ** -2 = Pull-up **

Example

Set I/O 1 high:

@16 31 1 1 (CR)

Response

ACK only

QuickControl Example

Immediate (Host) Command Only

CLP:Clear Program

Description

The Clear Program prepares the unit for downloading into the Program Buffer. First, the Program Buffer is cleared. Then the buffer pointer is set to the beginning of the buffer. This command is used prior to a Start Download command. It sets up the buffer to properly receive a program.

This command may also be used to end the program download initiated by a Start Download (SDL) command.

See Memory Model in User Manual for details on downloading programs

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CLP	Immediate Class B 8 (0x8) 1 word	NONE	NONE	NONE

Example

Clear the Program Buffer.

@16 8 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

AKC only

CPL:Clear Poll

See Also: POL:Poll

Description

This is a complement to the Poll (POL) command. This command is used to clear the Polling Status Word (PSW) bits (See Polling Status Word (PSW) in User Manual for bit definitions). When a status bit is set ("1") it will remain set until a Clear Poll (CPL) command is sent with the same bit set in its Clear Status Word parameter.

For example, if a POL command gets back a Polling Status Word(PSW) of "0x2000", bit 13 set (Program completed), of the PSW is set. To reset bit 13, the Clear Status Word parameter must be set to "0x2000". This will cause bit 13 to be re-set ("0"). All other bits in the PSW will be left unchanged if the corresponding clear bit is not set. New occurrences since the last poll will NOT be cleared (the PSW is double buffered). That is, the information must be read before it is cleared.

See Status Words in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CPL	Immediate Class A 1 (0x1) 2 words	Clear Status Word	U16	0 to 65535

Example

Clear only Bit #13 set in the Polling Status Word (Decimal 8192 = 0x2000 in Hexadecimal)

@16 1 8192 (CR)

Clear all the bits set in the Polling Status Word.

@16 1 65535 (CR)

QuickControl Example

Immediate (Host) Mode Command Only.

Response

ACK only

HLT:Halt

Description

This command immediately shuts down any motion in progress (hard stop), disables the single step mode, and then causes the motor to load and run the Kill Motor Recovery program. (see Kill Motor Recovery (KMR) command for details.)

This command stops the execution of all commands, programs and motions. When executed, it will stop any command or program in process. Unless the Kill Motor Recovery Program has been designated and the Kill Enable Driver (KED) has been enabled, the motor driver will be disabled. This allows the motor shaft to be manually spun.

Bit #10 of the Internal Status Word (ISW) is "set" to indicate that a Halt command was sent. This is useful for determining the cause of the motor shut down when using an internal Kill Motor Recovery program.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
HLT	Immediate Class A 2 (0x2) 1 word	NONE	NONE	NONE

Example

Halt any command, program or motion in process

@16 2 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

ACK only

Interpolated Move Write Queue (IMW)

See Also: **Profiled Move (PMV), Interpolated Move Start (IMS), QCI-TD044_InterpolatedMotion, Interpolated Move Clear (IMC)**

Description:

This Command writes data to the Interpolated Move Queue through the Serial Interface. This queue is a software FIFO (First in first out) buffer capable of holding data for up to four interpolated motion segments, the data for each segment consisting of four long words (32 bits each) of data. If the data is able to fit within the queue, it is accepted and the communication is acknowledged. If the queue is full, the request is answered with a NAK – Full response. This NAK is to be expected: it just indicates that the host is successfully keeping the queue filled. The same data should be sent again until it is positively Acknowledged.

The four long words of data associated with each Interpolated Move segment are:
Time ticks: Indicating the number to 120 microseconds time slices the segment is to last. A "0" indicates it is the last segment of the move.
Position. Unless the segment is intended to come to a halt at a given location, this should be full scale positive if the final velocity of the segment is positive, or full scale negative if the final velocity of the segment is to be negative.
Acceleration: Indicates the acceleration or deceleration to be used in reaching the requested velocity or position.
Velocity: Indicating the desired ending velocity magnitude (speed) of the segment, or maximum velocity to use if coming to a stop within this segment. Will only be reached if it is consistent with the starting velocity, the acceleration, and the segment time.

See "Scaling" in User Manual for details on scaling the parameters to engineering units.
See Interpolated Move in User Manual for details.

Command Info:

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
Interpolated Move Write Queue (IMW)	Immediate Class A 25 (0x19) 9 words	Time	S32	0 to +2,147,483,647
		Position	S32	-2,147,483,648 to +2,147,483,647
		Acceleration	S32	1 to 2,147,483,647
		Velocity	S32	0 to +2,147,483,647

Example:

Move to position -456 at acceleration 7890 at velocity 1234 and wait for 123 ticks before getting the next command.

@16 25 123 –456 7890 1234 (CR)

.

Response:

* 10 (CR) = ACK (data accepted)

! 10 0019 0006 (CR) = Negative Acknowledge (NAK) , Command 25 (0x19), Reason = Queue Full (data not accepted, retry later)

The NAK indicates that the data was rejected. This should commonly happen if the Host is keeping ahead of the controller unit. It means that the queue has been kept full, and to send the data again. Periodic polling should also be done to see that the move has not improperly ended due to the Host falling behind the consumption rate of the controller, and data not being available when needed.

QuickContol Example:

Immediate (Host) Mode Command Only

LPR:Load Program

Description

The Load Program transfers a program from the non-volatile memory to the Program Buffer. The number of words to be transferred is read from the location given in the NV Memory Address parameter. This count is automatically stored in the first word, along with a checksum, when the program is written into non-volatile memory.

The content in the first NV Memory Address of the program is the length in words of the program size and the checksum of the program. The first command is read from the address following the Length & Checksum word, with subsequent words transferred up to the size indicated in the Length.

During the load process, the data is used to calculate a checksum value. When the load is complete, the calculated checksum is compared to the stored checksum. If the checksums do not agree Bit #14 in the Polling Status Word is set ("1") to indicate a program load failure.

This command only transfers the program into the Program Buffer; it does not cause execution to begin. Once loaded into the Program Buffer a Run Program command must be issued to begin program execution. The program will remain in the buffer until removed by the Clear Buffer command or over loaded by another Load Program command.

See Memory Model in User Manual for details on downloading programs.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
LPR	Immediate Class B 14 (0x0E) 3 words	NV Memory Address	U16	Valid NV Memory
		Word Count The Count is typically set to "0"	U16	**0** = use count stored at first address location. **1 to** Program Buffer Size = read the literal word count.

Example

Load the program stored at NV Memory Address #110.

@16 14 110 0 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

ACK only

POL:Poll

See Also: PSW:Poll Status Word, CPL:Clear Poll

Description

This command is used to determine the condition of a unit. A Poll command can be executed at any time, including while the device is in motion. Executing this command will cause the addressed unit to return either an ACK (if no bits of the status are set), or the Polling Status Word (PSW). The PSW contains information about the current state of the device (see User Manual for definitions). The Poll command can be used when checking to see if a motion has completed. This is useful when a system must wait for the device to complete its operation before performing the next operation. The Polling Status Word bits are "Set" when the particular condition takes place. The bits are "cleared" using a Clear Poll (CPL) command. Note: Additional conditions that occur after a Poll will show up in the following Poll even if those bits have been cleared in an intervening Clear Poll command. (i.e. they cannot be cleared until they have been read - the data is double buffered).

See Status Words in User Manual for bit definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
POL	Immediate Class A 0 (0x0)*	NONE	NONE	NONE

*No command will also trigger the poll routine.

Example

Poll without command number

@16 (CR)

Poll with command number

@16 0 (CR)

QuickControl Example

Immediate (Host) Mode Command Only.

Response

ACK only or Polling Status Word. For a poll with a consistent response see Poll Status Word (PSW).

Response Example

Response with status

10 0000 2000 (CR)

Response without status

*10 (CR)

Polling Status Word (PSW)

The Polling Status Word is returned by the poll command and indicates the overall condition of the device. The meaning of each bit in the Polling Status Word (PSW) is explained in the table below. The description for each bit describes the device condition indicated by a "1" in that bit.

Bit#	Latched?	Definition	Description
15	Yes	I-Cmd Done	Immediate Command Done (i.e. Host Command).
14	Yes	NV Mem Error	There was a checksum error while reading data from or to the non-volatile memory. (SilverDust Rev 06 adds write protection to certain regions.)
13	Yes	P-Cmd Done	All commands active in the Program Buffer finished executing.
12	Yes	Command Error	There was an error associated with the command execution. Unreasonable parameter values or not support in this firmware
11	Yes	Input Found	The motion ended when the selected exit/stop condition was met.
10	Yes	Low/Over Volt	A low or over voltage error occurred.
9	Yes	Holding Error	Holding error limit set by the Error Limits (ERL) command was exceeded during a holding control state.
8	Yes	Moving Error	Moving error limit set with the ERL command was exceeded with the device in a moving control state.
7	Yes	Rx Overflow	Device serial receive (UART) buffer overflowed.
6	Yes	CKS Cond Met	A condition was met while executing a CKS command. One of the conditions set with the Check Internal Status (CKS) command was met.
5	Yes	Msg Too Long	The received message was too big for the Serial Buffer. Device rx packet > 31 bytes
4	Yes	Framing Error	There was a packet framing error in a received byte. Device rx byte with missing bit.
3	Yes	Kill Motor	The device was shut down due to one or more conditions set with the Kill Motor Condition (KMC) command (or KMX command for SilverDust Rev 06). Latched yes, but after CPL it clears even if there is still a KMC till present.
2	Yes	Soft Limit	A soft stop limit was reached as set by the Soft Stop Limit (SSL) command.
1	Yes	Rx Checksum	Device rx packet with an invalid checksum. Valid for 9 Bit Binary and Modbus only.
0	Yes	Aborted Pkt	There was a data error or a new packet was received before the last packet was complete.

POR:Poll With Response

See Also: POL:Poll, CPL:Clear Poll

Description

This command is the same as POL except that it response always has the same format. This might be easier to parse than the POL command for some host controllers.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
POR	Immediate Class A 27 (0x1B)	NONE	NONE	NONE

Example

Poll with command number

@16 27 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

Polling Status Word

Response Example

Response with status

10 0000 2000 (CR)

PUP:Protect User Program

Description

This Immediate mode command allows the user to protect a user selectable portion of the nonvolatile memory from the bottom (0) up to the user defined limit. The user specifies the first writable nonvolatile memory address, and a lockout code.

If the first writable location is set to 0, the memory is not protected, and the lockout code is also cleared.

To protect user program, the user must supply a lockout code and the first writable memory location. Supplying a location of 65535 will lock all of the user space. Note: this prevents all writes to EEPROM in the protected user space, including saving registers to non-volatile. If the program saves registers, it is necessary that these registers are saved to unprotected space (at or above the first writable location).

Note: SilverDust I-Grade devices are sent from the factory with the Factory Block protected (63488 or 0xF800 and up for I-Grade SilverDust, 65280 or 0xFF00 for the M-Grade SilverDust, and the Silver Nuggets). The M-Grade SilverDust units (non-CAN) are sent with the Factory Space unprotected by default, however, executing the PUP command, including executing it with both the lockout code and the starting location set to zero will enable protection of the Factory Block. In the case of setting both parameters to zero, the only effect will be to protect the Factory Space.

Updating the User Program (changing program or initialization) requires enabling the affected user space before downloading will be successful. This requires remembering the user assigned lockout code. It is thus highly advised to place it in the QCP remarks so that it is not forgotten.

NOTE: This is not intended as an absolute lockout against the determined, but as a means to prevent accidental or casual changes to the configuration.

If the device is busy, this command will NAK-Busy and not execute.

If the Lockout Code does not match, the command will NAK-Bad Lockout Code (NAK with a code of 9)

Note: A NAK –Bad Lockout Code error will be returned if an attempt is made to Store Program to a protected space.

Note: Attempting to save registers to protected space will set STATUS bits 14 and 12, as read by the POLL command, and the program will exit without completing the write.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PUP	Immediate Class B 33 (0x21) 1 word	Lockout Code	U16	0 to 65535
		1st NV Memory Address (first writeable address)	U16	0 to 65535

Example

Protect User Program below 1000.

@16 33 1234 1000 (CR)

Response

ACK only

QuickControl Example

Immediate (Host) Command

RIO:Read I/O States

Description

The I/O State Word (IOS) is available for reading back the states of miscellaneous I/O conditions. This word is dynamic and may change every servo cycle (120 usec.).

See Status Words in User Manual for bit definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RIO	Immediate Class A 21 (0x15) 1 word	NONE	NONE	NONE

Example

Read back the I/O State Word

@16 21 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

I/O State Code

Response Example

Indicates lines #4, 5, 6, & 7 are “High” and lines #1, 2 & 3 are also “High”

10 0015 F0F0 (CR)

I/O Status Word (IOS)

The I/O Status Word (IOS) indicates the states of the I/O lines, as well as several specific internal conditions.

The meaning of each bit in the I/O Status Word (IOS) is explained in the table below. The description for each bit, except Bit 7, indicates the condition indicated by a "1" in that bit.

Bit#	Latched?	Definition	Description
15	No	I/O #7	Status of I/O line #7 (normally high, low when triggered).
14	No	I/O #6	Status of I/O line #6 (normally high, low when triggered).
13	No	I/O #5	Status of I/O line #5 (normally high, low when triggered).
12	No	I/O #4	Status of I/O line #4 (normally high, low when triggered).
11	Yes	Reserved	
10	No	Delay Counter	Status of the Delay Counter. High when counting down, low when count is expired.
9	No	Holding Error	Holding error limit set with the Error Limits (ERL) command has been exceeded with the device in a holding control state.
8	No	Moving Error	Moving error limit set with the ERL command has been exceeded with the device in a moving control state.
7	No	Temperature Ok	Internal sensor determines over temperature condition and clears this bit when true. Bit is shared by drive enable lines on 34 frame servos. (disable = high, enable = low.)
6	No	I/O #3	Status of I/O line #3 (normally high, low when triggered).
5	No	I/O #2	Status of I/O line #2 (normally high, low when triggered).
4	No	I/O #1	Status of I/O line #1 (normally high, low when triggered).
3	No	Trajectory Generator Active	When the Trajectory Generator is active, the device is calculating motion. (i.e. a move is in progress)
2	No	External Index Found	An index mark on an external encoder has been detected.*
1	No	Internal Index Found	An index mark on an internal encoder has been detected.*
0	No	Index Found	An index mark has been detected on the selected encoder (external or internal).*

*Only one 120uSec cycle wide.

Note: Latched bits can be cleared using the CIS command.

RIS:Read Internal Status Word

See Also: CIS:Clear Internal Status

Description

The Internal Status Word (ISW) is used in the device to keep track of different conditions that are present in the motor. The Internal Status Word (ISW) can be cleared using the Clear Internal Status (CIS) command.

See Status Words in User Manual for bit definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RIS	Immediate Class A 20 (0x14) 1 word	NONE	NONE	NONE

Example

Read back the Internal Status Word

@16 20 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

Internal Status Word (ISW)

Response Example

Indicates Input #1, 2, 3 “High”, Last Calculation was Zero and Index Sensor was found.

10 0014 00F3 (CR)

Internal Status Word (ISW)

The meaning of each bit in the ISW is explained in the table below. The description for each bit indicates the condition indicated by a "1" in that bit. ISW is the upper word of register 3.

Bit#	Latched?	Definition	Description
15	-	Reserved	Reserved bit.
14	Yes	Low Voltage	Low voltage error has occurred. Defined by LVT command.
13	Yes	Over Voltage	Over voltage error has occurred. Defined by OVT command.
12	Yes	Wait Delay Exhausted	The wait delay timer has expired.
11	Yes	Input Found On Last Move	An input used as a stop condition was found during the last move. Latched but automatically cleared on next move.
10	Yes	Halt Command Sent	The halt command (HLT) was received by the device.
9	Yes	Holding Error	Holding error limit set with the Error Limits (ERL) command has been exceeded with the device in a holding control state.
8	Yes	Moving Error	Moving error limit set with the ERL command has been exceeded with the device in a moving control state.
7	No	Temperature Ok	Internal sensor determines over temperature condition and clears this bit when true. Bit is shared by drive enable lines on 34 frame servos. (disable = high, enable = low.)
6	No	I/O #3	Status of I/O line #3 (normally high, low when triggered).
5	No	I/O #2	Status of I/O line #2 (normally high, low when triggered).
4	No	I/O #1	Status of I/O line #1 (normally high, low when triggered).
3	No	Negative Calculation Result	Result of the last calculation was negative. CLC command.
2	No	Positive Calculation Result	Result of the last calculation was positive. CLC command.
1	No	Zero Calculation Result	Result of the last calculation was zero. CLC command.
0	Yes	Index Found	An index has been detected.

Latched bits can be cleared using the CIS command.

RPB:Read Program Buffer

Description

Reads the data that is currently contained in the Program Buffer. The specified number of words are read from the Program Buffer starting with the given address. Up to 8 words can be read at one time. To read the entire contents of the Program Buffer multiple reads are required. For details on memory management, see the User Manual section Basic Motion and Programming Fundamentals.

NOTE: When reading command codes from Program Buffer the MSB (Most Significant Bit) will be complemented. For example, if an MRV command is read from the Program Buffer, it will be read as a 0x07 instead of a 0x87.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RPB	Immediate Class A 6 (0x6) 3 words	Length (in words)	S16	1 to 8
		Address	S16	0 to Program Buffer Length

Example

Read the first 7 words from Program Buffer

@16 6 7 0 (CR)

QuickControl Example

Immediate (Host) Mode Command Only

Response

Requested number of words read from Program Buffer.

Response Example

10 0006 0007 0000 9C40 0002 7524 2000 0058 (CR)

RRG:Read Register

See Also: RRW:Read Register Write

Description

The Read Register command reads back data from a selected 32-Bit Data Register using the Serial Interface. Since it is an Immediate Mode, this command can be used at any time, even during program execution. Any Data Register can be read back using this command. Later versions allow up to 4 registers to be read back in a single operation.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RRG	Immediate Class A 12 (0x0C) 2 words	Data Register SD 31: Data Register 2 (optional) Data Register 3 (optional) Data Register 4 (optional)	U16 SD 31 S16 S16 S16	Standard Register Range SD 31: Optional arguments to read additional registers.

Example

Read back the motor's current position.

@16 12 1 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

Data Register data

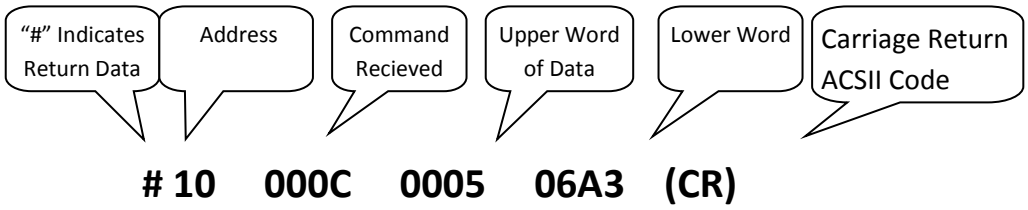
Response Example

Read Register command that requests the "Current Position" from device #16 (which is "10" in Hexadecimal); the last 8 digits represent the 32-bits of position data.

The current position = 329,379 in decimal

10 000C 0005 06A3 (CR)

The return data breaks down as follows:



RRW:Read Register Write

See Also: RRG:Read Register, WRI:Write Register, Immediate Mode
WRP:Write Register, Program Mode, WRX:Write Register Extended

Description

This command reads the given register then modifies it using the given operation and data. NOTE: The returned data is the value of the register prior to modification.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RRW	Immediate Class A 32 (0x20) 5 words	Operation	U16	0 = Set (OR) 1 = Clear 2 = AND 3 = XOR (Toggle) 4 = Add
		Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

QuickControl Example

Immediate (Host) Command Only

Example

Toggle (XOR) Bit 1 in data register #12.

@16 32 3 12 1 (CR)

Response

Assuming register 12 was originally contained zero:

10 0020 0000 0000 (CR)

Following the operation, register 12 would contain a 1.

RST:Restart

See Also: RSP:Restart, Program Mode

Description

The Restart command is provided to cause the device to do a “soft” reset of the processor and logic circuits. This causes the processor to jump to memory address zero as if the power were just cycled on. All configurations and settings are returned to power-up defaults. All registers are cleared but non-volatile memory is not affected.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RST	Immediate Class A 4 (0x4) 1 word	NONE	NONE	NONE

Example

Restart the processor. This is done immediately.

@16 4 (CR)

Response

There is no response due to the resetting of the processor

QuickControl Example

Immediate (Host) Mode Command Only

RUN:Run Program

Description

Executes the program that has been previously loaded into the Program Buffer. This command will clear the download mode, set the program pointer to "0" and start the program.

The Program Buffer can be filled using the Start Download command from the Host controller (see Start Download below). It can also be filled using the Load Program command that will move a program from the non-volatile memory into the Program Buffer (see Load Program above).

Any Command or Program remaining in the Program Buffer can be executed over again using this command. When in Host Mode, Program Mode commands sent to the device will remain in the buffer until another Program Mode command is sent or a Program is loaded. (Note: the STOP command will alter the command buffer if a motion is in process when the STOP command is sent). The Run Program command can be used to repeat the previous Program Mode command.

Sending this command while a Program or Command is executing will give a NAK – Busy response.

NOTE: Sending a Program Mode command while in Host mode actually loads that command into the start of the command buffer with an END command inserted behind it and then that (short) program is run.

See Memory Model in User Manual for details on downloading and running programs.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RUN	Immediate Class C 10 (0x0A) 1 word	NONE	NONE	NONE

Example

Run the Program or Command that was previously loaded into the Program Buffer.

@16 10 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

ACK only

RVN:Revision

Description

This command returns the revision date firmware, and the buffer sizes. The code revision date and buffer sizes of a device can be read back so that future upgrades can be dealt with through a software interface.

The response format is as follows:
<mon|day> <year> <rev> <ser size|pb size>

Data Type	Data Format	Example Shown Below
Month (mon)	1 Byte	"03" = March
Day (day)	1 Byte	"16" = 16th day
Year (year)	2 Bytes	"2016" = The year 2016
Revision (rev)	2 Bytes	"2A24" = Code rev 2A24
Serial Comm Buffer Size (ser size)	1 Byte	"0A" = 10 Words
Program Buffer Size (pb size)	1 Byte	"FF" = more then 255

*Note: Devices having a program buffer size of 255 or larger will report 255 words. Actual size for these devices may be read from register 210 high word.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RVN	Immediate Class A 5 (0x05) 1 word	NONE	NONE	NONE

Example

Read the revision code

@16 5 (CR)

QuickControl Example

Immediate (Host) Mode Command On

Response

Revision Code (8 Bytes)

Response Example

Revision code

10 0005 0316 2016 2A24 0AFF (CR)

SDL:Start Download

Description

This command puts the device into a program download mode**.** Program Mode commands that are sent after a Start Download command are automatically appended to the Program Buffer rather than being executed. Once in the Program Buffer, they can be executed as a program or stored to non-volatile memory. The program download mode is terminated by a Store Program (SPR), a Run Program (RUN) or a Clear Program (CLP) command.

Immediate Mode commands sent to the device when in download mode are not appended to the buffer. Depending on the command, it will be immediately executed or it will cause an error.

See Memory Model in User Manual for details on downloading programs.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SDL	Immediate Class B 9 (0x9) 1 word	NONE	NONE	NONE

Example

Put the device into the program download mode.

@16 9 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

ACK only

SPR:Store Program

Description

The Store Program command stores a program into Non-volatile Memory onboard the unit. The currently loaded program will be stored at the address number indicated in the address parameter of the command. A program must be downloaded in the Program Buffer before the Store Program is used. The program download mode is terminated by this command.

The length of the program (in words) and a Checksum are written to the indicated memory address, followed by the program. The length is used by the Load Program (LPR) or Load & Run Program (LRP) command to know the size of the program to load from non-volatile memory. Because the length is written to the first address location, add 1 word to overall length for keeping track of memory usage. The Checksum is used by the Load Program or Load & Run Program command to determine the data integrity. This prevents corrupted or partially overlapping programs from attempting execution. Programs of length 255 and longer have one additional word written to the second address in memory, to hold the larger size counter. The first size counter in the first word is set to 255 to indicate the large buffer mode. This extra word must be counted in the memory usage map if manually overriding the memory management in QuickControl.

This command leaves a background routine running until the programming of the non-volatile memory has completed. Once completed, Bit #15 in the Polling Status Word is set. Bit #14 is set if the command attempts to write beyond the allowed memory space. Execution time of this command varies depending on the number of words written.

See Memory Model in User Manual for details on downloading programs.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SPR	Immediate Class C 13 (0x0D) 2 words	NV Memory Address	U16	Valid NV Memory Range

Example

Store the currently loaded program into NV memory at address 1000.

@16 13 1000 (CR)

Response

ACK only

STP:Stop

Description

The Stop command exits the executing program or motion. If a motion is running, the Deceleration parameter sets the deceleration as follows: If the parameter is zero, the device uses the executing command's acceleration value for deceleration. If the parameter is positive, the device uses the given deceleration value. If the parameter is negative, the device does an immediate stop. The servo's target position value is set to the actual position. If the servo is not executing a motion, any Program Mode command executing is terminated and the servo returns to idle.

When the Stop command is sent, the Program Buffer is over-written (similar to a Clear Program (CLP) command). The Program Buffer must be loaded again (Load Program (LPR) or Load And Run Program (LRP)) for program execution.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
STP	Immediate Class F 3 (0x3) 3 words	Deceleration	S32	**-1** = Stop Immediate or **0** = Stop using previous Acceleration or 1 to 536,870,911

Example

Stop the device using
the previous commanded
acceleration.

@16 3 0 (CR)

Response

ACK only

QuickControl Example

Immediate (Host) Mode Command Only

VMI:Velocity Mode, Immediate Mode

See Also: VMP:Velocity Mode, Program Mode, PVC:Profile Velocity Continuous

Description

Accelerates the servo from the present velocity to the indicated velocity using the given acceleration. If the servo has an active move operation in progress, that motion is taken over from its current velocity, and ramps to the new velocity at the given acceleration rate. Any program operating is stopped and the contents of the command buffer are modified. This command is used when the velocity mode needs to be controlled from a Host controller. This command can only be used through the serial interface. See the Velocity Mode, Program Mode (VMP) command for embedding this type of command in a program. WARNING: As this command stops any sequence currently active, this command must not be issued until the initialization routine has had time to complete, otherwise the motor may not be completely configured when this command is detected.

NOTE: If the acceleration is negative, any accumulated position error is removed and the absolute value of the acceleration is then used.

See Scaling in User Manual for more details on native acceleration and velocity units.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
VMI	Immediate Class F 15 (0xF) 7 words	Acceleration	S32	-1 to -1,073,741,823 or 1 to 1,073,741,823
		Velocity	S32	-2,147,483,647 to +2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

Put the device into velocity mode
running at 200 RPM.

@16 15 200000 107374200 0 0(CR)

Response

ACK only

QuickControl Example

Immediate (Host) Mode Command Only

WRI:Write Register, Immediate Mode

Description

This command writes the given data into the selected 32 bit Data Register. Using the Serial Interface this command can be used at any time, even during program execution.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WRI	Immediate Class A 11 (0x0B) 4 words	Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

Example

Write the number “8000” to data register #12.

@16 11 12 8000 (CR)

Response

ACK only

QuickControl Example

Immediate (Host) Command Only

WRX:Write Register Extended

See Also: RRW:Read Register Write,WRI:Write Register, Immediate Mode
WRP:Write Register, Program Mode,CLX:Calculation Extended

Description

This command modifies the given register using the given operation and data.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WRX	Immediate Class A 30 (0x1E) 5 words Thread 1&2	Operation	U16	0 = Set (OR) 1 = Clear 2 = AND 3 = XOR (Toggle) 4 = Add
		Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

Example

Toggle (XOR) Bit 1 in data register #12.

@16 30 3 12 1 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

ACK only

ADL:ACK Delay

Description

ACK Delay sets a time delay for the device to wait before sending an Acknowledgement (ACK) or data after a command has been received. This delay is often needed to allow the host computer or communications hardware time to turn off transmit and prepare for the ACK / response.

When the serial interface is set to RS-232, a value of "0" causes the device to run in standard RS-232 mode (the Tx line is always driven). With a number of "1" or greater, the device will run in RS-232 multi-drop mode (the Tx line is tri-stated when not transmitting). For RS-485, the interface is always multi-drop, and ACK delay determines the turn-around time.

The parameter is a count that equates to a number of 120 microsecond (uSec) "ticks". For greater resolution, use a negative Delay Count to specify 40uSec ticks.

If Auto is checked, QuickControl will automatically set ADL at download depending on the previously downloaded Serial Interface (SIF) and Baud Rate (BRT) commands. An error message will be displayed if "auto" ADL is used in a program without SIF and BRT. Settings for ADL set to Auto are as follows:

RS-232: ADL=1 (enables RS-232 multi-drop mode)
RS-485: ADL=2.4ms+5*<character time period>

For Modbus® protocol, the ACK Delay configures the 2 character time period between the query frame and the response frame. See Application Note "QCI-AN038 Modbus Protocol" for details.

See Technical Document "QCI-TD053 Serial Communications" on our website for additional details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ADL	Program Class D 173 (0xAD) 2 words Thread 1&2	Delay Count in ticks 1 tick= 120 uSec For negative values: 1 tick = 40 uSec	S16	-32767 to 21845

Example

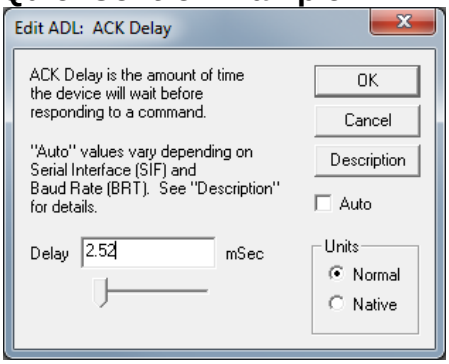
Delay ACK for 2.52 milliseconds
21 x 120 uSec= 2.52 ms

@16 173 21 (CR)

Response

ACK only

QuickControl Example



AHC:Anti-Hunt Constants

See Also: AHD:Anti-Hunt Delay, AHM:Anti-Hunt Mode

Description

Anti-Hunt Constants sets the thresholds used to determine if the position is sufficiently close to the target to allow the motor to go into and to stay in Anti-Hunt mode. The first parameter is the maximum error (in counts) allowed in the Anti-Hunt mode before the unit will revert to normal closed loop operation. The second parameter is the maximum error allowed to enter the Anti-Hunt mode. This command is primarily used for Hybrid Servo Motors; it should not be used for DC or brushless DC.

Setting the second parameter to a negative number (QuickControl: Check "Check Holding Currents") will cause a slightly different operation when going from no Anti-Hunt into Anti-Hunt (Closed => Open). Normally the device will not go into Anti-Hunt until the error is within the limit and the current torque (current) is less than the Open Loop Holding torque (current). When the error parameter is negative, the torque is not checked. This allows for zero holding current or "dead band" operation.

If the Torque Limits (TQL) Open Loop Holding and Open Loop Moving parameters have been set to zero, then the parameters in this command sets the limits of a conventional dead-band.

QuickControl Edit Mode:

- Default: (see table on right). (Default for Initialization Wizard).
- Disable Anti-Hunt.
- Edit the parameters manually.

Encoder Counts/ Rev (CPR)	Defaults Open to Close/ Close to Open	Max Recommended
4000	10/4	30
8000	20/8	60
16000	40/16	120

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
AHC	Program Class D 150 (0x96) 3 words Thread 1&2	Open to Closed	S16	0 to 140
		Closed to Open	S16	-140 to 140

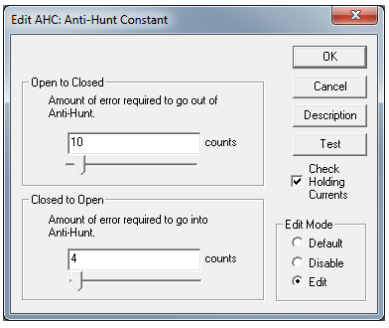
Example

Go into Anti-Hunt when within "4" counts of target. Go out of Anti-Hunt when "10" counts away

@16 150 10 4 (CR)

Response

ACK only



QuickControl Example

AHD:Anti-Hunt Delay

See Also: AHC:Anti-Hunt Constants
AHM:Anti-Hunt Mode

Description

After the conditions are met for Anti-Hunt as specified by the Anti-Hunt Constants (AHC) command, this Anti-Hunt Delay (AHD) specifies the amount of delay before going into Anti-Hunt. This is useful for allowing a system time to "settle" prior to going into Anti-Hunt, thus preventing system "chatter". See Anti-Hunt Constants (AHC) for more details.

Settling time is a system parameter, which must be analyzed under real working conditions. Using the Control Panel in QuickControl allows viewing of motion profiles for analyzing settling times.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
AHD	Program Class D 230 (0xE6) 2 words Thread 1&2	Delay Count in Ticks 1 Tick = 120usec.	U16	0 to 65535 Default =1250 ticks (150ms)

Example

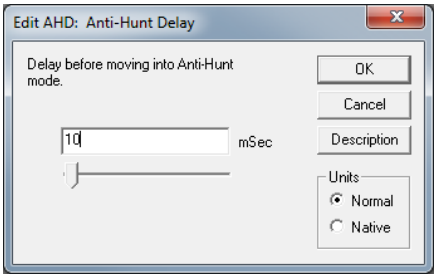
Allow Anti-Hunt 10 milliseconds after a motion is completed.

@16 230 83 (CR)

Response

ACK only

QuickControl Example



AHM:Anti-Hunt Mode

See Also: AHC:Anti-Hunt Constants
AHD:Anti-Hunt Delay

Description

The default mode of Anti-Hunt automatically switches from open loop to closed loop as soon as a motion begins, and then remains in closed loop for Anti-Hunt Delay time counts after the motion as completed and the position error is less than the Closed to Open parameter. Anti-Hunt Mode with Mode=1 bypasses the in motion check, allowing the servo to remain in open loop, even while moving, as long as the error is sufficiently low. A value of Mode=0 switches the Anti-Hunt function back to its default mode of operation.

With Mode=1, some Anti-Hunt Delay (AHD) is useful to keep from switching between moving and stopped while moving at low speeds.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
AHM	Program Class D Code (Hex): 219 (0xDB) 2 words Thread 1	Mode	S16	0 or 1 0 = only when stopped (Default) 1 = moving or stopped

Example

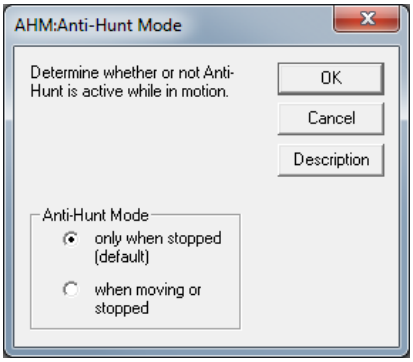
Allow Anti-Hunt Mode only while stopped.

@16 219 0 (CR)

Response

ACK only

QuickControl Example



BRT:Baud Rate

Description

This command is used to change the devices baud rate.

Negative values set the hard divisor for odd baud rates.

Divisor = 2.5MHz/(Baud Rate) -1

If this command is sent in Immediate Mode, the response will be at the new baud rate.

See Technical Document QCI-TD053 Serial Communications on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
BRT	Program Class D 174 (0xAE) 2 words Thread 1&2	Speed – Character bit rate NOTE: Negative, indicates bit period. Contact Factory for special baud rates.	S16	3 = 300 (baud) 12 = 1200 24 = 2400 48 = 4800 96 = 9600 192 = 19200 288 = 28800 384 = 38400 576 = 57600 (Default) 1000 =100000 1152 =115200 2304 =230400* 2500=250000** or -11 to -32767

* Not available for Modbus®, ** DMX only.

Example

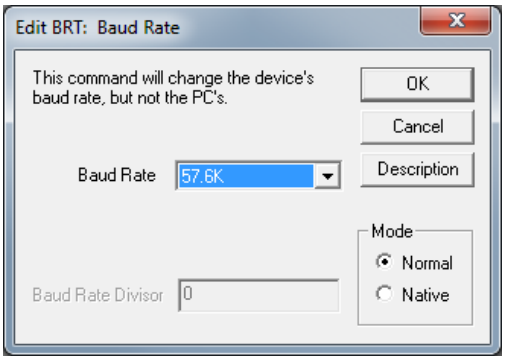
Set the baud rate for 57.6K.

@16 174 576 (CR)

Response

ACK only

QuickControl Example



CER:Command Error Recovery

See Also: KMR:Kill Motor Recovery, PLR:Power Low Recovery

Description

CER sets up options for recovery from a Command Error. Command Errors occur when the device is programmed to do something it cannot due. For example, a Command Error will occur if the servo is asked to move 1000 revs is 1ms. The required velocity is greater than 4000RPM. By default, Command Errors halt the program and set bit 12 in the Polling Status Word (PSW). When CER is used, the user can, instead, designate a recovery program to load and run anytime a Command Error occurs.

A value of 0 will disable this function. A value of –1 will run the code from non-volatile memory location 0.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CER	Program Class D 65 (0x41) 2 words Thread 1	Process	S16	0 = Do Nothing -1= Load and Run Program @NV Mem Adr 0 #### = Load and Run Program @ indicated NV Mem Adr

Example

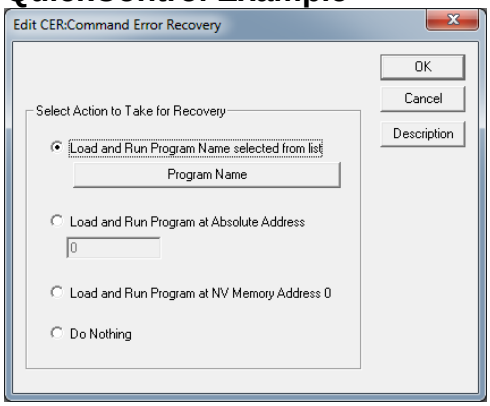
Recover command error from non-volatile memory location 1000

@16 65 1000 (CR)

Response

ACK only

QuickControl Example



CLM:Control Loop Mode

Description

This command sets the control loop of the servo to operate around either Position (default at power up) or Velocity.

In velocity mode, the servo loop is closed around velocity rather than position. The proportional gain term is disabled (zeroed out), and the integrator acts on the difference in velocities between the target velocity and the actual velocity. The anti-windup on the integrator is configured to smoothly recover from a motion stoppage without over-running the desired velocity.

NOTE: In Velocity Mode, the Error Limits kill motor condition must be disabled (see Kill Motor Conditions (KMC) command for details).

Command Info

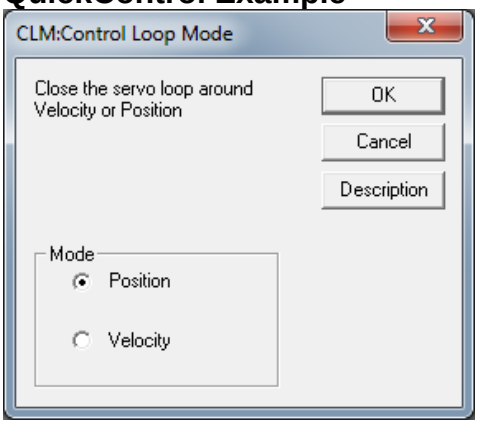
Command	Command Type/Num	Parameters	Param Type	Parameter Range
CLM	Program Class D 166 (0xA6) 2 Words Thread 1	Mode	U16	0 = Position mode (Default) 1 = Velocity mode

Example

@16 166 0 (CR)

Response
ACK only

QuickControl Example



CTC:Control Constants

See Also: CT2:Control Constants 2
FLC:Filter Constants, FL2:Filter Constants 2

Description

This command sets the various servo loop gain control constants. These are used in tuning the servo.

QuickControl stores a default set of parameters for each motor type (i.e. 23-3, 23H-1). If "Use Default For Device" is checked, QuickControl will use the default parameters and adjust Kp with respect to encoder resolution.

Kp = Kp(default) * Encoder Resolution (counts/rev) /4000

See Technical Document QCI-TD054 Servo Tuning on our website for servo loop and tuning procedures.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CTC	Program Class D 148 (0x94) 8 words Thread 1	Kv1: Velocity 1 Feedback Gain	U16	0 to 32767
		Kv2: Velocity 2 Feedback Gain	U16	0 to 32767
		Kvff: Velocity Feedfoward Gain	U16	0 to 32767
		Ka: Acceleration Feedback Gain	U16	0 to 32767
		Kaff: Acceleration Feedfoward Gain	U16	0 to 32767
		Kp: Proportional Gain	U16	0 to 32767
		Ki: Integrator Gain	U16	0 to 32767

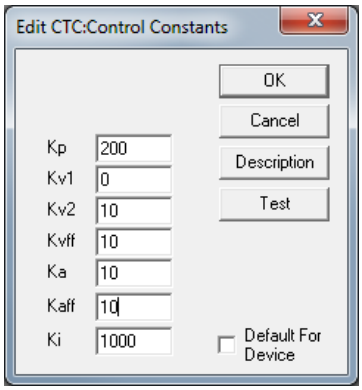
Example

@16 148 0 10 10 10 10 200 1000 (CR)

Response

ACK only

QuickControl Example



CT2: Control Constants 2

See Also: CTC:Control Constants
FLC:Filter Constants, FL2:Filter Constants 2

Description

This command sets the various servo loop gain control constants. These are used in tuning the servo. These correspond to the CTC constants, with one additional filtered acceleration term, Ka2, to accommodate a second simulated inertial damper.

Note: If used with the standard velocity filters (FLC), Ka2 is assumed as zero.

QuickControl stores a default set of parameters for each motor type (i.e. 23-3, 23H-1). If "Use Default For Device" is checked, QuickControl will use the default parameters and adjust Kp with respect to encoder resolution.

Kp = Kp(default) * Encoder Resolution (counts/rev) /4000

See Technical Document QCI-TD054 Servo Tuning on our website for details.

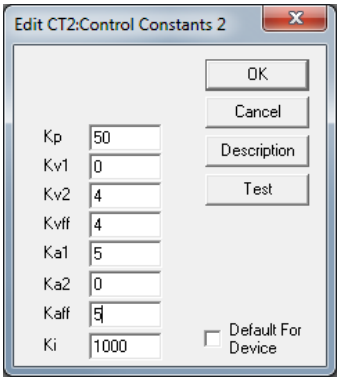
Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CT2	Program Class D 70 (0x46) 9 words Thread 1	Kv1: Velocity 1 Feedback Gain	U16	0 to 32767
		Kv2: Velocity 2 Feedback Gain	U16	0 to 32767
		Kvff: Velocity Feedfoward Gain	U16	0 to 32767
		Ka1: Acceleration 1 Feedback Gain	U16	0 to 32767
		Ka2: Acceleration 2 Feedback Gain	U16	0 to 32767
		Kaff: Acceleration Feedfoward Gain	U16	0 to 32767
		Kp: Proportional Gain	U16	0 to 32767
		Ki: Integrator Gain	U16	0 to 32767

Example

@16 70 0 4 4 5 0 5 50 1000 (CR)

QuickControl Example



Response

ACK only

DDB:Disable Done Bit

See Also: EDH:Enable Done High
EDL:Enable Done Low

Description

Disables the “Done” bit (I/O #1) on the servo. The “Done” bit indicates when the servo is running or idle (See Enable Done Bit for more details.) By default, the “Done” bit is disabled.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DDB	Program Class D 171 (0xAB) 1 word Thread 1&2	NONE	NONE	NONE

Example

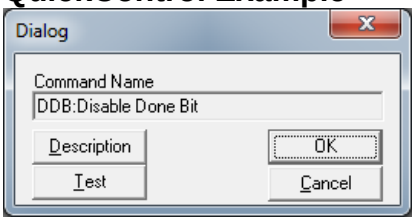
Disable usage of the “Done” bit

@16 171 (CR)

Response

ACK only

QuickControl Example



DEM:Disable Encoder Monitor

See Also: EEM:Enable Encoder Monitor, EMN:Encoder Monitor

Description

Turns off the Enable Encode Monitor mode. If the Enable Encode Monitor mode was set this command will take it out of the monitor mode and return the Digital I/O to normal operation.

Note: The SilverDust accepts this command (to provide back compatibility with previous initialization files), but this command does not affect anything as the Enable Encoder Monitor functionality was removed from the SilverDust. The I-Grade SilverDust, instead, has dedicated buffered encoder signals available without using up normal I/O lines.

Command Info

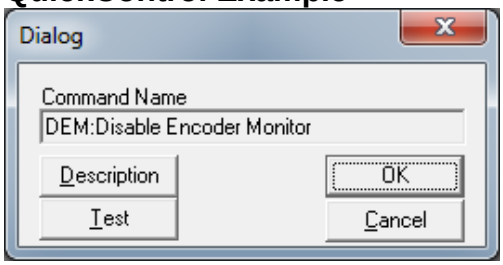
Command	Command Type/Num	Parameters	Param Type	Parameter Range
DEM SD n/a	Program Class D 171 (0xAB) 1 word	NONE	NONE	NONE

Example

Turn off monitoring of the Internal Encoder.

@16 171 (CR)

QuickControl Example



Response

ACK only

DIF:Digital Input Filter

Description

Sets up a filter time constant for any of the digital inputs. A "0" in the I/O line parameter causes all of the input filter constants to be changed at the same time. Selecting 1,2, 3, 4, 5, 6 or 7 for the I/O line changes only the selected line.

The Filter Constant is in "Ticks" (120 usec / tick). Setting the filter constant affects how long a digital state must be held for the device to “see” the given state. The filter does not require that the input be exclusively in the new state for the entire period, but just that it is in that state sufficiently long for the counter to expire.

For example, with the filter set to 8 Ticks (approximately 1 mS), and transitioning from low to high: 5 high states, followed by 2 low states (such as switch bounce / noise) require another 8-5+2 = 5 ticks of high before a high would be reported. 8 consecutive high levels are not required. This minimizes the effects of noise/contact bounce on the system.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DIF	Program Class D Code (Hex): 252 (0xFC) 3 Words Thread 1&2	I/O Line #	U16	0 = All Lines 1 to 7 (101 to 116 for SilverDust with extended IO)
		Filter Constant	U16	0 to 32767 Default: 83 ticks (10ms)

Example

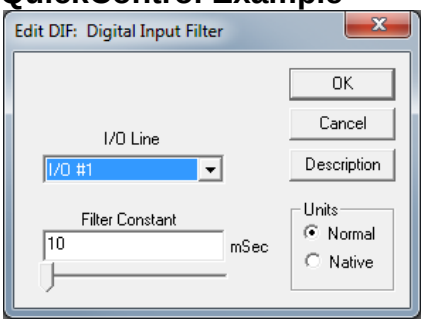
Filter Input #1 so that it must be either low or high for at least 10 milliseconds before the low or high state is accepted.

@16 252 1 83 (CR)

Response

ACK only

QuickControl Example



DIR:Direction

Description

Establishes the direction in which the servo will turn given a motion in a positive direction. Normally the device will turn Clockwise (when viewed from the shaft end of servo) when a positive distance or velocity number is used. A negative number will cause the servo to turn counter clockwise. Using the Direction command, this default operation can be reversed.

WARNING: DIR can only be used when the device is being initialized and before the Go Closed Loop (GCL) command is issued. If DIR is used after GCL the unit will fault with a sequence error. Typically, this command is only edited within the device Initialization Wizard while editing the initialization file "Factory Default Initialization.qcp".

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DIR	Program Class D 184 (0xB8) 2 words Thread 1	Mode	S16	0 = Normal (CW) (Default) 1 = Reverse (CLW)

Example

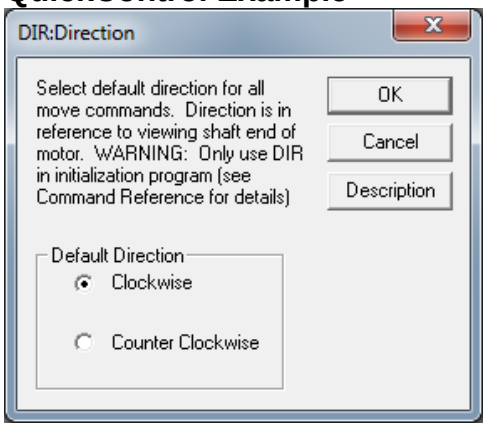
Clockwise

@16 184 0 (CR)

Response

ACK only

QuickControl Example



DLC:Dual Loop Control

See Also: SEE:Select External Encoder
SLC:Single Loop Control

Description

Configures the device to run in a Dual Loop control mode. In Dual Loop Control, the device servos its position based on an external or secondary encoder signal. Device commutation, velocity and acceleration feedback information is derived from the internal or primary encoder. Moving and holding error limits also use the secondary encoder for the Kill Motor Conditions.

When entering dual loop control the device sets its target position to actual position (position of secondary encoder) to prevent a sudden motion.

Use the Select External Encoder (SEE) command to set up the secondary encoder usage prior to using DLC.

NOTE: The Control Constants (CTC) typically need to be configured differently for single loop operation than for dual loop operation. The Velocity and Acceleration parameters for motions become related to secondary encoder counts rather than primary encoder units. The feedforward acceleration and velocity terms are relative to full speed in secondary encoder units while the feedback terms are relative to the primary encoder units, thus the feedback terms may need to be different from the feedfoward terms in order to minimize following error.

NOTE: For units supporting the SSI interface, the SSI port may be selected as the dual loop feedback position source by configuring the SSI command to select the SSI source as a dual loop source while the system is operating in single loop mode. When DLC is executed, the SSI position will be used as the feedback position.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DLC	Program Class D 243 (0xF3) 1 word Thread 1	NONE	NONE	NONE

Example

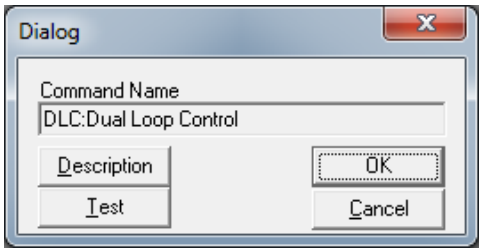
Configure the device for Dual Loop Control

@16 243 (CR)

Response

ACK only

QuickControl Example



DMD:Disable Motor Driver

See Also: EMD:Enable Motor Driver

Description

Disables the motor driver. The device will be unable to move when attempting any motion command. This is a software disable that can be overcome by the Enable Motor Driver (EMD) command (if the motor constants have already been set), or by setting the Motor Constants (MCT).

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DMD	Program Class D 228 (0xE4) 1 word Thread 1	NONE	NONE	NONE

Example

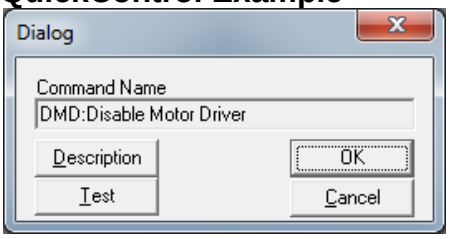
Disable the Motor Driver electronics

@16 228 (CR)

Response

ACK only

QuickControl Example



DMRM:DMX Register Map

Description

DMRM is a Combo-Command (see Combo-Command at the beginning of this manual for details) that maps incoming DMX512 serial data to SilverLode data registers. It consists of multiple Write Register Program (WRP) commands, encoding the DMX state machine.

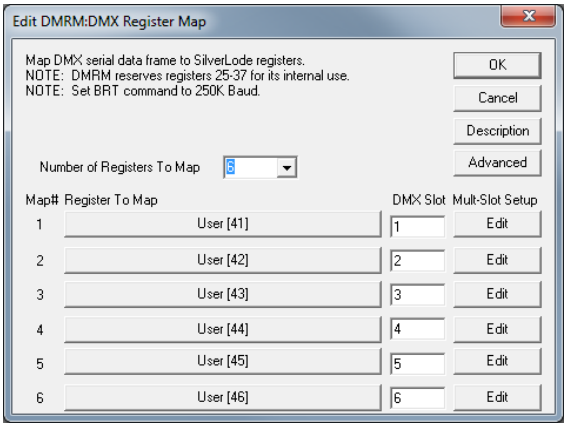
When DMRM is used, registers 25-37 are reserved for internal use.

See Application Note " QCI-AN045 DMX512 Protocol.doc" for more details including how to map more than 6 registers.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DMRM	Program Class COMBO D 40 words Thread 1&2	START Code	U8	0-255:0 (default)
		Comm Timeout in ticks 1 tick=120 uSec.	U32	
		Num Reg Maps	U16	1-6
		Reg Map1-6	6x U32	

QuickControl Example



DMT:Disable Multi-Tasking

See Also: EMT:Enable Multi-Tasking

Description

Disables the device's Multi-Tasking operation. See Enable Multi-Tasking for more information on multi-tasking operation.

DMT will immediately stop any active trajectory command (i.e. motion) with no ramp down.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DMT	Program Class D 226 (0xE2) 1 word Thread 1	NONE	NONE	NONE

Example

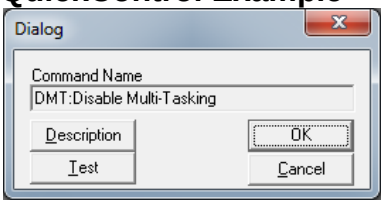
Disable the Multi-Tasking operation

@16 226 (CR)

Response

ACK only

QuickControl Example



EDH:Enable Done High

See Also: DDB:Disable Done Bit
EDL:Enable Done Low

Description

Enables a “Done” indication on the servo I/O Line #1. The “Done” indicates when the servo is idle and within the error limits. When the servo is idle (no pending commands and no active motions) and within the error limits, I/O #1 will be high (“1”), and the Green LED will be lighted, otherwise, I/O #1 will be low (“0”) and the Green LED will be dark.

Note, if multiple commands are in the Program Buffer, all of them must complete (and the error within limits) before the unit is “Done”.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EDH	Program Class D 251 (0xFB) 1 word Thread 1&2	NONE	NONE	NONE

Example

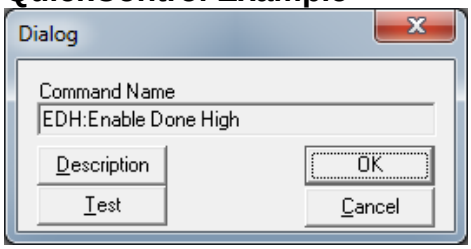
Enable usage of the “Done” indication by setting I/O line #1 High

@16 251 (CR)

Response

ACK only

QuickControl Example



EEM:Enable Encoder Monitor

See Also: DEM:Disable Encoder Monitor, EMN:Encoder Monitor

Description

The Enable Encoder Monitor command is used to output the Internal Encoder signals to the Digital I/O. It causes a buffered copy of the raw encoder signals to be output to three digital lines for external viewing. The Encoder A signal is output to I/O line #1, the Encoder B signal to I/O bit line #2 and the Encoder Index signal is output to I/O line #3.

These signals have the same output specifications as the generic digital outputs. I/O lines #1, #2, and #3 are not available in Bit Output mode (either set or clear) while the encoder outputs are enabled. Similarly, the Encoder outputs may not be enabled while any of the three I/O lines are in output mode. Either of these conflicts will cause a Command Error and will terminate the program. See Using I/O in the User Manual for more information on I/O usage and conflicts.

For the X-series, if the EEM command is enabled to output the encoder on IO # 1, 2, & 3, and the SEE command has the internal encoder output to IO #4, 5, 6, then the encoder signals to IO #1, 2, & 3 are ***inverted*** so as to provide differential encoder output signals.

To exit this mode, use the Disable Encoder Monitor command.

For using the encoder output for controlling or sending signals to other external devices see the modulo commands. These commands are designed to be more flexible in outputting encoder signals.

This command is NOT applicable for SilverDust. See Encoder Monitor (EMN) for a similar capability.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EEM SD n/a	Program Class D 170 (0xAA) 1 word	NONE	NONE	NONE

Example

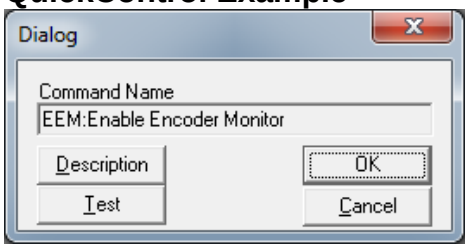
Turn on monitoring of the Internal Encoder.

@16 170 (CR)

Response

ACK only

QuickControl Example



EDL:Enable Done Low

See Also: DDB:Disable Done Bit
EDH:Enable Done High

Description

Enables a “Done” indication on the servo I/O Line #1. The “Done” indicates when the servo is idle and within the error limits. When the servo is idle (no pending commands and no active motions) and within the error limits, I/O #1 will be low (“0”), and the Green LED will be lighted, otherwise, I/O #1 will be high (“1”) and the Green LED will be dark.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EDL	Program Class D 187 (0xBB) 1 word Thread 1&2	NONE	NONE	NONE

Example

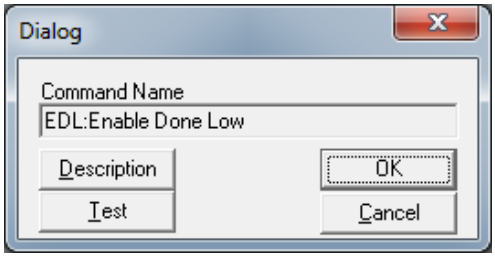
Enable usage of the “Done” indication by setting I/O line #1 Low

@16 187 (CR)

Response

ACK only

QuickControl Example



EMD:Enable Motor Driver

See Also: Disable Motor Driver (DMD)

Description

Enables the device motor driver. The driver is by default enabled by the initialization program, this command is only required if the driver has been disabled using the Disable Motor Driver (DMD) command or disabled by the Kill Motor operation or by an over voltage condition.

If the user is enabling the unit after it has been disabled, and any potential exists that the motor shaft has been rotated since the motor was disabled, then the user should make sure that the motor target and position are made equal before enabling the motor so as to prevent the motor from sudden rotations. This may be accomplished using either the Zero Target Position (ZTP) or the Set To Target Position (TTP) commands. The ZTP sets both the Target and Position to zero, while the TTP maintains the actual motor Position information, and merely sets the target to the current position so that no error exists when the motor is enabled to prevent unwanted motion. If it is necessary to restore the motor to its prior location, then save the target value before doing a STP command, and then do an absolute move using to the saved Target position.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EMD	Program Class D 227 (0xE3) 1 word Thread 1	NONE	NONE	NONE

Example

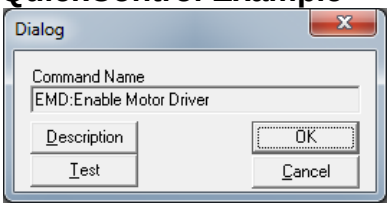
Enable the motor driver

@16 227 (CR)

Response

ACK only

QuickControl Example



EMN:Encoder Monitor

See Also: EEM:Enable Encoder Monitor,SEE:Select External Encoder

Description

The Encoder Monitor command is used to output the Internal or External Encoder signals to the Digital I/O. It causes a buffered copy of the raw encoder signals to be output to three digital lines for external viewing. The Encoder A signal is output to I/O line #4, the Encoder B signal to I/O bit line #5 and the Encoder Index signal is output to I/O line #6.

These signals have the same output specifications as the generic digital outputs. I/O lines #4, #5, and #6 are not available in Bit Output mode (either set or clear) while the encoder outputs are enabled. Similarly, the Encoder outputs may not be enabled while any of the three I/O lines are in output mode.

Note: Internally to the SilverDust, this and the Select External Encoder (SEE) command are the same command. This means that EMN also configures the SilverDust to read the encoder on I/O 4,5 and 6 to registers 200 and 201. The advanced parameter Index State specifies how the index pulse is processed (see SEE for details). If EMN and SEE are used in the same program, the last one executed will override any previous EMN or SEE commands.

Mode	
2	Output Internal Encoder A,B,Z to I/O #4,5,6
3	Output External Differential Encoder A,B,Z from SSI Port to I/O #4,5,6. This is only valid on a controller with an SSI port.
4	Disable Encoder Output
5	Output Internal Encoder A,B, to I/O #4,5
6	Output External Differential Encoder A,B from SSI Port to I/O #4,5. This is only valid on a controller with an SSI port.

NOTE: This command is NOT applicable for SilverNugget TBD, SilverDust MG, SilverDust IG and SilverDust IGB.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EMN SD 30	Program Class D 192 (0xC3) 4 words Thread 1	Mode	S16	2-6 (see above)
		Index State	S16	0 (not used at this time)
		Reserved	U16	0 (not used at this time)

Example

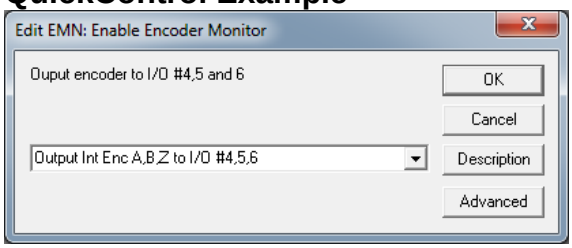
Turn on monitoring of the Internal Encoder.

@16 192 2 0 0 (CR)

Response

ACK only

QuickControl Example



EMT:Enable Multi-Tasking

See Also: DMT:Disable Multi-Tasking

Description

Enables the device's multi-tasking operation, which allows motion while executing a program. By default, the device does not continue internal program execution when performing a motion command or while executing in a mode command (i.e. VMP, VMI, SSD, RSD, VIM, TIM, PIM, PMC,…). Enable Multi-Tasking allows the device to continue thread 1 program execution after a motion command or mode has been started. This is separate from multi-threading which allows more than one program to run concurrently (thread 1 and thread 2).

See Multi-Tasking in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EMT	Program Class D 225 (0xE1) 1 word Thread 1	NONE	NONE	NONE

Example

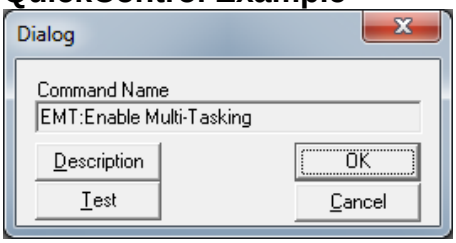
Enable multi-tasking operation

@16 225 (CR)

Response

ACK only

QuickControl Example



Initialization Commands

ERL:Error Limits

Description

The Error Limits command sets allowable position error before the Holding and/or Moving Error bits are set in various status words (see Status Words in User Manual for bit definitions). The Delay to Holding parameter specifies the time the device waits after a move is completed before it goes from moving torque to holding torque (see Torque Limits (TQL) command), and from applying moving error limits to holding error limits. This allows setting a larger window while moving and tightening the error window after the motion has completed and the delay time has exhausted.

A special "Drag" or clutch mode may be implemented by setting the error limits to negative values. The absolute value of the limit is still used to generate Holding and Moving status conditions, but the target is not allowed to get farther than the respective error limit from the servo position. This creates a slip clutch effect. NOTE: Since the Holding and Moving status bits are set in the Internal Status Word (ISW), these bits should be disabled in the kill motor condition commands KMC and/or KMX. See Error Limits and Drag Mode in User Manual for more details.

Note: If you are using QuickControl with the Drag Mode box checked, it will automatically (internally) negate the error limits for you. An error value of zero disables that error check.

Command Info

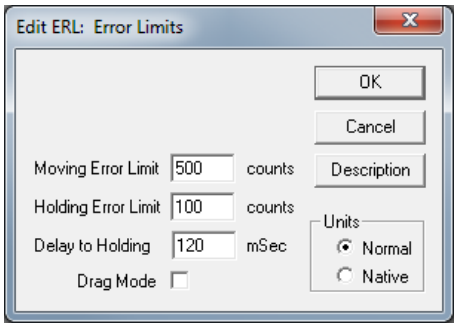
Command	Command Type/Num	Parameters	Param Type	Parameter Range
ERL	Program Class D 151 (0x97) 4 words Thread 1&2	Moving Limit	S16	-32768 to 32767 , Default = 0 QuickControl Default = 20000
		Holding Limit	S16	-32768 to 32767, Default = 0 QuickControl Default = 20000
		Delay to Holding (ticks)	S16	0 to 65535 ,Default = 100 ticks QuickControl Default = 120ms

Example

Allow 500 counts of error while moving and 100 counts of error when holding position. Allow 120 milliseconds before going into Hold mode with its tighter error limit.

@16 151 500 100 1000 (CR)

QuickControl Example



Response

ACK only

ETN:End Of Travel, Negative

See Also: SSL:Soft Stop Limits
ETP:End Of Travel, Positive

Description

End Of Travel, Negative (ETN) allows the user to choose the condition for end of travel in the negative direction. ETN will prevent the Trajectory Generator from commanding additional movement in the negative direction (normally CCW) if the corresponding condition(s) are met.

QuickControl's "standard" implementation of ETN allows the user to specify a single input for the negative end of travel limit. The "advanced" implementation, allows the user to specify multiple conditions (including inputs) from the Internal Status Word (ISW), the Internal Status Word 2 (IS2), and the Extended IO Word (XIO).

Parameter Details for Non-QuickControl Users

The 3 pairs of Enable/State words are for the status words ISW, IS2, and XIO. A “1” in at a particular bit of the Enable word enables the corresponding bit to be checked. The State word determines the level that is considered as “Active” and will prevent motion in the negative direction when enabled by the Enable bit.

When the Trajectory Generator detects any of the selected conditions selected by this command, negative changes (normally CCW) in the target position are prohibited. If a velocity or time based move is underway, the motion will not reach its intended destination. A profile move will continue execution until either terminated or until the cause of the CCW limit has been removed. Note that if an error “windup” has occurred, stopping the trajectory may not immediately stop the motion. Note that the trajectory generator may still operate in the position direction even if ETN is limiting motion in the negative direction.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ETN	Program Class D 67 (0x43) 7 words Thread 1	Condition Enable ISW	S16	0-65535
		Condition State ISW	S16	0-65535
		Condition Enable IS2	S16	0-65535
		Condition State IS2	S16	0-65535
		Condition Enable XIO	S16	0-65535
		Condition State XIO	S16	0-65535

Example

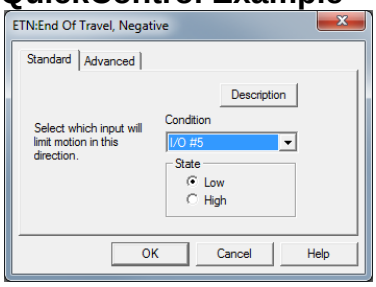
Prevent negative motion if input 5 is low

@16 67 0 0 8192 0 0 0 (CR)

Response

ACK only

QuickControl Example



ETP:End Of Travel, Positive

See Also: SSL:Soft Stop Limits
ETN:End Of Travel, Negative

Description

End Of Travel, Positive (ETP) allows the user to choose the condition for end of travel in the positive direction. ETP will prevent the Trajectory Generator from commanding additional movement in the positive direction (normally CW) if the corresponding condition(s) are met.

This is basically the same command as End Of Travel, Negative (ETN). See ETN for parameters details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ETP	Program Class D 66 (0x42) 7 words Thread 1	Condition Enable ISW	S16	0-65535
		Condition State ISW	S16	0-65535
		Condition Enable IS2	S16	0-65535
		Condition State IS2	S16	0-65535
		Condition Enable XIO	S16	0-65535
		Condition State XIO	S16	0-65535

Example

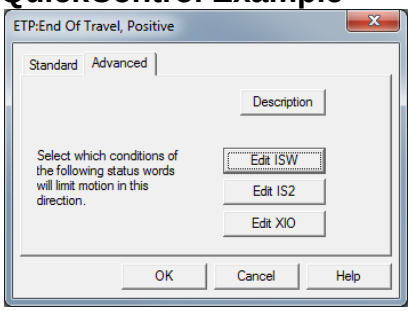
Prevent CW motion if IO1 is high or IO101 is low

@16 66 16 16 0 0 1 0 (CR)

Response

ACK only

QuickControl Example



FLC:Filter Constants

See Also: FL2:Filter Constants 2
CTC:Control Constants, CT2:Control Constants 2

Description

Filter Constants sets the cutoff frequency for the velocity and acceleration filters.

See Technical Document QCI-TD054 Servo Tuning on our website for details.

See Scaling in User Manual for details on converting filter values between Hz and native units. Also see Example below.

QuickControl stores a default set of parameters for each motor type (i.e. 23-3, 23H-1, ...). If "Use Default For Device" is checked, QuickControl will use the default parameters both now and at download time.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
FLC	Program Class D 169 (0xA9) 4 words Thread 1	Fv1: Velocity 1 Feedback Filter	S16	4096 to 32767
		Fv2: Velocity 2 Feedback Filter	S16	4096 to 32767
		Fa: Acceleration Feedback Filter	S16	4096 to 32767

Example

Set filters to roll of at 469, 413 and 117 Hz.

$23000 = 32768\ e^{-(469)2\pi(120uS)}$

$24000 = 32768\ e^{-(413)2\pi(120uS)}$

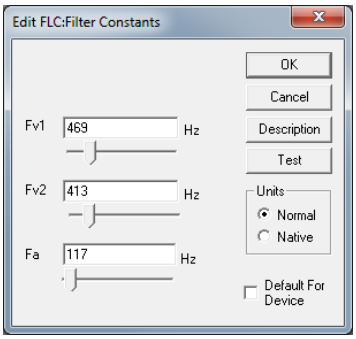
$30000 = 32768\ e^{-(117)2\pi(120uS)}$

@16 169 23000 24000 30000 (CR)

Response

ACK only

QuickControl Example



FL2:Filter Constants 2

See Also: FLC:Filter Constants
CTC:Control Constants, CT2:Control Constants 2

Description

Filter Constants 2 (FL2) changes the servo loops actual velocity and actual acceleration calculations from an "estimator" model to a more efficient "observer" model and enables the addition of a second acceleration feedback term.
FL2 overrides any previous Filter Constants (FLC) command and enables the use of CT2's Acceleration 2 Feedback Gain (Ka2) parameter.

NOTE. The only change to the underlining PVIA™ servo control loop is the addition of a second acceleration feedback term (Ka2) and the method of calculating the velocity and acceleration feedback terms.

See Technical Document QCI-TD054 Servo Tuning on our website for details.

QuickControl stores a default set of parameters for each motor type (i.e. 23-3, X23C-1, ...). If "Use Default For Device" is checked, QuickControl will use the default parameters both now and at download time.

Command Info

Command	Command Type/Num	Parameters	Param Type
FL2	Program Class D 68 (0x44) 7 words Thread 1	Kd: Damping Factor	S16
		Ksi: Stiffness Per Inertia Factor	S16
		Kaa: Anticipated Acceleration Factor	S16
		Fv2: Velocity 2 Feedback Filter	S16
		Fa1: Acceleration 1 Feedback Filter	S16
		Fa2: Acceleration 2 Feedback Filter	S16

Example

Fsi = 472Hz

$22958 = 32768\ e^{-(472)2\pi(120uS)}$

32768-22958 =9810

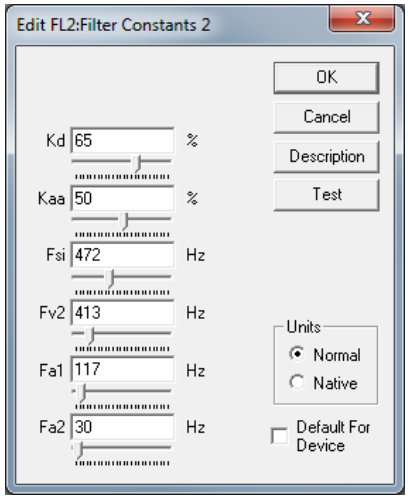
Fv2 = 413Hz,

$24000 = 32768\ e^{-(413)2\pi(120uS)}$

.....

Kaa = 50%
Kd = 65%

@16 68 21255 9810 4905 24000 30001 32035 (CR)



Response

ACK onlyQuickControl Example

GCD:Go Closed Loop – DC motor

Description

Puts non-hybrid servo into closed loop operation. This is typically only done one time during initialization. This command is used to put device into closed loop mode if the unit has been placed into open loop mode. This command DOES NOT sets the phase relationship between the rotor and the encoder for closed loop operation, as this is not needed for DC motors and voice coils. DO NOT USE with stepper / hybrid servo motors, use GCL instead.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
GCD	Program Class D 88 (0x58) 1 word Thread 1	NONE	NONE	NONE

Example

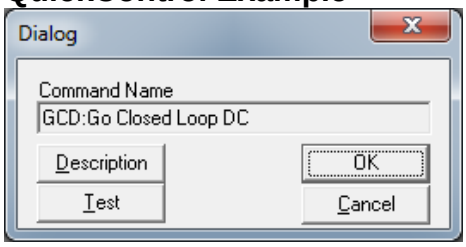
Put device into closed loop mode

@16 88 (CR)

Response

ACK only

QuickControl Example



GCL:Go Closed Loop

Description

Puts the device into closed loop operation. This is typically only done one time during initialization. This command is used to put device into closed loop mode if the unit has been placed into open loop mode. This command sets the phase relationship between the rotor and the encoder for closed loop operation. (See Initialization in the User Manual for more information.)

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
GCL	Program Class D 142 (0x8E) 1 word Thread 1	NONE	NONE	NONE

Example

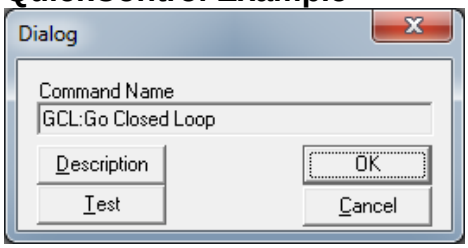
Put device into closed loop mode

@16 142 (CR)

Response

ACK only

QuickControl Example



Initialization Commands

GOC:Gravity Offset Constant

See Also: TQL:Torque Limits

Description

Establishes a value that compensates for the effects of gravity on the load that the servo is driving. This servo control parameter is designed to neutralize the effect of gravity on mechanisms that operate in other than horizontal orientation. It enables the servo control to operate consistently in both directions of servo rotation by creating a torque offset that counters the torque required to hold the load in position. The offset value is in torque units the same as the Torque Limits (TQL) command.

Depending on the direction of the torque applied to the servo shaft, the value can be set to a negative or positive value.

For QuickControl, if the Edit GOC dialog box "Normal" option is checked, QuickControl will automatically translate the percent torque to the native torque units at time of download.

Note: The Gravity offset value allows the system to smoothly switch in and out of Anti-Hunt operation by not requiring the error to build up or the integrator to ramp up to provide the torque needed to hold the load when switching from open loop to closed loop operation (given an appropriate value for the Gravity offset has been configured.)

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
GOC	Program Class D 237 (0xED) 2 words Thread 1	Gravity Offset	S16	-32767 to 32767 Default: 0

Example

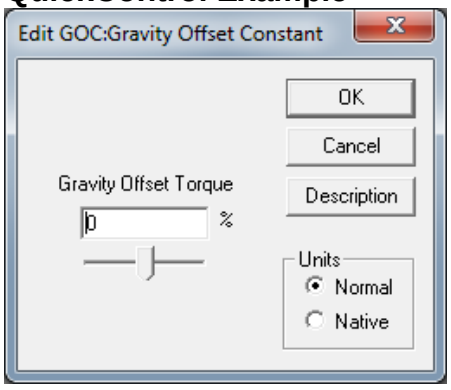
Set the Gravity Offset to 35% Torque for a 23-3 servo

@16 237 7000 (CR)

Response

ACK only

QuickControl Example



GOL:Go Open Loop

Description

Puts the device into open loop operation. This is the default servo power up mode. This command is used during servo initialization to aid in aligning the rotor to the encoder.

The command can also be used to force the servo into open loop mode. This is not recommended for normal operation, as the system performance is severely degraded.

If the servo is in Dual Loop Control (DLC) operation when this command is encountered, it is forced back into Single Loop Control.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
GOL	Program Class D 143 (0x8F) 1 word Thread 1	NONE	NONE	NONE

Example

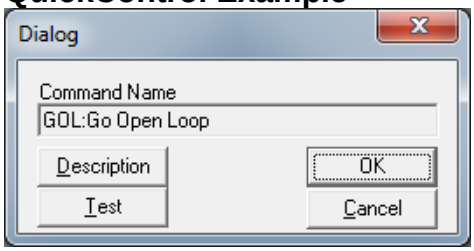
Put device into open loop mode

@16 143 (CR)

Response

ACK only

QuickControl Example



IDT:Identity

See Also: SMD:Set Mode (Respond to Group ID mod)

Description

The Identity command is used to set the Unit ID and Group ID addresses to which the device will respond. The device will accept and respond to any command addressed with the Unit ID. The device will accept commands sent to either the Group ID or to the Global ID (255), but no response will be sent as multiple units cannot respond at the same instant. No two units should have the same Unit IO when connected on the same network. Multiple units may share a common Group IO when they are on the same network. Do not set Unit ID and Group ID to the same value. Group ID may be set to zero (disabled) if not needed.

Note: If SMD:Respond to Group mode is enabled, the unit so configured will respond to both its own ID and to the Group ID. This allows 3rd party HMI's to receive a response to a message sent to the Group ID. Only one unit in the group should be so configured.

Identities need to be in the range of 1 to 254.

Set To ID Rotary Switch Checkbox (D2-IG8)

If checked on a controller with an ID Rotary Switch, the Unit ID field is set to 255 which causes the actual Unit ID to be calculated from the rotary switch and the Group ID using the following table. This allows a user to change the Unit ID without re-programming the device.

Unit ID as a function of ID Rotary Switch Setting and Group ID Table

	ID Rotary Switch Setting															
Group ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0
0 to 15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
16 to 31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
32 to 47	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
48 to 63	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
64 to 79	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
80 to 95	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
96 to 111	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
112 to 127	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
128 to 143	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
144 to 159	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144
160 to 175	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
176 to 191	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176
192 to 207	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
208 to 223	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208
224 to 239	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224
240 to 255	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240

Note: Group ID's evenly divisible by 16 should be avoided to prevent having the same Group ID and Unit ID.

Note: The ID Rotary Switch value is available via the lower 4 bits of CAN object 200Ah.

See Technical Document QCI-TD053 Serial Communications on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
IDT	Program Class D 155 (0x9B) 2 words Thread 1&2	Group/Unit ID Group ID = Upper Byte Unit ID = Lower Byte	U16	257 to 65278 Unit ID = 255: Unit ID=ID Rotary Switch Default: Unit ID=16 Group ID=20

Example

To Calculate number: Multiply the Group Identity times 256, then add the Unit Identity

Group = 20, Unit = 16

Identity = (20 * 256) + 16 = 5136

Group Identity of 20, Unit Identity of 16;

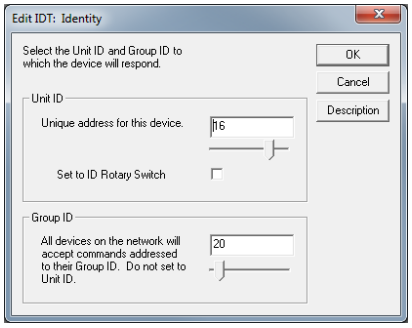
@16 155 5136 (CR)

Response

ACK only

If this command is sent in Immediate Mode, the response will be with the new Unit ID.

QuickControl Example



KDD:Kill Disable Driver

See Also: KED:Kill Enable Driver

Description

Disables the motor driver, when a Kill Motor Condition is met. If the device is moving, it will stop immediately in a rapid manner. The motor will be unable to move until re-enabled using the Enable Motor Driver (KMD) command. This is the default setting for the servo.

See Technical Document QCI-TD052 Shutdown and Recovery on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
KDD	Program Class D 183 (0XB7) 1 word Thread 1	NONE	NONE	NONE

Example

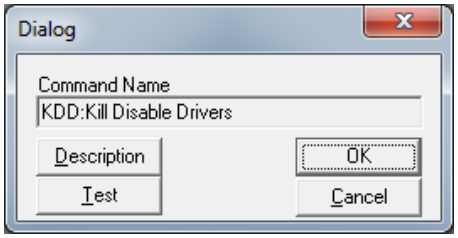
Disable the Motor Driver electronics when Kill Motor Conditions are met

@16 183 (CR)

Response

ACK only

QuickControl Example



KED:Kill Enable Driver

See Also: KDD:Kill Disable Driver
EMT:Enable Multi-Tasking

Description

Causes the device to leave the motor drivers enabled when a Kill Motor Condition is met. Normally the motor driver is disabled with a Kill Motor Condition, this command can be used to leave the driver enabled if continuing operation is required.

In order for this command to function, the device must be set up for multi-tasking operation. Without multi-tasking, the driver will be disabled when a Kill Motor Condition occurs.

See Technical Document QCI-TD052 Shutdown and Recovery on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
KED	Program Class D 182 (0xB6) 1 word Thread 1	NONE	NONE	NONE

Example

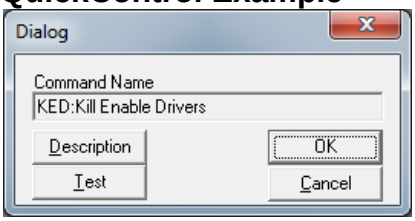
Leave the motor driver enabled

@16 182 (CR)

Response

ACK only

QuickControl Example



KMC:Kill Motor Conditions

See Also: KMR:Kill Motor Recovery
KMX:Kill Motor Conditions Extended

Description

The Kill Motor Conditions allows the user to select what conditions will allow a controlled shutdown of the unit. The Condition Enable word selects which bits in the Internal Status Word (ISW) will be evaluated (See Internal Status Word (ISW) in User Manual for bit definitions). Conditions are enabled by setting a "1" in the desired bit position of the Condition Enable binary word. See KMX for more kill motor conditions.

The Condition State word allows the user to specify the state of the selected conditions that will cause the device to do a controlled shutdown. Note: Over-voltage is always enabled whenever the driver is enabled to protect the drivers from over voltage. An over-voltage condition will always disable the drivers regardless of the of Kill Enable Drivers state. (SilverMax-X series will short the windings during the duration of the over voltage to prevent excessive regeneration.)

See Technical Document QCI-TD052 Shutdown and Recovery on our website for details.

Default has only Over Temperature enabled.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
KMC	Program Class D 167 (0xA7) 3 Words Thread 1	Condition Enable	U16	0 to 65535
		Condition State	U16	0 to 65535

Example

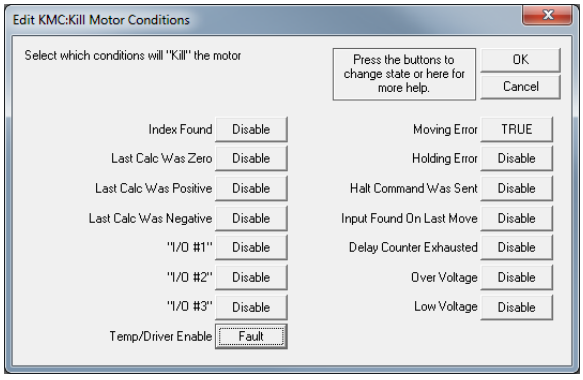
Shut down servo if any of the following conditions are met:
I/O#1 LOW (bit 4)
Over Temp (bit 7)
Moving Error (bit 8)

NOTE: Over Temp TRUE = 0.

Enable = $2^4 + 2^7 + 2^8 = 400$
State = $2^4*0 + 2^7*0 + 2^8*1 = 256$

@16 167 400 256 (CR)

QuickControl Example



Response

ACK only

KMX:Kill Motor Conditions Extended

See Also: KMR:Kill Motor Recovery
KMC:Kill Motor Conditions

Description

The Extended version of Kill Motor Conditions (KMC) provides 3 status and I/O words of conditions that may be selected to allow a controlled shutdown of the unit. The three Condition Enable words selects which bits in the respective registers will be evaluated; a "1" state is set for each bit which is to be evaluated, and a "0" bit for those bits which are to be ignored. The three words are ISW, IS2 and XIO. See User Manual for bit definition.

The Condition State word allows the user to specify the state of the selected conditions that will cause the device to do a controlled shutdown. Note: Over-voltage is always enabled whenever the driver is enabled to protect the drivers from over voltage. An over-voltage condition will always disable the drivers regardless of the of Kill Enable Drivers state. (SilverMax-X series will short the windings during the duration of the over voltage to prevent excessive regeneration.)

See Technical Document QCI-TD052 Shutdown and Recovery on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
KMX	Program Class D 220 (0xDC) 7 Words Thread 1	Condition Enable ISW	U16	0 to 65535
		Condition State ISW	U16	0 to 65535
		Condition Enable IS2	U16	0 to 65535
		Condition State IS2	U16	0 to 65535
		Condition Enable XIO	U16	0 to 65535
		Condition State XIO	U16	0 to 65535

Example

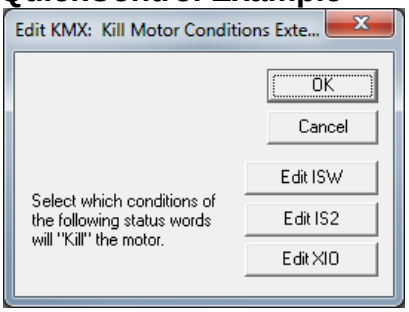
Shut down servo if:

I/O#1 LOW (bit 4)
Over Temp (bit 7)
Moving Error (bit 8)
NOTE: Over Temp TRUE = 0.

Enable ISW = $2^4 + 2^7 + 2^8 = 400$
State ISW = $2^4*0 + 2^7*0 + 2^8*1 = 256$

@16 167 400 256 0 0 0 0(CR)

QuickControl Example



Response

ACK only

KMR:Kill Motor Recovery

See Also: Kill Motor Conditions (KMC), Kill Motor Extended (KMX)

Description

Kill Motor Recovery sets up options for recovery from a device shut down. The Kill Motor Conditions (KMC) and Kill Motor Extended (KMX) establish conditions that will cause the device to shut down. Using Kill Motor Recovery the device can perform a standard or user defined process for re-initializing the device. User programs can be executed that have been previously stored in the non-volatile memory. (See Kill Motor Conditions for more detail).

Three options available:

1. "0" – Default: No recovery program designated. The device drops out of any motion or program that is currently executing and goes into an idle state. The drivers are disabled. At this point the device will sit with no current to the device.
2. "-1" – Normal operation: -1 is a special parameter value indicating to run the initialization program from non-volatile memory location "0"
3. "####" – Normal operation: The routine located at #### is loaded and executed.

NOTE: If QuickControl is polling the device when the shutdown occurs, it will display the cause of the fault providing the KMR program does not clear it too quickly. Because of this, it is recommended that the KMR program have short delay in it before clearing the fault. QCI suggests 100ms/axis.

See Technical Document QCI-TD052 Shutdown and Recovery on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
KMR	Program Class D 181 (0xB5) 2 words Thread 1	Process	S16	0 = Do Nothing -1 = Load and Run Program @ NV Mem adr 0. #### = Load and Run Program @ indicated NV Mem adr.

Example

After motor shutdown load and run "Fault Recovery" program which is stored at 542.

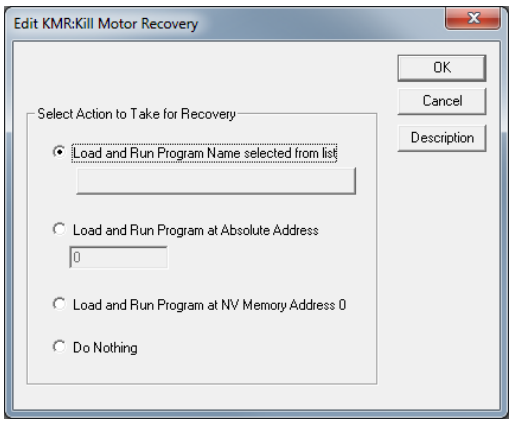
NOTE: In QuickControl, the user only needs to specify the program name. The address is calculated automatically.

@16 181 542 (CR)

Response

ACK only

QuickControl Example



LVP:Low Voltage Processor Trip

See Also: PLR:Power Low Recovery
LVT:Low Voltage Trip

Description

This command is only usable with units that provide separate power supply inputs for the processor and for the driver sections. This command allows the monitoring of the processor power supply for low voltages in the same way that a Low Voltage Trip (LVT) command monitors the driver (or, for single supply motors, the main power supply).

This command sets the input voltage that will trigger a Low Voltage status (Bit #14 in the Internal Status Word (ISW)) and subsequently the Power Low Recovery (PLR) routine (if configured). When a Low Voltage Processor Trip occurs the low voltage trip values, both driver and processor; are overwritten to zero to prevent multiple triggering.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
LVP	Program Class D 131 (0x83) 2 Words Thread 1	Voltage	U16	0 = Don't Check 10 to 48 Default: 0

Example

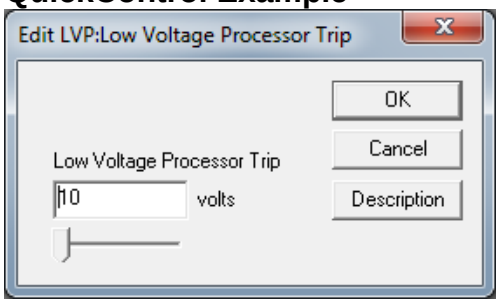
Set LVP to 10 volts

@16 131 10 (CR)

Response

ACK only

QuickControl Example



LVT:Low Voltage Trip

See Also: LVP:Low Voltage Processor Trip

Description

This command sets the input voltage (or driver Input voltage for units that have dual input power supplies) that will trigger a Low Voltage status (Bit #14 in the Internal Status Word (ISW)) and subsequently the Power Low Recovery (PLR) routine (if configured). When a Low Voltage Trip occurs the low voltage trip values associated with the Low Voltage Trip and Low Voltage Processor Trip commands are overwritten to zero to prevent multiple triggering.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
LVT	Program Class D 212 (0xD4) 2 Words Thread 1	Voltage	U16	0 = Don't Check 10 to 48 Default: 10V

Example

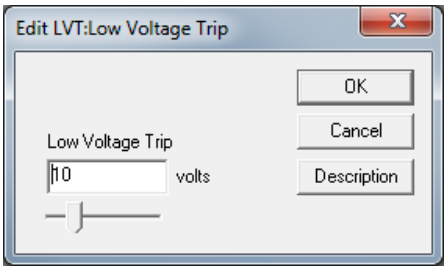
Set shut down at 10 volts

@16 212 10 (CR)

Response

ACK only

QuickControl Example



MCT:Motor Constants

Description

These constants are factory supplied for the selected motor at the requested power supply voltage. Normally these are set using QuickControl's Initialization Wizard. Executing this command also causes the motor driver to be “Enabled”.

The Edit MCT dialog box gives the user the following options:

- Auto: QuickControl will read the servo's voltage and line resistance (line resistance stored in servo during Initialization Wizard) at download time and set the parameters accordingly. This is the recommended default setting.
- Manual: The user selects the voltage. This option is useful when the voltage in the field is different than the voltage at time of download. Line resistance is still read from the servo at download time.
- Native: An advanced mode that should only be used at the direction of QuickSilver Controls.
- Custom Motor: Calculates motor parameters for a user supplied motor from properties found on the motor’s datasheet. See Custom Motors in User Manual for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MCT	Program Class D 168 (0xA8) 9 Words Thread 1	MC1	S16	0 to 32767
		MC2	S16	0 to 32767
		MC3	S16	0 to 32767
		MC4	S16	0 to 32767
		MC5	S16	0 to 32767
		MC6	S16	0 to 32767
		MC7	S16	0 to 32767
		MC8	S16	0 to 32767

Example

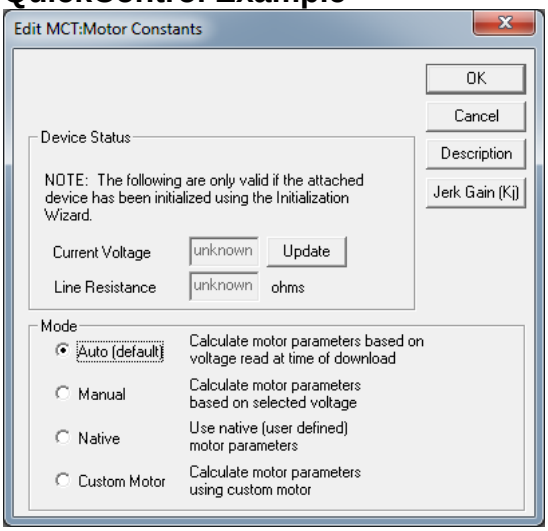
Set up a 23-5 for 24 volt operation

@16 168 1631 14843 31816 2057 1758 2329 32767 8213 (CR)

Response

ACK only

QuickControl Example



MTT:Maximum Temperature Trip

Description

Sets the temperature at which the device will shut down the servo. This is used to prevent internal over-heating of the servo electronics. The value is entered in degrees Celsius integer units. (Example "70" for 70 degrees Celsius). The maximum temperature error condition is OR-ed with the motor driver over temperature condition. Either active will cause an Over Temperature status condition in the Internal Status Word. The temperature can be read using the ANALOG READ INPUT command. Note that the SilverSterling, SilverMax – X and SilverNugget – X read the internal CPU temperature which is about 15C higher than the internal ambient temperature once temperatures have stabilized.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MTT	Program Class D 214 (0xD6) 2 words Thread 1	Temperature (°C)	U16	0 = Don't Check 1 to 100 Default: 0

Example

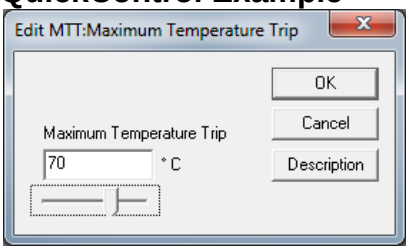
Set Servo to give an error at 70 degrees C

@16 214 70 (CR)

Response

ACK only

QuickControl Example



OLP:Open Loop Phase

Description

The Open Loop Phase is used to set initial motor phase prior to doing motor/encoder alignment. It is primarily used as an element in the algorithmic motor to encoder alignment routine. For positive values, this represents the micro-step position of the motor. Negative values are used to select half step positions – i.e. –2 sets the motor at 1 full step away from a value of 0. Please see the Initialization section in the User Manual for a more detailed description of the initialization process. This command is normally used only in the initialization procedure.

The use of negative values makes the operation independent of encoder resolution.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
OLP	Program Class D 152 (0x98) 2 words Thread 1	Phase Angle Count 1000 line encoder: 2000 line encoder: 4000 line encoder:	S16	 -7 to 79 -7 to 159 -7 to 319

Example

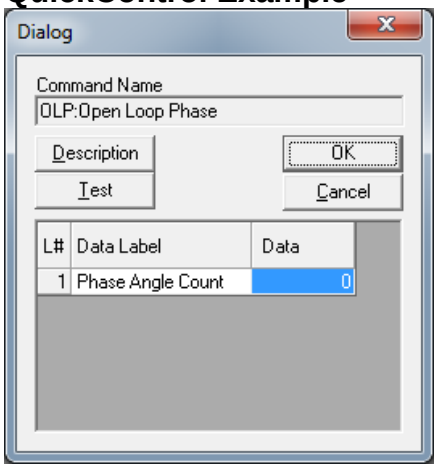
Set the open loop phase to “0”

@16 152 0 (CR)

Response

ACK only

QuickControl Example



OVT:Over Voltage Trip

Description

Sets the voltage at which the device will cause a motor shutdown. This command is mainly used to prevent over-voltage from the power regenerated during deceleration. The voltage value is entered in integer units (example: “48” for 48 volts). If an over-voltage condition is detected, a motor shutdown is executed that disables the motor driver to reduce regenerated power flowing into the power supply input which boosts the supply voltage. The SilverMax – X and SilverNugget – X short the motor windings while the voltage exceeds the limit to prevent further regeneration.

NOTE: The Kill Enable Driver (KED) command does not allow the motor driver to stay enabled when an Over Voltage Trip occurs. This condition always disables the motor driver.

The motor driver is disabled when this condition occurs and must be re-enabled using the Enable Motor Driver (EMD) command or by re-writing the Motor Constants (MCT).

The factory default is set at 52 volts. A power supply voltage that exceeds 52 volts may cause the motor to shutdown at power up. Unregulated power supplies with excessive voltage ripple can cause an over voltage trip, even though an average reading meter may report the voltage as within specification. The over voltage trip may also activate when doing rapid decelerations with large inertias, or using the device as a clutch without using a Clamp Module between the device and the power supply. (Note: the I-Grade SilverDust units have the clamp built in.)

In QuickControl, if Automatic is selected OVT will be set to 4V above the voltage used by the most recent MCT command. This is only determined at time of download.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
OVT	Program Class D 213 (0xD5) 2 words Thread 1	Voltage	U16	1 to 53 (52 = Default)

Example

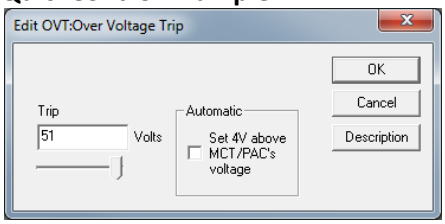
Shut down the Motor if the input voltage exceeds 52 volts

@16 212 52 (CR)

Response

ACK only

QuickControl Example



PAC:Phase Advance Constants

See Also: MCT:Motor Constants

Description

Sets the motor phase advance constants. These are motor type and power supply voltage dependent to optimize motor torque at high speed. Factory set for optimal performance. While PAC1 and PAC3 are basically constant for all motor types and voltages, PAC2 changes with both. Typically lower voltages requires a larger PAC2.

PAC uses the same QuickControl dialog box as MCT. For more details, see MCT.

The MCT and PAC commands are tightly coupled. When either of these commands is edited in a QuickControl QCP file, the other command is automatically updated.

Note: It is not recommended for user to alter this command from default. If there's a need to change, please consult with QuickSilver's Technical Support first.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PAC	Program Class D Code (Hex): 172 (0xAC) Thread 1	PAC1	S16	0 to 32767
		PAC2	S16	0 to 32767
		PAC3	S16	0 to 32767

Example

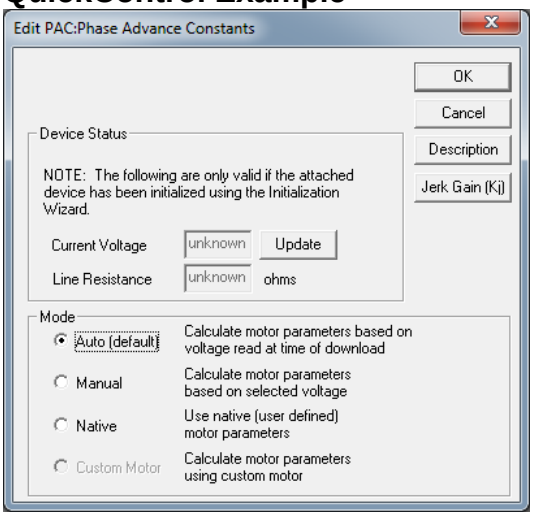
Phase advance for a 23-5

@16 172 5 160 37 (CR)

Response

ACK only

QuickControl Example



PLR:Power Low Recovery

See Also: LVT:Low Voltage Trip
LVP:Low Voltage Processor Trip

Description

This command designates which program will run if the power supplies voltage drops below that specified by the Low Voltage Trip (LVT) command or Low Voltage Processor Trip (LVP) commands.

The QuickControl edit PLR dialog box has four options:

1. Load and Run Program - Select a PLR program.

2. Load and Run Absolute Address - Enter the non-volatile memory address of the program you want to load and run for the PLR.

3. Load and Run Program at NV Memory Address 0 - Load and run the program stored at 0. By default, this is the initialization program.

4. Do Nothing – This default state indicates that no recovery program has been designated. The device drops out of any motion or program that is currently executing and goes into an idle state. Note: Bit 14 of the ISW word is set by the low voltage activity, and, if enabled, the Kill Motor Recovery will handle this condition.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PLR	Program Class D 208 (0xD0) 2 Words Thread 1	Process	S16	0 = Do Nothing -1 = Load and Run Program @ NV Mem adr 0. #### = LRP @ NV Mem adr.

Example

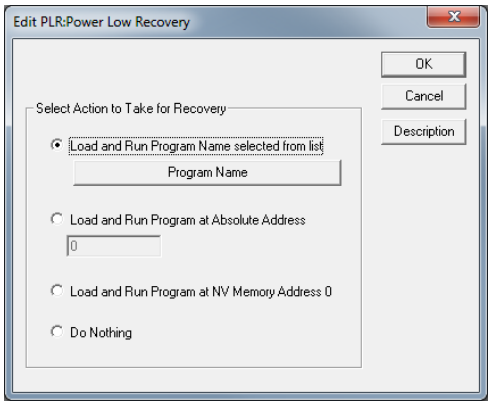
If power low condition exists load and run "Program Low Recovery" program which is stored at 568.

@16 208 568 (CR)

Response

ACK only

QuickControl Example



PRO:Protocol

Description
Allows the user to select the desired communications protocol.

If this command is sent in Immediate Mode, the response will be in the new protocol.

The lower byte of the parameter selects the desired protocol, while the upper byte selects the serial configuration. Note, for QCI 9 bit and DMX512 protocols, the serial configuration must be 2 stop bits and no parity.

See Technical Document "QCI-TD053 Serial Communications" on our website for more details on this command.

See Application Note "QCI-AN038 Modbus Protocol" for details on communicating with a Modbus® device including an example program.

See Application Note "QCI-AN045 DMX512 Protocol" for details on communicating with a DMX device.

Command Info

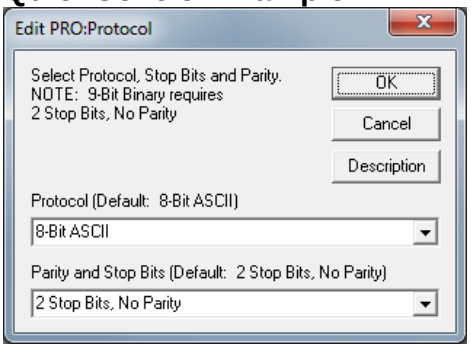
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PRO	Program Class D 185 (0xB9) 2 words Thread 1&2	Parity&Stop | Mode	S16	Mode (lower byte) 0 = 9-Bit 1 = 8-Bit (Default) 2 = Modbus® 3 = DMX512 Parity &Stop Bit 15: Stop bits, 0=>2bits, 1=>1 bit Bit 14: Enable Parity, 0=none, 1=enabled Bit 13: Odd Parity, 0=even, 1=odd

Example
Select the 8-Bit ASCII Protocol

@16 185 1 (CR)

Response
ACK only

QuickControl Example



SCF:S-Curve Factor

Description

The shape of motion profile acceleration can be set from linear to full s-curve. This command can be set at any time except for during a motion. SCF only affects the basic motion commands and their register based derivations (MRT, MRV, …).

SCF is not available in the Step & Direction (i.e. SSD), Profiled Move (i.e. PMC) , Input Mode (i.e. PIM) or the Velocity modes (i.e. VMP).

See S-Curve in User Manual for more information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SCF	Program Class D 195 (0xC3) 2 words Thread 1	Factor	S16	0 = Trapezoidal 1 to 32766 = s-curve 32767 = Full s-curve Default: 0

Example

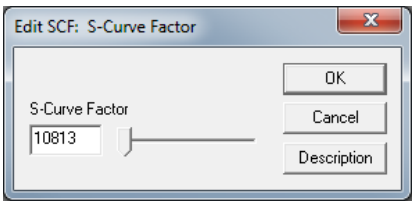
Use some S-Curve.

@16 195 10813 (CR)

Response

ACK only

QuickControl Example



SIF:Serial Interface

Description

Allows the user to select between RS-232 and RS-485 serial communications hardware interface. This command is usually used at power up as part of the initialization program. Care should be taken when using this command, as communications may be lost if the host controller is not compatible with the new hardware setting. Only 485 is allowed in SilverSterling and X-series SilverMax and SilverNugget.

QuickControl will automatically set this parameter at download if the box "Set to SIF currently being used by device" is checked. At download, QuickControl asks the device whether it is in RS-232 or RS-485 and then sets the SIF command accordingly. For RS-232 multi-drop, uncheck the box, set SIF to RS-232 and set ACK Delay (ADL) to some non-zero value (i.e. 5).

If this command is sent in Immediate Mode, the response will be in the new interface.

See Technical Document QCI-TD053 Serial Communications on our website for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SIF	Program Class D 186 (0xBA) 2 words Thread 1&2	Mode	S16	0 = RS-232 (Default) 1 = RS-485

Example

Set up the device to use RS-232 for the serial interface

@16 186 0 (CR)

QuickControl Example

Response

ACK only

SEF:Select Encoder Filter

Description

Selects the desired digital filter for the external encoder signals. The default (0)is a 150nS filter for SilverDust and 100ns filter for X-Series. The other option is a 300nS for the SilverDust/X-Series SilverNuget and SilverMax. The increased filter time may help applications using the external encoder or step/direction inputs in a noisy environment. The filter is applied to each external encoder interface line (all 3 I/O usable at same time). This filter only affects the external (secondary) encoder count, not any I/O that may also be looking at these same lines. This command is not available for SilverSterling as no external encoder input are available.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SEF SS – N/A	Program Class D 130 (0x82) 2 words Thread 1&2	Filter Enable	S16	X-series 0 = 100ns (Default) 1= 300nS SilverDust 0 = 150nS (Default) 1 = 300nS

Example

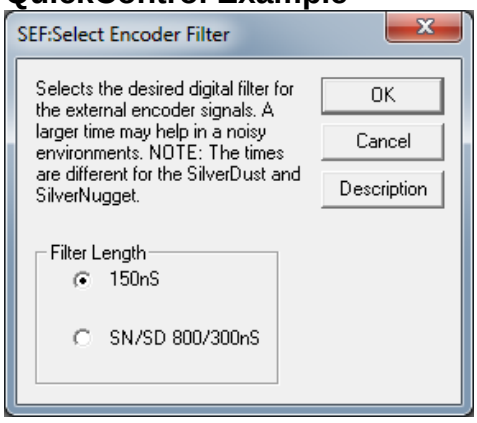
Set up the external (secondary) encoder filter for 800nS:

@16 130 1 (CR)

Response

ACK only

QuickControl Example



SLC:Single Loop Control

See Also: SEE:Select External Encoder
DLC:Dual Loop Control

Description

Configures the device to run in the standard single loop control mode. Encoder information for commutation, position, velocity and acceleration control is derived from the Internal Encoder.

If a motion is running, the servo Trajectory Generator must be shut down prior to executing this command or an error will result.

When entering single loop control, the device sets the current “Target” to the “Current position” (Internal Position from the Internal Encoder).

By default, the device starts up in Single Loop Control mode.

See the Dual Control Loop (DLC) command for cases where external encoder position control is required. Switching between Single Loop and Dual Loop modes usually requires changing the control loop tuning.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SLC	Program Class D 244 (0xF4) 1 word Thread 1	NONE	NONE	NONE

Example

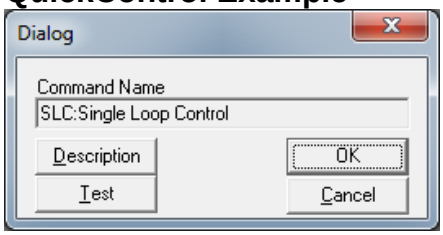
Configure for Single Loop Control

@16 244 (CR)

Response

ACK only

QuickControl Example



SMD:Set Mode

Description

Allows setting of various Mode Bits for operation. Additional modes will be added as needed. Some to these are defined as their own command name for easier use.

Modes

Mode 0: Clear/Set CAN limit Switches
Data value of 0 disables all limit switches, while a Data value of 1 sets the positive and negative limit switches to conform to the switches configured by Dictionary Object 2004h. See SilverLode CANopen User Manual for details. Only available on units with CAN.

Mode 1: Respond to Group ID Commands
Allows the user to configure a unit to respond to both its Group ID as well as to its Unit ID. This allows 3rd party HMI devices to get a response when sending a command to a group address. Only one unit in each group should have the mode enabled. Note: this operation only pertains to the 8-bit ASCII and the 9-bit Binary protocols. It does not apply to DMX or Modbus.

Mode 2: DC Motor Mode
Enable to drive DC motor/solenoid rather than for a stepper motor.
0=Stepper/hybrid servo, 1=DC motor or voice coil

Mode 3: Analog Feedback Mode (SilverDust only)
Use specified analog input as position feedback instead of motor encoder.

- 0: Disable
- 1: Position=IO6, Velocity=IO7
- 2: Position & Velocity from IGH inputs (not available on SilverSterling or X-Series)
- 3: Position=IO7, Velocity calculated from position
- 4: Position=IGH input, Velocity calculated from position (Not available on SilverSterling or X-Series)

Mode 4: Analog Input Filter (SilverDust only)
Sets analog filters for IO4-IO7 (analog inputs #1-#4). This is also used for the Analog Feedback Mode analog inputs (see Scaling, Filter in User Manual).

Mode 5: DC Motor PWM Filter
Sets analog filter on DC motor drive voltage (see Scaling, Filter in User Manual).

Mode 6: Analog velocity crossover (SilverDust only)
Sets Analog Velocity crossover frequency. Velocity estimate for frequencies higher than the crossover are derived from the analog velocity channel while lower frequency components are derived from the position feedback channel. This is provided to minimize differencing noise in the velocity estimate from position feedback at higher frequencies while removing offset voltage/drift issues from the analog velocity channel at lower frequencies (and DC).

(SilverDust) Mode 7: Read Pulse Width from IO6,107 ** different for SilverSterling and X-Series**
Parameter selects pre-scale value needed to reduce the total counts to less than 65535 for slower pulses. The actual prescale value is 2^n. A value of 0 restores register 200, 201 to being

external encoder counts. Register 200 holds the positive pulse time period for IO 106, Register 201 holds the positive pulse time period for IO 107. These correspond to the number CPU clock cycles divided by the prescaler value for the positive pulse (rising edge to falling edge).

(SilverSterling, X-series) Mode 7: Enable Analog feedback from selected analog, polynomial adjust (optional)

0=Disable, 1=analog feedback from IO4, 2=analog feedback from memory port, 3=polynomial adjust from IO4 ,4=polynomial adjust from analog memory port

Mode 8: Select filter for analog feedback (SilverSterling only)

Allow independent filtering for the selected feedback from Mode 7

Mode 9: Read PWM input IO1 as count into Register 200, with prescale (SilverSterling Only)

Parameter is prescale exponent – prescale = 2^n, a value of 0 disables this function

Mode 11: Enable 3 phase motor operation (0=disable, 1=enable) (SilverSterling only)

Mode 12: Enable Mosolver operation (0=disable, 1=enable) (SilverSterling only)

Mode 13: Enable Secondary Encoder with Mosolver (0=disable, 1=enable) (Silver Sterling only)

Note: needs internal jumpering from IO lines to encoder, and encoder receiver disable jumpered. Contact factory for special order.

Mode 15: Enable non-zero start and stop speeds for profiled move command

0=disable, 1=enable. Register 25 becomes starting speed, 26 becomes stopping speed allowing non-zero start and stopping speeds. See notes on PMV. This is also documented as command PMZ, although it is actually implemented as a mode command

Mode 20: Configure second SPI port, set clock rate, no clock delay (SPI memories) (SS 34 ; SX,MX 29)

0 = I/O, ;1= 100kHz, 2 = 250kHz, 3 =500kHz, 4 = 1MHz, 5 = 2MHz, 6 = 3MHz

Mode 21: Configure second SPI port, set clock rate, with clock delay (typical of IO) (SS 34 ; SX,MX 29)

0 = I/O, ;1= 100kHz, 2 = 250kHz, 3 =500kHz, 4 = 1MHz, 5 = 2MHz, 6 = 3MHz

Mode 22: Delay read until timeout or SOMI goes low (SilverSterling only) (SS 34 ; SX,MX 29)

Parameter is the number of ticks to wait Note: also pulses Chip select each 120us cycle to keep alive a watchdog for Bridge amplifier ADC board.

Note: Some of the other modes are used in customer specific code versions and are RESERVED for general applications.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SMD	Program Class D 86 (0x56) 3 words	Mode	U16	0
		Data	S16	see above

Example

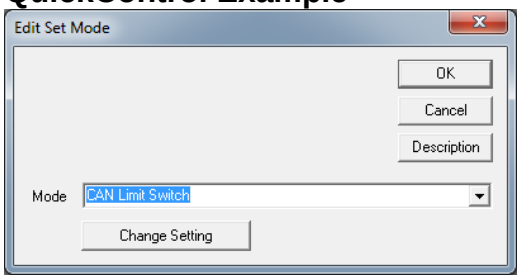
Configure for Respond to Group ID

@16 86 1 1 (CR)

Response

ACK only

QuickControl Example



SSI:SSI Port Mode

See Also: SEE:Select External Encoder

Description

Configure the Synchronous Serial Interface (SSI) Port Mode. SSI port is available as on option on selected SilverDust controllers. SSI Port connects to third party devices that support SSI function. Common uses for the SSI port are absolute encoders. This port may also be used to output internal single ended or differential quadrature encoder signals, as well as to input differential encoder signals (driving IO 4,5,6) through software selection. See SSI and SEE commands.

Mode

0 Input Differential A,B,Z from SSI port (must be configured via SEE to use).
NOTE: This is the power up default. That is, if the controller has an SSI port, the differential encoder signals (A,B,Z) may be wired to the port and will be available via the SEE command even if the SSI command is not executed.

1 Output Internal Encoder A,B,Z to SSI Port

2 Input Encoder (or other SSI compatible device) from SSI Port to Register 253

3 Input Encoder from SSI Port for use in dual loop control (see DLC command).

Options

Bits:0-4: Resolution: 0=not used, 8-31 bits
Note: Resolution is only used for SSI compatible inputs.
Bit:5: SSI data in Gray Code (QuickControl. press Advanced button to set).

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SSI SD 30 (SilverDust D2-IG8 only)	Program Class D 92 (0x5C) 4 words Thread 1&2	Mode	U16	See Above 0 = Default
		Options	U16	See Above
		Reserved	U16	0 (Reserved) Not used at this time

Example

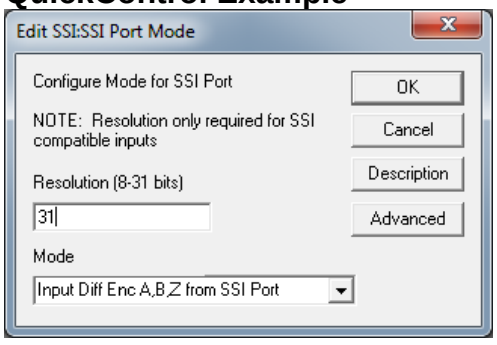
Input 31 bits from SSI encoder

@16 92 2 31 0(CR)

Response

ACK only

QuickControl Example



SSL:Soft Stop Limits

See Also: ETN:End of Travel, Negative, ETP:End of Travel, Positive

Description

Sets position limits for "End of Travel" control. Two registers are used to store the end limits. End of travel positions must be stored in the specified registers. The register selection sets aside two registers in succession. Any motion affecting the Target is limited so as to keep the target more than the first register value and less than the second register value. If the move parameter is beyond a limit, only motion in the direction toward that limit is allowed. The motion exceeding a given limit is ramped down to the point that the limit is encountered. Internally, the motion calculations continue, but their effect is not directed to the Target value. NOTE: This command affects the move commands only, not direct writing to Target Position register.

IS2:Bit 1 is set when SSL limits motion. Note, this is set as soon as a move command executes if that command attempts to move beyond a limit.

The limits consider the position as "Linear" rather than "Cyclic". If the position attempts to wrap-around (going past the full range values), the Soft Stop Limits will prevent this movement.

The first register is used for the lower limit, which is checked when the direction of a motion is negative. The second register is used for the upper limit, which is checked when the direction is positive. If the limits are set so that the Target is outside of the permitted range, only motions toward the permitted range are effective.

If the Lower Limit is set more positive than the Upper Limit, this will create a Dead Zone. If the servo's position is in the Dead Zone, it will not be able to move. No error checking is done on the Data Register values to prevent this condition.

Set Data Register parameter to 0 to disable SSL (QuickControl: Check "Disable Soft Stop Limits").

Command Info

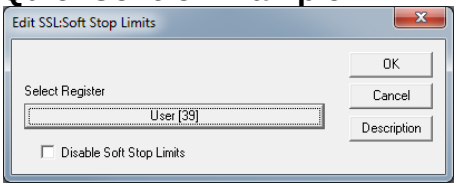
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
SSL	Program Class D 221 (0xDD) 2 words Thread 1&2	Starting Data Register (First of two)	U16	10 to 198 0 = Not Used In QuickControl check "Disable .."

Example

The device uses Data Registers 39 & 40 for end a travel position limits

@16 221 39 (CR)

QuickControl Example



Response

ACK only

T2K:Thread 2 Kill Conditions

See Also: T2S:Thread 2 Start

Description

Determines which conditions are excluded from causing a shutdown of Thread 2. By default, all of these conditions will shut down thread 2 unless excluded by use of the T2K command. Setting the corresponding bit to 1 will exclude the condition, setting the corresponding bit to 0 will allow the condition to shutdown Thread 2.

See Multi-Thread Operation in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
T2K	Program Class D Code (Hex): 77 (0x4D) 2 words Thread 1	Exclusions	U16	Bit 0 => Kill Motor Bit 1 => Over Voltage Driver Bit 2 => Under Voltage Driver Bit 3 => Under Voltage Processor Bit 4 => Halt Command Bit 5 => Stop Command Bits 6..15 Reserved

Example

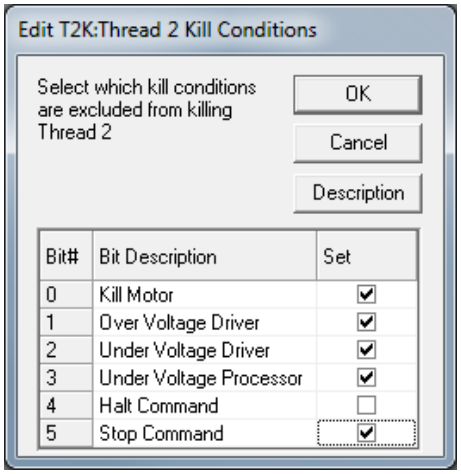
Configure Thread 2 to survive all but a Halt command. (Bits 0, 1, 2, 3, 5 set)

@16 77 0x2F (CR)
or
@16 77 47 (CR)

Response

ACK only

QuickControl Example



TQL:Torque Limits

Description

This command sets the torque limits for the different operating modes of the servo. The unit may be in either Open Loop or Closed Loop mode, and in either Moving or Molding mode. The four parameters supplied set the limits on the output torque for all four combinations: Closed Loop Holding, Closed Loop Moving, Open Loop Holding, and Open Loop Moving.

See Technical Document QCI-TD051 Torque Control on our website for details on this command.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
TQL	Program Class D 149 (0x95) 5 words Thread 1&2	Closed Loop Holding	U16	0 to 32767
		Closed Loop Moving	U16	0 to 32767
		Open Loop Holding	U16	0 to 32767
		Open Loop Moving	U16	0 to 32767

Example

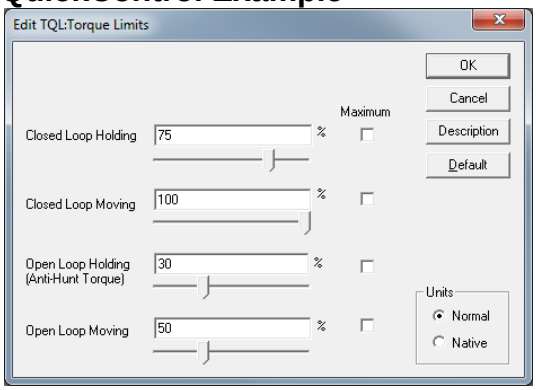
Set torque to:
Closed Loop Holding 75%
Closed Loop Moving 100%
Open Loop Holding 30%
Open Loop Moving 50%

@16 149 15000 20000 6000 10000(CR)

Response

ACK only

QuickControl Example



TRU:Torque Ramp Up

Description

Ramps up the torque limit values by the increment given up to the final value. This is used mainly during initialization. Only ramps up open loop torque limits. This command slowly brings up the Open Loop motor current to avoid a harsh or sudden movement during servo power up. This is done just prior to the algorithmic alignment of the motor rotor to the encoder. The increment sets how much current will be added each servo cycle (120usec).

The ramp up time is calculated by taking the final value divided by the increment times 120usec.
Example
20000/5 = 4000
4000 * 0.00012 = 480 milliseconds

For QuickControl, if the Edit TRU dialog box "Normal" option is checked, QuickControl will automatically translate the percent torque to the native torque units at time of download.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
TRU	Program Class D 222 (0xDE) 3 words Thread 1	Final Torque	S16	0 to 32767
		Increment per 120usec	S16	1 to 32767

Example

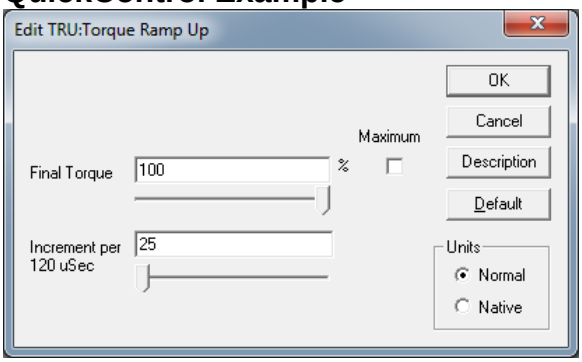
Set open loop current to 20000 (100%) in 4000 servo cycles (480 milliseconds)

@16 222 20000 5(CR)

Response

ACK only

QuickControl Example



VLL:Velocity Limits

Description

This command sets a limiter value within the servo control loop so as to limit the maximum velocity of the servo system. Both Moving and Holding limits are provided.

Note: Moving is defined by anytime the motor is in motion (Trajectory Generator active) and during the settling time as defined in the Error Limits (ERL) command.

Note: The trajectory will continue to change at the commanded rate, even if the physical motor has been limited by the velocity limit command. Use the Error Limits (ERL) command to either enable the "drag" mode, or combine with the error recovery commands to implement a shutdown if needed by the application.

Note: Bit 1 is set in the IS2 word if the velocity limit actually engages. This may be used to end a motion or to trigger an error recovery.

Note: The Gravity Offset Constant (GOC) is added following the velocity loop. Care must be exercised to verify that the GOC is not set so high as to override the velocity limit.

The velocity limits are given in SilverLode Actual Velocity Units (SAV) (see User Manual for details).

NOTE: The lower limit is 455 for motion to still be allowed.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
VLL	Program Class D 69 (0x45) 5 words Thread 1&2	Moving Limit	S16	0 to 32767 SAV
		Holding Limit	S16	0 to 32767 SAV

Example

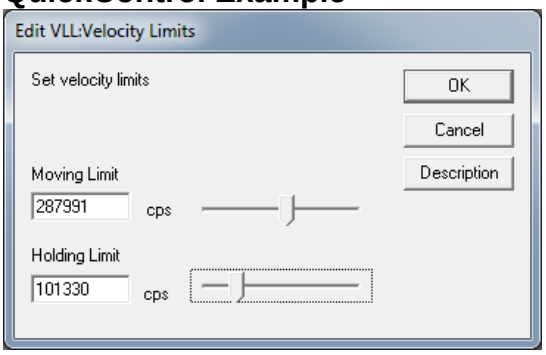
Set Moving Limit to 133333 cps and Holding Limit to 13330 cps

@16 69 16384 1638 (CR)

Response

ACK only

QuickControl Example



EGI:Electronic Gearing Mode, Interpolate

See Also: EGM:Electronic Gearing Mode,RSD:Registered Step & Direction
ERR:Error Limits, Remote

Description

The EGI command computes the local Position and Velocity from the register pair, designated by the Parameter Starting Data Register.. The register pair should be mapped via CAN to the master axis Target Position and Target Velocity register values, with the master axis sharing these two registers via on a single Transmit Channel (TPDO). The update time should be fixed, rather than change driven (Tx. Type 254, or Synchronous). The Master Target Position is linearly interpolated over the specified update interval to form the Local Target Position. The remote Target Velocity is copied from the source register for the segment period.

The Mode parameter will be used for future expansion. As of firmware rev 27, 0 is the only supported value. Cycle Count, is the number of 120uS ticks between data updates. If a fractional value results (i.e. a 1mS update would be 8.333 ticks) round up (9 ticks). Making this number larger will make this interpolation act as a low pass as the change will be spread over a longer period, but some lag will be introduced.

This command is a modal style motion command. It remains active until an over-ride style motion command is activated. The next command is executed immediately (120uS) following this command even if multi-tasking is not enabled.

Note: No Target Acceleration is calculated by this command. For best tracking between the master and slave units, set the Acceleration Feedforward in the Control Constants (CTC/CT2) command to zero.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EGI	Program Class D 91 (0x5A) 4 words Thread 1	Mode*	U16	0 = Absolute
		Starting Data Register (following register holds Velocity)	U16	11 to 198
		Cycle Count	U16	1 to 128 120 usec ticks

* Other modes will be added in future revs.

Example

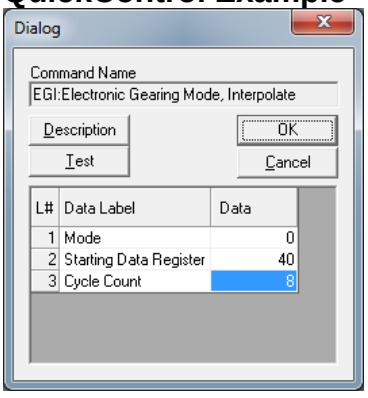
Follow the Position/Velocity as set in registers 150 and 151, with a 5ms update rate.

@16 91 0 150 41 (CR)

Response

ACK only

QuickControl Example



EGM:Electronic Gearing Mode

See Also: RSD:Registered Step & Direction
PVC:Profile Velocity Continuous

Description:

EGM provides high-resolution electronic gearing capability including the ability to smoothly transition between different gearing factors. During the move, any move parameter can be updated. With the parameter Starting Data Register=N, the move parameters are as follows:

Register N = Acceleration Limit Factor (AF)
Register N + 1 = Scale Factor (SF)

For a given Gear Ratio (GR), SF = Gear Ratio (GR) * 10,000,000.

This is powerful command beyond the scope of this document. See Application Note "QCI-AN019 Electronic Gearing" for details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EGM	Program Class D 93 (0x5D) 5 words Thread 1	Mode	U16	See Above 8=default
		Starting Data Register	U16	11 to 198
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

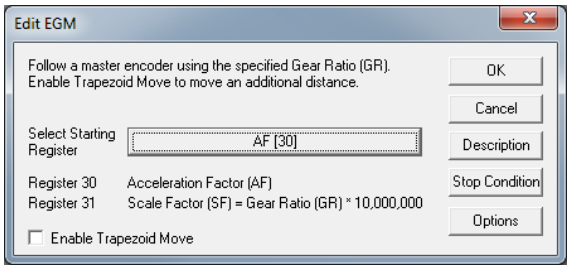
Put device into electronic gearing mode with Acceleration Factor in register 21 and Scale Factor in register 22:

@16 93 8 21 0 0(CR)

Response

ACK only

QuickControl Example



HSM:Hard Stop Move

Description

This command provides a way, while in multi-tasking operation, to execute a hard stop of any move or mode from within a program. A hard stop immediately halts the trajectory generator (motion commands) or stops the current mode, in either case the motor will come to an abrupt stop. In many situations, this may cause the motor to overshoot the stop position and oscillate until settled. More controlled stops can be accomplished by using the Velocity Mode which allows a user selectable deceleration to "0" velocity (stopped). The Profile Move Exit (PMX) command may similarly be used to halt an existing motion with a controlled deceleration.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
HSM	Program Class D 229 (0xE5) 1 word Thread 1	NONE	NONE	NONE

Example

Stop the device immediately.

@16 229 (CR)

Response

ACK only

QuickControl Example

Interpolated Move Start (IMS)

See Also: **Profiled Move (PMV), Interpolated Move Queue Clear (IMQ), QCI-TD044_InterpolatedMotion, Interpolated Move Write (IMW)**

Description:

This command provides a means generating an arbitrary motion from either nonvolatile memory, or from a host via the serial interface. Before issuing this command, Register #17 should be written to point to the source of the profile data, while Register #19 should contain a deceleration value to use if the data stream were to become interrupted. This command takes succeeding sets of 4 data values, and copies them, within a single normal control cycle, into the associated registers used by the profiled move command. It then waits the designated number of cycles (120 microseconds each) before loading the next set of data. At the same time, a Profiled Move operation is running in the background, using the given data. Each set of data represents a timed slice of the total motion, consisting of a constant acceleration period ramping to the new velocity, followed by a constant velocity period until the next set of data is loaded. If the given destination is reachable in the time slice given the other parameters, the target position will come to rest there until the next set of data is loaded that requires the motion to begin again. Complex moves involving multiple axis may be generated that may run from either internal Non-volatile memory, or from the Serial interface.

These two distinct modes of operation are selected by the value of the data in Register #17. A non-zero value indicates the address of the first of the four registers that will hold the data. Once this command is executed, the contents of the first of these registers will be copied to Register #18 to be used as a time countdown. The continuing operation of this command will decrement Register #18 each cycle. No external modification of Register #18 should be made while this mode is active.

The second register in this bank of four contains the target position for this time segment. If the segment is intended to end with the velocity non-zero, the value should be the approximate actual position + or - 1/2 full scale. This indicates the desired direction of travel. If the final velocity of this segment is zero, then the position should be the desired stopping position. This value is automatically copied to Register 20 for use by the profiled move operation. Note that the velocity for the last segment should not be set to zero, but left as the initial value for the segment.

The third register in this bank of four contains the acceleration or deceleration magnitude (positive values only) for this segment of the move. It is copied to Register # 21 and #23 for use by the profiled move operation.

The fourth register in this bank of four contains the speed (absolute value of the velocity) for this segment of the move. It is automatically copied to Register #22 for use by the profiled move operation.
After the four values have been copied, the upper word of Register #17 is set to “1” to indicate the data has been transferred and is now stale. If updating from internal memory, a multitasking program should be looping until this register has been modified. Next it needs to either point to the next set of data (Register 34, for example) if the data representing the next segment has already been loaded, or it needs to load the data representing the next segment of the move into the same set of Registers, then re-write Register #17 with the starting Register address. If the current movement segment ends while the upper half of Register #17 is non-zero, then the deceleration data in Register #19 is used to decelerate the motion to a stop. An error flag is set in bit 12 of the Status word to indicate a timeout in the data stream. The motion is stopped, but the program continues in operation. The final motion segment of a move is denoted by setting the segment time counter, the data in the first word of the four data words to zero. The final four words of segment information are copied into the respective profiled move operation registers, and the move continues as a profiled move.

This move may also be driven from the Serial Interface. The first method is to initialize the registers, as was explained above, but using the serial interface to write the registers and monitor Register # 17 to determine when the next data needs to be supplied. The preferred method is to use the Interpolated Move Queue Clear and Interpolated Move Write commands. In this mode, the Interpolated Move Clear, a program mode command, is used to clear our any existing data from the a four deep by four long word software FIFO specifically provided for this use. Next, Register #17 is set to zero (0) to indicate data will be drawn from the circular queue. Register #19 should be initialized with an appropriate deceleration value to use to stop the motion in the case of loss of communications. The Interpolated Start Move command may then be issued. The motion will not start until the first data set has been written to the software FIFO (circular queue). The queue may be kept full by the host via the Serial Interface. The buffering makes it easier to keep one or multiple axis fed with data. It also eliminates an extra register read to determine when data is required. (See Interpolated Move Write Queue (IMW) command for details.) The Interpolated move continues until either the Segment Time read is a zero, which terminates as detailed above, or until the queue is found empty when data is required, which uses the deceleration data in Register # 19 to bring the motion to an end, setting error bits as described above.

See Interpolated Motion Control in the User Manual for more details.

Command Info:

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
Interpolated Move Start (IMS)	Program Class D 253 (0xFD) 1 word	NONE	NONE	NONE

Register usage:

Register #17 Points to Register containing Segment Data, or “0” for Queued operation.
Register #18 Used internally to hold segment time countdown
Register #19 Holds data loss deceleration value
Registers # 20 to 24, as defined in the Profiled Move command (PMV**)**

Note: Register #17 is modified following each non-queued data transfer at the start of each segment.See notes above.

Example:

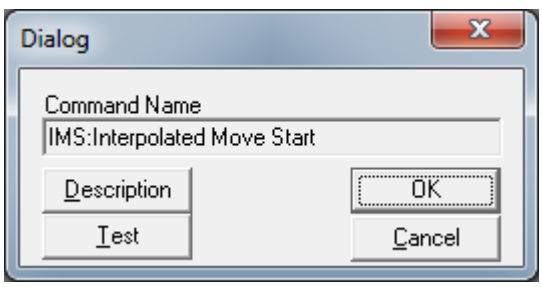
Start Interpolated Move.

@16 253 (CR)

SilverMax Response:

ACK only

QuickContol Example:



Interpolated Move Queue Clear (IMQ)

SeeAlso: **Profiled Move (PMV), Interpolated Move Start (IMS), QCI-TD044_InterpolatedMotion, Interpolated Move Write (IMW)**

Description:

This Command clears any data that may have been left in the Interpolated Move Queue. This queue is a software FIFO (First in first out) buffer capable of holding data for up to four interpolated motion segments, the data for each segment consisting of four long words (32 bits each) of data. If the data is able to fit within the queue, it is accepted and the communication is acknowledged. If the queue is full, the request is answered with a NAK – Full response. This just indicates that the host is successfully keeping the queue filled. The same data should be sent again until it is positively Acknowledged.

See Interpolated Move in User Manual for details.

Command Info:

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
Interpolated Move Queue Clear (IMQ) Rev 29	Program Class D 254 (0xFE) 1 word	NONE	NONE	NONE

Example:

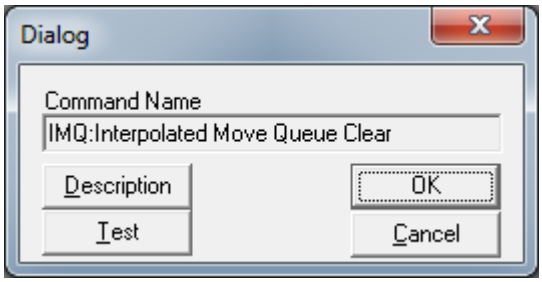
Stop the SilverMax immediately.

@16 254 (CR)

SilverMax Response:

ACK only

QuickContol Example:



MAT:Move Absolute, Time Based

Description

Move Absolute initiates a move to an absolute position with acceleration and total time.

See Basic Motion and Programming Fundamentals in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Scaling in User Manual for more details on native time units

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MAT	Program Class D 176 (0xB0) 9 words Thread 1	Position	S32	-2,147,483,648 to +2,147,483,647
		Acceleration Time (ticks)	U32	0 to 65534 (7.86 secs)
		Total Time (ticks)	U32	2 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

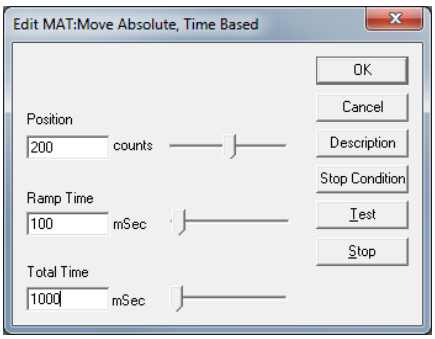
Move the device to position 200 in 1.0 seconds with a 0.1 second acceleration.

@16 176 200 83 8333 0 0(CR)

Response

ACK only

QuickControl Example



MAV:Move Absolute, Velocity Based

Description

Move Absolute initiates a move to an absolute position, using acceleration and velocity.

Note, acceleration to given velocity must be less than 7.86 seconds. That is:
Velocity/Acceleration < 7.86 (Profile Move - PMV – may be used for slower ramps)

See Basic Motion and Programming Fundamentals in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Scaling in User Manual for more details on native acceleration and velocity units.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MAV	Program Class D 134 (0x86) 9 words Thread 1	Position	S32	-2,147,483,648 to +2,147,483,647
		Acceleration	U32	1 to 1,073,741,823
		Velocity	U32	0 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

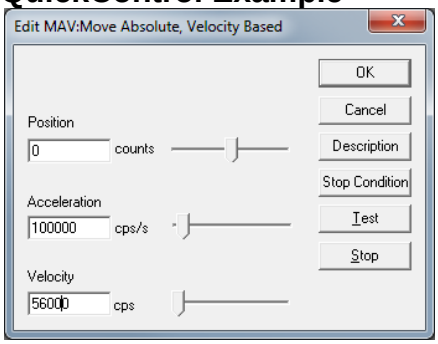
Move the device to position 0 at 56000 cps (see Scaling).

@16 134 0 96637 450971566 0 0(CR)

Response

ACK only

QuickControl Example



MRT:Move Relative, Time Based

Description

Move Relative initiates a distance move relative to the current target position, based on ramp and total time.

See Basic Motion and Programming Fundamentals in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Scaling in User Manual for more details on native time units

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MRT	Program Class D 177 (0xB1) 9 words Thread 1	Distance	S32	-2,147,483,648 to +2,147,483,647
		Ramp Time	U32	0 to 65534 (7.86 secs)
		Total Time	U32	2 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

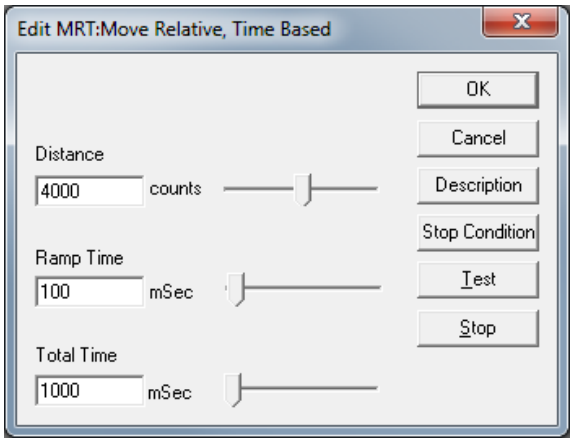
Move the device 4000 counts from its current position. Do the move in 1 second with a 0.1 second acceleration.

@16 177 4000 833 8333 0 0 (CR)

Response

ACK only

QuickControl Example



MRV:Move Relative, Velocity Based

Description

Move Relative initiates a distance move relative to the current target position.

Note, acceleration to given velocity must be less than 7.86 seconds. That is:

Velocity/Acceleration < 7.86

See Basic Motion and Programming Fundamentals in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Scaling in User Manual for more details on native acceleration and velocity units.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MRV	Program Class D 135 (0x87) 9 words Thread 1	Distance	S32	-2,147,483,648 to +2,147,483,647
		Acceleration	U32	1 to 1,073,741,823
		Velocity	U32	0 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

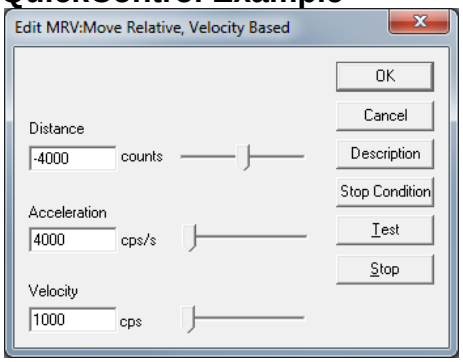
Move -4000 counts from its current position at 1000cps.

@16 135 -4000 3865 8053064 0 0(CR)

Response

ACK only

QuickControl Example



PIM:Position Input Mode

Description

Puts the device into a position control mode. Uses the contents of registers #12 -18 for position control processing.

See Application Note “QCI-AN047 Input Mode – Joystick” for details on using this command.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PIM	Program Class D Code (Hex): 216 (0xD8) Thread 1	Filter Constant	S16	0 to 32767
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

Position Input mode using a 117 Hz filter.

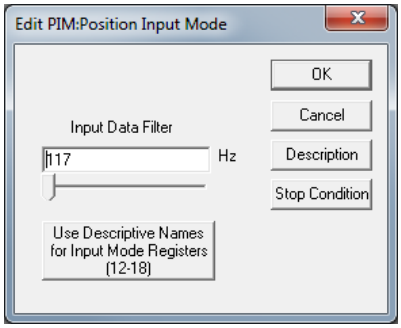
$30000 = 32768\ e^{-(117)2\pi(120uS)}$

@16 216 30000 0 0 (CR)

Response

ACK only

QuickControl Example



PMC:Profile Move Continuous

See Also: PMV:Profile Move, PMZ:Profile Move Zero

Description

The Profile Move commands are distinct from the other Motion commands in that the move parameters can be modified while the motion is in progress. A change in a move parameter updates the move immediately and can alter the move profile “real-time”.

The Profile Move Continuous puts the device into a move that does not end unless explicitly commanded (i.e. VMP, PMX), or stop condition. During the move, any move parameter can be updated either by a Host controller using the serial interface or by an internal program (**Multi-Tasking or Multi-Threading operation is required**).

With this feature, any motion profile shape can be accomplished by changing the appropriate parameter at the desired time. Five parameters are associated with this command. Each of the parameters is dedicated to a specified User Data Register. Modifying the contents of the Data Register modifies the parameter.

The following table shows the list of the parameters and their associated Data Register:

Register	Description	Data Range	Comment
20	**Position**	-2,147,483,648 to +2,147,483,647	This is an “Absolute” destination value.
21	**Acceleration**	2 to 1,073,741,823	Sets the acceleration rate that is used when increasing the move speed.
22	**Velocity**	0 to 2,147,483,647	The maximum speed that is allowed during a move
23	**Deceleration**	2 to 1,073,741,823	Sets the deceleration rate that is used when decreasing the move speed.
24	**Offset**	-2,147,483,648 to +2,147,483,647	A distance value to move that is added to the current position when a “Stop Condition” is encountered
25 **	**Starting Velocity**	0 to 2,147,483,647	The move will attempt to use this as a starting velocity if the move is sufficiently long to permit and PMZ is active
26 **	**Ending Velocity**	0 to 2,147,483,647	The move will attempt to use this as an ending velocity if the move is sufficiently long to permit and PMZ is active

** 25 and 26 are only used as start and stop velocities if PMZ command is active

Data Registers must be pre-loaded with the move parameters prior to issuing the Profile Move Continuous command.

Profiles Moves begin immediately after executing the command (within 120 usec.). The motor is accelerated using the Acceleration parameter until the maximum Velocity is reached. Deceleration begins when the distance of the move is such that the Absolute Position is achieved at the same time the motor has decelerated to "0" velocity. Depending on the parameters the maximum velocity may never be reached (Triangle Move).

During a Profile Move, the device is constantly recalculating its intermediate move values (every 120 usec.). This is done by taking the given move parameters, the current position and current velocity and adjusting what is required to hit the absolute position. This means that the device can even go from a Velocity Mode into a Profile Move without needing to stop first (Multi-Tasking operation is required). Remember that the move calculations are being done continually. Therefore, the parameters can be changed at any time and affect the motion in process.

The Acceleration and Deceleration parameters should typically be no greater than a ratio of 100:1 of each other (one value is no greater than 100 times the other) for numerical stability. For higher ratios user must verify proper operation.

The Position parameter can act as a Relative Distance value by using the Add To Register (ATR) command to increase or decrease the Position value. (See Add To Register for more details)

The Offset parameter is used to extend a move by the offset distance after a Stop Condition is encountered. In cases where a move needs to continue a prescribed distance past the point where a sensor triggers a stop, this parameter can be used to precisely control that offset distance to be moved. Note that the offset is automatically calculated as a distance in the direction of motion, thus a positive value gives a negative offset if the motion is going in a negative direction when the input is found. The Offset parameter allows easy trailing edge registration operations.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Profile Move Operation in User Manual for details.

Note: The Profiled motion commands combined with the "Drag" mode of the Error Limits will allow the user to reach the destination smoothly even if the rotor is restrained or torque limited, once the over torque condition has been removed.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PMC	Program Class D 240 (0xF0) 3 words Thread 1	Stop Enable	S16/U16	See above
		Stop State	S16/U16	See above

Example

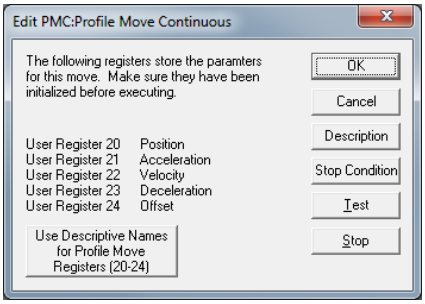
Put the device into a continuous Profile move.
Stop if Input #1 is high ("1").

@16 240 –1 1 (CR)

Response

ACK only

QuickControl Example



PMO:Profile Move Override

See Also: Profile Move Continuous (PMC)

Description

The Profile Move Override command allows a Profile Move Continuous to end when the Position is achieved. Normally the Move Continuous will not end until explicitly stopped by a Stop Condition or another command. The Override provides a graceful way to end the move so that the entire motion is completed with the motor stopping at the defined position. PMO will also override all other motions, including Step and Direction, if multi-tasking is enabled.

PMO operates exactly like the Profile Move command except that it does not wait for the previous motion to complete.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Profile Move Operation in User Manual for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PMO	Program Class D 249 (0xF9) 3 words Thread 1	Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

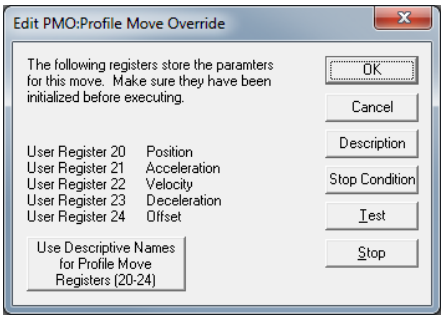
End the current Profile move when at “Position”.
Stop if Input #1 is high (“1”).

@16 249 –1 1 (CR)

Response

ACK only

QuickControl Example



PMV:Profile Move

See Also: PMC:Profile Move Continuous

Description

The Profile Move command works identical to the Profile Move Continuous except that when the Position is achieved, the move ends and the trajectory generator goes inactive. All of the parameters including the position can be changed while the move is executing. Once the move has ended, changing the parameters will have no effect.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Profile Move Operation in User Manual for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PMV)	Program Class D 241 (0xF1) 3 words Thread 1	Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

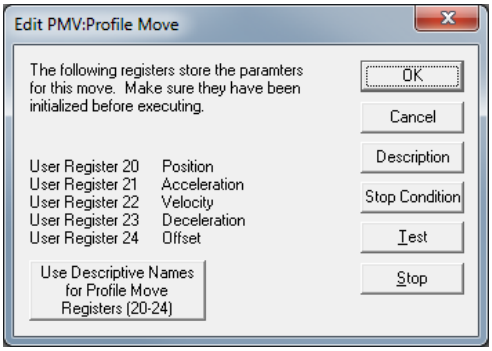
Start a Profile move. Stop if Input #1 is high (“1”).

@16 241 –1 1 (CR)

Response

ACK only

QuickControl Example



PMX:Profile Move Exit

See Also: PMC:Profile Move Continuous

Description

Exits the current Profile Move allowing the move to stop using the Deceleration parameter stored in Data Register #23. This command will work to stop any Motion, Profile Move or Mode (as long as register 23 has been initialized). The deceleration begins immediately and the profile destination will normally ***not*** be reached.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
PMX	Program Class D 242 (0xF2) 1 word Thread 1	NONE	NONE	NONE

Example

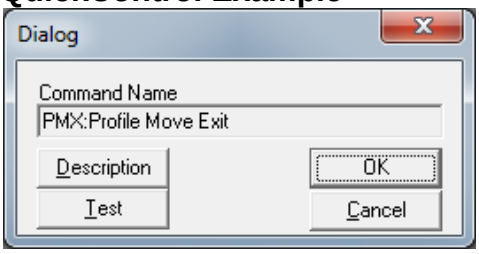
Exit the current move.

@16 242 (CR)

Response

ACK only

QuickControl Example



PMZ:Profile Move Zero

See Also: PMC:Profile Move Continuous

Description

This command allows the profile move commands to operate either with zero starting and ending velocities (default), or to have non-zero starting and ending velocities. A rapid stop may help break the flow in certain pump applications, but this command is not limited to those applications. The starting and ending velocities are used if the move is large enough to permit reaching these speeds according to the distance traveled. This command is implemented as mode 15 of the Set Mode commands.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
PMZ SS 25 Nx-X - all SM-X - all SD: none	Program Class D 86 (0x56) 3 words Thread 1 or 2	Mode	U16	15
		Enable	U16	0=disable 1=enable

Example

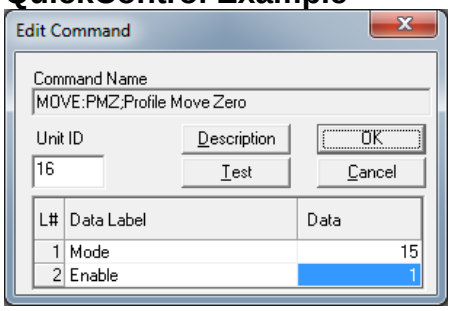
Enable non-zero starting and stopping velocity profiles.

@16 86 15 1 (CR)

Response

ACK only

QuickControl Example



PVC:Profile Velocity Continuous

See Also: VMI:Velocity Mode, Immediate Mode
VMP:Velocity Mode, Program Mode
EGM:Electronic Gearing Mode

Description:

PVC accelerates the servo to the register based velocity using the register based acceleration. During the move, any move parameter can be updated. With the parameter Starting Data Register=N, the move parameters are as follows:

Register N = Acceleration
Register N + 1 = Velocity

The bits of the Mode word allow the following combinations:

Bit #	Description
0	End Command When Stopped Set this bit to end the command when the servo is commanded to zero velocity (velocity register=0) and stops (actual velocity=0).
1	Override Existing Move Set this bit to have PVC override any existing move commands (requires multi-tasking enabled). By default, this bit=0, which means PVC will wait for the previous move to complete.
2	Must be set to 0.
3	Must be set to 0. NOTE: EGM has the same command number as PVC (cmd=93). The command is EGM if this bit=1 and PVC if this bit=0.

The Profile Velocity Continuous puts the device into a move that does not end unless explicitly commanded (i.e. VMP).

This command can also be used through the serial interface, however a NAK Busy will be reported when a Program or a motion command is executing.

Acceleration is normally a positive value regardless of velocity sign. If a stop condition is met with the Acceleration register set to 0, the servo will ramp to a fairly fast stop (approximately 1 second from full speed). Setting the acceleration to a negative value will cause the motion to ramp to a stop and the command to end using the magnitude of the acceleration to decelerate. As long as “End Command When Stopped” is not set, PVC allows the motion to be stopped and restarted just by writing to the registers.

See Scaling in User Manual for more details on native acceleration and velocity units.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PVC	Program Class D 93 (0x5D) 5 words Thread 1	Mode	U16	See Above 0=default
		Starting Data Register	U16	11 to 198
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

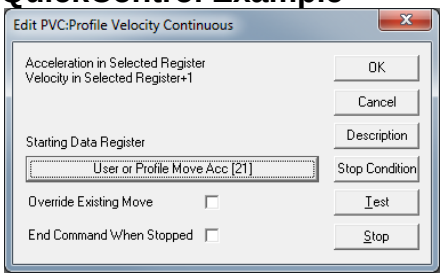
Put device into velocity mode running at acceleration in register 21 and velocity in register 22:

@16 93 0 21 0 0(CR)

Response

ACK only

QuickControl Example



RAT:Register Move Absolute, Time Based

See Also: MAT:Move Absolute, Time Based

Description

The Register Move Absolute performs an absolute move using a position value contained in the indicated User Data Register. This command works like the basic Move Absolute, Time Based (MAT) command in all other ways.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RAT	Program Class D 178 (0xB2) 9 words Thread 1	Data Register	U32	Standard Register Range
		Acceleration Time	U32	0 to 65534 (7.86 secs)
		Total Time	U32	2 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

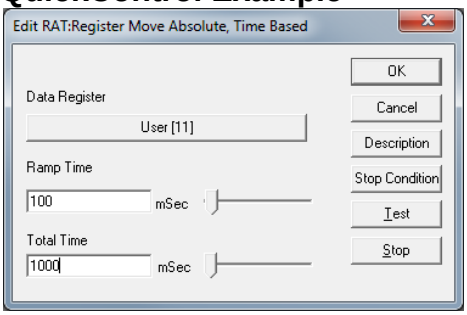
Move to position indicated by User Data Register #11 in 1000 mSec with a 100 mSec acceleration (see Scaling in User Manual).

@16 178 11 833 8333 0 0(CR)

Response

ACK only

QuickControl Example



RAV:Register Move Absolute, Velocity Based

See Also: MAV:Move Absolute, Velocity Based

Description

The Register Move Absolute performs an absolute move using a position value contained in the indicated User Data Register. This command works like the basic Move Absolute, Velocity Based (MAV) command in all other ways.

Note, acceleration to given velocity must be less than 7.86 seconds. That is:

Velocity/Acceleration < 7.86

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Scaling in User Manual for more details on native acceleration and velocity units.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
RAV	Program Class D 160 (0xA0) 9 words Thread 1	Data Register	U32	Standard Register Range
		Acceleration	U32	1 to 1,073,741,823
		Velocity	U32	0 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

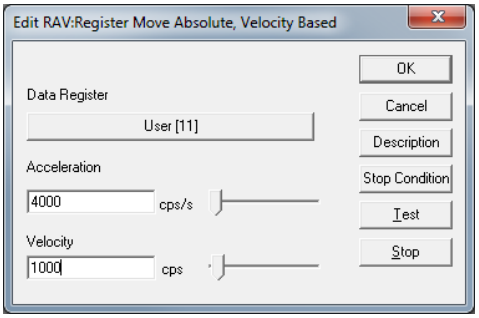
Move to position indicated by User Data Register #11 at vel=1000 cps and acc=4000cps/s.

@16 160 11 3865 8053064 0 0 (CR)

Response

ACK only

QuickControl Example



RRT:Register Move Relative, Time Based

See Also: Move Relative, Time Based (MRT)

Description

The Register Move Relative performs a relative move using a distance value contained in the indicated User Data Register. This command works like the basic Move Relative, Time Based (MRT) command in all other ways.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Motion Control Using Inputs and Registers in User Manual for more details.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
RRT	Program Class D 179 (0xB3) 9 words Thread 1	Data Register	U32	Standard Register Range
		Acceleration Time	U32	0 to 65534 (7.86 secs)
		Total Time	U32	2 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

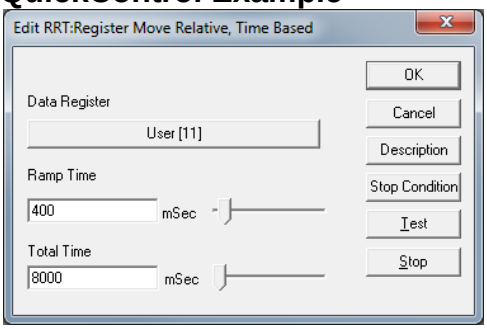
Move to position indicated by User Data Register #11. Do the move in 8 seconds with a 0.400 second acceleration.

@16 179 11 3333 66664 0 0(CR)

Response

ACK only

QuickControl Example



RRV:Register Move Relative, Velocity Based

See Also: MRV:Move Relative, Velocity Based

Description

The Register Move Relative performs a relative move using a distance value contained in the indicated User Data Register. This command works like the basic Move Relative, Velocity Based (MRV) command in all other ways.

Note, acceleration to given velocity must be less than 7.86 seconds. That is:

Velocity/Acceleration < 7.86

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Scaling in User Manual for more details on native acceleration and velocity units.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RRV	Program Class D 161 (0xA1) 9 words Thread 1	Data Register	U32	Standard Register Range
		Acceleration	U32	1 to 1,073,741,823
		Velocity	U32	0 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

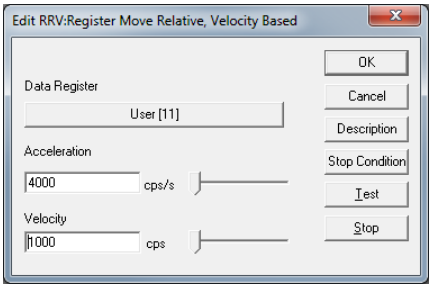
Move to position indicated by User Data Register #11 at vel=1000cps and acc=4000cps/s.

@16 161 11 3865 8053064 0 0 (CR)

Response

ACK only

QuickControl Example



SEE:Select External Encoder

See Also: SSI:SSI Port Mode, EMN:Encoder Monitor

Description

Selects the desired input format for an external or secondary encoder or step/direction input. Secondary encoders can be used by the "Step and Direction" commands, as well as the Dual Loop mode. If a secondary encoder is not being used, the inputs are ignored. The count since cleared or powered up is available in the register 200 - "External Encoder". A sensing of the designated index source causes the external encoder counter contents to be copied to register 201 - external encoder index. SEE does not tri-state the selected I/O unless outputting on these pins. If an I/O is already set LOW or HIGH it will remain that way after the SEE command, which may interfere with signals.

Index State: For single index encoders like QCI's M-Grade motor/encoders and third party standard encoders, set Index State to 0 (default) to detect the falling edge. Set to 1 to detect the index pulse on the rising edge of index channel. I-Grade motors use a 49/50 index pattern to allow for positive commutation values with a much smaller motion; set the Index State = CPR * 3/400. (For an 8000 count per revolution motor, this would be set to 3*(8000/200) = 120

Encoder Style 10 is a special mode intended for CANopen trajectory mirroring. The Target position is mapped into register 201, and the number of time ticks between data updates is set as Index State in 120uS ticks; the count should represent slightly longer than the actual time so that the calculated velocity does not drop to zero between updates. The value in register 200 is internally updated and the position increment is calculated by interpolating the value in 201 compared to the previous setting. This smooths out the position data when the CANopen position update rate is significantly slower than the control loop speed.

The SilverSterling has only one encoder input module. If a Mosolver is used, the encoder inputs may be internally jumpered to bring these inputs to the external I/O connector or to the mezzanine board. If the Mosolver is not being used, then encoder style 2 and 3 should not be used.

Note:The SilverSterling does not implement this command.

Parameters

Controller	Index Source	Index State	Encoder Style
SilverNugget X-Series, SilverMax X-Series	0=Index Source I/O #6 2=Enable internal encoder onto 4,5,6	See Above (ticks for style 10)	0 = A/B Quad on I/O #4 & 5 2 = Step & Dir on I/O #4 & 5 7 = make 4,5,6 back into IO 10 = interpolate position from register 201 (CANopen interface) *See Notes
SilverSterling **0,3 only for Mosolver with encoder brought out to I/O	0=Index Source I/O #6	See Above (ticks for style 10)	0 = A/B Quad internal (P1, 12C, 13C, 14C, with 18A grounded to disable the internal encoder receiver) 3 = Step & Dir on I/O #2 & 3 10 = interpolate position from register 201 (CANopen interface) *See Notes
SilverDust IG, MG	0=Index Source I/O #6	See Above	0 = A/B Quad on I/O #4 & 5 2 = Step & Dir on I/O #4 & 5 7 = make 4,5,6 back into IO

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SEE	Program Class D 192 (0xC3) 4 words Thread 1	Index Source	S16	0-1 (see above)
		Index State	S16	(see above)
		Encoder Style	U16	0-3 (see above)

Example

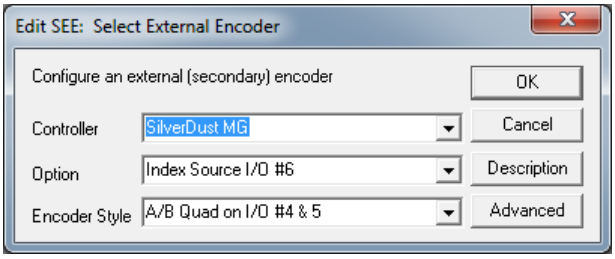
Set up the External encoder inputs for Index on input #6 and Step & Dir on #2 & #3.

@16 192 0 0 3 (CR)

Response

ACK only

QuickControl Example



RSD:Registered Step & Direction

See Also: EGM:Electronic Gearing Mode

Description

RSD causes the servo follow an incoming encoder (may be mapped via CAN) signal at a specified Gear Ratio (GR). This incoming signal is double low pass filtered to generate an estimate of the Target Velocity for the Velocity Feedforward term. The filter time constants used are the same as is used for the Velocity #1 and Velociy#2 Filters (see Filter Constant (FLC) command).

The parameter Scale Factor Register holds the Scale Factor (SF) and is calculated as follows:

SF = GR * SF1 where

> GR = <Change in Slave(CS)> (CM) >/<Change in Master(CM)>
> SF1 (1:1 Scale Factor) is a constant that depends on the servo's internal encoder.
> For details on GR, SF and SF1 see Application Note "QCI-AN019 Electronic Gearing".

SD(32)

Alternatively, CM and CS can be stored in the upper and lower words of the SF register. This allows for easy entry when both CM and CS are integers (i.e. GR=CS/CM=1:3).

CM may range from 1 to 32767 and CS may range from -32768 through 32767. A negative CS causes the slave to move in the opposite direction as the master.

See Application Note "QCI-AN019 Electronic Gearing".

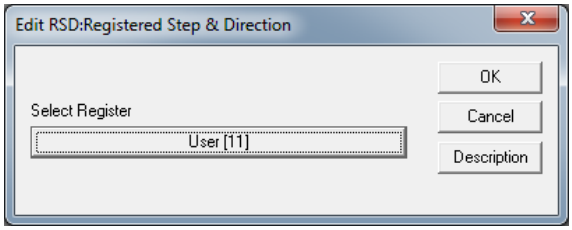
Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RSD	Program Class D 223 (0xDF) 2 words Thread 1	Scale Factor Register (SF)	U16	10 to 199

Example

Put the device into a electronic gearing mode using register #11 for the scaling value.

@16 223 11 (CR)



Response

ACK only

QuickControl Example

SSD:Scaled Step & Direction

See Also: RSD:Registered Step & Direction, EGM:Electronic Gearing Mode

Description

The Scaled Step & Direction (SSD) command causes the servo follow an incoming encoder or step and direction signal at a specified Gear Ratio (GR). This incoming signal is double low pass filtered to generate an estimate of the Target Velocity for the Velocity Feedforward term. The filter time constants used are the same as is used for the Velocity #1 and Velociy#2 Filters (see Filter Constant (FLC) command). This command is included for back compatibility. RSD and EGM are more advanced methods. NOTE: Registered Step and Direction (RSD) is much more flexible with the gearing and is preferred.

Note: The Select External Encoder (SEE) command must be issued prior to SSD to configure the inputs used by SSD.

The Scale Factor (SF) parameter is calculated as follows:

SF = GR * SF1

Where SF1 (1:1 Scale Factor) is a constant that depends on the servo's internal encoder as follows:

Internal Encoder CPR	SF1
4000	1024
8000	512
16000	256

A negative SF reverses direction of motor.

See Using Encoder Signals with Digital I/O in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SSD	Program Class D 180 (0xB4) 2 words Thread 1	Scale Factor(SF)	S16	-32767to 32767 (~2.88° per step clock)

Example

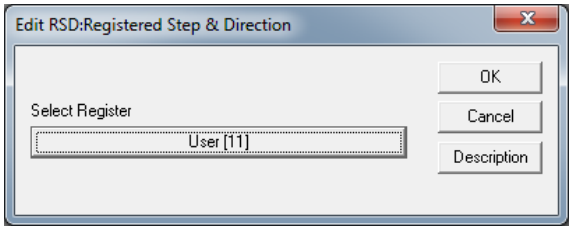
Put the device into a SSD mode with a 1:1 scale factor assuming a 4000 CPR encoder.

@16 180 1024 (CR)

Response

ACK only

QuickControl Example



TIM:Torque Input Mode

Description

Puts the device into a torque control mode. Uses the contents of data registers #12 -18 for torque control processing while the servo is moving.

See Application Note "QCI-AN047 Input Mode – Joystick" for details on using this command.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

See Technical Document QCI-TD051 Torque Control on our website for details on torque.

Note: In the absence of sufficient load or other feedback, this command may cause the motor to run at very high speeds.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
TIM	Program Class D 218 (0xDA) 4 words Thread 1	Filter constant	S16	0 to 32767
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

Torque Input mode using a 515 Hz filter. Exit if Input #1 is high ("1")

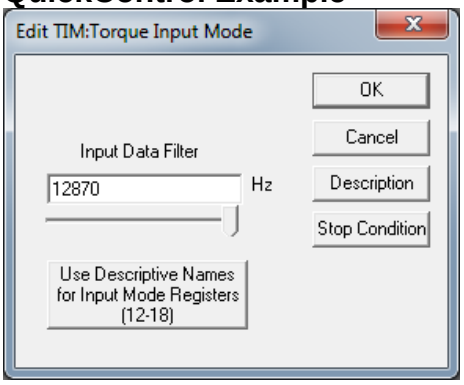
$30000 = 32768\ e^{-(117)2\pi(120uS)}$

@16 216 2222 –1 1 (CR)

Response

ACK only

QuickControl Example



VIM:Velocity Input Mode

Description

Puts the device into a velocity control mode. Uses the contents of registers #12 -18 for velocity control processing.

See Application Note "QCI-AN047 Input Mode – Joystick" for details on using this command.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
VIM	Program Class D 217 (0xD9) 4 words Thread 1	Filter Constant	S16	0 to 32767
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

Velocity Input Mode using a 117 Hz filter.

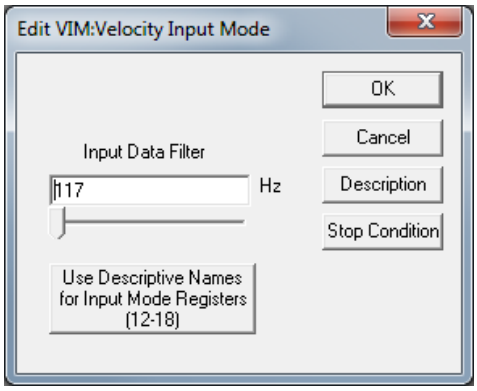
$30000 = 32768\ e^{-(117)2\pi(120uS)}$

@16 217 30000 –1 1 (CR)

Response

ACK only

QuickControl Example



VMP:Velocity Mode, Program Mode

See Also: VMI:Velocity Mode, Immediate Mode, PVC:Profile Velocity Continuous

Description

Accelerates the servo to the indicated velocity using the given acceleration. This command may be run from within a program. When this command is executed in a program, the motion will continue until the velocity reaches zero. Issuing the command with a non-zero velocity and stop on I/O enabled will allow the servo to run at velocity until the selected stop configuration is met; the velocity then ramps down to zero and the motion ends. This command can also be used through the serial interface, however a NAK Busy will be reported when a Program or a motion command is executing. See the Velocity Mode**,** Immediate Mode (VMI) command above for velocity mode using the serial interface. If multi-tasking is enabled, this command will take over any executing motion without the completion of that motion, and may be used to shut down a motion if the new velocity is zero.

See Scaling in User Manual for more details on native acceleration and velocity units.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
VMP	Program Class D 159 (0x9F) 7 words Thread 1	Acceleration	U32	1 to 1,073,741,823
		Velocity	S32	-2,147,483,648 to +2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

Put device into velocity mode running at (See Scaling):

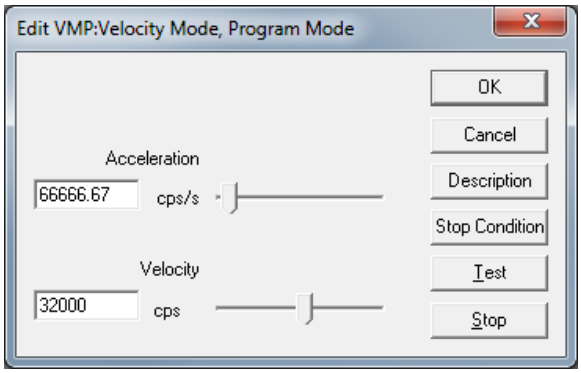
Vel = 32000 cps
Acc = 666666.67

@16 159 64425 257698038 0
0(CR)

Response

ACK only

QuickControl Example



XAT:Extended Register Move Absolute, Time Based

See Also: MAT:Move Absolute, Time Based

Description

The Extended Register Move Absolute performs an absolute position move using move parameters contained in the indicated User Data Registers. User can specify the starting Register.

The move parameters are retrieved from the User Data Registers in the Following order.
If Starting Data Register = N:

N = Position
N + 1 = Acceleration Time
N + 2 = Total Time

This command works like the basic Move Absolute, Time Based (MAT) command in all other ways.

See Register Based Motion Commands in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
XAT	Program Class D 236 (0xEC) 4 words Thread 1	Starting Data Register	U16	Standard Register Range
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

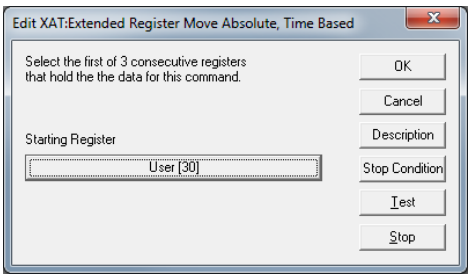
Move the device using parameters from User Data Registers #30-32.

@16 236 30 0 0(CR)

Response

ACK only

QuickControl Example



XAV:Extended Register Move Absolute, Velocity Based

See Also: MAV:Move Absolute, Velocity Based

Description

The Extended Register Move Absolute performs an absolute position move using move parameters contained in the indicated User Data Registers. User can specify the starting Register.

The move parameters are retrieved from the User Data Registers in the Following order.
If Starting Data Register = N:

N = Position
N + 1 = Acceleration
N + 2 = Velocity

This command works like the basic Move Absolute, Velocity Based (MAV) command in all other ways.

See Register Based Motion Commands in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
XAV	Program Class D 234 (0xEA) 4 words Thread 1	Starting Data Register	U16	Standard Register Range
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

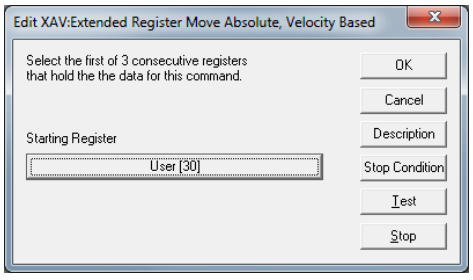
Move the device using parameters from User Data Registers #30-32.

@16 234 30 0 0 (CR)

Response

ACK only

QuickControl Example



XRT:Extended Register Move Relative, Time Based

See Also: MRT:Move Relative, Time Based

Description

The Extended Register Move Relative performs a relative distance move using move parameters contained in the indicated User Data Registers. User can specify the starting Register.

The move parameters are retrieved from the User Data Registers in the Following order.
If Starting Data Register = N:

N = Distance
N + 1 = Acceleration Time
N + 2 = Total Time

This command works like the basic Move Relative, Time Based (MRT) command in all other ways.

See Register Based Motion Commands in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
XRT	Program Class D 235 (0xEB) 4 words Thread 1	Starting Data Register	S16	Standard Register Range
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

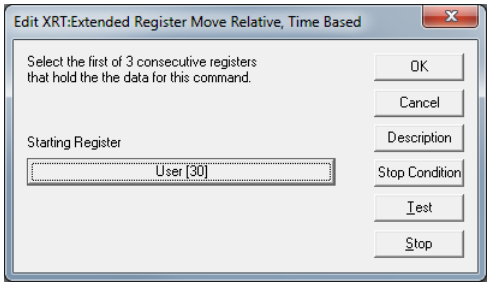
Move the device using parameters from User Data Registers #30-32.

@16 235 30 0 0 (CR)

Response

ACK only

QuickControl Example



XRV:Extended Register Move Relative, Velocity Based

See Also: MRV:Move Relative, Velocity Based

Description

The Extended Register Move Relative performs a relative distance move using move parameters contained in the indicated User Data Registers. User can specify the starting Register.

The move parameters are retrieved from the User Data Registers in the Following order.
If Starting Data Register = N:

N = Distance
N + 1 = Acceleration
N + 2 = Velocity

This command works like the Move Relative, Velocity Based (MRV) command in all other ways.

See Register Based Motion Commands in User Manual for more details.

See Using Inputs to Stop Motion in User Manual for Stop Enable and Stop State definitions.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
Extended XRV	Program Class D 233 (0xE9) 4 words Thread 1	Starting Data Register	S16	Standard Register Range
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example

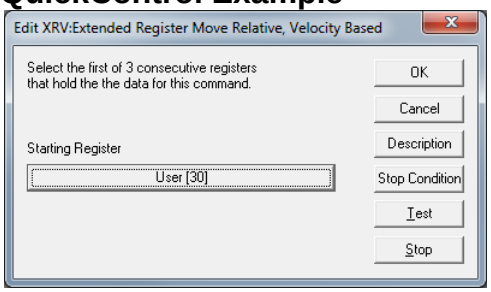
Move the device using parameters from User Data Registers #30-32.

@16 233 30 0 0 (CR)

Response

ACK only

QuickControl Example



ATR:Add To Register

See Instead: CLX:Calculation, Extended,CLD:Calculation Extended With Data

Description

The Add to Register command adds the included data into the selected 32 bit Data Register. This command is similar to the Write Register commands except it is designed to add to the existing value instead of overwriting it. The data parameter is "signed" so that a negative value can be added, which works as subtraction for decrementing.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ATR	Program Class D 248 (0x9A) 4 words Thread 1&2	Data Register	U16	Standard Register Range
		Data	S32	-2,147,483,648 to +2,147,483,647

Example

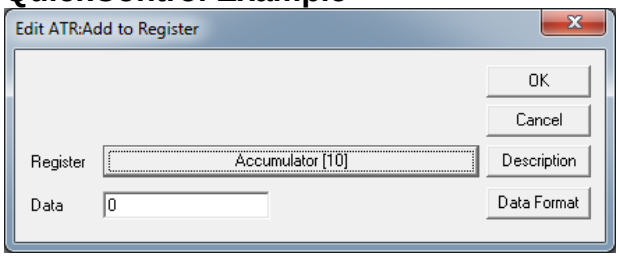
Add the number "1.5 SEC." to data register #10.

@16 248 10 12500 (CR)

Response

ACK only

QuickControl Example



CKS:Check Internal Status

Description

This command checks the conditions of the Internal Status Word in the same manner as does the Jump command. If the condition enabled is true, bit #6 of the Polling Status is set to “1”. A zero in the Condition Enable parameter unconditionally sets bit #6 of the Polling Status Word.

This command may be used to convey information from a program executing back to the host processor that is polling the device.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CKS	Program Class D 164 (0xA4) 3 words Thread 1&2	Condition Enable	U16	0 to 65535
		Condition State	U16	0 to 65535

Example

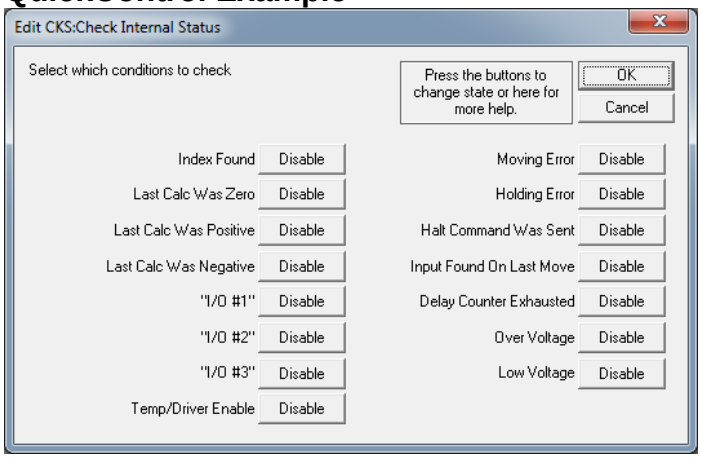
Check for a Last Calculation Was Positive and report to Host using Polling Status Word.

@16 164 4 4 (CR)

Response

ACK only

QuickControl Example



CME:Clear Max Error

Description

The Maximum Error (absolute value of the Position Error) is updated and latched each servo cycle. The value is limited to a single word, saturating at 32767 (0x7FFF) as a maximum value. This command allows the Maximum Error value to be reset to zero so that the Maximum Error for a new motion profile may be determined.

The Maximum Error value is stored in a Dedicated Data Register and may be read using the Read Register command.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CME	Program Class D 147 (0x93) 1 word Thread 1&2	NONE	NONE	NONE

Example

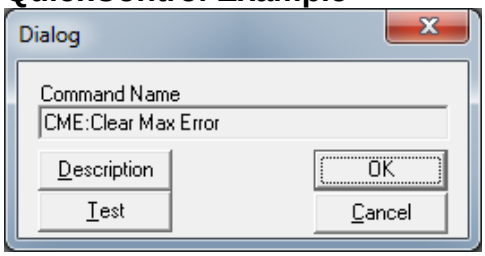
Clear the Maximum Error value.

@16 147 (CR)

Response

ACK only

QuickControl Example



DLT:Delay In Ticks

See Also: DLY:Delay

Description

The Delay In Ticks command sets a Delay Counter (Register 5) with the supplied parameter. The counter is decremented every servo cycle (120 microseconds). If the Tick Count is positive, the given value is used for the counter and a WDL command is automatically executed. If the value is negative, the absolute value of the parameter is loaded into the counter and the execution continues on to the next command in the Program Buffer

A Tick Count equals 120 microseconds in time. To convert to seconds multiply the Tick Count by 0.00012. A one second delay (rounded off) is 8333 Tick Counts .

See Program Flow Control Using Inputs and Data Registers in User Manual for general program flow control information and examples.

NOTE: DLT has the same command number as Delay (DLY). To the device there is no difference between these two commands. Only QuickControl distinguishes between DLT and DLY with DLT have units of ticks and DLY having units of milliseconds.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
DLT	Program Class D 140 (0x8C) 3 words Thread 1&2*	Delay Count 1 tick=120 uSec.	S32	-2,147,483,647 to 2,147,483,647

*Be cautious of using DLT/DLY in both Thread 1 and Thread 2. The Delay Counter (Register 5) is common to both threads. Each thread will reset this register when DLT/DLY is executed.

Example

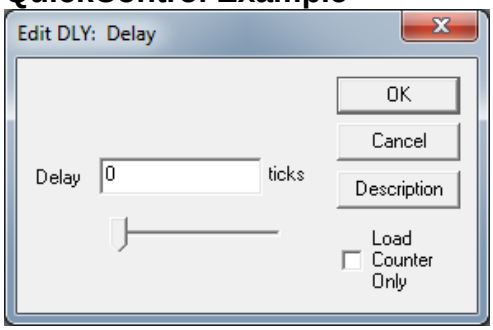
Cause the device to delay program execution by 500 ticks.

@16 140 500 (CR)

Response

ACK only

QuickControl Example



DLY:Delay

See Also: DLT:Delay In Ticks

Description

This command is the same as the Delay In Ticks (DLT) command. It has the same command number and parameters. The only difference is how QuickControl scales them. To the device they are exactly the same. QuickControl takes the parameter entered and multiplies it by 8.3333 to convert milliseconds to servo ticks (120uSec/tick).

See Program Flow Control Using Inputs and Data Registers in User Manual for general program flow control information and examples.

Command Info

Same as Delay In Ticks (DLT)

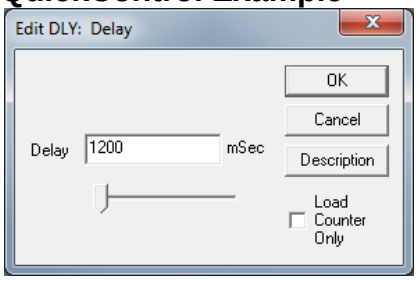
Example

Cause the device to delay program execution by 1.2 seconds.

1200mSec * 8.3333 = 10000 ticks

@16 140 10000 (CR)

QuickControl Example



Response

ACK only

END:End Program

Description

Programs typically end when the last line of the program is completed. If the program needs to end based on a Conditional Jump, the End Program command can be inserted in the program at the desired point. When this command is executed, the currently running program will stop executing and the motor will be placed in a Host Mode. Issuing any of the Override Commands can also stop programs.

NOTE: An END command is automatically inserted at the end of programs loaded in the Program Buffer.

This command does nothing if sent to the motor when it is in Host Mode.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
END	Program Class D 128 (0x80) 1 word Thread 1&2	NONE	NONE	NONE

Example

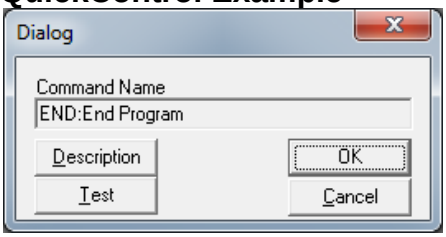
As part of a program, end program execution.

@16 128 (CR)

Response

ACK only

QuickControl Example



FOR:For

See Also: NXT:Next

Description

The FOR command initializes the Loop Register used for a "For Loop" and specifies the increment and final loop test value for the loop.

General

The Loop Register is updated by adding the Increment value when the Next (NXT) statement is evaluated. If the Increment is positive, the loop terminates when the new calculated Loop Register exceeds Final Value; if the Increment is negative, then the loop terminates when the new Loop Register value is less than Final Value.

FOR/NXT loops can be nested as shown. The inner loop is the FOR on line 3 and the line NXT on 5, the middle loop is the FOR on line 2 and the NXT on line 6 and the outer loop is the FOR on line 1 and the NXT on line 7.

Line# Oper	Label	Command
1:FOR		FOR "Loop C[29]" = 3 to 0 with inc=-1
2:FOR		FOR "Loop B[30]" = 3 to 0 with inc=-1
3:FOR		FOR "Loop B[30]" = 5 to 0 with inc=-1
4:DLY		Delay for 500 mSec
5:NXT		Next (FOR line 3)
6:NXT		Next (FOR line 2)
7:NXT		Next (FOR line 1)

See Program Flow Control in User Manual for parameter details.

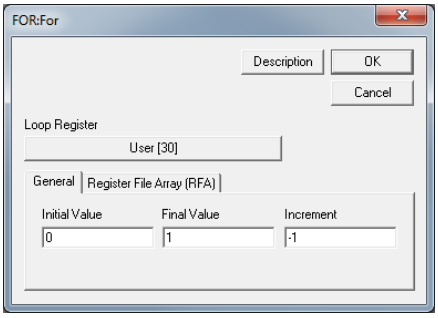
Register File Array (RFA)
This special tab is used to iterate though all the rows of the selected Register File Array (RFA). For each row of the RFA, the Loop Register will hold the starting address of that row.
See Application Note QCI-AN046 Indirect Addressing for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
FOR	Program Class E 209 (0xD1) 8 words Thread 1&2	Initial Value	S32	-2,147,483,647 to 2,147,483,647
		Final Value	S32	-2,147,483,647 to 2,147,483,647
		Increment	S32	-2,147,483,647 to 2,147,483,647
		Loop Register	U16	Standard Register range (Must be writable)

Example
FOR Loop Register 30 = 3 to 0 with -1 increment.
@16 209 3 0 -1 30 (CR)

QuickControl Example



Response

ACK only

JAN:Jump On AND I/O State

See Also: JRB:Jump On Register Bitmask

Description

The Jump on AND I/O State command allows looping and other conditional branching inside a program based on the condition of the I/O State Word (IOS).

The IOS Condition Enable selects which inputs will be used in the AND-ed evaluation. The IOS Condition State allows the user to specify the states (High "1" or Low "0") of the selected inputs that will cause a TRUE condition for each of the inputs. Setting both parameters to "zero" forces an unconditional jump to the specified Program Buffer location.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JAN	Program Class E 250 (0xEE) 4 words Thread 1&2	IOS Condition Enable	U16	0 to 65535
		IOS Condition State	U16	0 to 65535
		Program Buffer Address	U16	0 to Program Buffer Size

Example

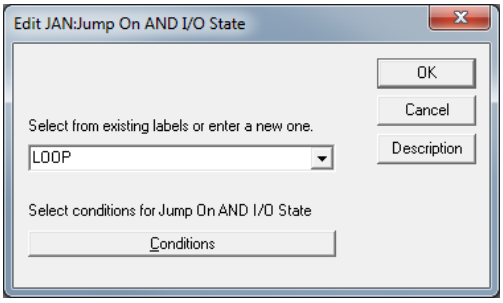
Jump to Program Buffer location 10 if digital inputs #4, #5, #6 and #7 are High ("1"). See I/O State Word (IOS) in User Manual for bit definitions.

@16 250 61440 61440 10 (CR)

Response

ACK only

Quick Control Example



JGE:Jump On Register Greater Or Equal

See Also: JRE:Jump On Register Equal

Description

The Jump On Register Greater or Equal command allows looping and other conditional branching inside a program based on the comparison of the contents of the given register with the value of the compare parameter.

Note: Internally JRE, JGE, JNE, JLT,JGR, JLE all share the same Command Number, with the difference indicated in the high byte of the Operation/Register parameter (see JRE command for details). For this command, the Operation/Register parameter is equal to the register number + 256.

See Program Flow Control in User Manual for more details.

See JRE command for parameter information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JGE	Program Class E 137 (0x89) 5 words Thread 1&2	Operation (High Byte) Register (Low Byte)	U16	Operation: =1 Register: Standard Register Range
		Value	S32	-2,147,483,648 to +2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

Example

Jump to Program Buffer location 10 if Register # 32 is greater or equal to “1200”

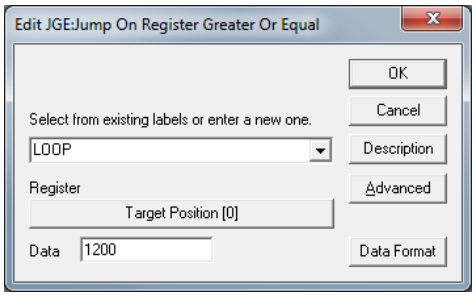
Operation/Register=256 + 32=288

@16 137 288 1200 10 (CR)

Response

ACK only

QuickControl Example



JGR:Jump On Register Greater Than

See Also: JRE:Jump On Register Equal

Description

The Jump On Register Greater Than command allows looping and other conditional branching inside a program based on the comparison of the contents of the given register with the value of the compare parameter.

Note: Internally JRE, JGE, JNE, JLT,JGR, JLE all share the same Command number, with the difference indicated in the high byte of the Operation/Register parameter (see JRE command for details). For this command, the Operation/Register parameter is equal to the register number + 256.

See Program Flow Control in User Manual for more details.

See JRE command for parameter information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JGR	Program Class E 137 (0x89) 5 words Thread 1&2	Operation (High Byte) Register (Low Byte)	U16	Operation: =4 Register: Standard Register Range
		Value	S32	-2,147,483,648 to +2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

Example

Jump to Program Buffer location 10 if Register # 32 is greater than “1200”

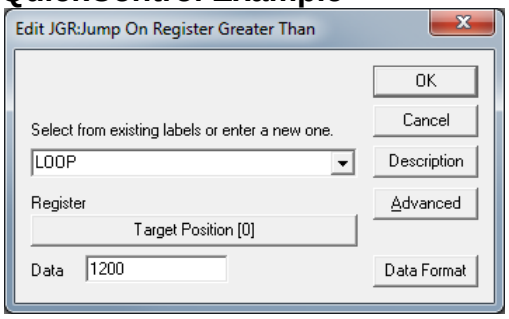
Operation/Register=1024 + 32=1056

@16 137 1056 1200 10 (CR)

Response

ACK only

QuickControl Example



JLE:Jump On Register Less or Equal

See Also: JRE:Jump On Register Equal

Description

The Jump On Register Less or Equal command allows looping and other conditional branching inside a program based on the comparison of the contents of the given register with the value of the compare parameter.

Note: Internally JRE, JGE, JNE, JLT,JGR, JLE all share the same Command Code, with the difference indicated in the high byte of the Operation/Register parameter (see JRE command for details). For this command, the Operation/Register parameter is equal to the register number + 512.

See Program Flow Control in User Manual for more details.

See JRE command for parameter information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JLE	Program Class E 137 (0x89) 5 words Thread 1&2	Operation (High Byte) Register (Low Byte)	U16	Operation: =5 Register: Standard Register Range
		Value	S32	-2,147,483,648 to +2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

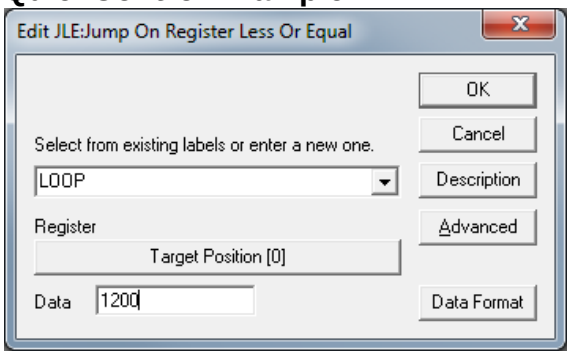
Example

Jump to Program Buffer location 10 if Register # 32 is less or equal to “1200”.

Operation/Register = 1280 + 32 = 1312

@16 137 1312 1200 10 (CR)

QuickControl Example



Response

ACK only

QuickControl Example

JLT:Jump On Register Less Than

See Also: JRE:Jump On Register Equal

Description

The Jump On Register Less Than command allows looping and other conditional branching inside a program based on the comparison of the contents of the given register with the value of the compare parameter.

Note: Internally JRE, JGE, JNE, JLT,JGR, JLE all share the same Command Code, with the difference indicated in the high byte of the Operation/Register parameter (see JRE command for details). For this command, the Operation/Register parameter is equal to the register number + 512.

See Program Flow Control in User Manual for more details.

See JRE command for parameter information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JLT	Program Class E 137 (0x89) 5 words Thread 1&2	Operation (High Byte) Register (Low Byte)	U16	Operation: =2 Register: Standard Register Range
		Value	S32	-2,147,483,648 to +2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

Example

Jump to Program Buffer location 10 if Register # 32 is less than to “1200”.

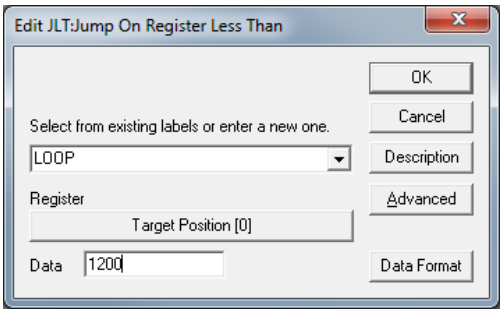
Operation/Register = 512 + 32 = 544.

@16 137 544 1200 10 (CR)

Response

ACK only

QuickControl Example



JMP:Jump

Description
The Jump command allows looping and other conditional branching inside a program based on the condition of the Internal Status Word (ISW)

See Program Flow Control in User Manual for parameter details.

Command Info

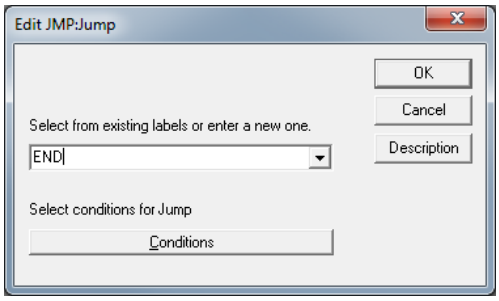
Command	Command Type/Num	Parameters	Param Type	Parameter Range
JMP	Program Class E 162 (0xA2) 4 words Thread 1&2	Condition Enable	U16	0 to 32767
		Condition State	U16	0 to 32767
		Program Buffer Address	U16	0 to Program Buffer Size

Example
Jump to Program Buffer location 0 last calculation was zero. See Internal Status Word (ISW) in User Manual for bit definitions.

@16 162 2 2 0 (CR)

Response
ACK only

QuickControl Example



JNA:Jump On NAND I/O State

See Also: JRB:Jump On Register Bitmask

Description

The Jump On NAND I/O State command allows looping and other conditional branching inside a program based on the condition of the I/O State Word (IOS).

The IOS Condition Enable selects which inputs will be used in the NAND-ed evaluation. The IOS Condition State allows the user to specify the states (High "1" or Low "0") of the selected inputs that will cause a TRUE condition for each of the inputs. If all the enabled inputs are TRUE a jump will NOT occur. This means that a jump will always occur when any of the conditions are FALSE. Setting both parameters to "zero" forces an unconditional jump to the specified Program Buffer location.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JNA	Program Class E 238 (0xEE) 4 words Thread 1&2	IOS Condition Enable	U16	0 to 65535
		IOS Condition State	U16	0 to 65535
		Program Buffer Address	U16	0 to Program Buffer Size

Example

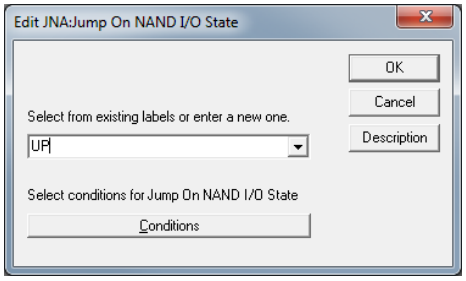
Don't jump to Program Buffer location 10 if digital inputs #4, #5, #6 and #7 are High ("1"). See I/O State Word (IOS) in User Manual for bit definitions.

@16 238 61440 61440 10 (CR)

Response

ACK only

QuickControl Example



JNE:Jump On Register Not Equal

See Also: JRE:Jump On Register Equal

Description

The Jump On Register Not Equal command allows looping and other conditional branching inside a program based on the comparison of the contents of the given register with the value of the compare parameter.

Note: Internally JRE, JGE, JNE, JLT,JGR, JLE all share the same Command Code, with the difference indicated in the high byte of the Operation/Register parameter (see JRE command for details). For this command, the Operation/Register parameter is equal to the register number + 512.

See Program Flow Control in User Manual for more details.

See JRE command for parameter information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JNE	Program Class E 137 (0x89) 5 words Thread 1&2	Operation (High Byte) Register (Low Byte)	U16	Operation: =3 Register: Standard Register Range
		Value	S32	-2,147,483,648 to +2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

Example

Jump to Program Buffer location 10 if Register # 32 is not equal to "1200"

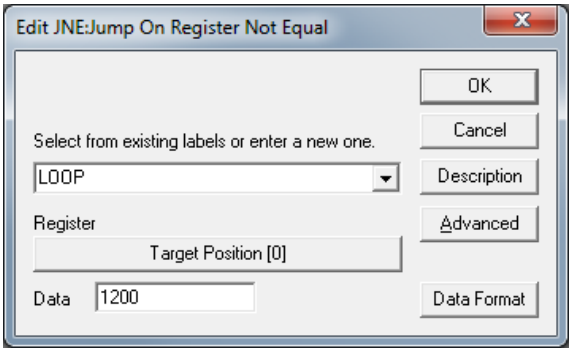
Operation/Register = 768+32= 800.

@16 137 800 1200 10 (CR)

Response

ACK only

QuickControl Example



JOI:Jump On Input

Description

The Jump On Input command allows looping and other conditional branching inside a program based on the condition of an Input. This command is actually the same number as the Jump command (JMP), however, by using a negative number for the first parameter the usage of the command changes.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JOI	Program Class E 162 (0xA2) 4 words Thread 1&2	Enable Code	S16	-1 to -14
		Enable State	S16	0 to 1 0 = “Low” 1 = “High”
		Program Buffer Address	U16	0 to Program Buffer Size

Example

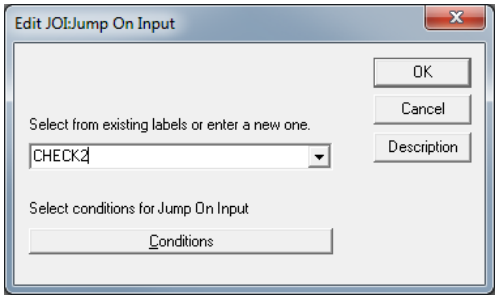
Jump to Program Buffer location 10 if digital input #5 is High “1”.

@16 162 -5 1 10 (CR)

Response

ACK only

QuickControl Example



JOR:Jump On OR I/O State

See Also: JRB:Jump On Register Bitmask

Description

The Jump On Inputs, OR-ed command allows looping and other conditional branching inside a program based on the condition of the I/O State Word (IOS).

The IOS Condition Enable selects which inputs will be used in the OR-ed evaluation. The IOS Condition State allows the user to specify the states (High "1" or Low "0") of the selected inputs that will cause a TRUE condition for each of the inputs. Setting both parameters to "zero" forces an unconditional jump to the specified Program Buffer location.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JOR	Program Class E 239 (0xEF) 4 words Thread 1&2	IOS Condition Enable	U16	0 to 65535
		IOS Condition State	U16	0 to 65535
		Program Buffer Address	U16	0 to Program Buffer Size

Example

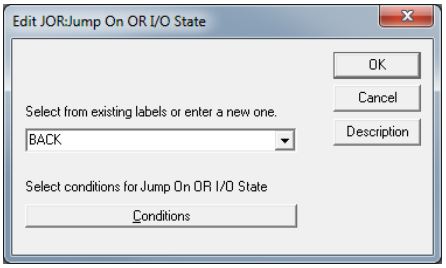
Jump to Program Buffer location 0 if input #1 or #2 or #3 is High ("1"). See I/O State Word (IOS) in User Manual for bit definitions.

@16 162 112 112 0 (CR)

Response

ACK only

QuickControl Example



JRB:Jump On Register Bitmask

See Also: PCB:Program Call On Register Bitmask

Description

The Jump On Register Bitmask command allows looping and other conditional branching inside a program based on the comparison of the given register with the value of a pair of 32 bit constants.

This command shares the same command number as Program Call On Register Bitmask (PCB) with the Type parameter (see below) differentiating the two.

The Operation parameter determines how the Register is compared to the two constants, Param 1/State and Param 2/Enable.

For bitwise operations (i.e. AND, OR,..), the Enable parameter selects which bits will be used in the comparison. The to be compared are set in the Enable parameter. The State parameter allows the user to specify the states (High "1" or Low "0") of the selected bits that will cause a jump.

Operation Description	Operation Value
AND Selected Bits	0
OR Selected Bits	1
NAND Selected Bits	2
NOR Selected Bits	3
Param 1 < Reg < Param 2	4
Param 1 <= Reg < Param 2	5
Param 1 < Reg <= Param 2	6
Param 1 <= Reg <= Param 2	7
Reg <= Param 1, Reg >= Param 2	8
Reg < Param 1, Reg >= Param 2	9
Reg <= Param 1, Reg > Param 2	10
Reg < Param 1, Reg > Param 2	11

See Program Flow Control in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JRB	Program Class E 89 (0x59) 4 words Thread 1&2	Type/Operation Type (High Byte) Operation (Low Byte)	U16	Type = 0 for JRB Operation: see above table
		Data Register	U16	Standard Register Range
		Param 1/State	S32	-2,147,483,648 to 2,147,483,647
		Param 2/Enable	S32	-2,147,483,648 to 2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

Example

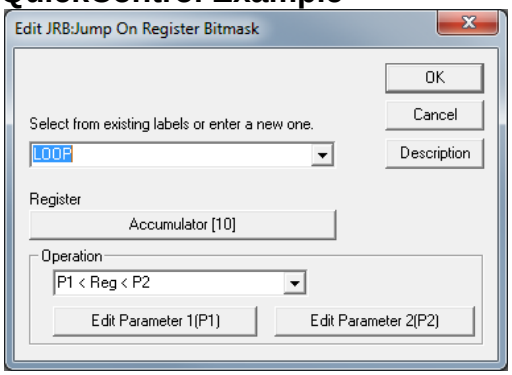
Jump to Program Buffer location 10 if Register # 10 > 0 and < 5.

@16 88 4 10 0 5 10 (CR)

Response

ACK only

QuickControl Example



JRE:Jump On Register Equal

Description

The Jump On Register Equal command allows looping and other conditional branching inside a program based on the comparison of the contents of the given register with the value of the compare parameter.

Note: Internally JRE, JGE, JNE, JLT (SD05 JGT, JLE) all share the same Command Number, with the difference indicated in the high byte of the Operation/Register Parameter.

The Operation is automatically handled for you by QuickControl. The following is provided for those not using QuickControl.

Operation (bits 0-3)	Equivalent Command	Operation/Register Parameter Value
0	JRE	Register #
1	JGE	Register # + 256
2	JLT	Register # + 512
3	JNE	Register # + 768
4 (SD05)	JGR	Register # + 1024
5 (SD05)	JLE	Register # + 1280

For SilverDust Rev 05+, the selected register may also be modified to simplify loop construction. Bits 4-7 of the Operation parameter are used for this feature.

Bit4: Disable/Enable
Bit5: Inc/Dec
Bit6: Post/Pre

See Program Flow Control in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
JRE	Program Class E 137 (0x89) 4 words Thread 1&2	Operation|Register	U16	Operation: = 0 Register: Standard Register Range
		Value	S32	-2,147,483,648 to 2,147,483,647
		Program Buffer Adr	U16	0 to Program Buffer Size

Example

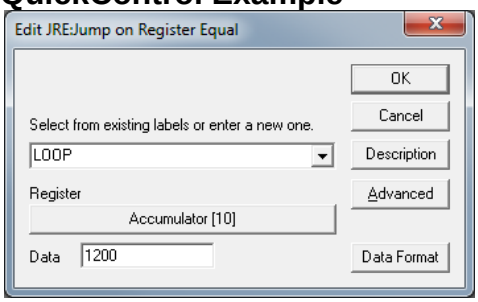
Jump to Program Buffer location 10 if Register # 32 is equal to "1200"

@16 137 32 1200 10 (CR)

Response

ACK only

QuickControl Example



LRP:Load And Run Program

Description

The Load and Run Program transfers a program from non-volatile memory to the Program Buffer and executes it. This command combines the function of the Load Program and the Run Program together in one command.

During the load process, the data is used to calculate a checksum value. When the load is complete, the calculated checksum is compared to the stored checksum. If the checksums do not agree, Bit #14 in the Polling Status Word is set ("1") to indicate a program load failure. (This may occur if data and/or programs overlap their usage in non-volatile memory.)

After a load is complete and no errors were encountered, a Run Program will be initiated starting the program and dropping the device into the Program Mode. Programs that contain errors will shut down the servo and exit execution when an error is encountered. Bit #12 (Program errors) of the Polling Status Word will be set indicating program execution error. The program will remain in the buffer until removed by the Clear Buffer command or over loaded by another Load Program command.

Note: One extra word of non-volatile memory is used to store the size and checksum of programs up to 254 words long. Two extra words of non-volatile memory are used for programs 255 words and longer (SD05).

If run from Thread 2, program loads into Thread 2 Program Buffer space. Non-Volatile memory can only be accessed by one thread at a time. Other thread will automatically wait to access Non-Volatile memory.

See Memory Model in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
LRP	Program Class D 156 (0x9C) 2 words Thread 1&2	NV Memory Address	U16	NV Memory Range

Example

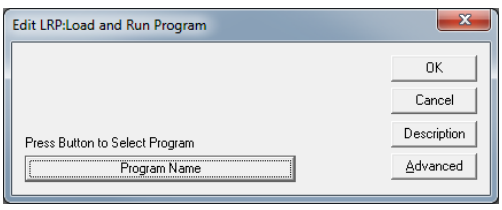
Load and Run the Program stored at NV Memory Address #150.

@16 156 150 (CR)

Response

ACK only

QuickControl Example



NXT:Next

See Also: FOR:For

Description

The Next (NXT) command closes the bottom of a nested FOR Loop.

NXT uses the following parameters from the FOR command:

FOR Parameters

- Final Value
- Increment
- Loop Register

NXT increments Loop Register by adding Increment and then compares Loop Register to Final Value. See FOR for details on evaluation and nesting.

The Program Buffer Address parameter points to the beginning of the nested FOR command. QuickControl automatically calculates this and simply displays the line number of the matching FOR command.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
NXT	Program Class E 210 (0xD2) 2 words Thread 1&2	Program Buffer Address	U16	0 to Program Buffer Size

Example

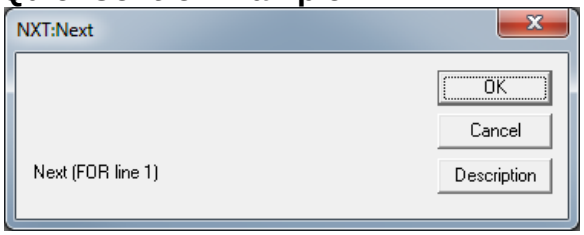
NEXT looping back to FOR on line 1 (Program Buffer Address 0)

@16 210 0 (CR)

Response

ACK only

QuickControl Example



PCI:Program Call On Input

See Also: PCL:Program Call

Description

The Program Call on Input command (PCI) works the same as Program Call (PCL) except the format of the call conditions. See Program Call (PCL) for details.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PCI	Program Class D 201 (0xC9) 4 words Thread 1&2	Enable Code	S16	0 or -14 (-116 with extended I/O)
		Enable State	S16	0 to 1
		Program Buffer Address	U16	0 to Program Buffer Size

Example

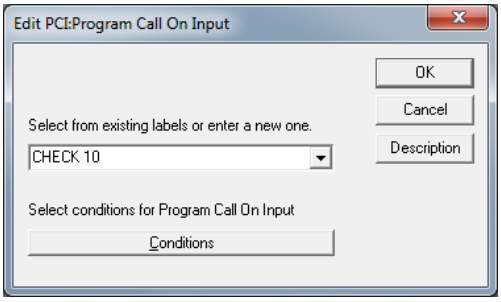
Call Program Buffer location #50 if digital input #2 is High "1".

@16 201 50 -2 1 (CR)

Response

ACK only

QuickControl Example



PCL:Program Call

See Also: PCI:Program Call On Input

Description

If the conditions are met, PCL jumps to the specified Program Buffer location (program label in QuickControl) and continues executing commands until a Program Return (i.e. PRT or PRI) command is encountered. A Program Return command causes the execution to continue at the command after the PCL.

Up to 8 PCL levels may be used (8 nested routines). If too many PCLs are executed without corresponding Program Returns the program will error, Stop execution and Bit #12 in the Polling Status Word will be set. The PCL and Program Return must both be in the Program Buffer (same QuickControl program). The stack is cleared each time a Load and Run Program (LRP) command is executed.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PCL	Program Class D Code (Hex): 201 (0xC9) 4 words Thread 1&2	Condition State	U16	0 to 32767
		Condition Enable	U16	0 to 32767
		Program Buffer Address	U16	0 to Program Buffer Size

Example

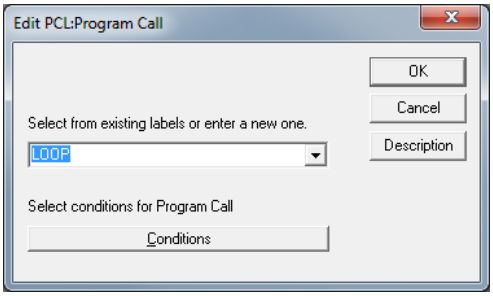
Call Program Buffer location #50 if digital input #1 is High “1”.

@16 201 50 16 16 (CR)

Response

ACK only

QuickControl Example



PCB:Program Call On Register Bitmask

See Also: JRB:Jump On Register Bitmask

Description

The Program Call On Register Bitmask (PCB) works the same as Program Call (PCL) except the format of the call conditions. See Program Call (PCL) for details.

This command shares the same command number as Jump On Register Bitmask (PCB) with the Type parameter (see below) differentiating the two.

The conditions for the call are the same as the JRB command. See JRB command for parameter definitions.

See Program Flow Control in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PCB	Program Class E 89 (0x59) 4 words Thread 1&2	Type/Operation Type (High Byte) Operation (Low Byte)	U16	Type = 1 for PCB Operation: see above table
		Data Register	U16	Standard Register Range
		Param 1/State	S32	-2,147,483,648 to 2,147,483,647
		Param 2/Enable	S32	-2,147,483,648 to 2,147,483,647
		Program Buffer Address	U16	0 to Program Buffer Size

Example

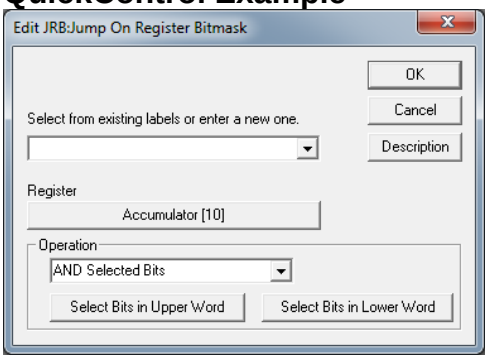
Jump to Program Buffer location 10 if Register # 10 > 0 and < 5.

@16 88 4 10 0 5 10 (CR)

Response

ACK only

QuickControl Example



PRI:Program Return On Input

See Also: PRT:Program Return

Description

The Program Return command is used as a complement to the Program Call command. Program execution continues at the command immediately following the Program Call. See Program Call (PCL) for details.

If a Program Return on Input is executed without a previous program called, the program will error, Stop execution and set Bit #12 in the Polling Status Word.

Placing a "0" in both parameters will cause an unconditional return.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PRI	Program Class D 202 (0xCA) 3 words Thread 1&2	Enable Code	S16	0 to –14 (-116 with extended I/O)
		Enable State	S16	0 or 1

Example

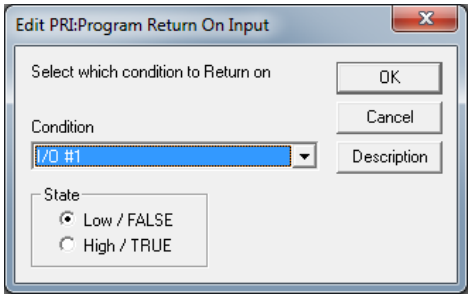
Return from Call if the Input #1 is Low ("0").

@16 202 -7 0 (CR)

Response

ACK only

QuickControl Example



PRT:Program Return

See Also: PRI:Program Return On Input

Description

The Program Return command is used as a complement to the Program Call command. Program execution continues at the command immediately following the Program Call. See Program Call (PCL) for details.

If a Program Return is executed without a previous Program Call, the program will error, stop execution and set Bit #12 in the Polling Status Word.

Placing a "0" in both parameters will cause an unconditional return.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PRT	Program Class D 202 (0xCA) 3 words Thread 1&2	Condition Enable	U16	0 to 32767
		Condition State	U16	0 to 32767

Example

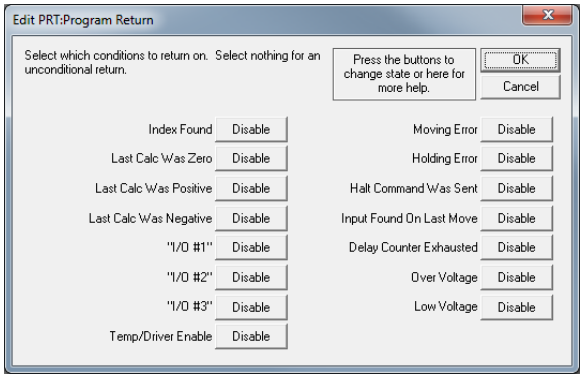
Return from Call if the last I/O #1 is High.

@16 202 16 16 (CR)

Response

ACK only

QuickControl Example



RSP:Restart, Program Mode

See Also: RST:Restart

Description

Allows the user to force a hardware restart of the controller. Program execution continues from the normal power on start-up sequence.

If executed immediately, no acknowledge is sent, as the processor resets.

See Memory Model in User Manual for details on downloading and running programs.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RSP	Program Class D 255 (0xFF) 1 word Thread 1&2	NONE	NONE	NONE

Example

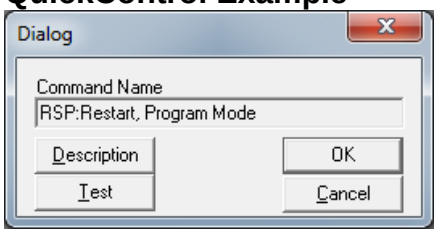
Restart the Processor.

@16 255 (CR)

Response

NONE

QuickControl Example



T1F:Thread 1 Force LRP

See Also: T2S:Thread 2 Start

Description

Multi-Threading Capability: Provides the ability for Thread 2 to force a program to load and run in Thread 1, in essence this forms a programmable motor recovery routine, with thread 2 being able to use multiple sources to cause multiple different recovery routines to be invoked. May also be used by a background task in thread 2 to select and run various motion commands in thread 1.

This command does NOT stop any motion happening in Thread 1. It changes Thread 1 to multitasking mode so that the new forced program can shutdown any existing motion under program control, and then executes a Thread 1 Load and Run Program (LRP) command.

See Multi-Thread Operation in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
T1F	Program Class D 75 (0x4B) 2 words Thread 2	NV Memory Address	U16	Start of Program in Non-Volatile memory.

Example

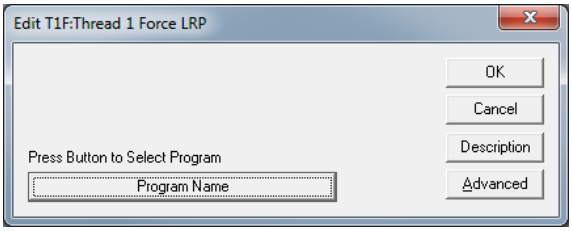
Load and Run program at non-volatile location 1000 in Thread 1.

@16 75 1000 (CR)

Response

ACK only

QuickControl Example



T2S:Thread 2 Start

See Also: T1F:Thread 1 Force LRP, T2K:Thread 2 Kill Conditions

Description

Multi-Threaded Capability: Start Thread 2, loading program from Non-Volatile memory into designated program buffer size carved from the top of Thread 1's program buffer. This command may also be used to override the existing program executing in Thread 2 while maintaining or changing the allocated buffer size. Finally, this program may be used to kill Thread 2 by setting the buffer allocation size to 0 (the Non-Volatile memory location will be ignored). QuickControl normally manages the total buffer memory use automatically; the maximum length thread 1 program plus the maximum length thread 2 program must be less than the command buffer size-1

Thread 2 Program Buffer is allocated from the top of Thread 1 Program Buffer. Thread 2 is allocated the indicated memory which equals Thread 2's Program Buffer size +1. Thread 1 buffer is decreased by the allocated amount plus 1 word. The buffer space is automatically restored when Thread 1 ceases operation. An END command will terminate Thread 2.

While Thread 2 is running, each thread alternates execution, causing each thread to execute at 240 microsecond ticks rather than 120 microsecond ticks. Bit 11 in the Internal Status Word 2 (IS2) is set while Thread 2 is active. This bit may be monitored by the Kill Motor Extended (KMX) to cause Thread 1 to react to the undesired termination of Thread 2.

See Multi-Thread Operation in User Manual for more details.

Note: This command is only available in Thread 1.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
T2S	Program Class D 76 (0x4C) 3 words Thread 1	NV Memory Address.	U16	Start of program in Non-Volatile Memory.
		Thread 2 Program Buffer Size	U16	0 (Kills Thread 2) 1 to 511 (up to half of Thread 1 Program Buffer) ** SD 38 increases the maximum thread 2 size to 901

Example

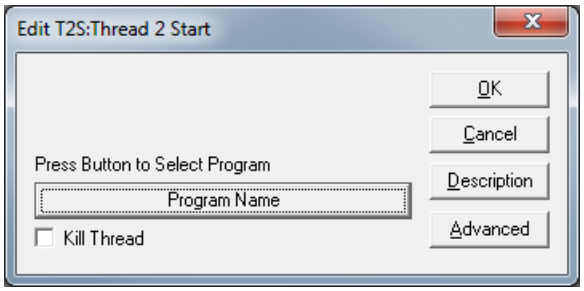
Start up thread 2 from Non-Volatile memory location 2000 allocating 200 words of Program Buffer space.

@16 76 2000 200 (CR)

Response

ACK only

QuickControl Example



WBE:Wait On Bit Edge

See Also: WBS:Wait On Bit State

Description

During program execution, the Wait on Bit Edge command causes the device to wait until the selected IO transition occurs. This is a very fast check that is done every servo cycle (120microseconds). Placing this command in a program will cause the program to wait on the current line until the input condition is met. There is no wait limit; therefore, this can put the device into an endless wait state. The I/O bit condition is edge triggered. The input must transition from High to Low for the Falling and Low to High for the Rising condition to be true. Many other conditions can be monitored in addition to the I/O. See QuickControl command for selection.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WBE	Program Class D 204 (0xCC) 3 words Thread 1&2	Enable Code	S16	1 to 71 (see QuickControl) (116 with extended I/O)
		Enable State	S16	0 = falling (High to Low) 1 = rising (Low to High)

Example

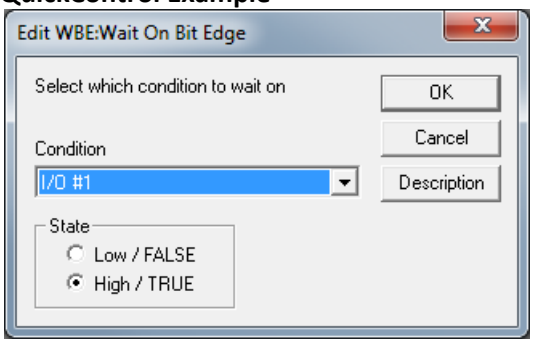
Cause program to wait until I/O #1 goes from Low to High.

@16 204 1 1 (CR)

Response

ACK only

QuickControl Example



WBS:Wait On Bit State

See Also: WBE:Wait On Bit Edge

Description

During program execution, the Wait on Bit State command causes the device to wait until a condition is true. This is a very fast check that done every servo cycle (120microseconds). Placing this command in a program will cause the program to wait on the current line until the input condition is met. There is no wait limit; therefore, this can put the device into an endless wait state.

The I/O bit condition is state triggered, if the condition is true when the command is encountered no waiting will occur. Note that many other conditions can be monitored by this command. See QuickControl for the other options.

See Program Flow Control in User Manual for parameter details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WBS	Program Class D 194 (0xC2) 3 words Thread 1&2	Enable Code	S16	1 to 71 (116 with extended I/O)
		Enable State	S16	0 = “Low” 1 = “High”

Example

Cause program to wait until I/O #1 is Low.

@16 194 1 0 (CR)

Response

ACK only

QuickControl Example

WDL:Wait Delay

See Also: DLY:Delay

Description

The Wait Delay command waits until the Delay Counter (Register 5) has decremented all the way to zero. Once it has reached zero, this command is exited and the next command in the Program Buffer is executed.

The Delay Counter is initialized using the Delay (DLY) command with a negative value parameter or by directly writing to register 5. This causes the counter to begin the count down to zero. When the count has expired the Wait Delay exits and allows the program to continue. (See Delay command above for more details.) The Delay counter may also be written with any of the register manipulation commands, either from the Serial Interface or from the program.

This command is useful when a timer needs to be set before a series of other commands are executed with a wait at the end. This allows a program or sub-routine to execute with precise timing.

NOTE: WDL should not be used in both Thread 1 and Thread 2 programs at same time. Each thread will be reset the counter (Register 5) when DLY is executed.

See Program Flow Control in User Manual for details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WDL	Immediate Class D 141 (0x8D) 1 word Thread 1&2*	NONE	NONE	NONE

*Be cautious of using WDL in both Thread 1 and Thread 2. The Delay Counter (Register 5) is common to both threads.

Example

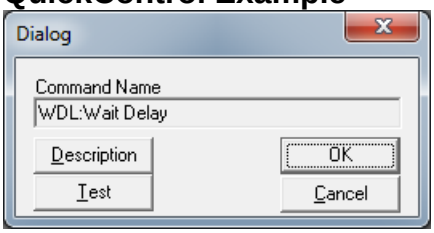
Cause program to wait until Delay Count is expired.

@16 141 (CR)

Response

ACK only

QuickControl Example



ACR:Analog Continuous Read

See Also: ARI:Analog Read Input, ARX:Analog Continuous Read Extended

Description

The Analog Continuous Read does continuous read (every 120uSec) of a selected analog channel into the given register. Reading and filtering of all channels into dedicated registers occur continuously in the background (only one ACR at a time).

The internal Analog to Digital Converter (ADC) is a 10-bit version, which yields approximately 0.005 volts per ADC count for SilverNugget and 0.0033 volts per ADC count for SilverDust. The input is filtered (5ms) and scaled up to a 15-bit value, but the resolution remains the same. Note that the maximum reading corresponds to 32*1023 = 32736.

SD35: An additional, low pass filter can be added by setting the upper 12 bits of the first parameter to the upper 12 bits of Filter Value (Fv) (see Scaling, Filter in User Manual). When using QuickControl, just check Enable and enter the filter value(Hz).

Analog Channel

0 = (Disable)
1 = Analog #1
2 = Analog #2
3 = Analog #3
4 = Analog #4
5 = Analog #1 and Analog #2
6 = Analog #3 and Analog #4
7 = V+ (non-calibrated)
8 = Temperature (ADC counts)
9 = V+ Scale Factor
10 = Processor V+
11 = Driver Temperature
12 = Processor V+ Scale Factor

SD35
13 = Velocity (SAV)
14 = Torque (STU)
15 = Position Error (counts)

See Application Note “QCI-AN023 Analog Inputs” for more information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ACR	Program Class D Code (Hex): 207 (0xCF) 3 words Thread 1&2	Filter[12] | Analog Chan#[4]	S16	See above
		Data Register	S16	Standard Register Range

Example

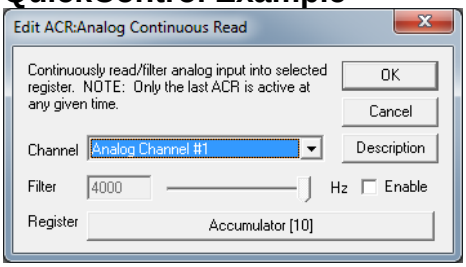
Configure Analog input #4 to do a continuous read to Data Register #26.

@16 207 4 26 (CR)

Response

ACK only

QuickControl Example



ARI:Analog Read Input

See Also: ACR:Analog Continuous Read, ARX:Analog Continuous Read Extended

Description

The Analog Read Input does a single read of a selected Analog Channel into given register. A reading is taken only once and transferred into the selected register.

The internal Analog to Digital Converter (ADC) is a 10-bit version, which yields approximately 0.005 volts per ADC count for SilverNugget and 0.0033 volts per ADC count for SilverDust. The input is filtered (5ms) and scaled up to a 15-bit value, but the resolution remains the same. Note that the maximum reading corresponds to 32*1023 = 32736.

Analog Channel

0 = (Disable)
1 = Analog #1
2 = Analog #2
3 = Analog #3
4 = Analog #4
5 = Analog #1 and Analog #2
6 = Analog #3 and Analog #4
7 = V+ (non-calibrated)
8 = Temperature (ADC counts)
9 = V+ Scale Factor
10 = Processor V+
11 = Driver Temperature
12 = Processor V+ Scale Factor SD35
13 = Velocity (SAV)
14 = Torque (STU)
15 = Position Error (counts)

See Application Note “QCI-AN023 Analog Inputs” for more information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ARI	Program Class D 193 (0xC1) 3 words Thread 1&2	Analog Channel #	S16	See above.
		Data Register	S16	Standard Register Range

Example

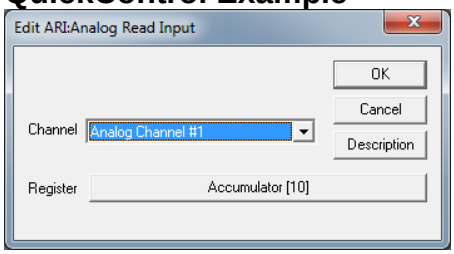
Read into data register #10 the V+ input voltage.

@16 193 7 10 (CR)

Response

ACK only

QuickControl Example



ARX:Analog Continuous Read Extended

See Also: ARI:Analog Read Input

Description

The Analog Continuous Read Extended does a continuous read (every 120uSec) of a selected analog channel into the given register. Reading and filtering of all channels into dedicated registers occur continuously in the background (only one ARX at a time). The X-series SilverMax and SilverNugget have additional channels which may be accessed via the ARX command, earlier designs do not have this command.

The internal Analog to Digital Converter (ADC) is a 12-bit version, which yields approximately 0.0008 volts per ADC count for X-Series controllers. The input is filtered (5ms) and scaled up to a 15-bit value, but the resolution remains the same. Note that the maximum reading corresponds to 8*4095 = 32760. An additional, low pass filter can be added by setting the filter parameter to the Filter Value (Fv) (see Scaling, Filter in User Manual). When using QuickControl, just check Enable and enter the filter value(Hz).

Analog Channel

0 = Disable function
1 = ANALOG #1 = IO4 = ADC 2
2 = ANALOG #2 = IO5 = ADC 3
3 = ANALOG #3 = IO6 = ADC 4
4 = ANALOG #4 = IO7 = ADC 4
5 = ANA1-ANA2 = IO4-IO5 = ADC2-ADC3
6 = ANA3-ANA4 = IO6-IO7 = ADC4-ADC5
7 = V+
8 = Temperature
9 = Voltage scale factor
10 = processor V+ for HC
11 = driver temp for HC
12 = processor V+ calibration for 1v for HC
13 = velo2
14 = requested torque
15 = position error - limited to 15 bits plus sign
16 = IO1 ANA
17 = IO2 ANA
18 = IO3 ANNA
19 = IO4 ANNA
20 = IO5 ANNA
21 = IO6 ANNA
22 = IO7 ANNA
23 = 0-10v input
24 = IO1-IO2
25 = IO3-IO4
26 = Reserved
27 = Reserved
28 = Analog input from motor Memory (when in analog)

See Application Note “QCI-AN023 Analog Inputs” for more information.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ARX SX and NX only	Program Class D Code (Hex): 95 (0x5F) 4 words Thread 1&2	Analog Chan#	S16	See Above
		Filter	S16	
		Data Register	S16	Standard Register Range

Example

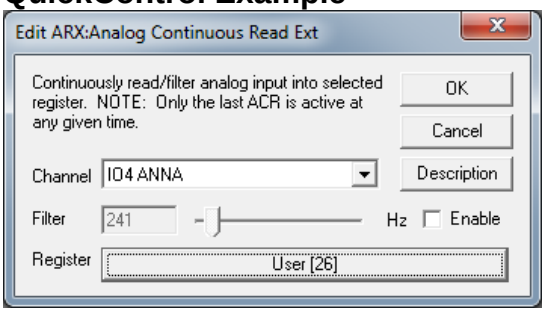
Configure Analog input #4 to do a continuous read to Data Register #26.

@16 95 4 0 26 (CR)

Response

ACK only

QuickControl Example



CIO:Configure I/O

See Also: CII:Configure I/O, Immediate Mode, COB:Clear Output Bit
SOB:Set Output Bit

Description

Configures the selected digital I/O bit for input or output. When setting as an output the logic level state is also set. Each I/O bit is individually set using this command, the power-up default is all I/O bits are inputs. This prevents I/O conflicts.

Note: Extended I/O 101 through 116 are open collector only: Mode 0 (Clear) drives the output low (driver on), while –1 and 1 both turn off the driver and put the device in input mode. Extended I/O allows the input to be read back even when driven. This allows for fault detection – that is, if the output has been driven low, but the input reads high, the output has shut down due to overcurrent. (Note: user needs to either delay the time period of the input filter or set the input filter to zero to prevent false error reports due to the delay of the filtering of the input signals.)

X-Series SilverMax and SilverNuggets also have programmable pull-up/pull-down resistors that are configured by this command. These should only be configured when the I/O is in input mode to maximize available drive current for output mode. Pull inactive puts the driver to the 2.2k resistor in a tristate condition. The pull resistor should also be inactive for analog reading to maximize accuracy.

See Input and Output Functions in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CIO	Program Class D 188 (0xBC) 3 words Thread 1&2	I/O Line #	S16	I/O Line #
		Mode	S16	-1 = Input 0 = Clear (Low) 1 = Set (High) X-Series adds: -2 = Pull-up -3 = Pull-down -4 = Pull inactive

Example

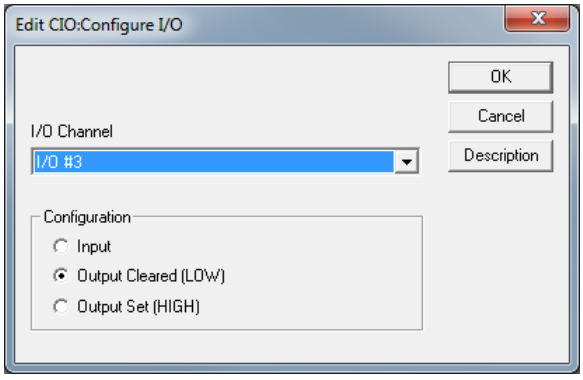
Set I/O bit #3 as output “Low”.

@16 188 3 0 (CR)

Response

ACK only

QuickControl Example



COB:Clear Output Bit

See Also: CIO:Configure I/O,SOB:Set Output Bit

Description

Clears the selected Digital I/O bit to a logic Low (“0”) condition (Output = 0 volts). If the I/O was configured as an input this will reconfigure the bit as an output and clear it to logic Low (“0”).

See Input and Output Functions in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
COB	Program Class D 206 (0xCE) 2 words Thread 1&2	I/O Line #	S16	I/O Line #

Example

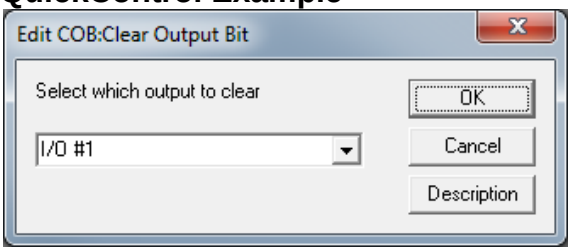
Clear I/O bit #1 to a low (“0”) state.

@16 206 1 (CR)

Response

ACK only

QuickControl Example



PCP:Position Compare

See Also: PLS:Programmable Limit Switch

Description

This command enables a background routine which compares the current position to the user supplied "Trigger Position". When the current position crosses over the trigger position, I/O #1 is toggled. If the crossing was in the positive direction, then the "Modulo" register is added to the old trigger position to form a new trigger position; if the crossing was in the negative direction, then the "Modulo" register is subtracted from the old trigger position to form a new trigger position.

If the "Modulo" value is zero, then the "Trigger Position" is not modified, and the I/O bit will represent a straight comparison of actual (measured) position versus "Trigger Position".

The "Trigger Position" is stored in the first of two User Data Registers, the "Modulo" value is stored in the second of the two User Data Registers. I/O #1 must be configured as an output with the desired starting state prior to running this command.

The First Data Register = "Position"
The Second Data Register = "Modulo"

This command is accomplished using a software compare (updated every 120 usec.) and therefore may have a small delay of 120 microseconds from a compare to the actual I/O #1 change of state.

Note: Once this command is executed, the background routine runs until a PCP with a Data Register=0 is issued.

Command Info

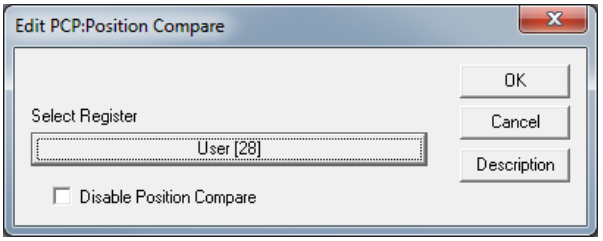
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PCP	Program Class D 245 (0xBF) 2 words Thread 1&2	Data Register	U16	0 = Disable Usage Standard Register Range

Example

Enable Position Compare using Data Register #28 for the Position compare value and #29 for the Modulo value.

@16 245 28 (CR)

QuickControl Example



Response

ACK only

PLS:Programmable Limit Switch

See Also: PLT:Programmable Limit Trigger

Description

The PLS command allows user to construct a data table up to any number of pre-defined of trigger points. The pre-defined trigger points will be stored in the selected user registers in units of encoder counts. The flexibility of the command also enables user to choose the I/O to trigger and the modulo point where the cycle repeats.

Every 120 uSec the controller compares the current position with the trigger point position. When the transition point is reached, the state of the I/O will transition automatically.

See QCI-AN050 Programmable Limit Switch for a detailed explanation of this command include example programs.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
PLS	Program Class D 78 (0x46) 5 words Thread 1&2	IO Line #	U16	Any Output 0=Disable
		Initial State	U16	0 or 1
		Starting Data Register	U16	Standard Register Range
		Number of Triggers	U16	Limited by number of available registers.

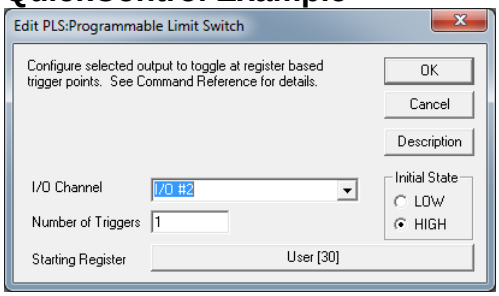
Example

@16 78 2 1 30 4 (CR)

Response

ACK only

QuickControl Example



PLT:Programmable Limit Trigger

See Also: PLS:Programmable Limit Switch

Description

Programmable Limit Trigger (PLT) allows enabling/disabling I/O triggering set up by PLS. PLT also provides additional flexibility by changing the number of trigger points without resetting the modulo counter. For example, if the application requires 4 triggers from 10000 – 20000 counts and only 2 triggers from 20000-30000 counts. Please note that in order to use PLT, prior PLS command setup is required.

See QCI_AN050 Programmable Limit Switch for a detailed explanation of this command include example programs.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PLT	Program Class D 79 (0x47) 4 words Thread 1&2	IO Line #	U16	Any Input 0=disable
		Initial State	U16	0 or 1
		Number of Triggers	U16	Limited by number of available registers.

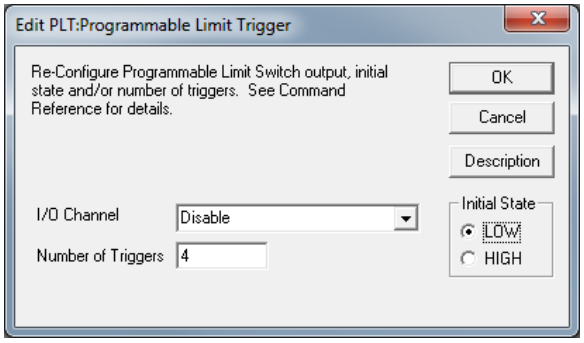
Example

Disable PLS
@16 79 0 0 4 (CR)

Response

ACK only

QuickControl Example



PWO:PWM Output

Description

The PWM Output command outputs a Pulse Width Modulated (PWM) signal to IO2. While this signal is active, all other I/O commands to this output are ignored. The PWM signal is dynamically modified every 120 microseconds according to the real time contents of the given register without further intervention until disabled.

The PWM signal is a 25KHz (40uS) total time, with a 50% duty cycle corresponding to a zero input, a solid high output corresponding to 32767 input, and a solid low output corresponding to –32768 input. The input may be taken from either the high or low word of any register.

Mode 0 is used to disable the PWM output, and returns IO2 to its previously commanded state.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
PWO	Program Class D 129 (0x81) 3 words Thread 1&2	Register	U16	Standard Register Range
		Mode	S16	0 = Disable 1 = High Word 2 = Low Word

Example

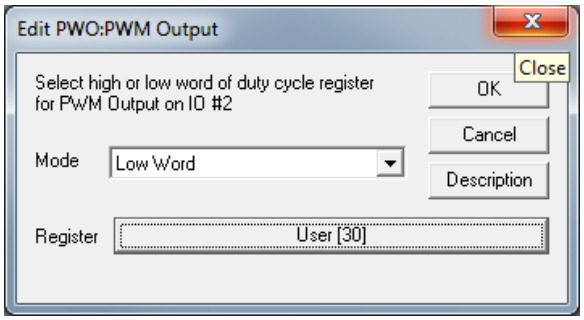
Enable PWM using Register 30 Low word.

@16 129 30 2 (CR)

Response

ACK only

QuickControl Example



SOB:Set Output Bit

See Also: CIO:Configure I/O,COB:Clear Output Bit

Description

Sets the selected Digital I/O bit to a logic High (“1”) condition. If the I/O was configured as an input this will reconfigure the bit as an output and set it to logic High (“1”).

See Input and Output Functions in User Manual for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SOB	Program Class D 205 (0xCD) 2 words Thread 1&2	I/O Line #	S16	I/O Line #

Example

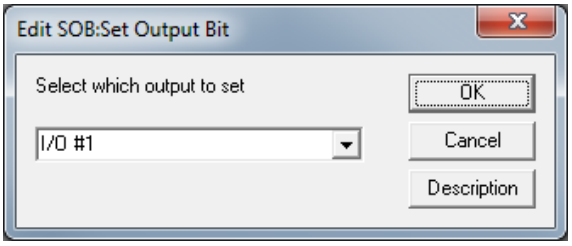
Set I/O bit #1 to a High (“1”) state.

@16 205 1 (CR)

Response

ACK only

QuickControl Example



SP2:SPI Port 2

See Also: SMD command

Description

Allows sending and receiving data from the second (external) SPI port to communicate with memories and I/O

Operations is the number of cycles to repeat in the background for each execution. Negative values indicate the number of cycles to implement, but to auto increment either the source or destination register. The source register will auto incremented unless is it is a special value of either -1 or 0, and then the destination register will be auto incremented instead.

The source register is normally where the data to be sent will be taken from. 0 is a special value that sends all zeros, and -1 is a special value which will send all 1's . These are useful when reading back data.

The destination register is where the received data will be stored. 0 is a special value indicating do not save the data.

Bytes is the number of bytes to send from each register for each operation. Data is right aligned if not all (4) bytes of the register are sent. Data is sent Big-Endian (most significant byte first).

ChipSelect: chipselect operation prior to sending the first block:

0 = Do nothing to the Chip Select signal

1 = pulse high and then leave low (falling edge chipselect)

2 = pulse low and then leave high (rising edge chipselect) (CS inverted from CS* signal)

Handshake: 0=none, 1 = wait for MISO to go low before starting each cycle (used to synchronize with certain ADC circuits.

This command is used with SMD modes 20,21,22 to configure the timing and the clock as well as to do a long hand shake with continued Chipselect operations to keep a watch-dog retriggerable monostable from timing out.

Contact the Factory for more information

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SP2 SS 34 E series 29	Program Class D 224 (0xE0) 7 words Thread 1&2	Operations	S16	-32767 - 32767 #
		Source	S16	-1,0,11 to 199
		Destination	S16	0, 11 to 199
		Bytes	S16	1 to 4
		Chip Select	S16	0 to 2
		Handshake	S16	0 or 1

Example

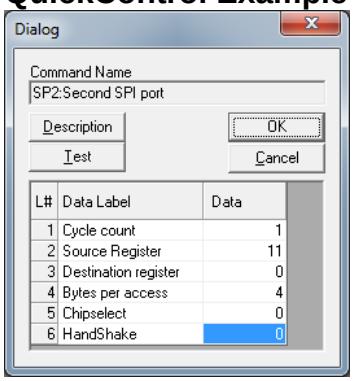
Send 1 cycle of 4 bytes of data from register 10 receiving data to register 11, do noe change the chipselect, and do not wait for handshake signal.

@16 224 1 10 11 4 0 0 (CR)

Response

ACK only

QuickControl Example



CLC:Calculation

See Also: CLX:Calculation, Extended,CLD:Calculation Extended With Data

Description

The Calculation command provides basic math, logic and other function using Data Registers. The command uses two parameters (combined into a single word), Operation and Data Register, to perform all of its defined operations. Several of the operations have two Operands to perform the calculation. When two Operands are required, Data Register #10 is used as one of the operands while the selected Data Register is used for the second operand. Typically, Data Register #10 Accumulator is used as the destination for a two-operand operation. For single Operand operations, the selected Data Register is used as the source and/or destination. Limited to registers 0-255.

Data Register #10 is typically used as an accumulator but may also be the Selected Data Register.

The standard Multiply operations operate on the entire 32 bit word, but only return the 32LSB of the result.

The Divide command takes a 32 signed Dividend and a 16 bit positive divisor (up to 32767), and produces a signed 32 bit quotient.

When performing math functions the read only data registers can be used as the selected data register. Data cannot be saved or written to these registers due to their read only nature. User data registers can be used for any purpose as they are designed for both read and write operations. (See appendix in User Manual for definitions of Data Registers.)

Calculations affect the conditions of the Internal Status Word. Depending on the result of an operation one of three different conditions will occur (zero, positive, negative). See Internal Status Word in the User Manual for more details.
NOTE: There are two related Calculation Commands, Calculation (CLC) and Calculation Two Word (CTW). CLC requires byte combination of the Operation and Register parameters, whereas CTW breaks these into separate parameters. CLC only uses 2 words in the Program Buffer while CTW uses 3 words. As CLC requires the combination of two bytes into a word, it may be too difficult to use in applications programmed without QuickControl (i.e. host programming). CTW is also limited to registers 0-255.

The following is a summary of the CLC operations. For more details see Technical Document “QCI-TD026 Calculation Command”.

Operation Parameter Definitions

NOTE: Acc = Accumulator (register 10)

Code	Operation
0	Clear (Reg=0)
1	Add (Acc = Acc + Reg)
2	Sub (Acc = Acc - Reg)
3	Copy (Acc = Reg)
4	Increment (Reg = Reg +1)
5	Decrement (Reg = Reg -1)
6	Absolute Value (Reg = ABS(Reg))
7	Sub Target Position (Targ-Reg, Pos-Reg) Subtracts the register from both the Target and Position regs
8	Copy (Reg = Acc)
9	Copy Word, Sign Extend (Acc = HI(Reg)) Loads the high word of register(sign extend) into register #10.
10	Copy Word, Sign Extend (Acc = LO(Reg)) Loads the low word of register(sign extend) into register #10.
11	AND (Acc = Acc AND Reg)
12	OR (Acc = Acc OR Reg)
13	XOR (Acc = Acc XOR Reg)
14	Div - S32/U16 Bit (Acc = Acc/LO(Reg)) Divide signed 32 bit long word of Register #10 by the positive valued of low word of selected Data Register. 32 bit result is placed Register #10
15	Mult - Unsigned (Acc = Acc * Reg) Unsigned multiply of register #10 32 bit long word and 32 bit long word of selected register. 32 LSB of result is placed in Register #10. (User must keep terms appropriate such that the result fit in a 32 bit result field.
16	Mult - Signed (Acc = Acc * Reg) Signed multiply of Register #10 32 bit long word of and the signed 32 bit long word of selected register. 32 bit LSB of result is placed in Register #10. User must select values that limit the signed product to fit in 32 bits.
17	Acc H-Word = LO(Reg) Replace the upper word of Register #10 with the low word of the selected register
18	Copy Reg Ref (Acc = reg#, reg#=value of Reg Loads Register #10 with the contents of the Register addressed by the data within the Selected Register (selected register is a pointer to the data location).
19	Copy Reg Ref (reg# = Acc, reg#=value of Reg Copies Register #10 contents to the Register addressed by the data within the Selected Register (selected register is a pointer to the data save location).
20	Copy Word (Reg H-Word = LO(Acc)) Copies the Low word of Register #10 to the High word of the selected register. Used to write to half of a combined word register.
21	Copy Word (Reg L-Word = LO(Acc)) Copies the Low word of Register #10 to the Low word of the selected register. Used to write to half of a combined word register
22	Shift Reg Left Performs a 32 bit Left Shift of the selected Register.

23	Shift Reg Right w/ Sign Extend Performs a 32 bit sign extended right shift of the selected Register. Implements a signed divide by 2.
24	Shift Reg Right w/o Sign Extend Performs a 32 bit right shift of the selected Register. Implements an unsigned divide by 2.
25	Modulo 32 % 16 Bit (Acc % LO(Reg)) Performs a modulo (remainder) calculation using the signed 32 bits of Register 10, with the positive (only) divisor being the lower word of the selected Data Register. Note the remainder will always be positive, following the standard Modulo format.
26	Max (Acc = Max(Acc,Reg)) The larger of the values of Register 10 and the selected Register are stored to Register 10
27	Min (Acc = Min(Acc,Reg)) The smaller of the values of Register 10 and the selected Register are stored to Register 10
28	Sub (Acc = Reg - Acc) Note: Sames as code 2 but with parameters swapped.
29	Negative (Reg = -Reg)
30	Add (Acc = HI(Reg) + Acc) Take the high word of the selected register, sign extend it and add it to Register 10
31	Add (Acc = LO(Reg) + Acc) Take the low word of the selected register, sign extend it and add it to Register 10

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CLC	Program Class D 165(0xA5) 2 words Thread 1&2*	Operation = Upper Byte Data Register = Lower Byte	U16	Operation: See Previous Table Data Register: Standard Register Range

*Thread 2 maintains its own copy of the Accumulator (Register 10) and the zero/positive/negative bits. See Multi-Thread in User Manual for more details.

Example

Decrement Accumulator

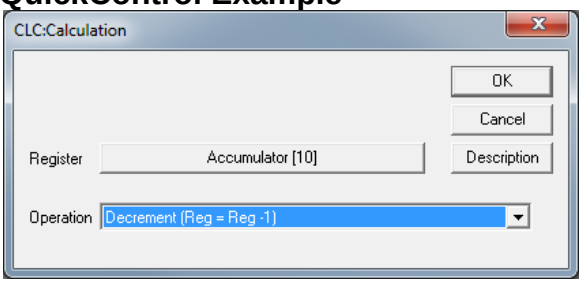
Operation/Register = 5 * 2^8 + 10 = 1290

@16 165 1290 (CR)

Response

ACK only

QuickControl Example



CLD:Calculation Extended With Data

See Also: CLX:Calculation Extended,CLC:Calculation

Description

The Calculation with Data command provides basic math, logic and other function using Data Registers with the second Parameter being a constant. This command allows one source register, one Constant and one result register. The result register may be the same as the source register, if wanted. If the operation only needs a single register, then the source register is used.

If bit 8 of the Result Register parameter is set, the Result Register will be pre-saved to the Accumulator (register 10) prior to the operation. In QuickControl, this bit is set when the "Pre-save…" check box is checked.

The standard Multiply operations operate on the entire 32 bit word, but only return the 32LSB of the result.

Mixed Multiply performs a 32 x 32 multiply with the middle 32 bits returned, essentially dividing the result by 65536. The result may be viewed as multiplying an integer by a mixed fraction in the range of - 32768 to ~ 32767.99985 with a resolution of 1/65536 or ~.000015.

The extended multiply provides a 32x32 multiply with 64 bits returned in two adjacent user registers.

The Divide command takes a 32 signed Dividend and a 16 bit positive divisor (up to 32767), and produces a signed 32 bit quotient. MOD takes the same parameters, but returns the standard (positive) modulo value.

A 64bit/32 bit Divide returns a 32 bit result. Divisor should be positive.

When performing math functions the read only data registers can be used as input parameters. The result register must be writable. (See Data Register Commands in User Manual for details and definitions of Data Registers.)

Calculations affect the conditions of the Internal Status Word. Depending on the result of an operation one of three different conditions will occur (zero, positive, negative). Thread 2 maintains its own copy of these bits so conditional tests on zero/positive/negative is thread independent. See Internal Status Word and Multi-Threading in the User Manual for more details.
NOTE: There are two related Calculation Commands, Calculation Extended (CLX) and Calculation with Data (CLD). CLD uses a 32 bit constant as the second parameter.

Operation Parameter Definitions

Firmware Rev.	Code	Operation
SD 05	0	Add (Result = Param 1 + Param 2)
SD 05	1	Sub (Result = Param 1 - Param 2)
SD 05	2	Copy (Result = Param 1)
SD 05	3	Absolute Value (Result = ABS(Param 1))
SD 05	4	Copy Word, Sign Extend (Result= HI(Param 1))
SD 05	5	Copy Word, Sign Extend (Result= LO(Param 1)
SD 05	6	AND (Result = Param 1 AND Param 2)
SD 05	7	OR (Result = Param 1 OR Param 2)

SD 05	8	XOR (Result = Param 1 XOR Param 2)
SD 05	9	Div - S32/U16 Bit (Result = Param 1/LO(Param 2)) Note: divisor must be positive – up to 32767
SD 05	10	Mult - Unsigned (Result = Param 1 * Param 2) Returns the lower 32 bits of the product; both parameters are considered as positive numbers
SD 05	11	Mult - Signed (Result = Param 1 * Param 2) Returns the lower 32 bits of the product; both parameters are considered as signed numbers
SD 05	12	Mult - Signed 64Bit (Result = (Param 1 * Param 2)>>16) Effectively performs signed mixed fractional math if user scales up one of the parameters by 65536.
SD 05	13	Copy Words (Result = LO(Param 1)<<16 + LO(Param 2))
SD 05	14	Copy Words (Result = HI(Param 1)<<16 + LO(Param 2))
SD 05	15	Copy Words (Result = HI(Param 1)<<16 + HI(Param 2))
SD 05	16	Modulo 32 % 16 Bit (Result = Param 1 % LO(Param 2)) Note: divisor must be positive – up to 32767 This is the remainder function of the Divide operation
SD 05	17	Max (Result = Max of Param 1 or Param 2)
SD 05	18	Min (Result = Min of Param 1 or Param 2)
SD 05	19	Negative (Result = -Param 1)
SD 05	20	Sub (Result = Param 2 - Param 1) Note: Same as code 1 but parameter order reversed.
SD 05	21	Div - S32/U16 Bit (Result = Param 2/LO(Param 1)) Note: divisor must be positive – up to 32767 Note: Same as code 9 but parameter order reversed.
SD 05	22	Modulo 32 % 16 Bit (Result = Param 2 % LO(Param 1)) Note: divisor must be positive – up to 32767 This is the remainder function of the Divide operation Note: Same as code 16 but parameter order reversed.
SD 06	23	Add (Result = HI(Param 1) + Param 2) Take the high word of the first parameter, sign extend it and add it to the Constant
SD 06	24	Add (Result = LO(Param 1) + Param 2) Take the low word of the first parameter, sign extend it and add it to the Constant

Queue Operators

	25	Queue Init (Param 1:Base, Param 2:Size) Initialize queue starting at Param 1 Register. Set max size to Param 2. Head=Tail=0
	26	Queue-Cmd Err Push Head (Param 1: Head <= Param 2) Pushes Param 2 unto Head of Param 1 queue Result = Size of Queue Used Command Error occurs if queue overflows
	27	Queue-Cmd Err Pop Head (Param 1:Head => Result) Pops Head of Param 1 queue to Result Command Error occurs if queue underflows
	28	Queue-Cmd Err Push Tail (Param 1: Tail <= Param 2) Pushes Param 2 unto Tail of Param 1 queue Result = Size of Queue Used Command Error occurs if queue overflows
	29	Queue-Cmd Err Pop Tail (Param 1:Tail => Result) Pops Tail of Param 1 queue to Result Command Error occurs if queue underflows
	30	Queue-Cmd Err Read Element (Result = Param 1:Element[Param 2]) Reads Param 2 element of Param 1 queue into Result Command Error occurs if read out of bounds
	31	Queue Push Head (Param 1: Head <= Param 2) Pushes Param 2 unto Head of Param 1 queue Result = Size of Queue Used Zero Flag=Success, Neg Flag=Overflow
	32	Queue Pop Head (Param 1:Head => Result) Pops Head of Param 1 queue to Result Zero Flag=Success, Neg Flag=Underflow
	33	Queue Push Tail (Param 1: Tail <= Param 2) Pushes Param 2 unto Tail of Param 1 queue Result = Size of Queue Used Zero Flag=Success, Neg Flag=Overflow
	34	Queue Pop Tail (Param 1:Tail => Result) Pops Tail of Param 1 queue to Result Zero Flag=Success, Neg Flag=Underflow
	35	Queue Read Element (Result = Param 1:Element[Param 2]) Reads Param 2 element of Param 1 queue into Result Zero Flag=Success, Neg Flag=Out of Bounds
	100	Mult - Unsigned 64Bit (Result(U64)= Param 1 * Param 2) 64 bit result stored in two consecutive registers.
	101	Dev - U64/U32 (Result = Param 1(U64)/Param 2) 64 bit numerator stored in two consecutive registers.
SD 61 SS 30 SX,NX - all	102	Root – U64 return U32 square root

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CLD	Program Class D 200(0xC8) 6 words Thread 1&2*	Param 1 - Register	U16	Data Register: Standard Register Range
		Param 2 - Data	S32/ U32	32 bit constant (treated signed or unsigned according to operation)
		Operation	U16	See Operation Definitions above
		Option = Upper Byte 0x01 = Pre-save Result Register to Accumulator Result Register = Lower Byte	U16	Writable Data Register: Standard Register Range

*Thread 2 maintains its own copy of the Accumulator (Register 10) and the zero/positive/negative bits. See Multi-Thread in User Manual for more details.

Example

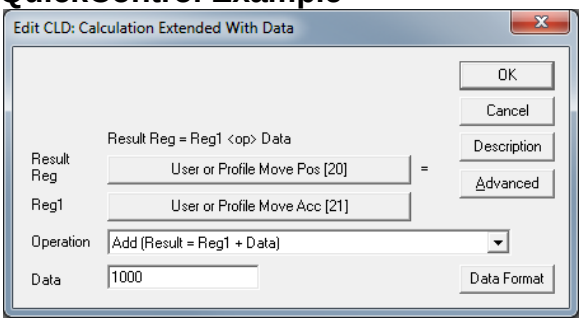
Register 20=Register 21 + 1000

@16 200 21 1000 0 20 (CR)

Response

ACK only

QuickControl Example



CLX:Calculation Extended

See Also: CLD:Calculation Extended With Data,CLC:Calculation

Description

The Calculation Extended command provides basic math, logic and other function using Data Registers. This command allows up to two source registers and one destination register. The destination register may be the same as a source register, if wanted. If the operation only needs a single register, then the first source register is used.

This is basically the same command as Calculation Extended with Data (CLD) except CLD's 2nd parameter (Param 2 - Data) is replaced with a second data registers (Param 2 - Register). This allows operations on two registers to be saved to a third register.

See Calculation Extended With Data (CLD) for Operation parameter definitions and notes on how operations are performed.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CLX	Program Class D 158(0x9E) 5 words Thread 1&2*	Param 1 - Register	U16	Data Register: Standard Register Range
		Param 2 - Register	U16	Data Register: Standard Register Range
		Operation	U16	See Operation Definitions in CLD command
		Option = Upper Byte 0x01 = Pre-save Result Register to Accumulator Result Register = Lower Byte	U16	Writable Data Register: Standard Register Range

*Thread 2 maintains its own copy of the Accumulator (Register 10) and the zero/positive/negative bits. See Multi-Thread in User Manual for more details.

Example

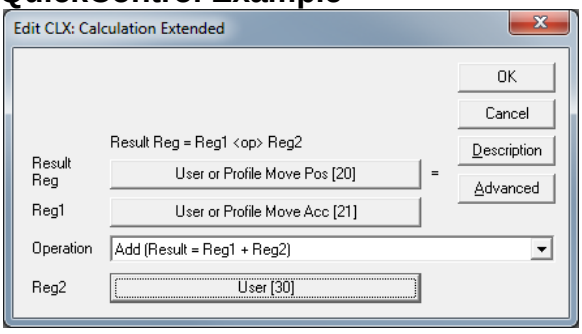
Register 20=Register 21 + Register 30

@16 158 21 30 0 20 (CR)

Response

ACK only

QuickControl Example



MMA:Motor Memory Access

Description

The Motor Memory Access command allows the user to access the on-board memory in the I-Grade Motors when using an I-Grade controller. The first 14 pages of memory (0 to 13) are reserved for Factory use, and are locked against writing by a user. Pages 14 and 15 are provided for the user to store up to 4 registers of information (Two registers may be stored per Page). Each Page operation transfers 2 user registers either to or from the Motor Memory.

The Motor Memory is communicated to through the Encoder/Driver cable; to prevent noise from the driver from affecting the memory operations, the motor driver must be disabled. (Running the MMA command with the driver enabled will result in a sequence error.)

The Motor Memory also has a 64bit permanent serial number which may be accessed via the MMA command. This number is set by the chip vendor and cannot be selected or altered. When reading the serial number, the Page argument is ignored.

Note: The user pages of the Motor Memory do not have an automatic checksum. The user may implement their own checksum if needed. The data is, however read back multiple times and verified to avoid communications errors.

Note: The success or failure of this operation is checked by testing the Zero, Positive, and Negative bits in the Internal Status word:

Zero = successful operation
Positive = unsuccessful operation – errors detected
Negative = no motor memory detected

Note: Motor Memory is only accessible using I-Grade motors with I-Grade SilverDust controllers.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MMA	Program Class D 133 0x85) 4 words Thread 1	Mode	U16	0 = Read 1 = Write 2 = Read Serial #
		Register	U16	10 to 198
		Page	U16	0 to 15 for read 14 or 15 for write

RLM:Register Load Multiple

See Also: RSM:Register Store Multiple (RSM)

Description

Loads an array of data from the selected non-volatile memory address to an array of data registers. A checksum value is verified to insure good data. For arrays of more than 1 register, the Data Registers targeted must all lie in the range of 10 to 199. Single register loads may be done to any writable register.

During the load process, the data is used to calculate a checksum value. When the load is complete, the calculated checksum is compared to the stored checksum. If the checksums do not agree bits #14 & #12 in the Polling Status Word are set ("1") to indicate a register load failure.

The Non-Volatile Memory may be indirectly addressed by putting the wanted address into Register # 10, and then using a NV Memory Address of zero. See Application Note "QCI-AN046 Indirect Addressing" for more details.

Use a negative Starting Data Register to allow the program to continue even if a Command Error occurs. A "zero" bit in the ISW word indicates success, while a "negative" indicates an error occurred and the data is no good.

NOTE: If this command is used in QuickControl's "Normal Mode", many of the complexities go away.

See Application Note "QCI-AN048 Register Files" for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RLM	Program Class D 197 (0xC5) 4 words Thread 1&2*	Number of Registers	S16	1 to 10
		Starting Data Register	S16	Standard Data Range > 10 0 for indirect addressing Negative to suppress Command Error
		NV Memory Address	U16	Max Size at NV memory

*NV mem can only be accessed by one thread at a time. Other thread will automatically wait to access.

Example

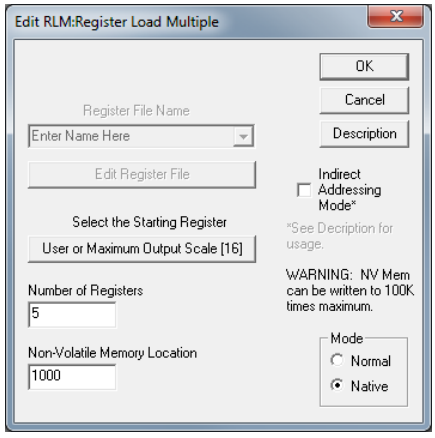
Sequentially Load 5 data registers starting at #16 with the data from NV memory address 1000

@16 197 5 16 1000 (CR)

Response

ACK only

QuickControl Example



RLN:Register Load Non-Volatile

See Instead: RLM:Register Load Multiple

Description

Loads data from the selected non-volatile Memory address into the selected Data Register. A Checksum value is verified to insure good data. Data may only be loaded into writable registers.

See Non-Volatile Memory in User Manual for details on loading and storing data.

The loading process is the same as used by the Register Load Multiple with only one register being loaded. The data selected must be stored using the Register Load Non-volatile or the Register Load Multiple using "1" for the Number of Registers.

See RLM for more parameter information.

NOTE: If this command is used in QuickControl's "Normal Mode", many of the complexities go away.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RLN	Program Class D 199 (0xC7) 3 words Thread 1&2*	Data Register	S16	Standard Data Range > 10 0 for indirect addressing Negative to suppress Command Error
		NV Memory Address	U16	Max Size of NV Memory

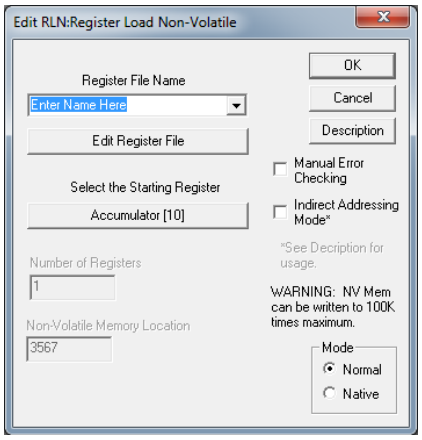
*Non-Volatile memory can only be accessed by one thread at a time. Other thread will automatically wait to access Non-Volatile memory.

Example

Load data register #12 with data from NV memory address 1000.

@16 199 12 1000 (CR)

QuickControl Example



Response

ACK only

RSM:Register Store Multiple

See Also: RLM:Register Load Multiple (RLM)

Description

Stores data from an array of Data Registers to the selected NV Memory address. A Checksum value is calculated from the array and stored with the array. Data from the selected Data Registers is stored sequentially to NV Memory. Data is also copied from the Data Registers sequentially. See Memory Model in User Manual for more details.
Multiple register stores must only be done from registers in the range 10 to 199. Single register stores may be done from any register.

The NV Memory may be indirectly addressed by putting the wanted address into Register # 10, and then using a NV Memory Address of zero. See Application Note "QCI-AN046 Indirect Addressing" for more details.

NOTE: If a rapidly changing data register is stored to NV Memory, the data has the possibility of being inaccurate. The device performs two 16 bit writes from the 32 bit data register to NV Memory. If the “data” in the register changes before the second 16 bit write cycle, then it will be incorrect. It is advisable to copy the data from the changing register to a user register and then storing the user register to NV Memory.

NOTE: If this command is used in QuickControl's "Normal Mode", many of the complexities go away.

NOTE: SD05: A Command Error will result if attempts are made to write to protected sections of Non-Volatile memory bits 12 and 14 will be set in the STATUS word.

See Application Note QCI-AN048 Register Files on our website for details.

Setting register number parameter to a negative value causes program to continue even if a Command Error occurs. A “zero” bit in the ISW word indicates success, while a “negative” indicates an error occurred and the data is no good.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RSM	Program Class D 196 (0xC4) 4 words Thread 1&2*	Number of Registers	S16	1 to 10
		Starting Data Register	S16	Standard Data Range 0 for indirect addressing
		NV Memory Adr	U16	Max Size at NV Memory

*NV mem can only be accessed by one thread at a time. Other thread will automatically wait to access.

Example

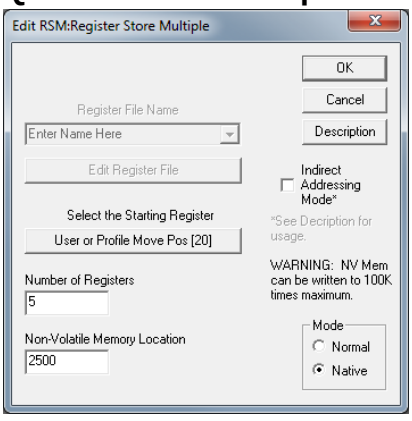
Store 5 data registers starting at #20 to NV memory starting at address 2500

@16 196 5 20 2500 (CR)

Response

ACK only

QuickControl Example



RSN:Register Store Non-Volatile

See Instead: RSM:Register Store Multiple

Description

Stores data from a Data Register to the selected Non-volatile Memory address. A Checksum value is calculated from the data and stored with the data.

The storing process is the same as used by the Register Store Multiple with only one register being stored. The data selected may be loaded using the Register Load Non-volatile or the Register Load Multiple using "1" for the Number of Registers.

See command Register Store Multiple (RSM) for more details.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
RSN	Program Class D 198 (0xC6) 3 words Thread 1&2*	Data Register	S16	Standard Data Range 0 for indirect addressing Negative to suppress Command Error
		NV Memory Address	U16	Max Size at NV Memory

*Non-Volatile memory can only be accessed by one thread at a time. Other thread will automatically wait to access Non-Volatile memory.

Example

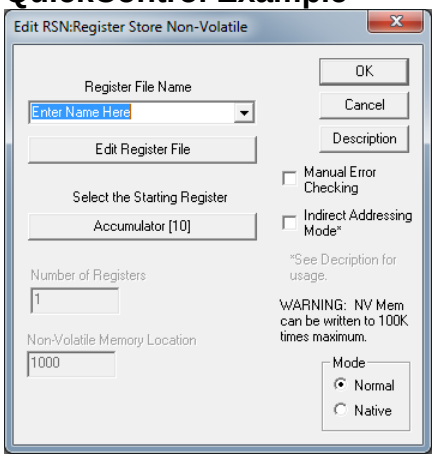
Store data register #1 to NV memory address 1612.

@16 198 1 1612 (CR)

Response

ACK only

QuickControl Example



WCL:Write Command Buffer Longword

See Also: WCW:Write Command Buffer Word

Description

This command allows program space starting at the selected program buffer location to be overwritten with the 32-bit data in the selected register. This allows for self-modification of the command parameters within the program buffer. Any of the command parameters can be dynamically modified within the program. This command specifically is intended to modify 32 bit parameters. Extreme care should be used when writing any self-modifying code to prevent unwanted outcomes. The QuickControl tool has support for this command, which simplifies its application, and enforces consistency checks. However, values being transferred are dynamic, based on the contents of the selected register; the range of the data is *not* verified at transfer, so undesired results may be obtained if out of range parameters are assembled into the program buffer, including Sequence Error shutdowns. This command does allow for great flexibility by allowing any of the parameters to be made register based.

In QuickControl, first enter the line to be modified and place a label on the same line. Then insert the WCL command, and reference the label to select the desired command, select the register and the parameter.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WCL	Program Class D 138 (0X8A) 3 words Thread 1&2	Data Register	U16	Standard Register Range
		Program Buffer Address	U16	Valid NV Memory Range

Example

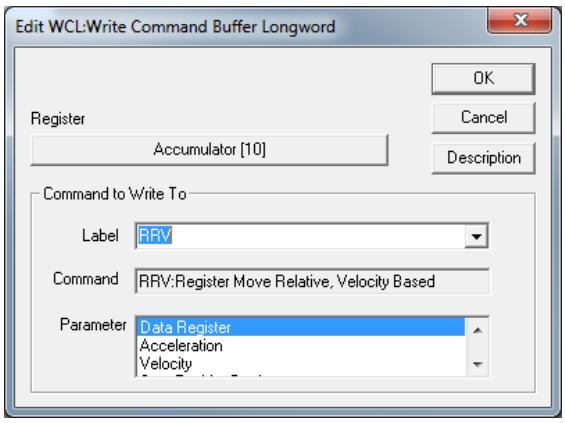
Overwrites the Command Buffer contents at locations 122 and 123 with the contents of Register #10.

@16 138 10 122 (CR)

Response

ACK only

QuickControl Example



WCW:Write Command Buffer Word

See Also: WCW:Write Command Buffer Longword

Description

This command allows program space starting at the selected program buffer location to be overwritten with the lower word 16-bit data in the selected register. This allows for self-modification of the command parameters within the program buffer. Any of the command parameters can be dynamically modified within the program. This command specifically is intended to modify 16 bit parameters. Extreme care should be used when writing any self-modifying code to prevent unwanted outcomes. The QuickControl tool has support for this command, which simplifies its application, and enforces consistency checks. However, values being transferred are dynamic, based on the contents of the selected register; the range of the data is ***not*** verified at transfer, so undesired results may be obtained if out of range parameters are assembled into the program buffer, including Sequence Error shutdowns. This command does allow for great flexibility by allowing any of the parameters to be made register based.

In QuickControl, first enter the line to be modified and place a label on the same line. Then insert the WCW command, and reference the label to select the desired command, select the register and the parameter.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
WCW	Program Class D 139 (0X8B) 3 words Thread 1&2	Data Register	S16	Standard Register Range
		Program Buffer Location	S16	Valid NV Memory Range

Example

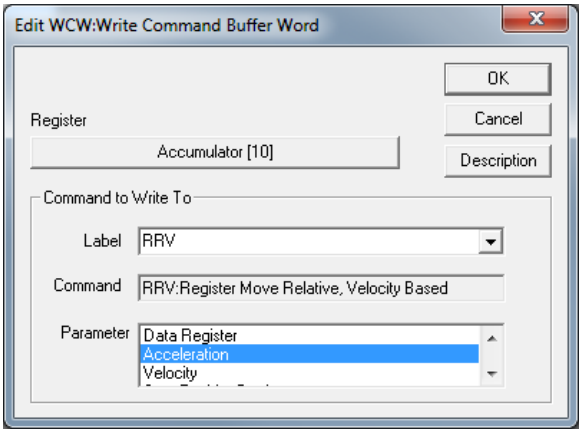
Overwrites the program buffer contents at locations 122 with the contents of the lower word of Register #10.

@16 138 10 122 (CR)

Response

ACK only

QuickControl Example



WRF:Write Register File

Description

WRF has the same command number as Write Register, Program Mode (WRP). WRF allows QuickControl to provide properties of Register Files and Register File Arrays.

See Application Note QCI-AN048 Register Files on our website for details.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
WRF	Program Class D 154 (0x9A) 4 words Thread 1&2	Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

Example

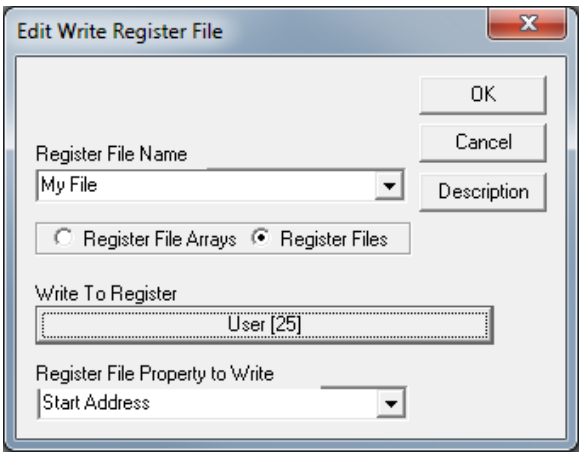
Write the number “1” to data register #10.

@16 154 10 1 (CR)

Response

ACK only

QuickControl Example



WRP:Write Register, Program Mode

See also: WRF:Write Register File

Description

The Write Register command writes the included data into the selected 32-bit Data Register. This command is similar to Write Register, Immediate Mode except it is designed to be embedded in a program and cannot be used through the serial interface while a command or program is being executed.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
WRP	Program Class D 154 (0x9A) 4 words Thread 1&2	Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

Example

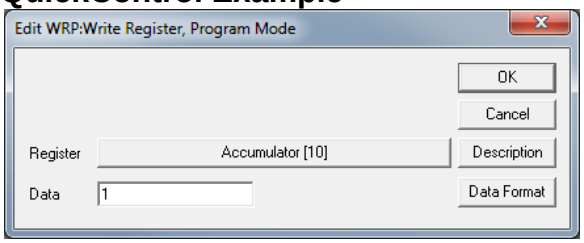
Write the number “1” to data register #10.

@16 154 10 1 (CR)

Response

ACK only

QuickControl Example



CIS:Clear Internal Status

Description

The Internal Status Word (ISW) is used to indicate different conditions or states in the device (see Internal Status Word (ISW) in User Manual for details). Several of the conditions are "latched" and therefore are persistent even after the condition has changed. The CIS command is used to clear the latched conditions in the ISW.

This command should be used after a Kill Motor condition has occurred before normal operation can be restored.

SD08: This command also clears the latched bits in the Internal Status Word 2 (IS2). (See User Manual for more details.)

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CIS	Program Class D 163 (0xA3) 1 word Thread 1&2	NONE	NONE	NONE

Example

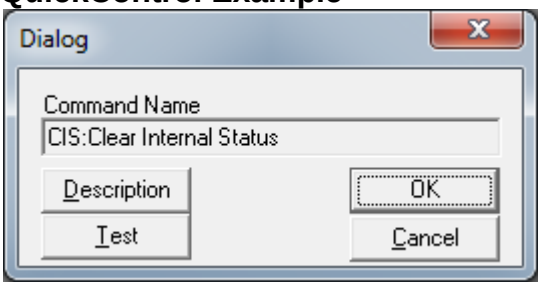
Clear the Internal Status Word.

@16 163 (CR)

Response

ACK only

QuickControl Example



TTP:Target To Position

See Also: ZTG:Zero Target, ZTP:Zero Targe and Position

Description

This command copies the current Position value into the Target register. This is useful for removing errors when an obstruction is encountered without losing track of position. This allows the next motion to move and ramp as expected rather than having to unwind the accumulated error. This is useful for homing against a hard stop where error is intentionally introduced, and for removing error before enabling the motor drivers after they have been disabled.

The Target value is updated by the Trajectory Generator, the Step & Direction mode or one of the Input Modes. The servo loop uses the Target value as the input position parameter. If the motor is unable to achieve the Target position windup will occur. This command removes the windup error.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
TTP	Program Class D 146 (0x92) 1 word Thread 1	NONE	NONE	NONE

Example

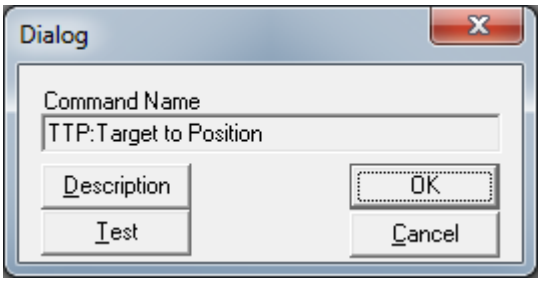
Sets the Target to the current position.

@16 146 (CR)

Response

ACK only

QuickControl Example



ZTG:Zero Target

See Also: TTP:Target to Postion,, ZTP:Zero Target and Position

Description

This command subtracts the current Target position from both the Target and the position register, resulting in a zero Target value, with no change in the actual position or servo action of the system. This is useful for homing routines to denote the current target location as “Zero” so that all other locations can be defined as an offset from “Zero”.

This command does not remove any windup, whatever Position Error exists before this command will remain. To zero the Target and clear the Position Error use the Zero Target & Position command.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ZTG	Program Class D 144 (0x90) 1 word Thread 1	NONE	NONE	NONE

Example

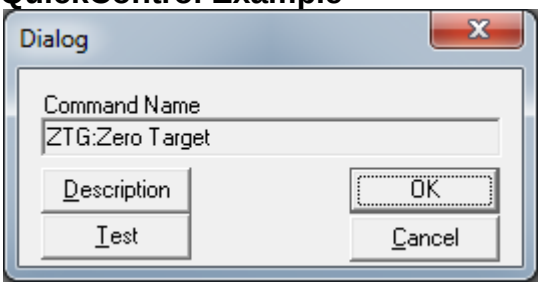
Sets the Target to zero (“0”) does not clear the position error.

@16 144 (CR

Response

ACK only

QuickControl Example



ZTP:Zero Target and Position

See Also: TTP:Target to Postion,, ZTG:Zero Target

Description

This command zeros both the Target register and the Position register. This removes any Position Error that may exist. This is useful for homing routines to denote the current location as “Zero” so that all other locations can be defined as an offset from “Zero”.

This command removes any Windup that may exist from a previous motion.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
ZTP	Program Class D 145 (0x91) 1 word Thread 1	NONE	NONE	NONE

Example

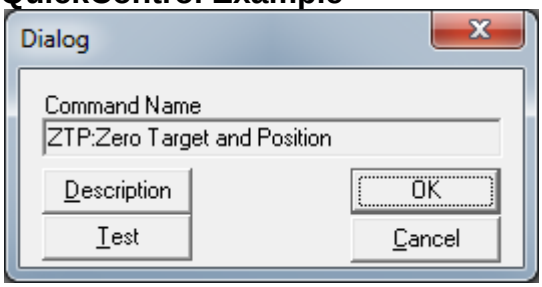
Sets the Target & Position to zero (“0”).

@16 145 (CR)

Response

ACK only

QuickControl Example



CAN Commands

Please see the CAN User Manual for more information on CAN, CANopen, and initialization and use of CAN. These are available on all except D2-MG (unless D2-MG CAN option is ordered).

CAN Baud Rate (CBD)

Description

The CAN Baud Rate command sets the CAN baud rate from the standard list of CANopen Baud Rates (see below). The power-on default CAN baud rate is 1Mb/Sec. Note that Baud rate 5 is “Reserved” in the CANopen implementation; a 100 kb/sec rate is included for compatibility with other CAN systems which use that baud rate.

BAUD Rotary Switch

One controllers with a BAUD Rotary Switch, setting CBD to 255 ("Rotary Switch" in QuickControl) will cause the CAN Baud Rate to be read from the BAUD Rotary Switch. This allows a user to change the CAN Baud Rate without re-programming. The BAUD Rotary Switch information is available on bits 4-7 of CAN object 200Ah.

Command Info

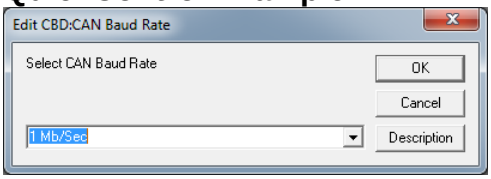
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
CAN Baud Rate (CBD)	Program Class D Code (Hex): 71 (0x47) 2 words	Baud Rate	U16	0 = 1 Mb/Sec 1 = 800 kb/sec 2 = 500 kb/sec 3 = 250 kb/sec 4 = 125 kb/sec 5 = 100 kb/sec *Reserved 6 = 50 kb/sec 7 = 20 kb/sec 8 = 10 kb/sec 255 = Get Baud from BAUD Rotary Switch 0-8: Same as above 9-F: 1Mb/Sec.

Example

Configure baud rate to 1 Mb/sec.

@16 71 0 (CR)

QuickControl Example



Response

ACK only

CAN Connect to Remote (CCTR)

Description

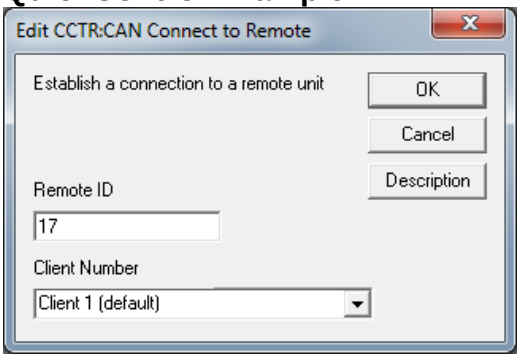
This command configures the CAN SDO client 1 or client 2 on the local unit to communicate with the default SDO server on the selected unit. This configuration is required to select the remote node prior to using the **CAN Dictionary Access Remote (CDR)** command.

The CCTR command sets the appropriate parameters in object 1280h (Client 1) or 1281h (Client 2) in the local Data Dictionary. CCTR is a Combo Command, internally consisting of three **CAN Dictionary Access, Local (CDL)** commands.

Command Info

Command Name	Command Type/Num	Parameters	Parameter Range
CAN Connect to Remote (CCTR)	Program Class COMBO D Code 18 words	Remote ID	Select the CAN ID with which to establish communications 1 to 127 (1 to 7fh)
		Client Number	Select which local client to use for communications

QuickControl Example



CAN Dictionary Access, Local (CDL)

See Also CAN Dictionary Access, Remote (CDR)

Description

The CAN Dictionary Access Command provides read/write access to the local CAN Data Dictionary Objects. Read access copies the Data Dictionary Object value to a User Register. Write Access copies the value from a Register to the selected Data Directory Object, or alternately, from a Constant to the selected Data Dictionary Object.

The Data Dictionary contains all objects accessible from CAN; some of these must be configured by a controller serving as a Master prior to accessing CAN. The Data Dictionary is accessed via a 16-bit Index and an 8-bit Sub-Index.

See CANopen User manual for a detailed listing and explanation of supported Data Dictionary Objects.

Note: An invalid access will generate a Command Error and halt the program

Note: Time of Day Objects require two Registers.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
CAN Dictionary Access, Local (CDL)	Program Class D 72 (0x48) 6 words	Mode	U16	0 = Read (Dictionary => Register) 1 = Write (Register => Dictionary) 2 = Write (Constant => Dictionary)
		Data Register or Constant	S32	Holds either Register number or Data: Register Number (Modes 0 or 1) 32-bit Constant (Mode 2)
		Index	U16	0 to FFFFh (0 to 65536)
		Sub-Index	U16	0 to FFh (0 to 255)

Example

Set Communications Cycle Time (Sync period) to 1000 microseconds. Data Dictionary 1006h Sub-Index 00h

@16 72 2 1000 0x1006 0x00 (CR)
or
@16 72 2 1000 4102 0 (CR)

Response

ACK only

Second Example

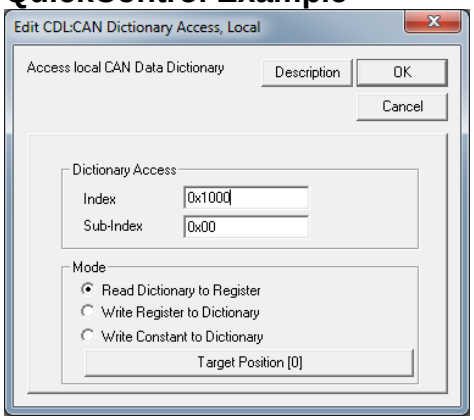
Read Heartbeat Status of first Heartbeat Consumer (Data Dictionary 2005h Sub-Index 01h) into User Register 30

@16 72 0 30 0x2005 0x1(CR)
or
@16 72 0 30 8197 1(CR)

Response

ACK only

QuickControl Example



CAN Dictionary Access, Remote (CDR)

See Also CAN Dictionary Access, Local (CDL)

Description

The CDR command provides a means to access another Node's Data Dictionary. This command will not operate properly until the SDO (Service Data Object) Client Communication objects in the local Data Dictionary have been initialized to specify which node is to be contacted (see Combo Cmd CCTR). Unless the node has been previously configured (either locally by its internal user program) or by a Master node, the initial communications must be configured to use the remote Node's default SDO server COB-ID's. These must be configured using the CAN Dictionary Access, Remote (CDR) command to set the local SDO Client Communications via Object 1280h (Client 1) or 1281h (Client2). These do not need to be reconfigured as long as the local Node is communicating with the same remote Node, but do need to be reconfigured to communicate (with the same client) to a different Node. With two SDO clients available, a Master Node may communicate to up to two slave/peer nodes without reconfiguring the SDO client communications each time.

Mode: Specifies the type of action requested for this SDO communication. Mode 0 Performs an upload (Read remote) to Register, Mode 1 performs a download (write remote) from register, Mode 2 performs a download (write remote) from constant. See the Errors section below for more details. QuickControl determines this automatically when using the Remote Output tab.

Data Register or Constant: For Mode types 0 and 1, this parameter specifies the starting register to use, with additional data taken from/delivered to subsequent registers, as needed. For Mode type 2 (Download/write Constant), the second parameter is a 32 bit constant. If the transfer is greater than four bytes, additional registers will be used, in an ascending order. Strings will be transferred to Registers low byte of the lowest numbered Register, up to the high byte, to the next register low byte, and so on. Mode 2, Constant style downloads, are limited to no more than four bytes. QuickControl determines this automatically when using the Remote Output tab.

Index and Sub Index: These parameters specify the entry in the remote Data Dictionary being accessed. Note that these addresses are normally specified in Hexadecimal, and many of the values written are specified in Hexadecimal for consistency with CANopen convention. QuickControl determines this automatically when using the Remote Output and Remote Register Access tabs.

Byte Count: In the case of a download (write) action, this is the number of bytes available for transfer, which may exceed those required by the object accessed in the remote Node. (Data is sent low byte first; an 8-bit transfer from a 32-bit source will only transfer the lowest byte, even if four bytes were specified as being available.) In the case of an upload (read) action, the byte count specifies the maximum number of bytes to transfer from the remote node so as not to exceed the local register space reserved for the transfer. For single register transfers - upload or download - this parameter may be set to 4. When uploading strings, setting bit 15 in addition to the number of bytes will allow up to the number of bytes to be transferred without an error if more byte of data are available (i.e. only read up to first x bytes of the string). QuickControl determines this automatically when using the Remote Output and Remote Register Access tabs.

Timeout: This parameter specifies the number of cycles to wait for the remote node to complete the SDO transfer before the local node times out. This is needed to prevent the user program from hanging on a remote node not present, excessive bus usage, etc. This value is dependent upon the bus loading as well as the baud rate and the size of the transfer. At 1Mb/sec, a value of 40 (4.8 ms) should normally be

sufficient if the bus is not overly loaded, and the transfer is up to four bytes. You may need to experiment to determine the setting for your configuration. In QuickControl, this is edited using the Advanced button.

Client: This parameter specifies which local SDO client to use. The use of more than one client allows access to more than one node without reconfiguring the SDO client communication parameters. Each Node has at least one SDO server (the SilverLode CANopen code provides two) to service SDO client requests. Each server/client connection is a one-to-one mapping, that is, each client may only access one server and each server may only service one client. In QuickControl, this is edited using the Advanced button.

Error Bits

A time-out may occur with this command if the remote node does not respond within the specified time period. If a timeout occurs, the command will terminate, but the ISW “Positive” condition bit will be set (testable with the Jump command testing for Positive) while the “Zero” and “Negative” bits will be cleared. This allows the user code to determine that the command timed out. An error may also occur if attempting to write to a read-only variable, or attempting to access an object that does not exist. If this type of error occurs, the command will terminate, but the ISW “Negative” condition bit will be set, while the “Positive” and “Zero” bits will be cleared.

If the command terminates normally, the ISW “Zero” bit is set, and the “Positive” and “Negative” bits are cleared. A jump on Positive or Negative to an error recovery routine after each CDR command should be used if the data sent or received is critical.

Accumulating Error Bits

Setting bit 2 in the Mode word (i.e. actions 4,5,6) does not alter the action, but allows accumulation of the returned ISW "error" bits to allow a single test at the end of a series of CDR commands (as long as no other register type commands – such as a calculation command – have been executed). To implement this, the first CDR command would not have the accumulate bit (bit2) set in the Mode word, so as to clear out any prior settings of the ISW Zero, Negative, and Positive bits, replacing them with the results of the first CDR command. The rest of the CDR commands would have the accumulate bit set. At the end of a series of CDR commands, a Jump on Negative would detect any disallowed operations (such as attempting to write to a read-only object), and a Jump on Positive would indicate if there were any timeouts (as would be caused by a busy bus, an improperly wired bus, or a remote module not powered up/initialized for CAN). If neither of these bits were set, then all of the series of CDR commands succeeded. In QuickControl, this is edited using the Advanced button.

See Service Data Objects (SDO) section in the CAN Data Dictionary.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
CAN Dictionary Access, Remote (CDR)	Program Class D 80 (0x50) 9 words	Mode	U16	Dictionary refers to Remote Dictionary 0 = Read (Dictionary => Register) 1 = Write (Register => Dictionary) 2 = Write (Constant => Dictionary) Setting Bit 2 (i.e. 4,5,6) does same function, but accumulates status bits.
		Data Register or Constant	S32	Register (Actions 0 or 1) 32 bit constant (Mode 2)
		Index	U16	Remote Node Index
		Sub-Index	U16	Remote Node Sub-Index
		Byte Count	U16	Number of Bytes to transfer
		Timeout	U16	Number of 120uS ticks before Timeout occurs.
		Client	U16	Local SDO client to use, 1 or 2

Example:

Upload remote unit Actual Position (Register 1) via Data Dictionary Object Index 2101h Sub-Index 00h, 4.8ms timeout using Client 1. Results are written to Register 30.

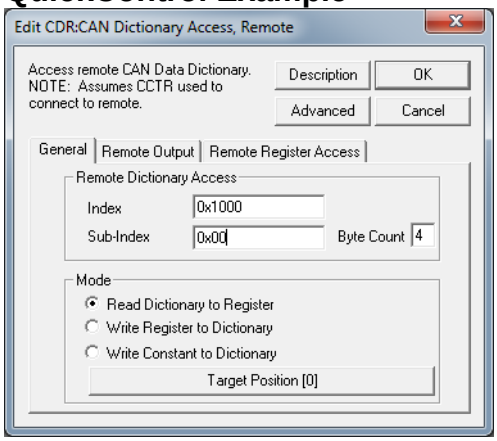
@16 80 0 30 0x2101 0 4 40 1 (CR)

or
@16 80 0 30 8449 0 4 40 1(CR)

Response

ACK only

QuickControl Example



CAN Identity (CID)

Description

CAN Identity for the first time sets the CANopen CAN ID and starts up CAN frame processing; it also results in an initialization of the COB-ID's (including those previously configured). This command cannot be processed (will produce a command error) if the
CAN NMT state is "Operational" or "Stopped"; it will only work in "Pre-Operational" or prior to configuring CAN (CAN initialization state).

Setting the CAN ID after it has been previously set only changes the current CAN CAN ID; to force a re-initialization of the COB-ID's, it is necessary to negate the ID value (i.e. –1 to –127).

The CAN CAN ID may be set explicitly (1 to 127) or it may be set to the lower 7 bits Node's Serial ID by setting the ID to zero (0). If using the lower 7 bits of the Serial ID, do not use Serial ID 128 as this would result in an invalid CAN CAN ID of 0, which is reserved for broadcast (will produce a command error). The user must assure the resulting CAN ID values are unique within a system as duplicate CAN ID values will cause communications errors.

Command Info

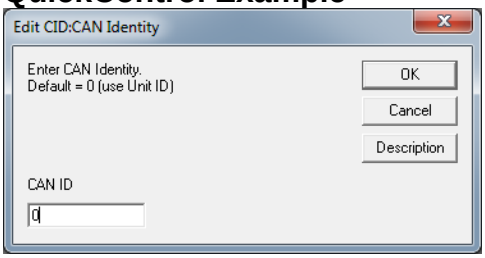
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
CAN Identity (CID)	Program Class D 73 (0x49) 2 words	CAN CAN ID	S16	0 (use lower 7 bits of Serial ID) 1-127 (Set and initialize) -1 to –127 (Set)

Example

Set CAN CAN ID to 0

@16 73 0 (CR)

QuickControl Example



Response

ACK only

CAN Set NMT State, Local (CNL)

See Also CAN Set NMT State, Remote (CNR)

Description

Transitions the local NMT (Network Management) State. Used by Peer or Master mode Nodes to change between NMT states “Pre-Operational”, “Operational”, “Stopped”, and to Re-initialize Communications parameters.

The NMT State of each node determines what types of CAN communications are allowed to take place. Some Data Dictionary Objects may only be written while in the Pre-Operational State (see CAN Data Dictionary).

See CNR for state definitions.

Note: The Transition Request Value is limited to the four documented values. The NMT State is transitioned to the requested state, but the Transition Value **does not** correspond to the resulting NMT State. See table

See **Network Management (NMT) Objects** in Chapter 3 for more details.

Command Info

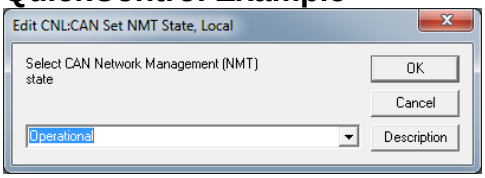
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
CAN Set NMT State, Local (CNL)	Program Class D 74(0x4A) 2 words	NMT State Transition Request	U16	1 = Transition to Operational 2 = Transition to Stopped 128 = Go Pre-Operational 130 = Reset Communications Parameters (when done transition to Pre-Operational) (NMT State 127)

Example

Transition to NMT State “Operational”

@16 74 1 (CR)

QuickControl Example



Response

ACK only

CAN Set NMT State, Remote (CNR)

See Also CAN Set NMT State, Local (CNL)

Description

Transitions the Network Management (NMT) State for one or all other nodes on the CANopen Network. Used by the Master Node to transition other nodes to the desired NMT state “Pre-Operational”, “Operational”, “Stopped”, and to Re-initialize Communications parameters of the target node(s) or to Reset the target node(s).

The NMT State of each node determines what types of CAN communications are allowed take place. Some Data Dictionary Objects may only be written while in the Pre-Operational State (see CAN Data Dictionary).

Pre-Operational (NMT State 127)

Access to NMT State dependent Data Dictionary Objects allowed. NMT communications allowed. PDO communications not allowed. SDO communications allowed.

Operational (NMT State 5)

All communication modes allowed

Stopped (NMT State 4)

Only NMT communications allowed.

Reset Communications

Sets Communications parameters (COB-ID’s) back to their default value, and then transitions to “Pre-Operational”

Reset Node

Forces a full hardware reset of the selected node(s), the nodes should return to Pre-Operational when done.

NOTE: Resetting a Node causes the Node to temporarily revert to RS-232 mode, single drop until the initialization has reached a certain point. This may cause the serial communications to temporarily “drop out”. This is normal.

The NMT management commands structures in CANopen require that only one Master Node **produce** Commands and zero or more Nodes **consume** them. There is no direct handshake mechanism, however, if the consumer node has its heartbeat configured, then the heartbeat will reflect the new state on its next transmission.

If the transmit buffer is free to transmit (no other pending transmission from a CNL command), this command returns only the Zero flag set. If the prior transmission has not yet successfully completed, the command terminates with only the Negative flag

set. The Jump conditional command may be used to retry the command or to enter an error recovery routine.

Note: The Transition Request Value is limited to the five documented values. The NMT State is transitioned to the requested state, but the Transition Value **does not** correspond to the resulting NMT State. See table below.

See **Network Management (NMT) Objects** in the CAN Data Dictionary Document for more details.

Command Info

Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
CAN Set NMT State, Remote (CNR)	Program Class D 81(0x51) 3 words	CAN ID	U16	0 = Broadcast (All Nodes) 1-127 = Nodes 1 to 127
		NMT State Transition Request	U16	1 = transition to Operational (NMT State 5) 2 = transition to Stopped (NMT State 4) 128 = Go Pre-Operational (NMT State 127) 129 = Reset Node (when done transition to Pre-Operational) (NMT State 127) 130 = Reset Communications Parameters (when done transition to Pre-Operational) (NMT State 127)

Example

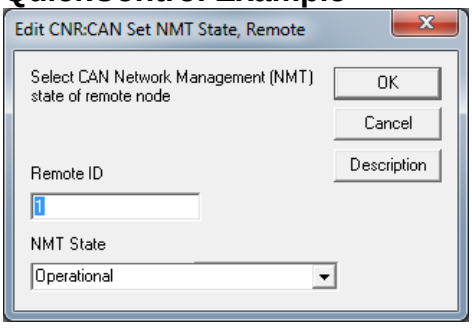
Transition Node 16 to NMT state “Operational”.

@16 81 16 1 (CR)

Response

ACK only

QuickControl Example



CAN Register Map, Local (CRML)

Description

This combo-command configures CAN to receive the selected Producer Data Object (PDO) into a user register (or registers). Multiple nodes may be configured to simultaneously consume the PDO data produced by any remote node.

PDO data streams may be used to dynamically share a register contents from a producer (sending node) to zero or more consumers (receiving nodes).

The PDO identifier is selected by choosing the Node number and transmit channel of the PDO producer. The local receive channel merely selects which local resource is used to receive the data; any receive channel not already in use may be used. The data is deposited into the selected register whenever it is received.

The default configuration maps the receive PDO data onto a single register, configured for Asynchronous (immediate) update.

The Advanced button allows selecting Rx Type (Asynchronous or Synchronous), as well the editing the register mapping. Synchronous Rx Type holds the received data until the next SYNC (Synchronization) frame is sent, allowing all nodes to simultaneously update their data from multiple data sources (as well as sampling the new data to be sent synchronized to the SYNC frame, if the Transmit PDO data set to type synchronous).

The Edit Register Mapping button on allows for finer mapping of the PDO data. The incoming data may be directed to up to two destinations. The destinations may be long words, word, 24 bit data, or 8-bit bytes. The data may be written, used to set bits (OR function) or used to clear bits (AND with NOT of data) in the designated registers.

Additionally, under the advanced button on the Edit Can Register Mapping panel, the data may be also be manually mapped to CAN Directory Objects by specifying the desired index, subindex, and number of bits to be mapped to the selected object (number of bits must correspond to size of object). The Mode pull down box must be set to manual to manually map this data.

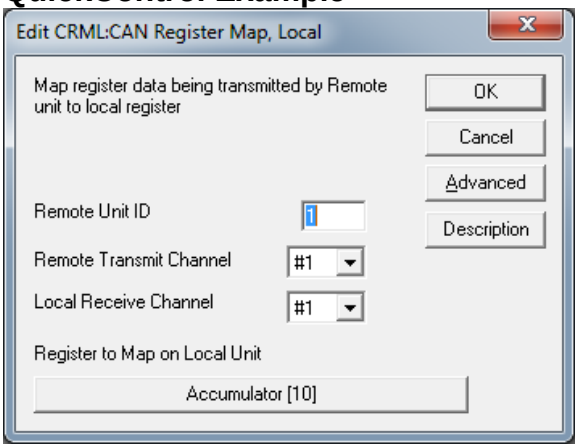
The CRMLcommand sets the appropriate parameters in objects 1400h and 1600h (Rx channel 1), 1401h and 1601h (Rx Channel 2), 1402h and 1602h (Rx channel 3), or 1403h and 1603h (Rx Channel 4) in the local Data Dictionary. CRML is a Combo Command, internally consisting of seven **CAN Dictionary Access, Local (CDL)** commands.

Note: The Receive and Transmit PDO objects may have up to 4 objects and up to 64 bits mapped to them if configured manually. The CRML combo command is limited to the more common configurations, allowing up to 2 objects to be mapped. See Data Dictionary for information on manual mapping.

Command Info

Command Name	Command Type/Num	*Parameters*	Parameter Range
CAN Register Map Local (CRML)	Program Class COMBO D Code 42 words	Remote Unit ID	Select the node ID of the remote unit producing the data. 1 to 127 (1 to 7fh)
		Remote Tx Channel	Select the transmit channel used by the remote unit. 1 to 4
		Local Rx Channel	Choose the desired receive channel (does not need to match Tx channel) 1 to 4
		Register to Map	Local user register to be updated with the received data. Register must be writable.
		Advanced	See above notes

QuickControl Example



CAN Register Map, Remote (CRMR)

Description

This combo-command configures another node via CAN to receive the selected Producer Data Object (PDO) into its user register (or registers). Multiple nodes may be configured to consume the PDO data produced by the producer node.

This combo-command performs a function very similar to the **CAN Register Map Local (CRML)**, except that instead of configuring the local node, a remote node is being configured to receive data. The configuration is done via the CAN bus, using SDO operations.

PDO data streams may be used to dynamically share a register contents from a producer (sending node) to zero or more consumers (receiving nodes).

The unit to be configured must first be selected via the **CAN Connect to Remote (CCTR)** command. (CRMR defaults to Client 1, but may use either client via the advanced options).

The PDO identifier is selected by choosing the Node number and transmit channel of the PDO producer. The receive channel merely selects which local resource of the node being configured is used to receive the data; any receive channel not already in use may be used. The data is deposited into the selected register of the selected node whenever it is received.

The default configuration maps the receive PDO data onto a single register, configured for Asynchronous (immediate) update. CRMR uses Client 1 as its default SDO channel, and a 1 second timeout per operation. A failure of communications will cause this combo-command to repeat until successful.

The Advanced button allows selecting Rx Type (Asynchronous or Synchronous), as well the editing the register mapping. Synchronous Rx Type holds the received data until the next SYNC (Synchronization) frame is sent, allowing all nodes to simultaneously update their data from multiple data sources (as well as sampling the new data to be sent synchronized to the SYNC frame, if the Transmit PDO data set to type synchronous).

The Edit Register Mapping button on allows for finer mapping of the PDO data. The received PDO data may be directed to up to two destinations. The destinations may be long words, word, 24 bit data, or 8-bit bytes. The data may be written, used to set bits (OR function) or used to clear bits (AND with NOT of data) in the designated registers.

The Edit SDO Communications Parameters button under the advanced tab allows selection of either Client 1 or Client 2 operation, as well as setting the SDO timeout.

Additionally, under the advanced button on the Edit Can Register Mapping panel, the data may be also be manually mapped to CAN Directory Objects by specifying the desired index, subindex, and number of bits to be mapped to the selected object (number of bits must correspond to size of object). The Mode pull down box must be set to manual to manually map this data.

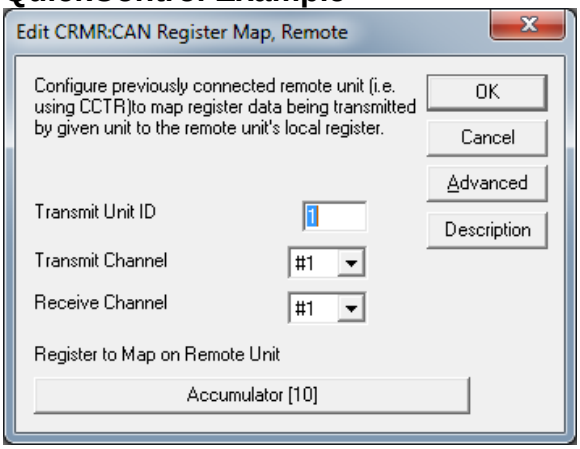
The CRMR command sets the appropriate parameters in objects 1400h and 1600h (Rx channel 1), 1401h and 1601h (Rx Channel 2), 1402h and 1602h (Rx channel 3), or 1403h and 1603h (Rx Channel 4) in the selected node's Data Dictionary. CRMR is a Combo Command, internally consisting of seven **CAN Dictionary Access, Local (CDL)** commands and one **Jump (JMP)** command.

Note: The Receive and Transmit PDO objects may have up to 4 objects and up to 64 bits mapped to them if configured manually. The CRML combo command is limited to the more common configurations, allowing up to 2 objects to be mapped. See Data Dictionary for information on manual mapping.

Command Info

Command Name	**Command Type/Num**	*Parameters*	**Parameter Range**
CAN Register Map Local (CRML)	Program Class COMBO D Code 46 words	Remote Unit ID	Select the node ID of the remote unit producing the data. 1 to 127 (1 to 7fh)
		Remote Tx Channel	Select the transmit channel used by the remote unit. 1 to 4
		Local Rx Channel	Choose the desired receive channel (does not need to match Tx channel) 1 to 4
		Register to Map	Local user register to be updated with the received data. Register must be writable.
		Advanced	See above notes

QuickControl Example



CAN Transmit Register, Local (CTRL)

Description

This command configures the selected local Transmit PDO to broadcast the selected register data.

The register to be transmitted is selected, as well as the PDO transmit channel to be configured.

The default configuration selects 32 bits from the given register, has a Transmission Type of 255: Asynchronous, set to transmit:

1) when the unit first goes into operation state (or the Transmit PDO is configured if dynamically configured (already in operational mode).
2) whenever the data changes
3) At least every 200 milliseconds (so that the state is refreshed)
4) But not more than every 2 milliseconds (so constantly changing data will not overload the bus.

Via the Advanced tab, the transmit type may be selected to be Synchronous (0 through 240 SYNC cycles, with 0 indicating to send synchronously only on change. The Inhibit time determines how fast back to back transmissions may occur. If using synchronous mode, the inhibit time may be set to 0. The event timer determines minimum frequency of transmission. In synchronous mode, the transmission will still be delayed until the next SYNC signal. Normally, the event timer is set to 0 (disabled) in synchronous mode.
The starting SYNC # is used to delay the given number of sync cycles before transmitting. This may be further enhanced by configuring Data Dictionary Object 1019h for the least common factor of the various Synchronous transmission times. See Object 1019h and Synchronous Communications sections for more details.

The transmit channel combined with the node number determine the priority of the data frame. The lower 7 bits of the frame address (by default) are the transmitting CAN ID, while the upper 5 bits grow in value as the transmit channel is increased. The frame identifier for Tx channel 1 is 180h + CAN ID, Tx channel 2 is 280h + CAN ID, Tx channel 3 is 380h + CAN ID, Tx channel 4 is 480h + CAN ID. The frame with the lowest identifier has the highest transmission priority over the CAN bus.

Internally, the **CCTR** combo-command consists of 11 **CDL** commands and one **CLD** command which configure data dictionary objects 1800h and 1A00h for Tx channel 1, objects 1801h and 1A01h for Tx channel 2, objects 1802h and 1A02h for Tx channel 3, or objects 1803h and 1A03h for Tx channel 4.

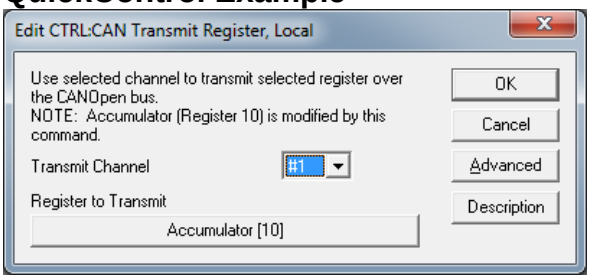
Note: transmissions will not begin until the transmitting unit is in NMT state Operational. Units configured to receive data will not react to PDO data until they are set to NMT-state Operational. If synchronous mode is configured, one of the nodes must be configured to produce a SYNC signal by configuring objects 0x1005 (bit 30 must be set on producer), and 1006h (sync time in microseconds).

Command Info

Command Name	Command Type/Num	Parameters	Parameter Range

CAN Transmit Register Local (CTRL)	Program Class COMBO D Code 72 words	Tx Channel	1 to 4 lowest numbered channel has highest priority for node. For the same channel, lowest numbered node has highest priority.
		Data Register	Selects the Data register to transmit
		Advanced options: 2nd transmit channel, Transmit type, inhibit time, event timer, starting sync.	See description above for details

QuickControl Example



CAN Transmit Register, Remote (CTRR)

Description

The **CAN Transmit Register, Remote (CTRR)** combo–command is used to configure a remote node to transmit data via a PDO object.

This combo-command performs a function very similar to the **CAN Register Map Local (CRML)**, except that instead of configuring the local node, a remote node is being configured to transmit data. The configuration is done via the CAN bus, using SDO operations.

PDO data streams may be used to dynamically share a register contents from a producer (sending node) to zero or more consumers (receiving nodes).

The unit to be configured must first be selected via the **CAN Connect to Remote (CCTR)** command. (CTRR defaults to Client 1, but may use either client via the advanced options).

The register to be transmitted is selected via the pull down menu, as well as the PDO transmit channel to be configured.

The default configuration selects 32 bits from the given register, has a Transmission Type of 255: Asynchronous, set to transmit:

1) when the unit first goes into operation state (or the Transmit PDO is configured if dynamically configured (already in operational mode).
2) whenever the data changes
3) At least every 200 milliseconds (so that the state is refreshed)
4) But not more than every 2 milliseconds (so constantly changing data will not overload the bus.

Via the Advanced tab, the transmit type may be selected to be Synchronous (0 through 240 SYNC cycles, with 0 indicating to send synchronously only on change. The Inhibit time determines how fast back to back transmissions may occur. If using synchronous mode, the inhibit time may be set to 0. The event timer determines minimum frequency of transmission. In synchronous mode, the transmission will still be delayed until the next SYNC signal. Normally, the event timer is set to 0 (disabled) in synchronous mode.
The starting SYNC # is used to delay the given number of sync cycles before transmitting. This may be further enhanced by configuring Data Dictionary Object 1019h for the least common factor of the various Synchronous transmission times. See 1019h and Synchronous Communications sections for more details.

Under the SDO Client Parameters tab of the Advanced panel, the SDO Client number may be selected, as well as the SDO timeout period (for each CDR command).

The transmit channel combined with the node number determine the priority of the data frame. The lower 7 bits of the frame address (by default) are the transmitting CAN ID, while the upper 5 bits grow in value as the transmit channel is increased. The frame identifier for Tx channel 1 is 180h + CAN ID, Tx channel 2 is 280h + CAN ID, Tx channel 3 is 380h + CAN ID, Tx channel 4 is 480h + CAN ID. The frame with the lowest identifier has the highest transmission priority over the CAN bus.

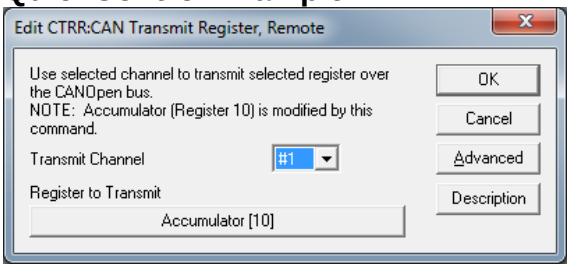
Internally, the **CCTR** combo-command consists of 11 **CDR** commands, one **CLD** command, and one **JMP** command which configure data dictionary objects 1800h and 1A00h for Tx channel 1, objects 1801h and 1A01h for Tx channel 2, objects 1802h and 1A02h for Tx channel 3, or objects 1803h and 1A03h for Tx channel 4. The Jump command repeats the sequence in the case of errors or timeouts. The jump may be manually modified to vector to an error recovery routine.

Note: transmissions will not begin until the transmitting unit is in NMT state Operational. Units configured to receive data will not react to PDO data until they are set to NMT-state Operational. If synchronous mode is configured, one of the nodes must be configured to produce a SYNC signal by configuring objects 1005h (bit 30 must be set on producer), and 1006h (sync time in microseconds).

Command Info

Command Name	Command Type/Num	Parameters	Parameter Range
CAN Transmit Register, Remote (CTRR))	Program Class COMBO D Code 109 words	Tx Channel	1 to 4 lowest numbered channel has highest priority for node. For the same channel, lowest numbered node has highest priority.
		Data Register	Selects the Data register to transmit
		Advanced options: 2nd transmit channel, Transmit type, inhibit time, event timer, starting sync., SDO client, timeout.	See description above for details

QuickControl Example



Appendix A: Registers

Data registers can be dedicated to a specific purpose, optionally dedicated or continuously available for user data. They can be designated as Read Only or Read & Write. Data registers are 32 bits in length (long word) and numbered from 0 to 255 (Not all registers are implemented – varies by product and code revision). Many are designed to operate as two independent 16 bit data registers with each 16 bit word containing discrete data. Data is refreshed internally every servo cycle (120 microseconds). It can be sent or retrieved serially through a host controller or used internally by programs downloaded into the device.

Dedicated Data Registers (0-10)

This type of data register is dedicated to a specific the device function. the device uses this data extensively for many internal operations. Some data registers contain factory specific data that directly affects the servomotor operation. Modifications to this type of data may cause the servo to operate unexpectedly.

The table below provides information on dedicated data registers 0 through 10. These specific registers are used frequently when programming and operating the device.

Data Register	Type	Dedicated Data Register Description High word = (HW) <¦> Low word = (LW)
0	R/W*	Target Position; calculated position data from trajectory generator
1	R/W	Actual Position; current internal encoder position count
2	R/W	Last Index Position; encoder position count of the last internal index trigger
3 ‡	R	Internal Status Word (ISW) (HW) <¦> Reserved (LW)
4	R/W	Last Trigger Position; encoder position count when last stop condition was satisfied.
5	R/W	Delay Counter; clock ticks for the internal delay counter (1 tick = 120 usec). This is the register used by the Delay (DLY) command.
6 ‡	R	Max Position Error (HW) <¦> Current Position Error (LW)
7 ‡	R	Velocity 1; current vel. 1st filter (HW) <¦> Velocity 2; current vel. 2nd filter (LW)
8	R	Reserved – Integrator value
9 ‡	R	Reserved; (HW) <¦> Torque; current torque value (LW)
10	R/W	Accumulator; calc. results, reg. copy/save buffer, indirect addressing pointer

‡ Data register contains two independent 16 bit data words.

R = Read Only: R/W = Read and Write
R/W* = Read and Write (Write only using CLC command with Offset Target/Position operation)

User Data Registers and Optionally Dedicated Data Registers (11-199)

Data registers 11 through 40 (SilverDust Rev 06 11 through 199) are defined as user data registers by default. These read/write registers can be used by all the device commands that are associated with user data registers. When the device is programmed to operate in an Input Mode, registers 12 through 18 become dedicated to the Input Mode operation. When Profile Move commands are implemented, registers 20 through 24 become dedicated to the Profile Move operation. Registers 11, 19, and 25 through 40 are always available for user data.

Data Register	Type	Default Use	Optional Dedicated Command Use	Dedicated Use Description

11	R/W	User Data		
12	R/W	User Data	*Input Mode	Input Source Data
13	R/W	User Data	*Input Mode	Input Offset
14	R/W	User Data	*Input Mode	Input Dead band
15	R/W	User Data	*Input Mode	Maximum Scale/Limit
16	R/W	User Data	*Input Mode	Maximum Output Scale
17	R/W	User Data	*Input Mode	Output Offset
18	R/W	User Data	*Input Mode	Output Rate of Change Limit
19	R/W	User Data		
20	R/W	User Data	*Profile Move	Absolute Position
21	R/W	User Data	*Profile Move	Acceleration
22	R/W	User Data	*Profile Move	Velocity
23	R/W	User Data	*Profile Move	Deceleration
24	R/W	User Data	*Profile Move	Offset (pos. from input stop)
25	R/W	User Data		
thru	R/W	User Data		
40 (198)	R/W	User Data		
199	R/W	User Data	*Default mapped I/O	Jump/stop on Mapped I/O

R/W = Read and Write

Registers for Optional Dedicated Command Use:

*Input Modes - Position Input Mode (PIM), Velocity Input Mode (VIM), and Torque Input Mode (TIM).
*Profile Move Commands - Profile Move (PMV), Profile Move Continuous (PMC), Profile Move Override (PMO), and Profile Move Exit (PME).

Appendix A: Registers

Dedicated Data Registers (200+)

The table below provides information on dedicated data registers 200+. These registers are utilized for advanced operations, complex programming, troubleshooting, and factory specific settings.

Data Reg	Type	Dedicated Data Register Description High word = (HW) <¦> Low word = (LW)
200	R/W	External Encoder Position; total count value from external encoder signals
201	R/W	External Index Position; count value of last external index trigger
202		Reserved
203	R	Reserved <¦> Internal Status Word 3 (IS3)
204	R	Target Acceleration (calculated internally by trajectory generator)
205	R	Target Velocity (calculated internally by trajectory generator)
206 ‡	R/W	Closed Loop Holding Torque (HW) <¦> Closed Loop Moving Torque (LW) Register based version of TQL command.
207 ‡	R/W	Open Loop Holding Torque (HW) <¦> Open Loop Moving Torque (LW) See 206 above
208 ‡	R/W	Error Limit Moving (HW) <¦> Error Limit Holding (LW) Register based version of ERL command.
209 ‡	R	Sense Mask; shaft rotation dir for +data (HW) <¦> IO Status Word (IOS) (LW) The direction Sense Mask (HW) defines the direction of shaft rotation (clockwise or counter clockwise) in relation to positive data parameters of position and velocity.
210 ‡	R	Program Buffer Size (HW) <¦> Program Buffer Start (LW)
211 ‡	R/W	Kill Motor Conditions (HW) <¦> Kill Motor States (LW) May be used to determine the cause of a triggered Kill Motor operation. These registers are written internally whenever the Kill Motor operation is activated. They may be overwritten to zero to make conditional testing of a triggering event easier. The Kill Motor Conditions are latched. See TD052 for more details.
212 ‡	R	Analog Input 1 (HW) <¦> Analog Input 2 (LW) Contains the filtered ADC readings for Analog Input1 (HW) and Analog Input 2 (LW). Allows these inputs to be monitored from the serial port without program involvement or stopping program operation.
213 ‡	R	Analog Input 3 (HW) <¦> Analog Input 4 (LW); raw data Same as 212, but for Analog Input 3 (HW) and Analog Input 4 (LW).
214 ‡	R	Driver Voltage (HW) <¦> Processor Temp (LW); raw data Contains the filtered ADC readings for the Main V+ drive voltage (HW) and the Controller/Processor temperature (LW). Temperature data is in a raw format and requires scaling for degree C output.
215 ‡	R	Process Volt (HW) <¦> Driver Temp (LC) Contains filtered ADC readings for HC processor voltage (HW) and HC driver temperature (LW). HC processors and drivers are used in the SilverNugget N3. Note that the calibration for the processor power is different than that for the driver power. The HC driver temperature does not follow the same scaling equation as for the processor temperature. The HC driver temperature can be approximated using Temp (centigrade) = (ADC value-2230)/228. Calculation is accurate to ± 3C between 5C and 100C.
216 ‡	R	Max Driver Voltage; raw data (HW) <¦> Drive Cal (counts/volt) (LW) Maximum driver voltage in ADC counts (HW) and Drive V+ calibrate data in ADC

		counts/volt (LW). The CAI command stored in the factory memory block initializes the data in this register
217 ‡	R/F	Max Driver Temp (HW) <¦> Processor Volt Cal (counts/volt) (LW) Maximum driver temperature in ADC counts for HC series (HW) and Processor V+ Calibration for the HC series, counts/volt (LW). Data is initialized by factory block.
218 ‡	R/W	Reserved
219 ‡	R	GroupID:Unit ID (HW) <¦> Reserved (LW)
220 ‡ thru 226 ‡	R/W	DIF I/O Filter Data for all seven I/O lines. DIF I/O Line Filter Constant; 0 = no filter (HW) <¦> DIF I/O Line Count (LW). [Note: If data is positive, I/O line is considered high, if negative, low. Counter will count up with high levels, and down with low levels, jumping to +/- filter count when it crosses zero count value (hysteresis).]
227	R/F	Reserved
228 ‡	R/W	Reserved
229 ‡	R/F	STEP4 <¦> Reserved STEP4 is set by the Initialization Wizard based on the encoder's counts/rev (CPR). STEP4 = Encoder CPR/50 To calculate encoder CPR within a program (QCP) read STEP4 from the upper word of reg 229 and multiply by 50.
230 ‡	R/F	Reserved
231	R/F	Reserved
232 ‡	R	Reserved
233	R/F	Reserved

Data Reg	Type	Series	Dedicated Data Register Description High word = (HW) <¦> Low word = (LW)
234	R		Encoder CPR (HW) <|> Encoder Modulo Position (LW) CPR is the counts per revolution of internal encoder Modulo Position - rotary location: zero = index, count is modulo CPR; This is only valid after the index has been found at least one time.
235	R/F		Reserved
236	R		Internal Status Word 2 (IS2) (HW) <|> Internal Status Word 4 (IS4) (LW)
237	R/W		Reserved
238	R/W		Extended IO Word (XIO) de-bounced input (HW) <|> XIO output driver enable Debounced XIO input values <|> XIO open collector output transistors enabled Input bits 0 to 15 correspond to IO 101 to 116; 0 for low, 1 for 1.5v or higher Output bits 0 to 15 correspond to IO 101 to 116; 0 = transistor off, 1=transistor on
239	R/W		Reserved
240	R/W		Kill Motor Cond IS2 <|> XIO The Kill Motor Conditions are latched. See TD052 for more details.
241	R/W		Max Motor Temp in 1/16 °C <|> Motor Temp in 1/16 °C Available in SilverDust IG/IGB with sensor equipped motors. Max=0 disables the over temperature check
242	R/W		Reserved
243	R/W		Command Error <|> Trajectory State Stores the cause of a command error when one occurs. Save trajectory state as well in case the Command Error was as a result of changing registers used by the trajectory generator (such as the VIM command). Command Errors 1. Not enough writable registers, invalid register 2. Not able to perform requested motion in requested time 3. No motion was pre-calculated (or since cleared) 4. Command prohibited in current state 5. Invalid Program Buffer location 6. Unable to perform action 7. Invalid Selection: Mode, sub-command, I/O bit 8. Parameter out of range 9. Internal Error - bad calibration data 10. Invalid command number fetched from buffer 11. Stack Space Error - excess calls or returns 12. Bad EEPROM Access Trajectory States: 0x81 Not enough writable registers, invalid register 0x84 Command prohibited in current state

			• 0x88 Parameter out of range
244	R/W		Count Up Timer (ms) 32 bit free running up counting millisecond timer. Increments once per millisecond from power application. Automatically rolls over. Value is user writable.
245	R/W		Count Down Timer (ms) 32 bit down counting millisecond timer. Sets a flag bit is IS2 when it reaches 0. Stops counting when it reaches zero. Value is user writable.
246‡	R		CAN_ERR_REG|CAN_STATE CAN_ERR_REG same as CAN object 1001h Lower 8 bits of CAN_STATE are CAN NMT state, upper bits reserved. See CANopen User Manual
247‡	R/C		CANESR|CANGSR 2406 hardware registers 7106h:7107h – (see CANopen Manual for details). Hardware status registers for the CAN subsystem. They are Read/Clear or Read Only (see below). Read/Clear indicates that writing a 1 to the designated bit clears the bit. CANESR = CAN Error Status Register 1 = indicated error has occurred Bit 8 = Form Error Flag (RC) Bit 7 = Bit Error Flag (RC) Bit 6 = Stuck at Dominant (RC) Bit 5 = CRC Error (RC) Bit 4 = Stuff Error (RC) Bit 3 = Acknowledge Error (RC) Bit 2 = Bus-Off Status (0=normal operation) (RO) Bit 1 = Error Passive mode (0=normal) (RO) Bit 0 = Warning Status (1=at least one error counter reached 96) (RO) RC = read, write a 1 to clear, RO = Read only, writes to bit are ignored CANGSR = CAN Global Status Register Bit 5 = SMA = Suspend Mode Acknowledge (0=normal) Bit 4 = CCE = Change Configuration Enable (0=normal) Bit 3 = PDA = Power Down Mode Ack. (0=normal) Bit 2 = Reserved Bit 1 = RM = Receive mode = CAN module is receiving a frame Bit 0 = TM = Transmit mode = CAN module is transmitting a frame.
248	R/W		Thread 2 Accumulator Thread 2 local copy of Register 10 (Allows access to Thread 2 register 10 via Serial/CAN
249‡	R/W		Cmd Err LOADADD | Reserved Copy of EEPROM load address prior to Command Error Recovery command (so Command Error Recovery routine knows the original Command Error)

250‡	R/W		Thread 2 LOADADD | Reserved Thread 2 EEPROM load address.
251‡	R		Thread 2 Program Buffer Start | Size Start of Program Buffer for Thread 2 Size of Program Buffer for Thread 2
252‡	R		Reserved
253	R	D2-IG8	SSI position – only available on D2-IG8 with SSI option
254	R	D2-IG8	CAN switches – only available on D2-IG8

‡ Data register contains two independent 16 bit data words.

R = Read Only: R/W = Read and Write; R/F = Read/Factory Writable (SilverDust Rev 06)

Note: Use caution when writing to 200 level data registers as some retain factory specific data. Changing the data in specific registers may cause operation problems with the device. Some registers labeled R/F may be user Read Only; these will eventually be set to user Read Only.

The following registers do not work with the CLC and CTW commands.

64000	R/F		Microsecond counter – updated every 40uS
64001‡	R		KV1|KV2
64002‡	R		KVF|KA1
64003‡	R		KA2|KAF
64004‡	R		KP|KI
64005‡	R		KSI|KOD
64006‡	R		KAO|VBACK
64007‡	R		AF1|AF2
64008‡	R		STATUS|CMD_ST ** CMD_ST reflect the thread from which this is accessed. These should only be accessed from the serial or CAN ports if only one thread is active. Status Bits: Bit 0 = aborted packet - data errors or new address frame Bit 1 = checksum error/Parity error (8 bit protocols only) Bit 2 = soft limits exceeded Bit 3 = overtemp shutdown / Killmask shutdown Bit 4 = Framing Error Bit 5 = Too long of message Bit 6 = Check Bit return Status (if condition found) Bit 7 = Receiver overflow Bit 8 = Motion Error (Moving) Bit 9 = Position Error (Stopped) Bit 10 = Voltage Shutdown Bit 11 = Move terminated with sensor found Bit 12 = Sequence errors Bit 13 = Sequence complete (into anti hunt mode) Bit 14 = Foreground Command errors Bit 15 = Foreground Command Done

64009‡	R		Reserved
64010‡	R		MSTATE_STOR|MSTATE2 (MSTATE from thread 1, accessible by thread 2|MSTATE of thread 2 accessible by thread 1)
64011‡	R		CMD_ST2|CMD_ST_STOR (Pointers to Command buffer for thread 2, pointer to command buffer for thread 1.)
64012‡	R	No SD	VP * 10| Vclmp * 10
64013‡	R	No SD	TempF|TempC (0.1F, 0.1C increments)
64014‡	R	No SD	5v supply * 100| 3.3 supply * 100
64015‡	R	No SD	Reserved
64016‡	R	No SD	Reserved
64017‡	R	No SD	Reserved
64018‡	R	No SD	Reserved
64018‡	R	No SD	Reserved
64020‡	R	No SD	Reserved
64021‡	R	No SD	Reserved
64022‡	R	No SD	Reserved
64023‡	R	No SD	Reserved
64024‡	R	No SD	Reserved
64025‡	R	No SD	Reserved
64026‡	R	No SD	Reserved
64027‡	R	No SD	Reserved
64028‡	R	No SD	Reserved
64029‡	R	No SD	Reserved
64030	R/F	No SD	Product Code
64031	R/F	No SD	Serial Number
64032‡	R	X only	ADCIO1|ADCIO2
64033‡	R	X only	ADCIO3|ADCIO4
64034‡	R	X only	ADCIO5|ADCIO6
64035‡	R	X only	ADCIO7|0-10v input

Appendix B: Error Codes

Command Error Code Descriptions

Cmd Err	Thread#	Description
1	1	Invalid register or not enough writable registers
2	1	Not able to perform requested motion in requested time
3	1	No motion was pre-calculated
4	1	Command prohibited in current state
5	1	Invalid command buffer location
6	1	Unable to perform action because already active
7	1	Invalid Selection: Mode, subcommand, I/O bit
8	1	Parameter out of range
9	1	Internal error or bad calibration data
10	1	Invalid command number found in command buffer
11	1	Stack space problem - underflow/overflow
12	1	Bad EEPROM access
13	1	CAN Dictionary location not writable
14	1	CAN Dictionary location not readable
15	1	No such Data Dictionary entry
16	1	CAN Dictionary internal consistency problem (please report to factory)
17	1	Thread 2 only command trying to execute in thread 1
18	1	CAN Dictionary - Write not successful due to system state
19	1	CAN Dictionary - byte count invalid
65	2	Invalid register or not enough writable registers
66	2	Not able to perform requested motion in requested time
67	2	No motion was pre-calculated
68	2	Command prohibited in current state
69	2	Invalid command buffer location
70	2	Unable to perform action because already active
71	2	Invalid selection: Mode-Subcommand-I/OBit
72	2	Parameter out of range
73	2	Internal error or bad calibration data
74	2	Invalid command number found in command buffer
75	2	Stack space problem - underflow/overflow
76	2	Bad EEPROM access
77	2	CAN Dictionary location not writable
78	2	CAN Dictionary location not readable
79	2	No such Data Dictionary entry
80	2	CAN Dictionary internal consistency problem (please report to factory)
81	2	Thread 2 only command trying to execute in thread 1
82	1	CAN Dictionary - Write not successful due to system state
83	1	CAN Dictionary - byte count invalid
129	0	Invalid register or not enough writable registers
132	0	Command prohibited in current state
136	0	Parameter out of range
193	3	Too many bytes called out for a single register must be 1 to 3 - DMX
200	3	Destination register not valid - DMX"

Appendix C: Conversion Data

Inertia - To convert from A to B, multiply by the constant in table

A \ B	oz-in²	oz-in-s²	lb-in²	lb-in-s²	N-m-s²	g-cm²	kg-m²	kgf-m-s²
oz-in²	1	$2.59*10^{-3}$	$6.25*10^{-2}$	$1.6188*10^{-4}$	$1.8289*10^{-5}$	182.9	$1.8289*10^{-5}$	$1.86*10^{-6}$
oz-in-s²	386.09	1	24.131	$6.25*10^{-2}$	$7.0612*10^{-3}$	$7.0612*10^{4}$	$7.0612*10^{-3}$	$7.2*10^{-4}$
lb-in²	16	$4.1441*10^{-2}$	1	$2.5901*10^{-3}$	$2.9262*10^{-4}$	2926.2	$2.9262*10^{-4}$	$2.9839*10^{-5}$
lb-in-s²	6177	16	386.09	1	0.11298	$1.1298*10^{6}$	0.11298	$1.1521*10^{-2}$
N-m-s²	$5.4678*10^{4}$	141.62	3417.4	8.8512	1	$1*10^{7}$	1	0.10197
g-cm²	$5.4678*10^{-3}$	$1.4162*10^{-5}$	$3.4174*10^{-4}$	$8.8512*10^{-7}$	$1*10^{-7}$	1	$1*10^{-7}$	$1.0197*10^{-8}$
kg-m²	$5.4678*10^{4}$	141.62	3417.4	8.8512	1	$1*10^{7}$	1	0.10197
kgf-m-s²	$5.3621*10^{5}$	1388.8	$3.3513*10^{4}$	86.801	9.8067	$9.8067*10^{7}$	9.8067	1

Power - To convert from A to B, multiply by the constant in table

A \ B	Watt	HP	N-m-RPS	oz-in -RPM	ft-lb-RPM	ft-lb/sec	N-m/sec
Watt	1	$1.341*10^{-3}$	0.1592	1352	7.042	0.7375	1
HP	745.7	1	118.7	$1.0083*10^{6}$	5251.4	549.93	745.7
N-m-RPS	6.283	$8.426*10^{-3}$	1	8496	44.25	4.634	6.283
oz-in -RPM	$7.396*10^{-4}$	$9.918*10^{-7}$	$1.177*10^{-4}$	1	$5.208*10^{-3}$	$5.454*10^{-4}$	$7.396*10^{-4}$
ft-lb-RPM	0.142	$1.904*10^{-4}$	$2.26*10^{-2}$	192	1	0.1047	0.142
ft-lb/sec	1.356	$1.818*10^{-3}$	0.2158	1833	9.549	1	1.356
N-m/sec	1	$1.341*10^{-3}$	0.1592	1352	7.0423	0.7375	1

Torque - To convert from A to B, multiply by the constant in table

A \ B	ft-lb	in-lb	oz-in	N-m	kgf-m	kgf-cm	gf-cm
ft-lb	1	12	192	1.3558	0.13825	13.825	$1.3825*10^{4}$
in-lb	$8.333*10^{-2}$	1	16	0.113	$1.1521*10^{-2}$	1.1521	1152.1
oz-in	$5.2083*10^{-3}$	$6.25*10^{-2}$	1	$7.0615*10^{-3}$	$7.2006*10^{-4}$	$7.2006*10^{-2}$	72.006
N-m	0.73757	8.8509	141.61	1	0.10197	10.197	$1.0197*10^{4}$
kgf-m	7.2331	86.798	1388.8	9.8067	1	100	$1*10^{5}$
kgf-cm	$7.2331*10^{-2}$	0.86798	13.888	$9.8067*10^{-2}$	$1*10^{-2}$	1	1000
gf-cm	$7.2331*10^{-5}$	$8.6798*10^{-4}$	$1.3888*10^{-2}$	$9.8067*10^{-5}$	$1*10^{-5}$	$1*10^{-3}$	1

Additional Conversion Data

Length	1 inch = 0.0254 meters	Temperature	°F = [°C • (9/5)] + 32
Mass	1 ounce = 0.02835 kilograms		
Velocity	1 revolution/second (rps) = 60 revolutions/minute (rpm)		

Command Set - Numeric/TLA List

NOTE: Some commands share the same command number. This occurs when a command accepts alternate parameters or has multiple uses.

(1) See SilverLode CANopen User Manual
+ Combo-Command: See Combo-Command at the beginning of this manual for details.

Sorted By Command Number

Cmd Num	Acronym (TLA)	Command Name	Reference: Page Number or Tech Doc
+	CCTR	CAN Connect to Remote	198
+	CRML	CAN Register Map, Local	208
+	CRMR	CAN Register Map, Remote	210
+	CTRL	CAN Transmit Register, Local	212
+	CTRR	CAN Transmit Register, Remote	214
+	DMRM	DMX Register Map	49
0	POL	Poll	9
1	CPL	Clear Poll	6
2	HLT	Halt	9
3	STP	Stop	30
4	RST	Restart	23
5	RVN	Revision	23
6	RPB	Read Program Buffer	21
8	CLP	Clear Program	6
9	SDL	Start Download	27
10	RUN	Run Program	26
11	WRI	Write Register, Immediate Type	32
12	RRG	Read Register	23
13	SPR	Store Program	29
14	LPR	Load Program	9
15	VMI	Velocity Mode, Immediate Mode	31
20	RIS	Read Internal Status Word	19
21	RIO	Read I/O States	16
25	IMW	Interpolated Mode Write Queue	9
27	POR	Poll With Response	14
30	WRX	Write Register Extended	33
31	CII	Configure I/O, Immediate Mode	6
32	RRW	Read Register Write	24
33	PUP	Protect User Program	16
64	ADX	ACK Delay Extended	QCI-TD053
65	CER	Command Error Recovery	39
66	ETP	End Of Travel, Positive	59
67	ETN	End Of Travel, Negative	58
68	FL2	Filter Constants 2	61

69	VLL	Velocity Limits	94
70	CT2	Control Constants 2	42
71	CBD	CAN Baud Rate	197
72	CDL	CAN Dictionary Access, Local	199
73	CID	CAN Identity	204
74	CNL	CAN Set NMT State, Local	205
75	T1F	Thread 1 Force LRP	158
76	T2S	Thread 2 Start	159
77	T2K	Thread 2 Kill Conditions	91
78	PLS	Programmable Limit Switch	169
79	PLT	Programmable Limit Trigger	169
80	CDR	CAN Dictionary Access, Remote	201
81	CNR	CAN Set NMT State, Remote	206
86	SMD	Set Mode Data	86
89	JRB	Jump On Register Bitmask	147
89	PCB	Program Call On Register Bitmask	154
92	SSI	SSI Port Mode	89
93	EGM	Electronic Engineering Mode	93
93	PVC	Profile Velocity Continuous/Follower	112
128	END	End Program	135
129	PWO	PWM Output	171
130	SEF	Select Encoder Filter	83
131	LVP	Low Voltage Processor Trip	73
134	MAV	Move Absolute, Velocity Based	102
135	MRV	Move Relative, Velocity Based	104
137	JGE	Jump On Register Greater Or Equal	138
137	JGR	Jump On Register Greater Than	139
137	JLE	Jump On Register Less or Equal	140
137	JLT	Jump On Register Less Than	141
137	JNE	Jump On Register Not Equal	144
137	JRE	Jump On Register Equal	149
138	WCL	Write Command Buffer Long Word	187
139	WCW	Write Command Buffer Word	189
140	DLT	Delay In Ticks	130
140	DLY	Delay	130
141	WDL	Wait Delay	162
142	GCL	Go Closed Loop	62
143	GOL	Go Open Loop	65
144	ZTG	Zero Target	194
145	ZTP	Zero Target And Position	195
146	TTP	Set Target To Position	193
147	CME	Clear Max Error	132
148	CTC	Control Constants	39
149	TQL	Torque Limits	92
150	AHC	Anti-Hunt Constants	35
151	ERL	Error Limits	57

154	WRF	Write Register File	190
154	WRP	Write Register, Program Mode	191
155	IDT	Identity	62
156	LRP	Load And Run Program	150
158	CLX	Calculation Extended	182
159	VMP	Velocity Mode, Program Mode	125
160	RAV	Register Move Absolute, Velocity Based	116
161	RRV	Register Move Relative, Velocity Based	118
162	JMP	Jump	142
162	JOI	Jump On Input	145
163	CIS	Clear Internal Status	192
164	CKS	Check Internal Status	130
165	CLC	Calculation	174
166	CLM	Control Loop Mode	40
167	KMC	Kill Motor Conditions	70
168	MCT	Motor Constants	75
169	FLC	Filter Constants	60
170	EEM	Enable Encoder Monitor	52
171	DDB	Disable Done Bit	43
171	DEM	Disable Encoder Monitor	206
173	ADL	ACK Delay	34
174	BRT	Baud Rate	38
176	MAT	Move Absolute, Time Based	73
177	MRT	Move Relative, Time Based	103
178	RAT	Register Move Absolute, Time Based	115
179	RRT	Register Move Relative, Time Based	117
180	SSD	Scaled Step And Direction	119
181	KMR	Kill Motor Recovery	72
182	KED	Kill Enable Driver	69
183	KDD	Kill Disable Driver	68
184	DIR	Direction	46
185	PRO	Protocol	81
186	SIF	Serial Interface	83
187	EDL	Enable Done Low	52
188	CIO	Configure I/O	162
192	EMN	Encoder Monitor	55
192	SEE	Select External Encoder	119
193	ARI	Analog Read Input	198
194	WBS	Wait On Bit State	161
195	SCF	S-Curve Factor	82
196	RSM	Register Store Multiple	185
197	RLM	Register Load Multiple	183
198	RSN	Register Store to Non-volatile	187
199	RLN	Register Load From Non-volatile	185
200	CLD	Calculation Extended with Data	178
201	PCI	Program Call On Input	152

Appendix D: Commands by TLA

201	PCL	Program Call	153
202	PRI	Program Return On Input	154
202	PRT	Program Return	154
204	WBE	Wait On Bit Edge	160
205	SOB	Set Output Bit	172
206	COB	Clear Output Bit	167
207	ACR	Analog Continuous Read	196
208	PLR	Power Low Recovery	79
209	FOR	For	136
210	NXT	Next	151
212	LVT	Low Voltage Trip	73
213	OVT	Over Voltage Trip	77
214	MTT	Maximum Temperature Trip	75
215	CTW	Calculation Two Word	QCI-TD019
216	PIM	Position Input Mode	105
217	VIM	Velocity Input Mode	119
218	TIM	Torque Input Mode	123
219	AHM	Anti-Hunt Mode	37
220	KMX	Kill Motor Conditions Extended	71
221	SSL	Soft Stop Limits	83
223	RSD	Registered Step & Direction	121
225	EMT	Enable Multi-Tasking	55
226	DMT	Disable Multi-Tasking	50
227	EMD	Enable Motor Driver	51
228	DMD	Disable Motor Driver	46
229	HSM	Hard Stop Move	97
230	AHD	Anti-Hunt Delay	36
231	PCM	Pre-Calculate Move	QCI-TD019
232	PCG	Pre-Calculated Go	QCI-TD019
233	XRV	Extended Reg Move Relative, Velocity Based	129
234	XAV	Extended Reg Move Absolute, Velocity Based	127
235	XRT	Extended Reg Move Relative, Time Based	128
236	XAT	Extended Reg Move Absolute, Time Based	126
237	GOC	Gravity Offset Constant	64
238	JNA	Jump On NAND I/O State	143
239	JOR	Jump On OR I/O State	146
240	PMC	Profile Move Continuous	106
241	PMV	Profile Move	110
242	PMX	Profile Move Exit	111
243	DLC	Dual Loop Control	46
244	SLC	Single Loop Control	83
245	PCP	Position Compare	168
248	ATR	Add To Register	130
249	PMO	Profile Move Override	109
250	JAN	Jump On AND I/O State	137
251	EDH	Enable Done High	51

252	DIF	Digital Input Filter	43
253	IMS	Interpolated Move Start	98
254	IMQ	Interpolated Move Queue Clear	100
255	RSP	Restart, Program Mode	157

Index

Index

Index

Index

Index