SilverLode™ Servo Family Command Reference

Revision 5.00
6 April 2009
For QuickControl Rev 5.00

Table of Content

How to Use This Manual

The Command Reference contains a detailed description of every command for the SilverLode™ Product Family which includes the SilverNugget™ and SilverDust™. It should be used as a reference not as a tutorial. For general information on QuickControl, please refer to the SilverLode User Manual.

The 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) SilverLode 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.

User must re-run the Configuration Wizard and Initialization Wizard in QuickControl right after replacing either motor, encoder, and/or driver; motor must be removed from load prior to powering up system after changing any of these elements to prevent potential injury or damage.

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.

The SilverNugget N3 must be wired with a voltage clamp (i.e. QCI-CLCF-04) between the N3 and the Driver power supply; the SilverNugget N2 and SilverDust D2 may require a clamp, according to the application. The voltage clamp must be placed close enough to the SilverLode servo controller/driver module to guarantee that the voltage difference between the module and the clamp at maximum current never

exceeds 1.5 Volts. (This includes the drop across both the power and ground conductors.)

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

! User shall limit current to the SilverNugget N3 units to no more than 35A, or shall fuse power to the SilverNugget N3 using a slow acting fuse rated at not more than 35A.

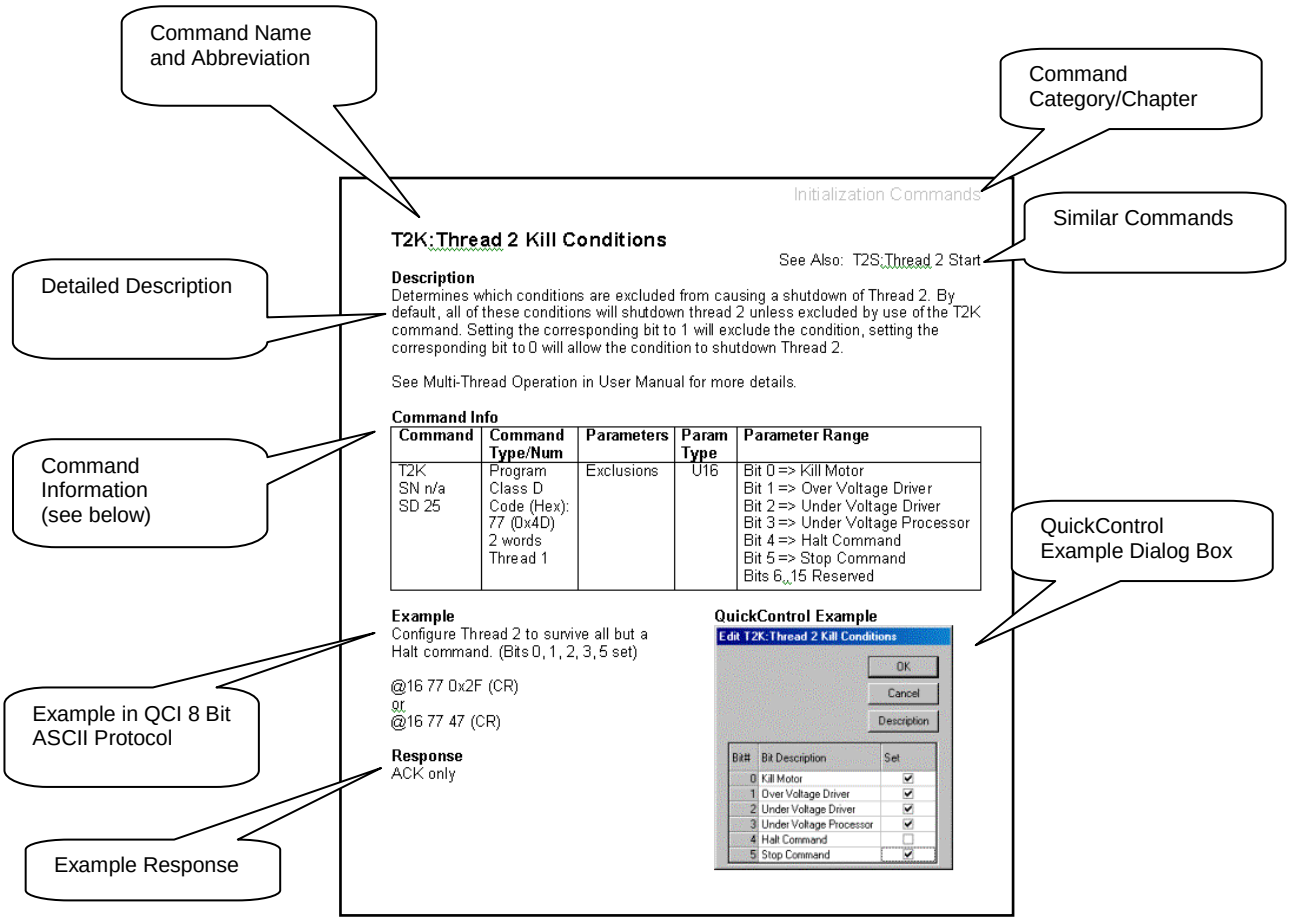
Trademarks

QuickControl is a registered trademarks and property of QuickSilver Controls, Inc. SilverLode, SilverNugget, SilverDust, Anti-Hunt, and PVIA are trademarks and property of QuickSilver Controls, Inc. All other names and trademarks cited are property of their respective owners, 2006.

Copyright

The SilverLode servo family's embedded software, electronic circuit board designs, embedded CPLD logic, and this User Manual are Copyright © 1996-2006 by QuickSilver Controls, Inc.

Page Anatomy



Command Information

Command Name

- Name of command and its three-letter acronym.
- First firmware revision this command appeared. Blank implies the command was available on the first SilverNugget. Otherwise the SilverLode products are listed with the revision this command was first available.
    - SilverNugget(SN), SilverDust(SD)
    - all: available in all revisions of this product
    - n/a: not available for this product
        - Example SN n/a, SD 05
          Not available on SilverNugget
          Available on SilverDust Rev 05 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.
- Commands numbers are given in decimal and hexadecimal format. In the above example the command number is 215 (0xD7). 215 decimal and D7 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 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. Overwrite the buffer and take over motion and command processing.)

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 the 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 were introduced in QuickControl Rev 4.4. Combo-Commands provide a macro like program construct in which user selections cause the parameters of multiple native commands to be simultaneously edited. All native commands have three letter acronyms, where as 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.

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.

How to Use This Manual

The Command Reference contains a detailed description of every command for the SilverLode™ Product Family which includes the SilverNugget™ and SilverDust™. It should be used as a reference not as a tutorial. For general information on QuickControl, please refer to the SilverLode User Manual.

The 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) SilverLode 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.

User must re-run the Configuration Wizard and Initialization Wizard in QuickControl right after replacing either motor, encoder, and/or driver; motor must be removed from load prior to powering up system after changing any of these elements to prevent potential injury or damage.

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.

The SilverNugget N3 must be wired with a voltage clamp (i.e. QCI-CLCF-04) between the N3 and the Driver power supply; the SilverNugget N2 and SilverDust D2 may require a clamp, according to the application. The voltage clamp must be placed

close enough to the SilverLode servo controller/driver module to guarantee that the voltage difference between the module and the clamp at maximum current never exceeds 1.5 Volts. (This includes the drop across both the power and ground conductors.)

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

! User shall limit current to the SilverNugget N3 units to no more than 35A, or shall fuse power to the SilverNugget N3 using a slow acting fuse rated at not more than 35A.

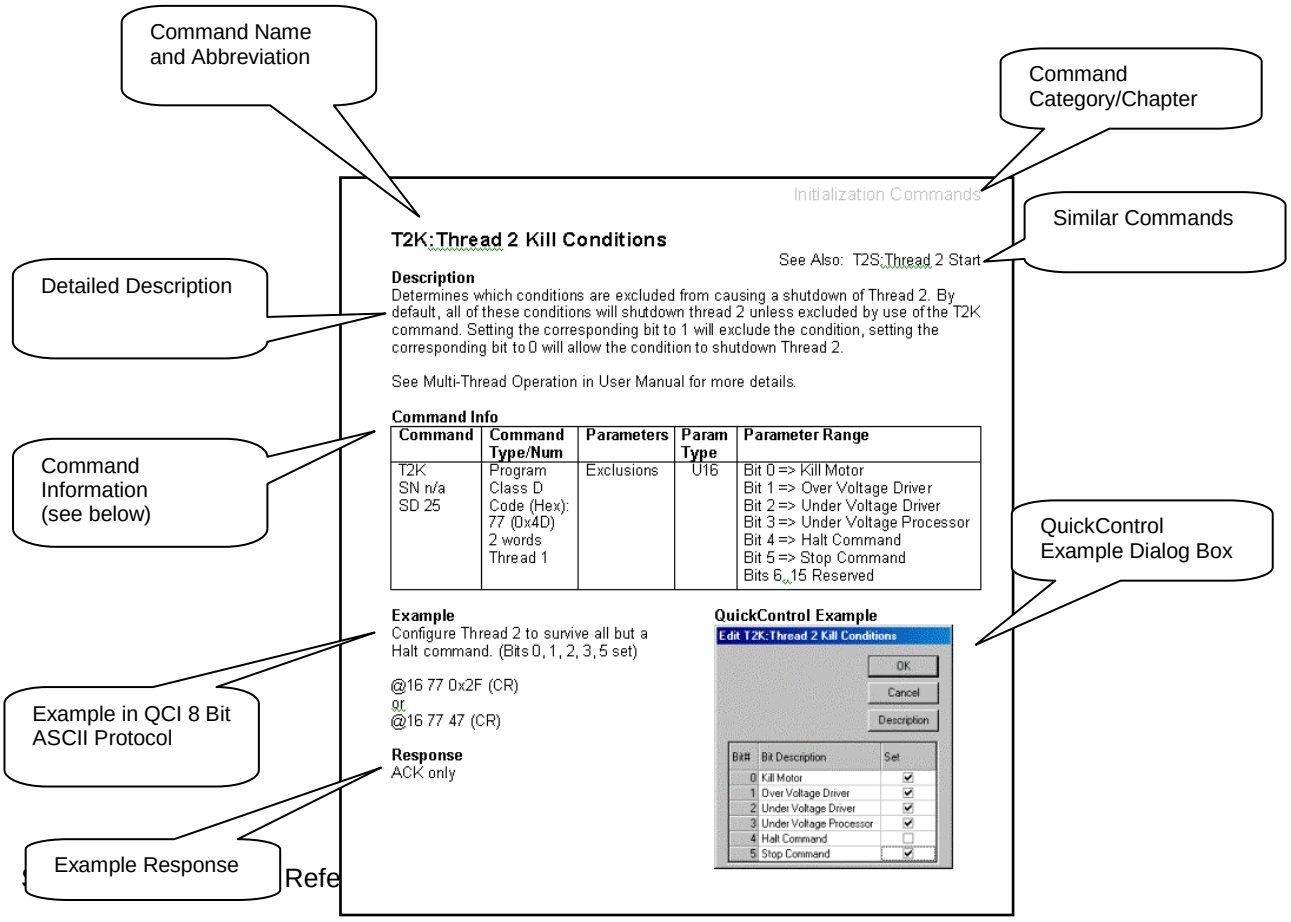
Trademarks

QuickControl is a registered trademarks and property of QuickSilver Controls, Inc. SilverLode, SilverNugget, SilverDust, Anti-Hunt, and PVIA are trademarks and property of QuickSilver Controls, Inc. All other names and trademarks cited are property of their respective owners, 2006.

Copyright

The SilverLode servo family's embedded software, electronic circuit board designs, embedded CPLD logic, and this User Manual are Copyright © 1996-2006 by QuickSilver Controls, Inc.

Page Anatomy



Command Information

Command Name

- Name of command and its three-letter acronym.
- First firmware revision this command appeared. Blank implies the command was available on the first SilverNugget. Otherwise the SilverLode products are listed with the revision this command was first available.
    - SilverNugget(SN), SilverDust(SD)
    - all: available in all revisions of this product
    - n/a: not available for this product
        - Example SN n/a, SD 05
          Not available on SilverNugget
          Available on SilverDust Rev 05 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.
- Commands numbers are given in decimal and hexadecimal format. In the above example the command number is 215 (0xD7). 215 decimal and D7 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 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. Overwrite the buffer and take over motion and command processing.)

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 the 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 were introduced in QuickControl Rev 4.4. Combo-Commands provide a macro like program construct in which user selections cause the parameters of multiple native commands to be simultaneously edited. All native commands have three letter acronyms, where as 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.

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.

Status Commands

Status commands are used to retrieve information from the device. These commands can all be used while the device is executing a motion or a program.

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)

Response
ACK only

QuickControl Example
Immediate (Host) Mode Command 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)

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 SD 05 SN na	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)

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)

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)

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)

QuickControl Example
Immediate (Host) Mode Command Only

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 stripped off. 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)

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	"11" = November
Day (day)	1 Byte	"16" = 16th day
Year (year)	2 Bytes	"1998" = The year 1998
Revision (rev)	2 Bytes	"0108" = Code rev 108
Serial Comm Buffer Size (ser size)	1 Byte	"0A" = 10 Words
Program Buffer Size (pb size)	1 Byte	"34" = 52 Words

*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 Only

Response
Revision Code (8 Bytes)

Response Example
Revision code

10 0005 1116 1998 0108 0A34 (CR)

Initialization Commands

These include all the commands to setup communication, modify servo performance, set error limits conditions and other miscellaneous initialization commands.

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 prepare for the ACK.

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).

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 indicates the 2 character time period used to indicate a new frame. See Application Note "QCI-AN038 Modbus Protocol" for details.

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

Command Info

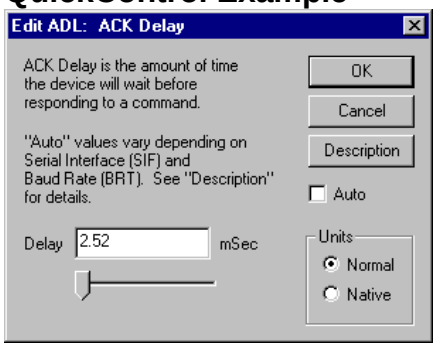
Command	Command Type/Num	Parameters	Param Type	Parameter Range
ADL	Program Class D 173 (0xAD) 2 words Thread 1&2	Delay Count in ticks 1 tick= 120 uSec For negative values: 1 tick = 40 uSec	S16	-32767 to 21845

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

@16 173 21 (CR)

Response
ACK only

QuickControl Example



AFM:Analog Feedback Mode

See Also: SMD: Set Mode Data

Description

The Analog Feedback Mode command allows setting of various configuration values associated with Analog feedback control used by QCI-D2-IGH system. The AFM command is provided within QuickControl, but is actually an alias of the SMD command. See the SMD command for modes and parameters details.

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.

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

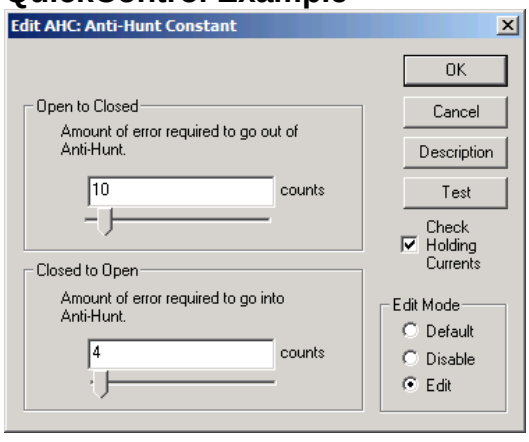
Go into Anti-Hunt when within "4" counts of target. Go out of Anti-Hunt when "10" counts away

@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

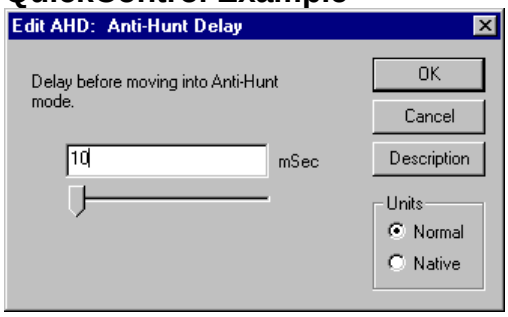
Command	Command Type/Num	Parameters	Param Type	Parameter Range
AHD	Program Class D 230 (0xE6) 2 words Thread 1&2	Delay Count in Ticks 1 Tick = 120usec.	U16	0 to 65535 Default =1250 ticks (150ms)

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

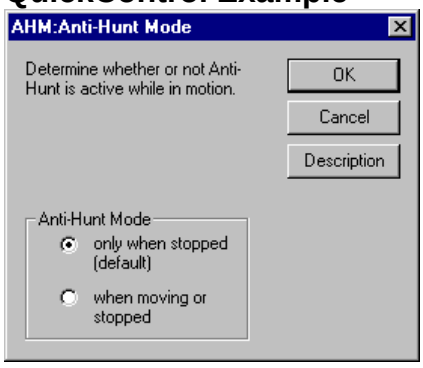
Allow Anti-Hunt Mode only while stopped.

@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.	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

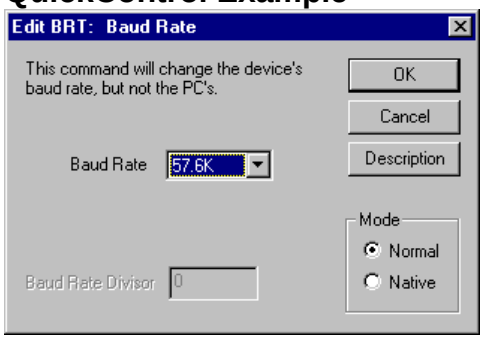
* Not available for Modbus®, ** DMX only.

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 SD 08	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

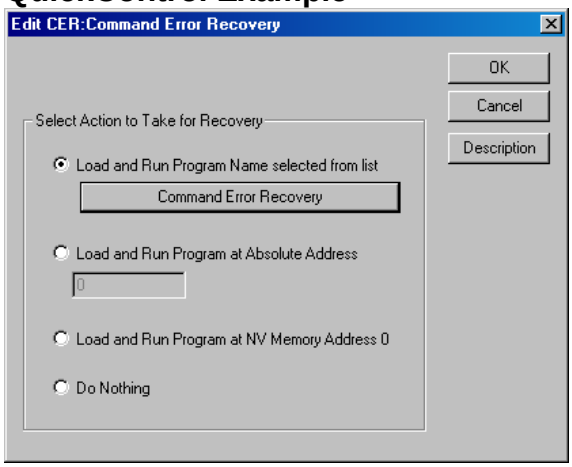
Recover command error from non-volatile memory location 1000

@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).

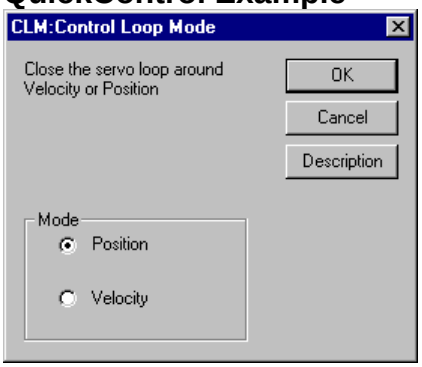
Command Info

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 details.

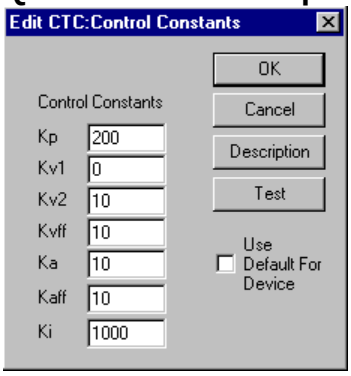
Command Info

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 ignored.

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 (SD08)	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

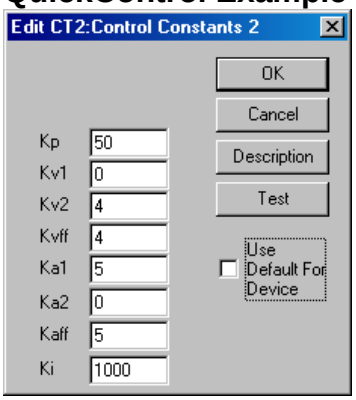
Example

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

Response

ACK only

QuickControl Example



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

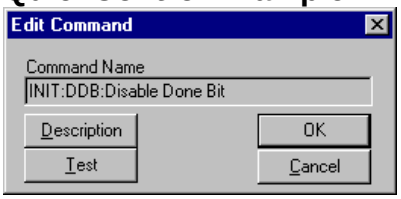
Command	Command Type/Num	Parameters	Param Type	Parameter Range
DDB	Program Class D 171 (0xAB) 1 word Thread 1&2	NONE	NONE	NONE

Example
Disable usage of the “Done” bit

@16 171 (CR)

Response
ACK only

QuickControl Example



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

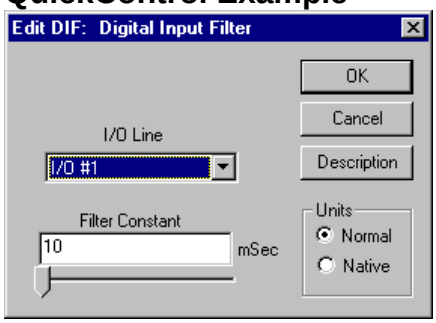
Command	Command Type/Num	Parameters	Param Type	Parameter Range
DIF	Program Class D Code (Hex): 252 (0xFC) 3 Words Thread 1&2	I/O Line #	U16	0 = All Lines 1 to 7 (101 to 116 for SilverDust with extended IO)
		Filter Constant	U16	0 to 32767 Default: 83 ticks (10ms)

Example
Filter Input #1 so that it must be either low or high for as 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

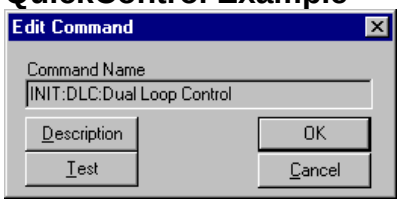
Command	Command Type/Num	Parameters	Param Type	Parameter Range
DLC	Program Class D 243 (0xF3) 1 word Thread 1	NONE	NONE	NONE

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, 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

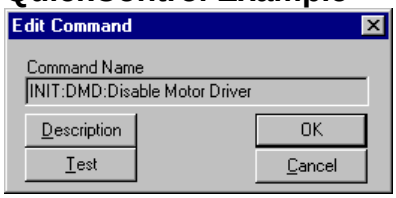
Disable the Motor Driver electronics

@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.

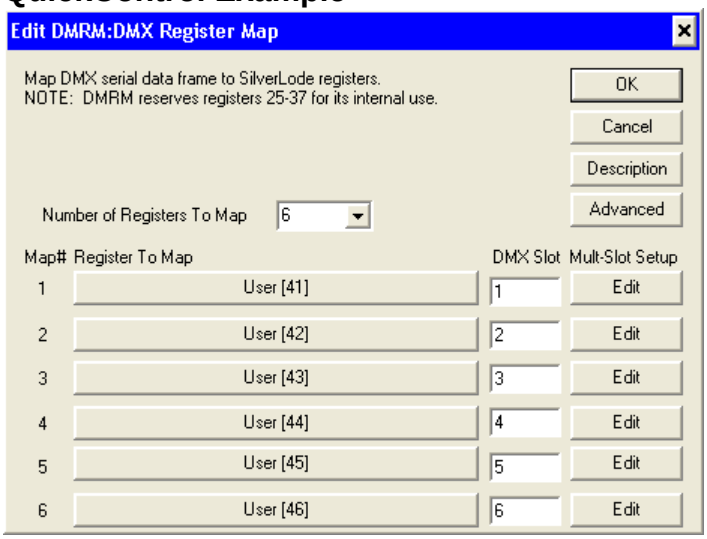
When DMRM is used, registers 25-37 are reserved for internal use.

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 SN n/a SD 34	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 command (i.e. motion) with no ramp down.

Command Info

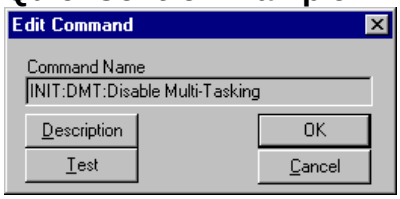
Command	Command Type/Num	Parameters	Param Type	Parameter Range
DMT	Program Class D 226 (0xE2) 1 word Thread 1	NONE	NONE	NONE

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”.

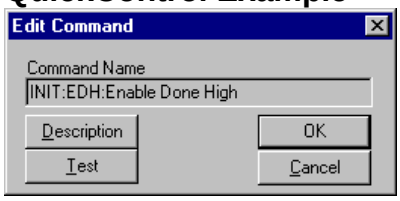
Command Info

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)

QuickControl Example



Response
ACK only

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.

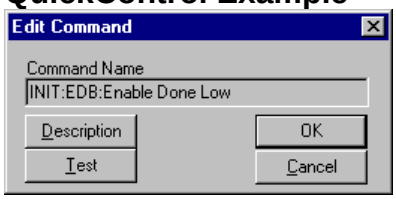
Command Info

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)

QuickControl Example



Response
ACK only

EMD:Enable Motor Driver

See Also: Disable Motor Driver (DMD)

Description

Enables the device motor driver. The driver is by default enabled, 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 Target Position (STP) commands. The ZTP sets both the Target and Position to zero, while the STP 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

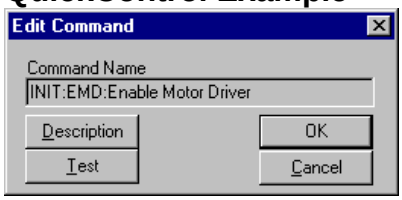
Enable the motor driver

@16 227 (CR)

Response

ACK only

QuickControl Example



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 causes the device to continue program execution after a motion command or mode has been started.

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

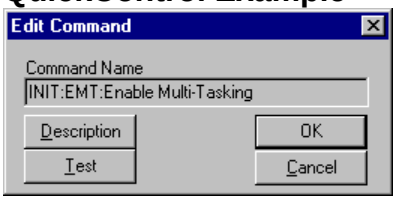
Enable multi-tasking operation

@16 225 (CR)

Response

ACK only

QuickControl Example



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).

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.

Command Info

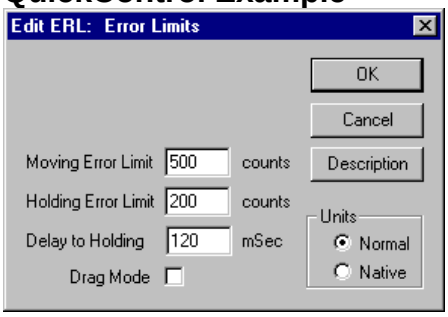
Command	Command Type/Num	Parameters	Param Type	Parameter Range
ERL	Program Class D 151 (0x97) 4 words Thread 1&2	Moving Limit	S16	-32768 to 32767 , Default = 0 QuickControl Default = 20000
		Holding Limit	S16	-32768 to 32767, Default = 0 QuickControl Default = 20000
		Delay to Holding (ticks)	S16	0 to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

Response
ACK only

QuickControl Example



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

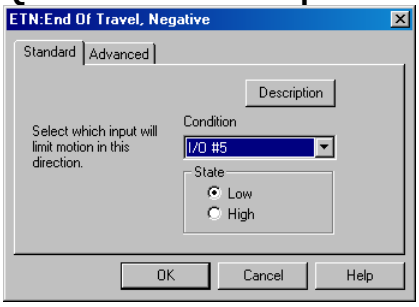
Command	Command Type/Num	Parameters	Param Type	Parameter Range
ETN SD 08	Program Class D 67 (0x43) 7 words Thread 1	Condition Enable ISW	S16	0-65535
		Condition State ISW	S16	0-65535
		Condition Enable IS2	S16	0-65535
		Condition State IS2	S16	0-65535
		Condition Enable XIO	S16	0-65535
		Condition State 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 SD 08	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

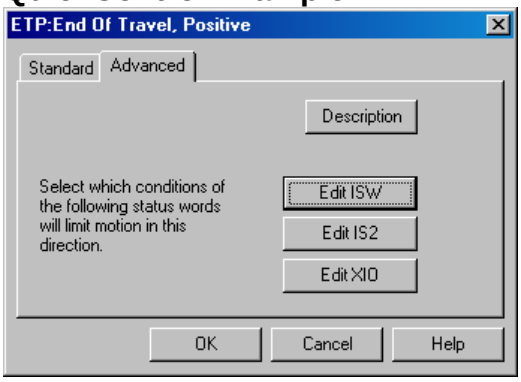
Prevent CW motion if IO1 is high or IO101 is low

@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 Hz to native units.

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

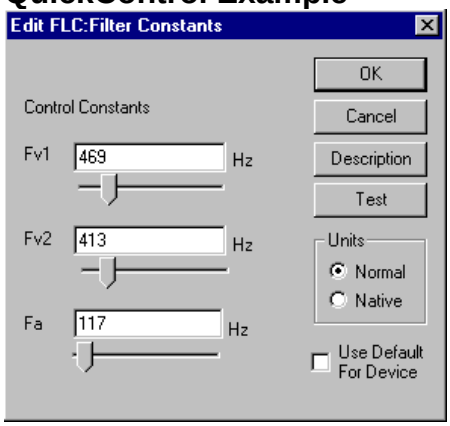
Command	Command Type/Num	Parameters	Param Type	Parameter Range
FLC	Program Class D 169 (0xA9) 4 words Thread 1	Fv1: Velocity 1 Feedback Filter	S16	4096 to 32767
		Fv2: Velocity 2 Feedback Filter	S16	4096 to 32767
		Fa: Acceleration Feedback Filter	S16	4096 to 32767

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, 23L-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 SD 08	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,

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

…..

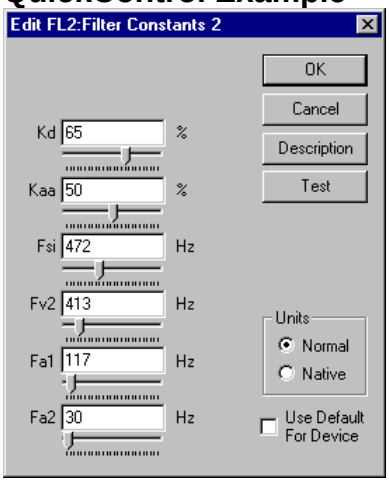
Kaa = 50%
Kd = 65%

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

Response

ACK 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

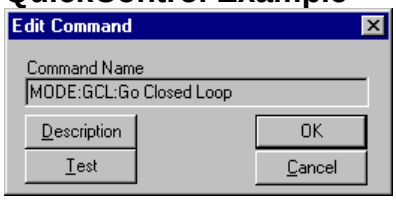
Command	Command Type/Num	Parameters	Param Type	Parameter Range
GCL	Program Class D 142 (0x8E) 1 word Thread 1	NONE	NONE	NONE

Example
Put device into closed loop mode

@16 142 (CR)

Response
ACK only

QuickControl Example



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

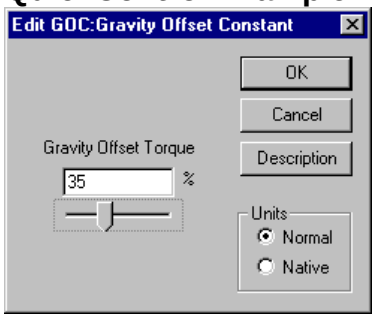
Set the Gravity Offset to 35% Torque for a 23-3 servo

@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

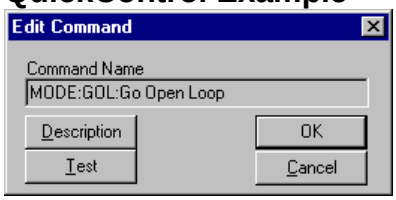
Command	Command Type/Num	Parameters	Param Type	Parameter Range
GOL	Program Class D 143 (0x8F) 1 word Thread 1	NONE	NONE	NONE

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

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;

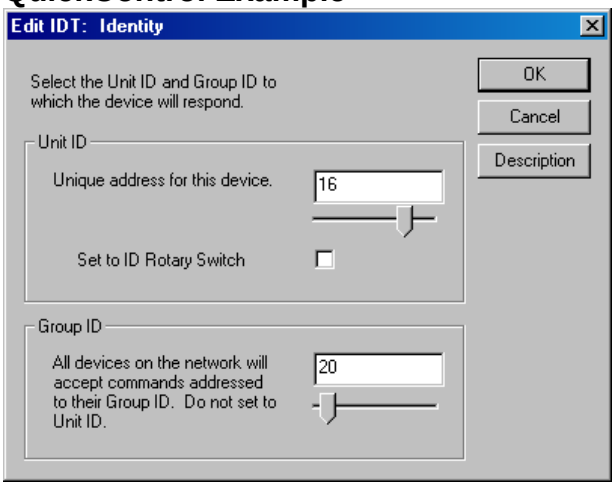
@16 155 5136 (CR)

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

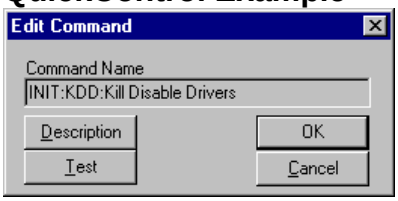
Command	Command Type/Num	Parameters	Param Type	Parameter Range
KDD	Program Class D 183 (0XB7) 1 word Thread 1	NONE	NONE	NONE

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

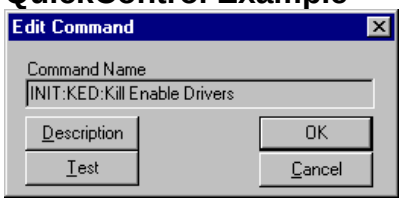
Command	Command Type/Num	Parameters	Param Type	Parameter Range
KED	Program Class D 182 (0xB6) 1 word Thread 1	NONE	NONE	NONE

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.

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

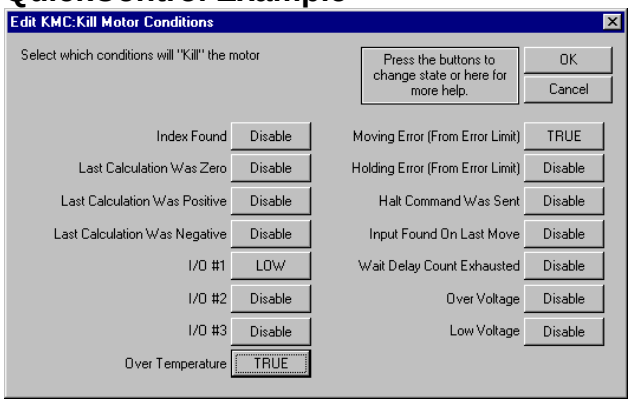
Shut down servo if any of the following conditions are met:
I/O#1 LOW (bit 4)
Over Temp (bit 7)
Moving Error (bit 8)

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.

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 SN n/a SD 05	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

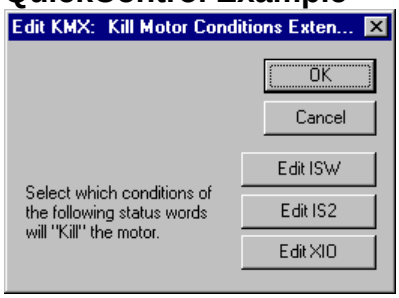
Shut down servo if:

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)

Description

Kill Motor Recovery sets up options for recovery from a device shut down. The Kill Motor Conditions (KMC) establishes 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.

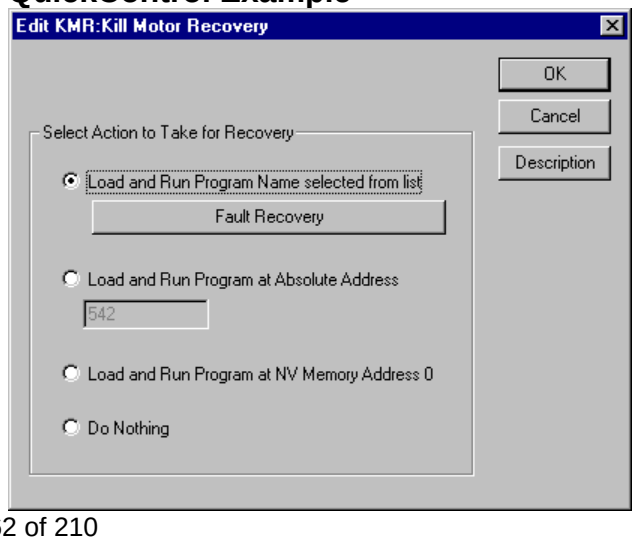
NOTE: In QuickControl, the user only needs to specify the program name. The address is calculated automatically.

@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

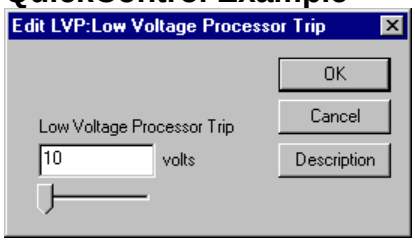
Set LVP to 10 volts

@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

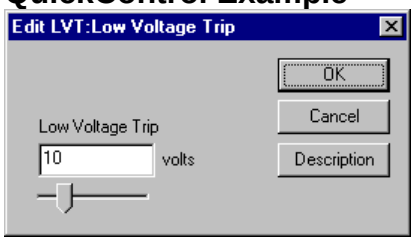
Set shut down at 10 volts

@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

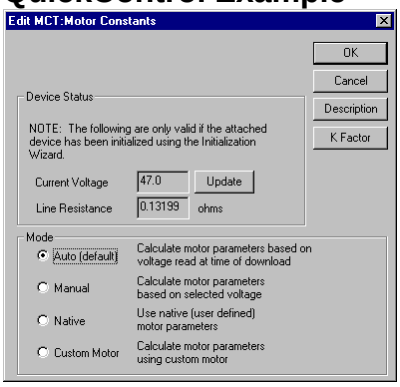
Set up a 23-5 for 24 volt operation

@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.

Command Info

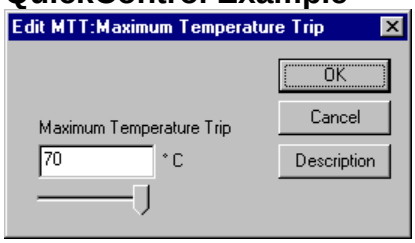
Command	Command Type/Num	Parameters	Param Type	Parameter Range
MTT	Program Class D 214 (0xD6) 2 words Thread 1	Temperature (°C)	U16	0 = Don’t Check 1 to 80 Default: 0

Example
Set Servo to give an error at 70 degrees C

@16 214 70 (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.

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

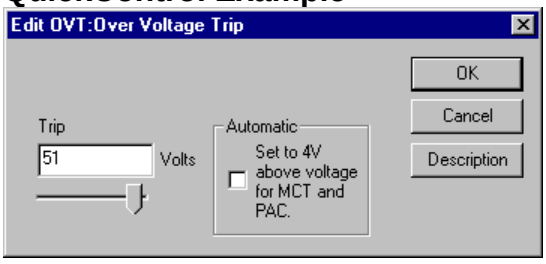
Command	Command Type/Num	Parameters	Param Type	Parameter Range
OVT	Program Class D 213 (0xD5) 2 words Thread 1	Voltage	U16	1 to 53 (52 = Default)

Example
Shut down the Motor if the input voltage exceeds 52 volts

@16 212 52 (CR)

Response
ACK only

QuickControl Example



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

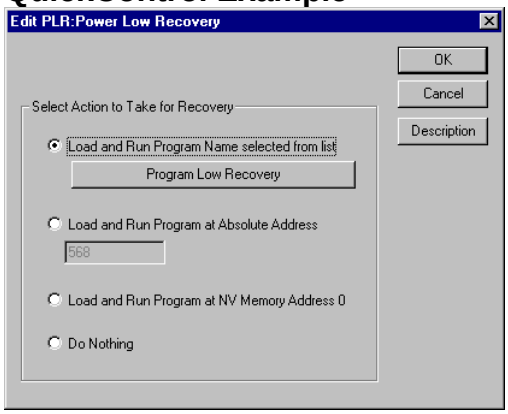
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PLR	Program Class D 208 (0xD0) 2 Words Thread 1	Process	S16	0 = Do Nothing -1 = Load and Run Program @ NV Mem adr 0. #### = LRP @ NV Mem adr.

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

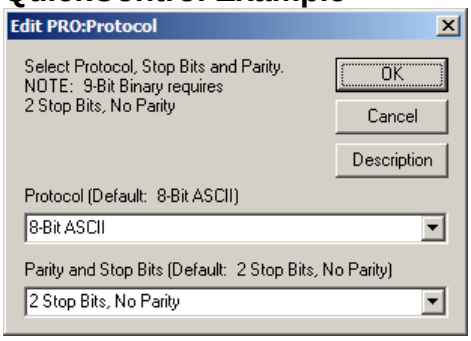
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PRO	Program Class D 185 (0xB9) 2 words Thread 1&2	Parity&Stop | Mode	S16	Mode (lower byte) 0 = 9-Bit 1 = 8-Bit (Default) 2 = Modbus® 3 = DMX512 Parity &Stop Bit 15: Stop bits, 0=>2bits, 1=>1 bit Bit 14: Enable Parity, 0=none, 1=enabled Bit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

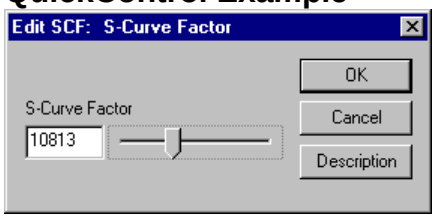
Command	Command Type/Num	Parameters	Param Type	Parameter Range
SCF	Program Class D 195 (0xC3) 2 words Thread 1	Factor	S16	0 = Trapezoidal 1 to 32766 = s-curve 32767 = Full s-curve Default: 0

Example
Use some S-Curve.

@16 195 10813 (CR)

Response
ACK only

QuickControl Example



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 inputs. If an I/O is already set LOW or HIGH it will remain that way after the SEE command.

Index State: For single index encoders like QCI's M-Grade motor/encoders, set Index State to 0 (default) or 1 to detect the index pulse on the falling or rising edge respectively. SilverDust controllers (Rev 05) allow the selecting of a 49/50 encoder index (i.e. I-Grade motor/encoder) by setting Index State = CPR * 3/400.

Note: Internally to the SilverDust, this and the Encoder Monitor (EMN) command are the same command. If EMN and SEE are used in the same program, the last one executed will override any previous EMN or SEE commands.

Parameters

Controller	Index Source	Index State	Encoder Style
SilverNugget	0=Index Source I/O #6 1=Index Source I/O #3	See Above	0 = A/B Quad on I/O #4 & 5 1 = Step Up/Dn on I/O #4 & 5 2 = Step & Dir on I/O #4 & 5 3 = Step & Dir on I/O #2 & 3
SilverDust MG	0=Index Source I/O #6	See Above	0 = A/B Quad on I/O #4 & 5 3 = Step & Dir on I/O #2 & 3
SilverDust IGx	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

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

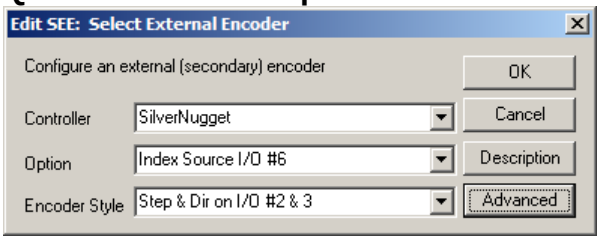
Set up the External encoder inputs for Index on input #6 and Step & Dir on #2 & #3.

@16 192 0 0 3 (CR)

Response

ACK only

QuickControl Example



SEF:Select Encoder Filter

Description
Selects the desired digital filter for the external encoder signals. The default is a 150nS filter. The other option is an 800nS filter for the SilverNugget and 300nS for the SilverDust. 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.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
SEF SN all SD 02	Program Class D 130 (0x82) 2 words Thread 1&2	Filter Enable	S16	SilverNugget 0 = 150nS (Default) 1= 800nS 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

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.

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

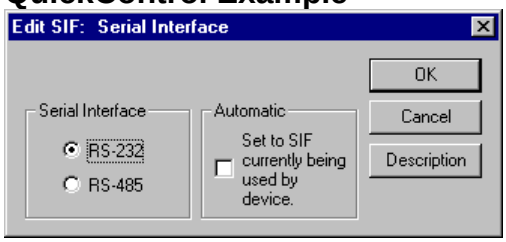
Command	Command Type/Num	Parameters	Param Type	Parameter Range
SIF	Program Class D 186 (0xBA) 2 words Thread 1&2	Mode	S16	0 = RS-232 (Default) 1 = RS-485

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

@16 186 0 (CR)

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

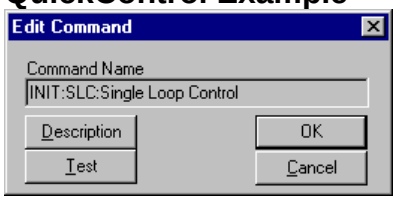
Command	Command Type/Num	Parameters	Param Type	Parameter Range
SLC	Program Class D 244 (0xF4) 1 word Thread 1	NONE	NONE	NONE

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.

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.

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.

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

- 0: Disable
- 1: Position=IO6, Velocity=IO7
- 2: Position & Velocity from IGH inputs
- 3: Position=IO7, Velocity calculated from position
- 4: Position=IGH input, Velocity calculated from position

Mode 4: Analog Input Filter
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
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).

Command Info

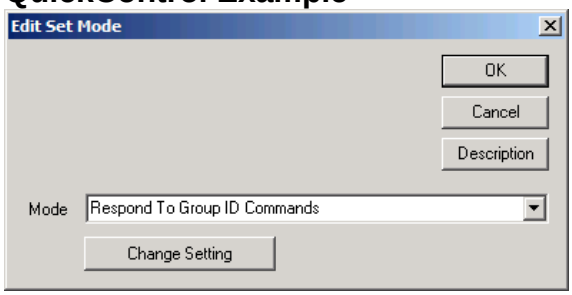
Command	Command Type/Num	Parameters	Param Type	Parameter Range
SMD	Program Class D 86 (0x56) 3 words	Mode	U16	0
		Data	S16	see above

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 SN n/a SD 30	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

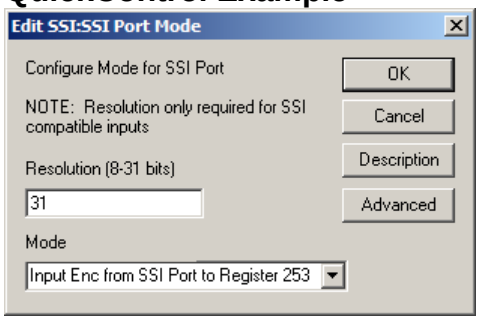
Input 31 bits from SSI encoder

@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

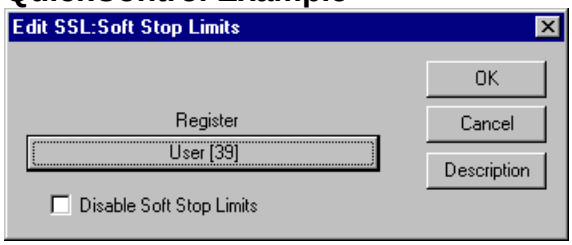
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
SSL	Program Class D 221 (0xDD) 2 words Thread 1&2	Starting Data Register (First of two)	U16	SN 10 to 39 SD 10 to 198 0 = Not Used In QuickControl check "Disable .."

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

@16 221 39 (CR)

Response
ACK only

QuickControl Example



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 shutdown 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 SN n/a SD 25	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

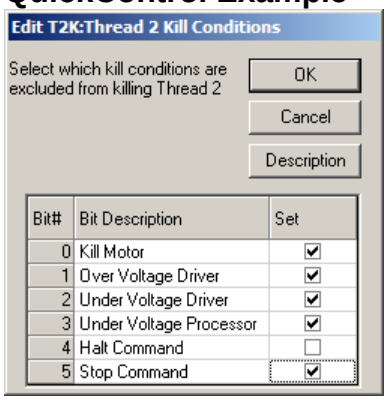
Configure Thread 2 to survive all but a Halt command. (Bits 0, 1, 2, 3, 5 set)

@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

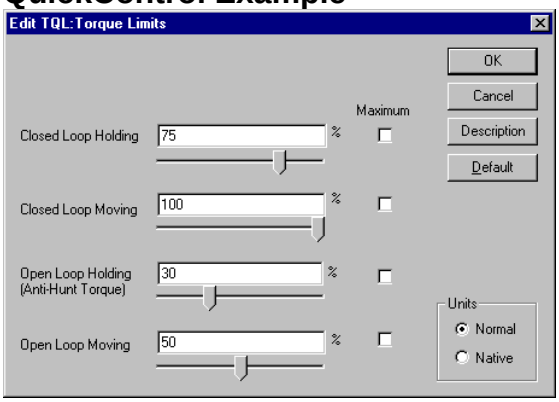
Command	Command Type/Num	Parameters	Param Type	Parameter Range
TQL	Program Class D 149 (0x95) 5 words Thread 1&2	Closed Loop Holding	U16	0 to 32767
		Closed Loop Moving	U16	0 to 32767
		Open Loop Holding	U16	0 to 32767
		Open Loop Moving	U16	0 to 32767

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



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

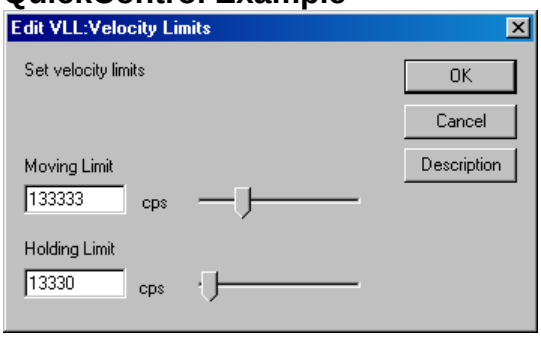
Command	Command Type/Num	Parameters	Param Type	Parameter Range
VLL SN n/a SD 08	Program Class D 69 (0x45) 5 words Thread 1&2	Moving Limit	S16	0 to 32767 SAV
		Holding Limit	S16	0 to 32767 SAV

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

@16 69 16384 1638 (CR)

Response
ACK only

QuickControl Example



Motion & Profile Move Commands

Motion & Profile Move commands make up the set of commands that use the Trajectory Generator to perform simple or complex motions.

EGM:Electronic Gearing Mode

See Also: RSD:Registered Step & Direction

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 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

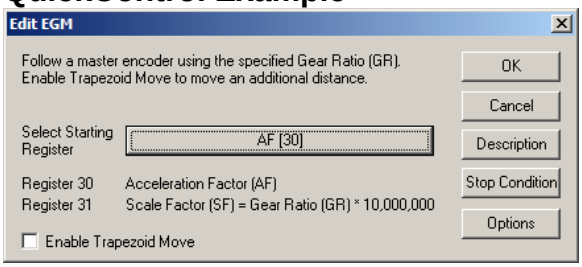
Command	Command Type/Num	Parameters	Param Type	Parameter Range
EGM SN n/a SD 30	Program Class D 93 (0x5D) 5 words Thread 1	Mode	U16	See Above 8=default
		Starting Data Register	U16	11 to 198
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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



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)

Response
ACK only

QuickControl Example
Immediate (Host) Mode Command Only

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

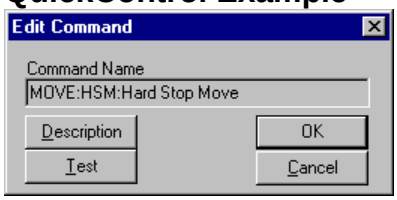
Command	Command Type/Num	Parameters	Param Type	Parameter Range
HSM	Program Class D 229 (0xE5) 1 word Thread 1	NONE	NONE	NONE

Example
Stop the device immediately.

@16 229 (CR)

Response
ACK only

QuickControl Example



MAT:Move Absolute, Time Based

Description
Move Absolute initiates a move to an absolute position.

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

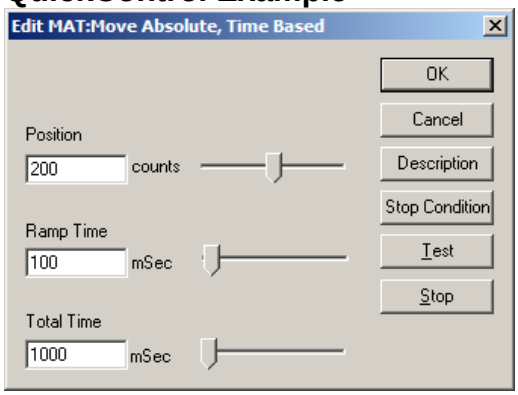
Command	Command Type/Num	Parameters	Param Type	Parameter Range
MAT	Program Class D 176 (0xB0) 9 words Thread 1	Position	S32	-2,147,483,648 to +2,147,483,647
		Acceleration Time	U32	0 to 65534 (7.86 secs)
		Total Time	U32	2 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

Example
Move the device 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.

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

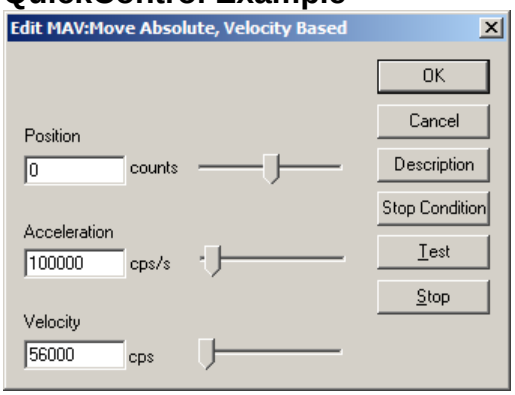
Command	Command Type/Num	Parameters	Param Type	Parameter Range
MAV	Program Class D 134 (0x86) 9 words Thread 1	Position	S32	-2,147,483,648 to +2,147,483,647
		Acceleration	U32	1 to 1,073,741,823
		Velocity	U32	0 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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.

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

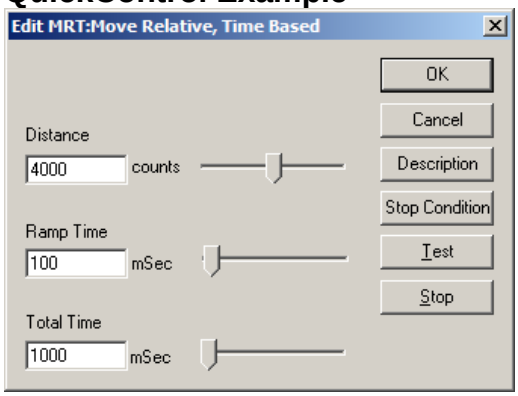
Command	Command Type/Num	Parameters	Param Type	Parameter Range
MRT	Program Class D 177 (0xB1) 9 words Thread 1	Distance	S32	-2,147,483,648 to +2,147,483,647
		Ramp Time	U32	0 to 65534 (7.86 secs)
		Total Time	U32	2 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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

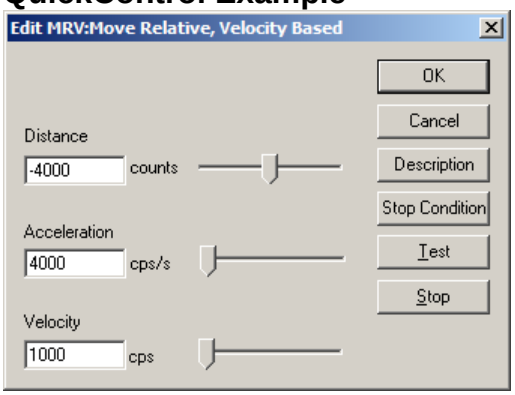
Command	Command Type/Num	Parameters	Param Type	Parameter Range
MRV	Program Class D 135 (0x87) 9 words Thread 1	Distance	S32	-2,147,483,648 to +2,147,483,647
		Acceleration	U32	1 to 1,073,741,823
		Velocity	U32	0 to 2,147,483,647
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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

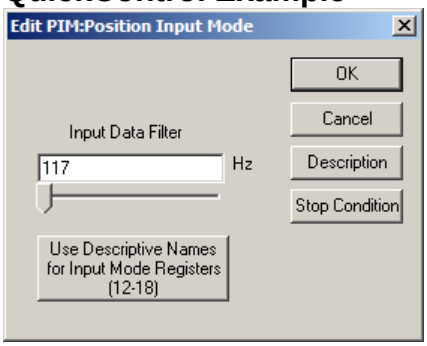
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PIM	Program Class D Code (Hex): 216 (0xD8) Thread 1	Filter Constant	S16	0 to 32767
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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

Description

The Profile Move commands are distinct from the 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). 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 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

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 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 negative if the direction of motion is negative when the input is found. The Offset parameter allows 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

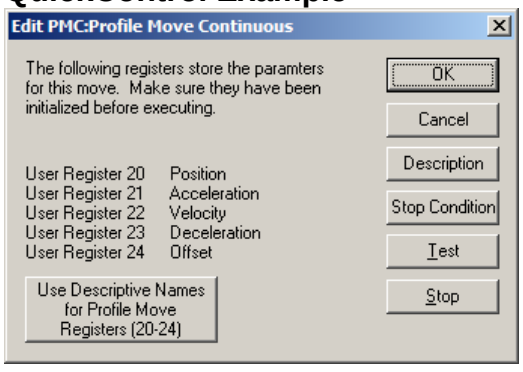
Put the device into a continuous Profile move. Stop if Input #1 is high (“1”).

@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

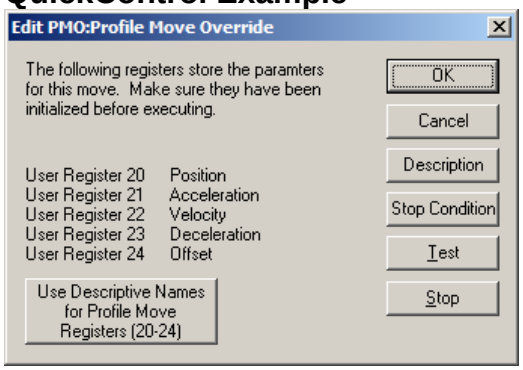
End the current Profile move when at “Position”. Stop if Input #1 is high (“1”).

@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

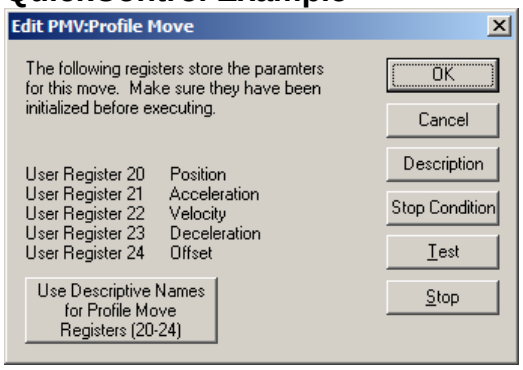
Start a Profile move. Stop if Input #1 is high (“1”).

@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

PVC:Profile Velocity Continuous

See Also: VMI:Velocity Mode, Immediate Mode
VMP:Velocity Mode, Program 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 SN n/a SD 30	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

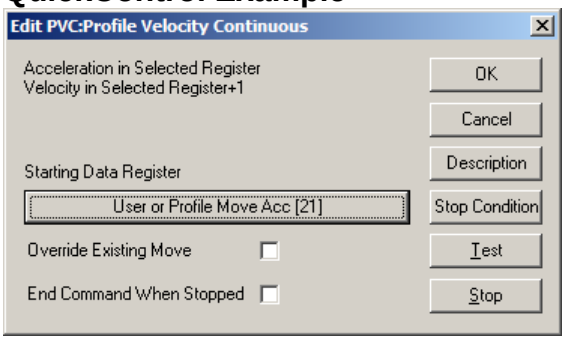
Put device into velocity mode running at acceleration in register 21 and velocity in register 22:

@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

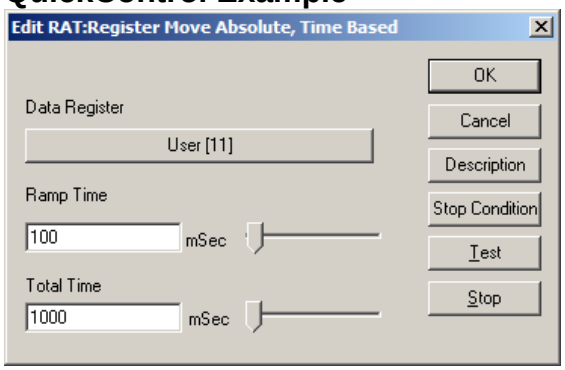
Move to position indicated by User Data Register #11 in 1000 mSec with a 100 mSec acceleration (see Scaling in User Manual).

@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

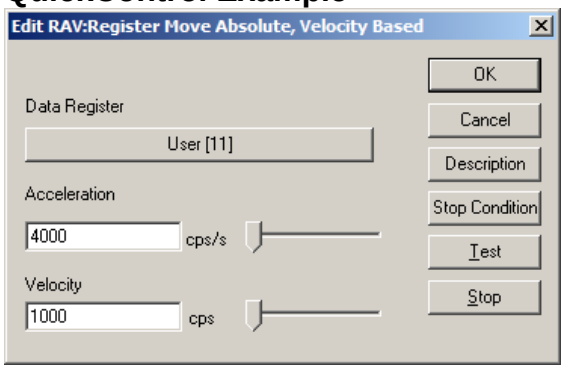
Move to position indicated by User Data Register #11 at vel=1000 cps and acc=4000cps/s.

@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

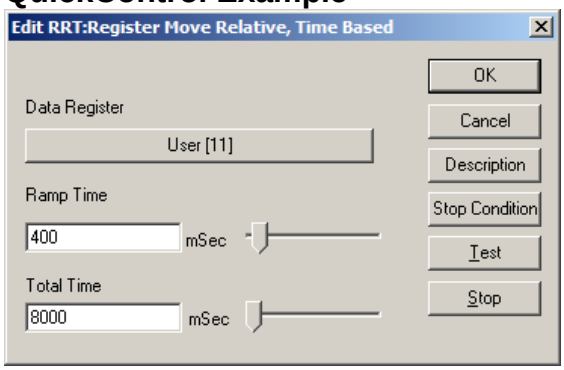
Move to position indicated by User Data Register #11. Do the move in 8 seconds with a 0.400 second acceleration.

@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

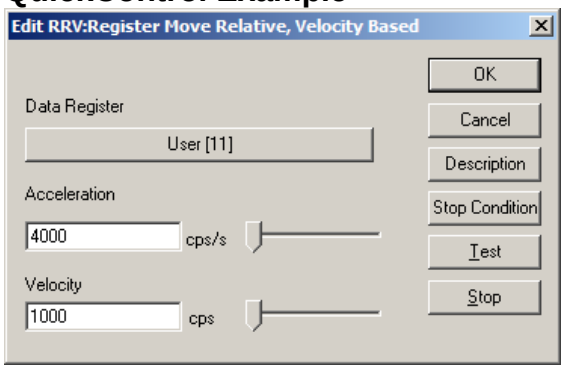
Move to position indicated by User Data Register #11 at vel=1000cps and acc=4000cps/s.

@16 161 11 3865 8053064 0 0 (CR)

Response

ACK only

QuickControl Example



RSD:Registered Step & Direction

See Also: EGM:Electronic Gearing Mode

Description

RSD 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).

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 40 (SilverNugget) 10 to 199 (SilverDust)

Example

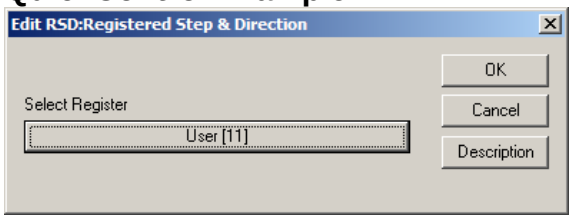
Put the device into a electronic gearing mode using register #11 for the scaling value.

@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).

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

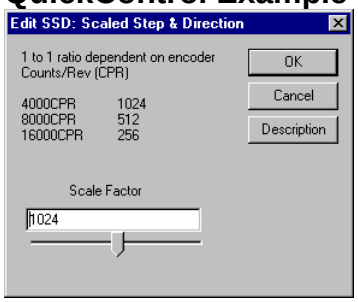
Put the device into a SSD mode with a 1:1 scale factor assuming a 4000 CPR encoder.

@16 180 1024 (CR)

Response

ACK only

QuickControl Example



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.

QuickControl Example
Immediate (Host) Mode Command Only

Response
ACK only
@16 3 0 (CR)

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

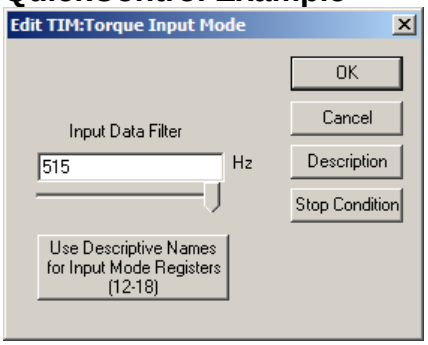
Command	Command Type/Num	Parameters	Param Type	Parameter Range
TIM	Program Class D 218 (0xDA) 4 words Thread 1	Filter constant	S16	0 to 32767
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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

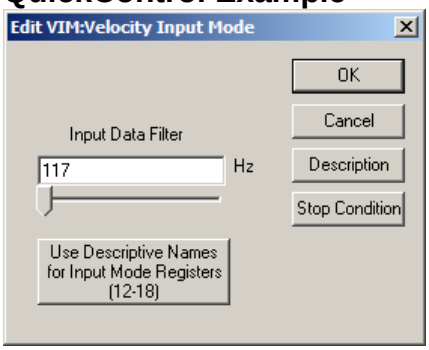
Command	Command Type/Num	Parameters	Param Type	Parameter Range
VIM	Program Class D 217 (0xD9) 4 words Thread 1	Filter Constant	S16	0 to 32767
		Stop Enable	S16/U16	See Above
		Stop State	S16/U16	See Above

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



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.

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

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 with out the completion of that motion, and may be used to shutdown 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):

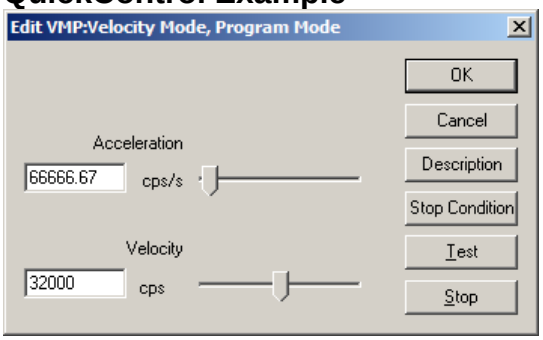
Vel = 32000 cps
Acc = 666666.67

@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

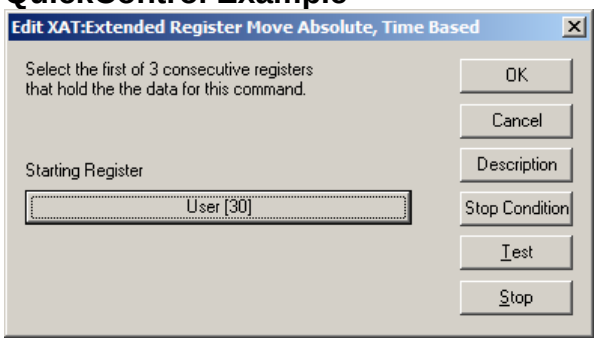
Move the device using parameters from User Data Registers #30-32.

@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

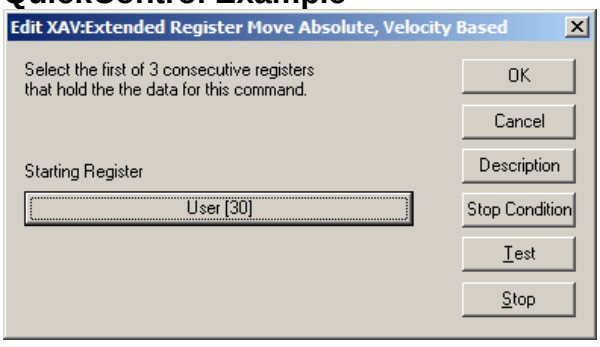
Move the device using parameters from User Data Registers #30-32.

@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

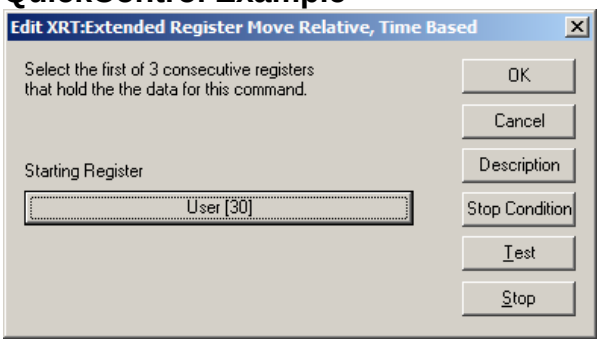
Move the device using parameters from User Data Registers #30-32.

@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

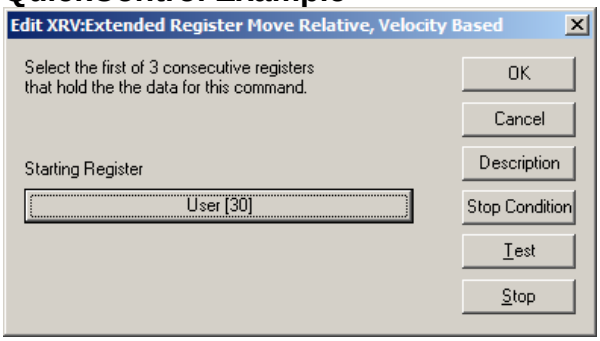
Move the device using parameters from User Data Registers #30-32.

@16 233 30 0 0 (CR)

Response

ACK only

QuickControl Example



Program Flow Commands

Commands that modify the flow or execution of a program.

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)

Response
AKC only

QuickControl Example
Immediate (Host) Mode Command Only

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)

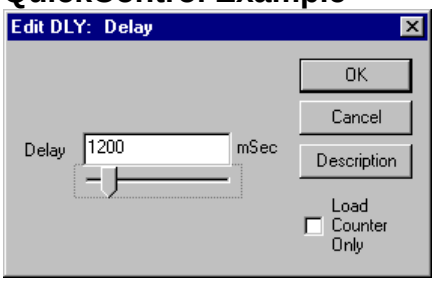
Example
Cause the device to delay program execution by 1.2 seconds.

1200mSec * 8.3333 = 10000 ticks

@16 140 10000 (CR)

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

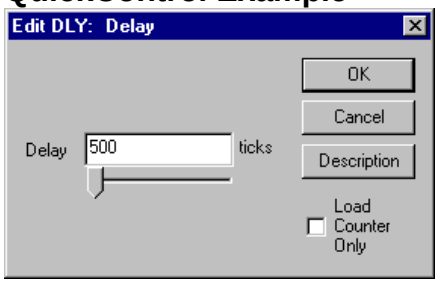
Cause the device to delay program execution by 500 ticks.

@16 140 500 (CR)

Response

ACK only

QuickControl Example



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

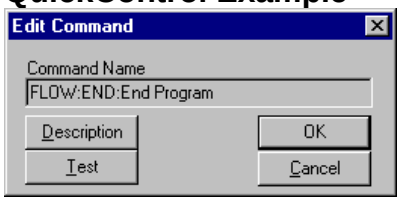
Command	Command Type/Num	Parameters	Param Type	Parameter Range
END	Program Class D 128 (0x80) 1 word Thread 1&2	NONE	NONE	NONE

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.

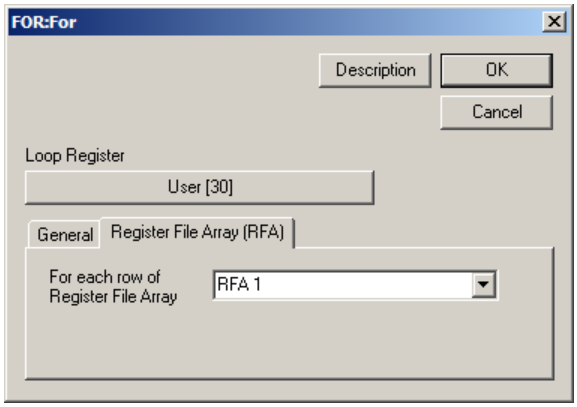
General

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 A[31]" = 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

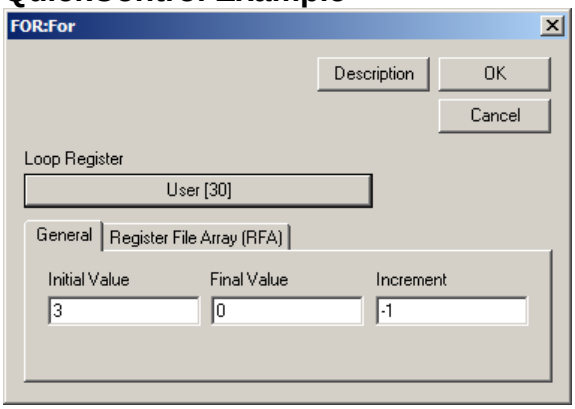
Command	Command Type/Num	Parameters	Param Type	Parameter Range
FOR SN n/a SD 04	Program Class E 209 (0xD1) 8 words Thread 1&2	Initial Value	S32	-2,147,483,647 to 2,147,483,647
		Final Value	S32	-2,147,483,647 to 2,147,483,647
		Increment	S32	-2,147,483,647 to 2,147,483,647
		Loop Register	U16	Standard Register range (Must be writable)

Example
FOR Loop Register 30 = 3 to 0 with -1 increment.

@16 209 3 0 -1 30 (CR)

Response
ACK only

QuickControl Example



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

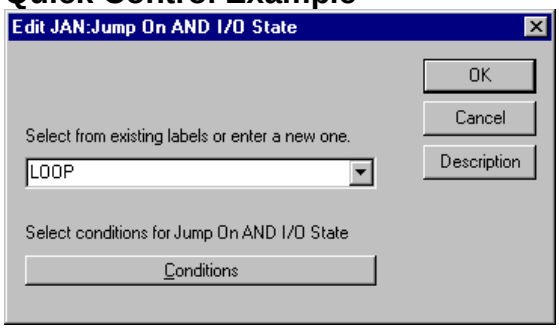
Jump to Program Buffer location 10 if digital inputs #4, #5, #6 and #7 are High (“1”). See I/O State Word (IOS) in User Manual for bit definitions.

@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"

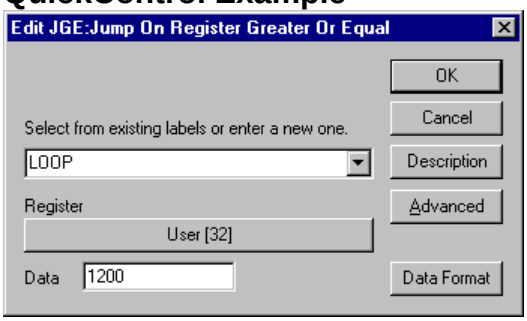
Operation/Register=256 + 32=288

@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 SD 05	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”

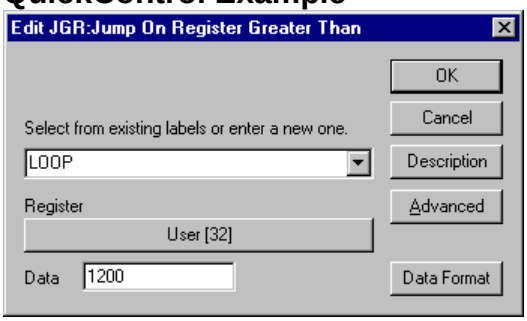
Operation/Register=1024 + 32=1056

@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 SN na SD 05	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”.

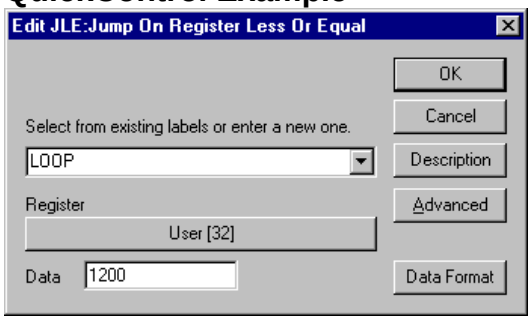
Operation/Register = 1280 + 32 = 1312

@16 137 1312 1200 10 (CR)

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”.

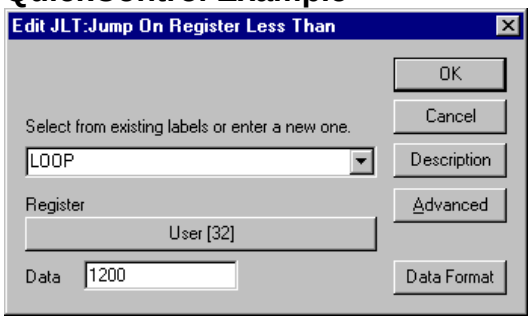
Operation/Register = 512 + 32 = 544.

@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

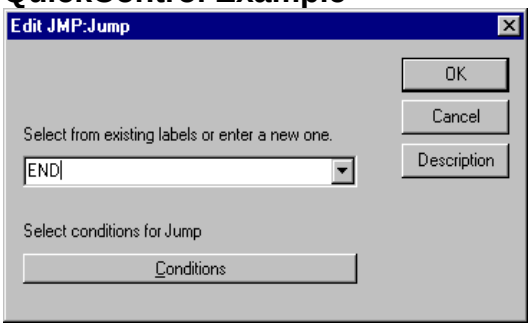
Command	Command Type/Num	Parameters	Param Type	Parameter Range
JMP	Program Class E 162 (0xA2) 4 words Thread 1&2	Condition Enable	U16	0 to 32767
		Condition State	U16	0 to 32767
		Program Buffer Address	U16	0 to Program Buffer Size

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

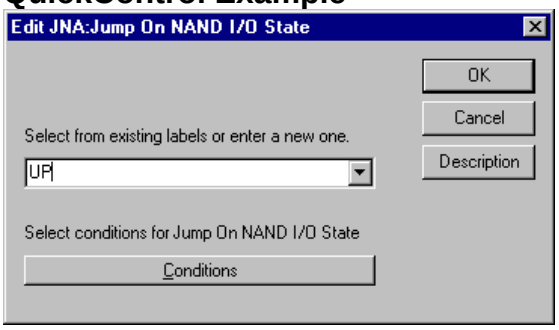
Don’t jump to Program Buffer location 10 if digital inputs #4, #5, #6 and #7 are High (“1”). See I/O State Word (IOS) in User Manual for bit definitions.

@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”

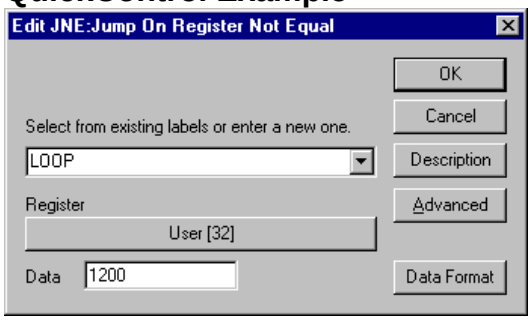
Operation/Register = 768+32= 800.

@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

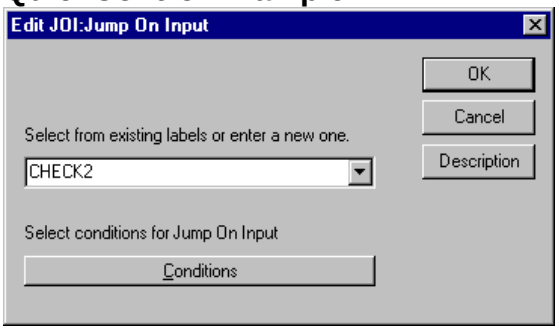
Command	Command Type/Num	Parameters	Param Type	Parameter Range
JOI	Program Class E 162 (0xA2) 4 words Thread 1&2	Enable Code	S16	-1 to -14
		Enable State	S16	0 to 1 0 = “Low” 1 = “High”
		Program Buffer Address	U16	0 to Program Buffer Size

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

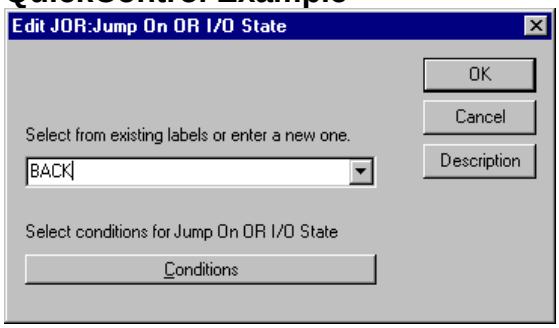
Jump to Program Buffer location 0 if input #1 or #2 or #3 is High (“1”). See I/O State Word (IOS) in User Manual for bit definitions.

@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

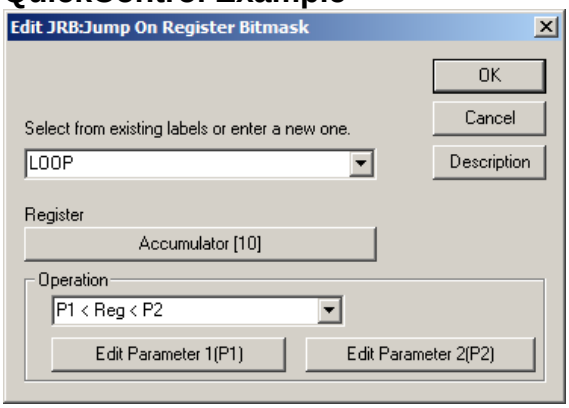
Command	Command Type/Num	Parameters	Param Type	Parameter Range
JRB SN n/a SD 27	Program Class E 89 (0x59) 4 words Thread 1&2	Type/Operation Type (High Byte) Operation (Low Byte)	U16	Type = 0 for JRB Operation: see above table
		Data Register	U16	Standard Register Range
		Param 1/State	S32	-2,147,483,648 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

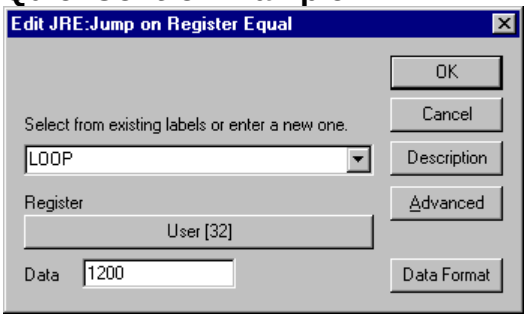
Command	Command Type/Num	Parameters	Param Type	Parameter Range
JRE	Program Class E 137 (0x89) 4 words Thread 1&2	Operation|Register	U16	Operation: = 0 Register: Standard Register Range
		Value	S32	-2,147,483,648 to 2,147,483,647
		Program Buffer Adr	U16	0 to Program Buffer Size

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



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)

Response
ACK only

QuickControl Example
Immediate (Host) Mode Command Only

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

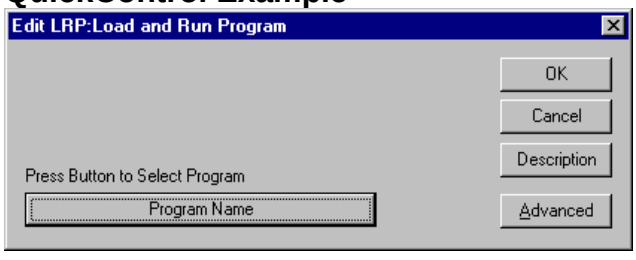
Command	Command Type/Num	Parameters	Param Type	Parameter Range
LRP	Program Class D 156 (0x9C) 2 words Thread 1&2	NV Memory Address	U16	NV Memory Range

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 SN n/a SD 04	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

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 SN n/a SD 27	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

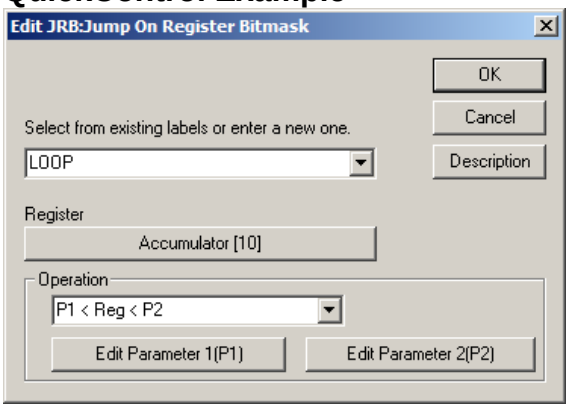
Jump to Program Buffer location 10 if Register # 10 > 0 and < 5.

@16 88 4 10 0 5 10 (CR)

Response

ACK only

QuickControl Example



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

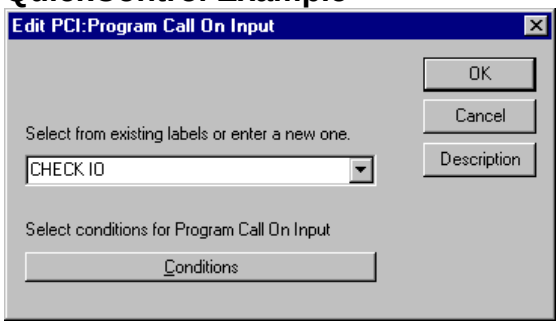
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PCI	Program Class D 201 (0xC9) 4 words Thread 1&2	Enable Code	S16	0 or -14 (-116 with extended I/O)
		Enable State	S16	0 to 1
		Program Buffer Address	U16	0 to Program Buffer Size

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.

Only one PCL can be executed at one time (no nested routines). If a second PCL is executed before a Program Return 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).

SilverDust (SD05) provides 8 levels of stack. Note that the stack is cleared whenever a LRP (Load and Run Program) 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

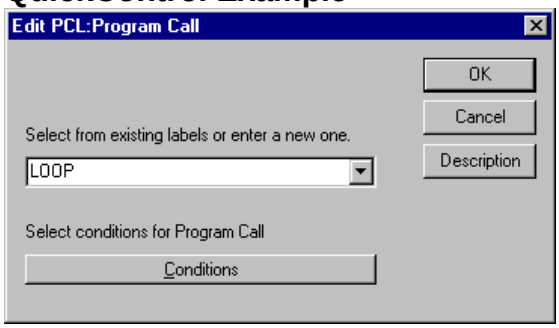
Call Program Buffer location #50 if digital input #1 is High “1”.

@16 201 50 16 16 (CR)

Response

ACK only

QuickControl Example



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
PR)	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

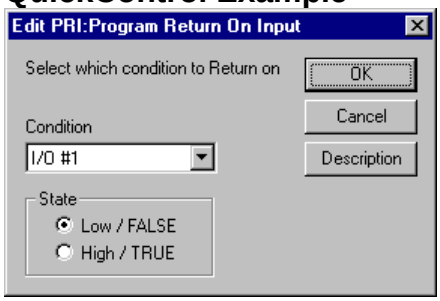
Return from Call if the Input #1 is Low (“0”).

@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

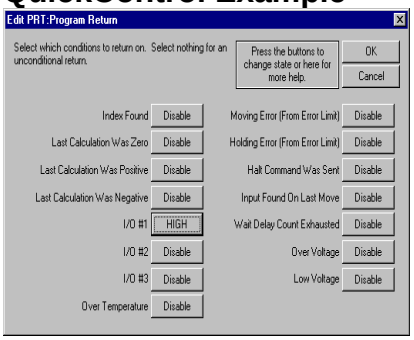
Return from Call if the last I/O #1 is High.

@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

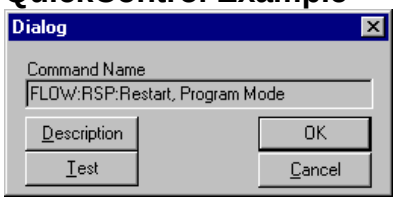
Command	Command Type/Num	Parameters	Param Type	Parameter Range
RSP SN n/a SD 06	Program Class D 255 (0xFF) 1 word Thread 1&2	NONE	NONE	NONE

Example
Restart the Processor.

@16 255 (CR)

Response
NONE

QuickControl Example



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)

Response
ACK only

QuickControl Example
Immediate (Host) Command Only

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)

Response
ACK only

QuickControl Example
Immediate (Host) Command 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

QuickControl Example
Immediate (Host) Command Only

T1F:Thread 1 Force LRP

See Also: T2S:Thread 2 Start

Description
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.

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

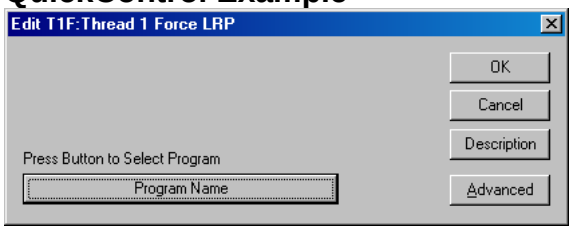
Command	Command Type/Num	Parameters	Param Type	Parameter Range
T1F SN n/a SD 25	Program Class D 75 (0x4B) 2 words Thread 2	NV Memory Address	U16	Start of Program in Non-Volatile memory.

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

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).

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 2 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

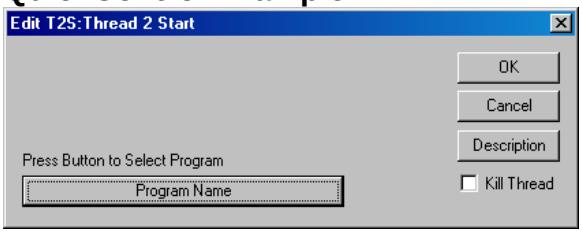
Command	Command Type/Num	Parameters	Param Type	Parameter Range
T2S SN n/a SD 25	Program Class D 76 (0x4C) 3 words Thread 1	NV Memory Address.	U16	Start of program in Non-Volatile Memory.
		Program Buffer Size	U16	0 (Kills Thread 2) 1 to 511 (up to half of Thread 1 Program Buffer) ** SD 38 increases the 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)

QuickControl Example



Response

ACK only

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 a condition is true. 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.

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 14 (116 with extended I/O)
		Enable State	S16	0 = falling (High to Low) 1 = rising (Low to High)

Example

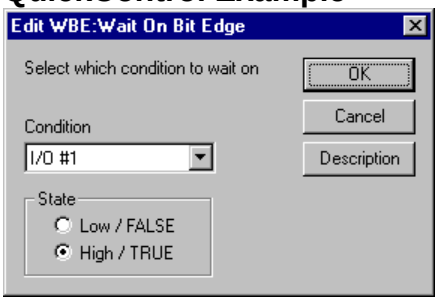
Cause program to wait until I/O #1 goes from Low to High.

@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.

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 14 (116 with extended I/O)
		Enable State	S16	0 = “Low” 1 = “High”

Example

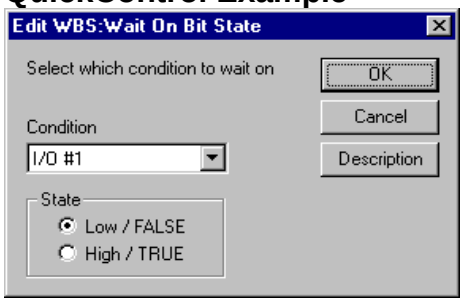
Cause program to wait until I/O #1 is Low.

@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

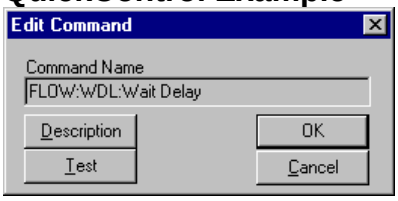
Command	Command Type/Num	Parameters	Param Type	Parameter Range
WDL	Immediate Class D 141 (0x8D) 1 word Thread 1&2*	NONE	NONE	NONE

*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)

QuickControl Example



Response
ACK only

I/O Commands

Commands to read inputs and set outputs.

ACR:Analog Continuous Read

See Also: ARI:Analog Read Input

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]	U16	See above
		Data Register	U16	Standard Register Range

Example

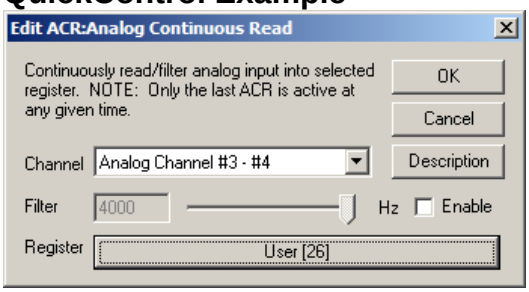
Configure Analog input #4 to do a continuous read to Data Register #26.

@16 207 4 26 (CR)

Response

ACK only

QuickControl Example



ARI:Analog Read Input

See Also: ACR:Analog Continuous Read

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 #	U16	See above.
		Data Register	U16	Standard Register Range

Example

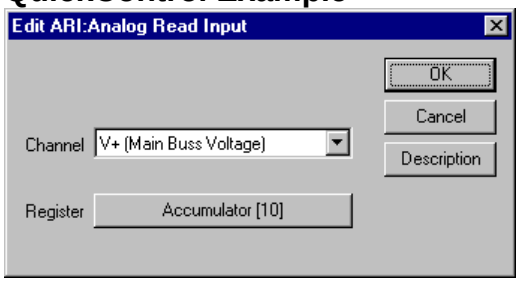
Read into data register #10 the V+ input voltage.

@16 193 7 10 (CR)

Response

ACK only

QuickControl Example



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.

This command is only available on the SilverDust.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CII SN n/a SD 05	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) *

Example

Set I/O 1 high:

@16 31 1 1 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

ACK only

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 shutdown 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.)

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)

Example

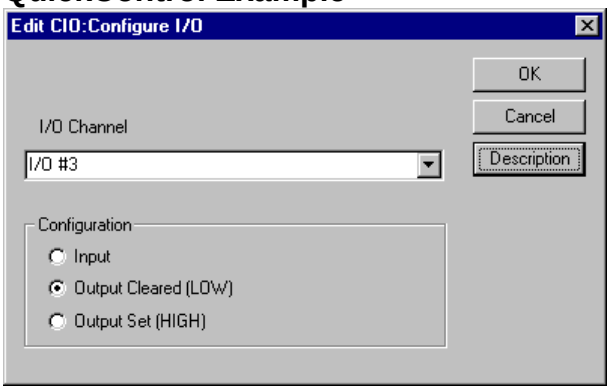
Set I/O bit #3 as output “Low”.

@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

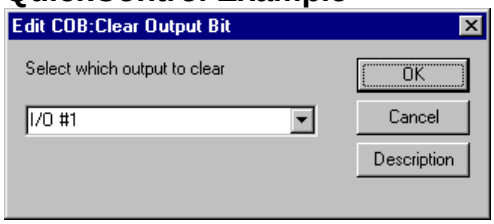
Command	Command Type/Num	Parameters	Param Type	Parameter Range
COB	Program Class D 206 (0xCE) 2 words Thread 1&2	I/O Line #	S16	I/O Line #

Example
Clear I/O bit #1 to a low (“0”) state.

@16 206 1 (CR)

Response
ACK only

QuickControl Example



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

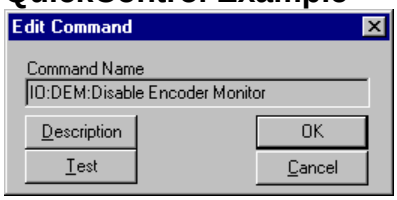
Command	Command Type/Num	Parameters	Param Type	Parameter Range
DEM SN all SD n/a	Program Class D 171 (0xAB) 1 word	NONE	NONE	NONE

Example

Turn off monitoring of the Internal Encoder.

@16 171 (CR)

QuickControl Example



Response

ACK only

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.

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

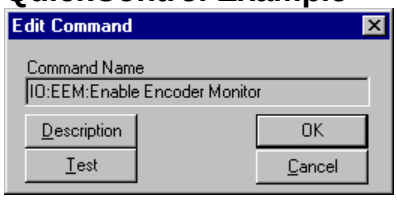
Command	Command Type/Num	Parameters	Param Type	Parameter Range
EEM SN all SD n/a	Program Class D 170 (0xAA) 1 word	NONE	NONE	NONE

Example

Turn on monitoring of the Internal Encoder.

@16 170 (CR)

QuickControl Example



Response

ACK only

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, SilverDust MG, SilverDust IG and SilverDust IGB.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
EMN SN n/a 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

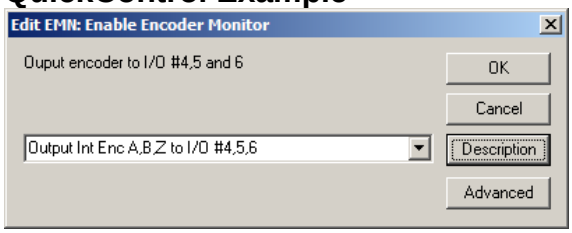
Turn on monitoring of the Internal Encoder.

@16 192 2 0 0 (CR)

Response

ACK only

QuickControl Example



MDC:Modulo Clear

See Also: MDS:Modulo Set, MDT:Modulo Trigger

Description

The Modulo Clear takes the device out of modulo output mode and frees up Digital I/O bits #6 & #7 for normal usage.

Note: The SilverDust accepts this command (to provide back compatibility with previous initialization files), but this command does not affect operation, as the modulo outputs are not available in the SilverDust units (and therefore do not need to be cleared to free up the I/O lines).

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MDC SN all SD n/a	Program Class D 190 (0xBE) 1 word	NONE	NONE	NONE

Example

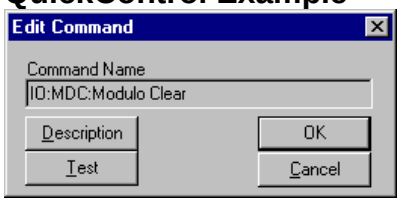
Turn off modulo output.

@16 190 (CR)

Response

ACK only

QuickControl Example



MDS:Modulo Set

See Also: MDC:Modulo Clear,MDT:Modulo Trigger

Description

The Modulo Set command is used to enable, select the source, divisor, and format of the modulo outputs. These outputs make use Digital I/O bits #6 & #7 (See I/O Configuration Chart), and require that these bits to be configured as Inputs before issuing this command.

Either the internal or an external encoder may be selected as the source for modulo output. This allows the modulo counter to be used with external sources.

The output can be configured in three different ways: 1) A/B quadrature, which is the normal output mode for an encoder, 2) Step up & Step Down, which gives a square wave output on I/O bit #6 when the encoder is counting positive or on I/O bit #7 when the encoder is counting negative, and 3) Step and Direction, which gives a square wave output on I/O bit #6 and a Direction output on I/O bit #7.

The Count sets up a divider that is the Modulo Count. In Step output modes, the square wave rate equals the encoder rate divided by the count.

To exit this mode, use the Modulo Clear (MDC) command. See Using I/O in the User Manual for more information on modulo usage.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MDS SN all SD n/a	Program Class D 189 (0xBD) 4 words	Count	S16	1 to 32 (External) (1 to 8 on 34HC)
		Encoder Source	S16	0 = Internal 1 = External
		Output Format	U16	0 = A/B Quad 1 = Step Up/Dn 2 = Step & Dir

Example

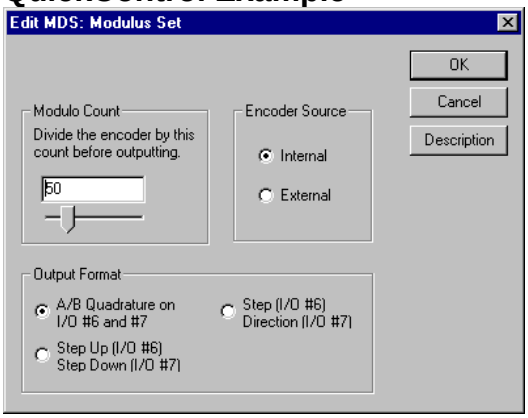
Divide internal encoder counts by 50 and output in A/B Quadrate format.

@16 189 50 0 (CR)

Response

ACK only

QuickControl Example



MDT:Modulo Trigger

See Also: MDS: Modulo Set, MDC:Modulo Clear

Description

The Modulo Trigger allows digital I/O #1 to act as a gating or triggering signal.

Trigger mode #0 disables modulo output until I/O #1 goes from logic Low (“0”) to High (“1”). Mode #1 enables modulo output for continuous operation. Mode #2 will gate the modulo output whenever I/O #1 is high (“1”).

Mode #0 is edge triggered and can be used to as a one shot trigger. Mode #2 is used as a one shot reset for mode #0. Mode #0 is set up by first going into mode #2. If I/O #1 is low, this will disable modulo output, which resets the one shot. Mode #0 can now be set and will wait until I/O #1 goes from Low to High before enabling modulo output.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
MDT SN all SD n/a	Program Class D 191 (0xBF) 2 words	Trigger Mode	S16	**0** = Disable until I/O #1 is High **1** = Enable **2** = Gate modulo using I/O #1

Example

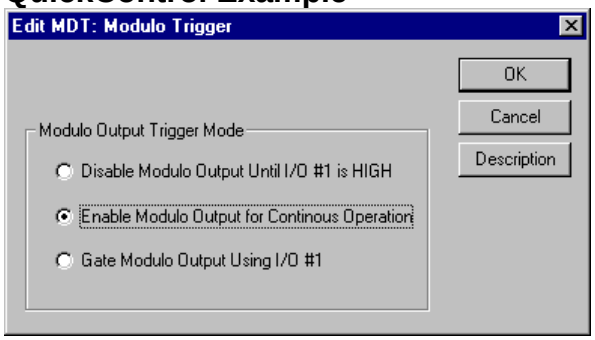
Enable continuous modulo output.

@16 191 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

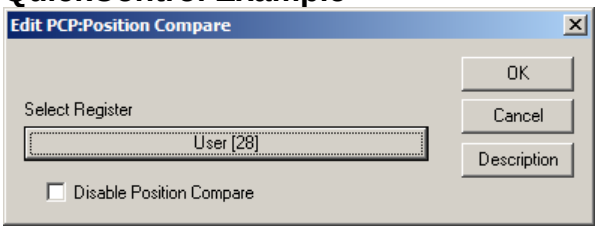
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PCP	Program Class D 245 (0xBF) 2 words Thread 1&2	Data Register	U16	0 = Disable Usage Standard Register Range

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, PCP:Position Compare

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 SN n/a SD 15	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.

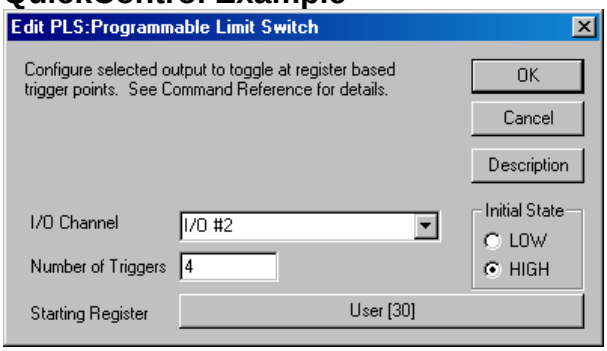
Example

@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 SN n/a SD 15	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.

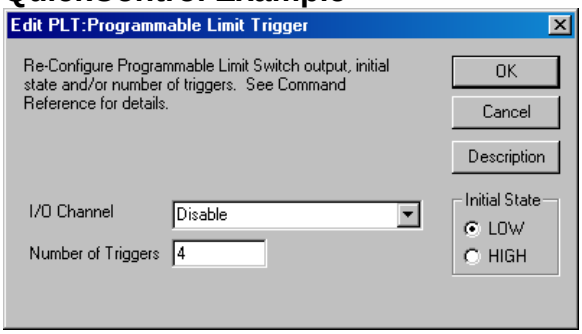
Example

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

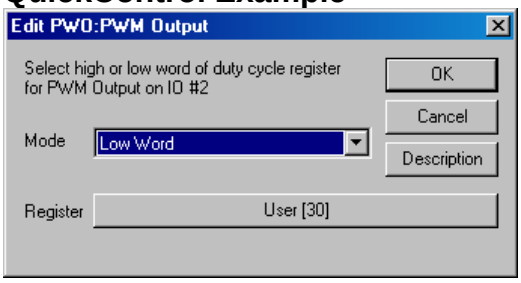
Command	Command Type/Num	Parameters	Param Type	Parameter Range
PWO SN n/a SD 05	Program Class D 129 (0x81) 3 words Thread 1&2	Register	U16	Standard Register Range
		Mode	S16	0 = Disable 1 = High Word 2 = Low Word

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

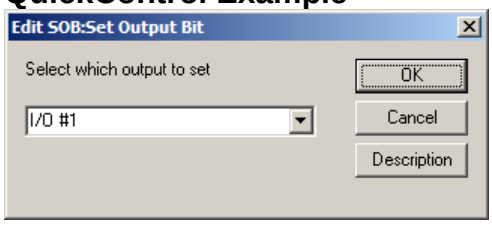
Command	Command Type/Num	Parameters	Param Type	Parameter Range
SOB	Program Class D 205 (0xCD) 2 words Thread 1&2	I/O Line #	S16	I/O Line #

Example
Set I/O bit #1 to a High (“1”) state.

@16 205 1 (CR)

Response
ACK only

QuickControl Example



Data Register Commands

A number of 32-bit Data Registers have been allocated for special purposes. Data Registers are used as data storage locations that may be used and modified by a Host controller or by the device internal functions. They provide data storage for the distance and position parameters for Register motion profile commands. They can also be used by the Calculation commands as data variables for more complex calculations, such as shortest paths for rotary motions. The Input Modes use data registers for Offset and Scaling factors.

Some of the special registers are split into upper and lower words, having unrelated functions, calculation functions are provide to allow modification to just the wanted word.

See User Manual appendix for Data Register definitions.

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.

Data Register #10 is typically used as an accumulator but may also be the Selected Data Register.

The 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. MOD takes the same parameters, but returns the standard (positive) modulo value.

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).

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)

Firmware Rev.	Code	Operation
all	0	Clear (Reg=0)
all	1	Add (Acc = Acc + Reg)
all	2	Sub (Acc = Acc - Reg)
all	3	Copy (Acc = Reg)
all	4	Increment (Reg = Reg +1)
all	5	Decrement (Reg = Reg -1)
all	6	Absolute Value (Reg = ABS(Reg))
all	7	Sub Target Position (Targ-Reg, Pos-Reg) Subtracts the register from both the Target and Position regs
all	8	Copy (Reg = Acc)
all	9	Copy Word, Sign Extend (Acc = HI(Reg)) Loads the high word of register(sign extend) into register #10.
all	10	Copy Word, Sign Extend (Acc = LO(Reg)) Loads the low word of register(sign extend) into register #10.
all	11	AND (Acc = Acc AND Reg)
all	12	OR (Acc = Acc OR Reg)
all	13	XOR (Acc = Acc XOR Reg)
all	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
all	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.
all	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.
all	17	Acc H-Word = LO(Reg) Replace the upper word of Register #10 with the low word of the selected register
all	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).
all	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).
all	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.
all	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
all	22	Shift Reg Left Performs a 32 bit Left Shift of the selected Register.
all	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.
all	24	Shift Reg Right w/o Sign Extend Performs a 32 bit right shift of the selected Register. Implements an unsigned divide by 2.
all	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.
SN n/a SD 05	26	Max (Acc = Max(Acc,Reg)) The larger of the values of Register 10 and the selected Register are stored to Register 10
SN n/a SD 05	27	Min (Acc = Min(Acc,Reg)) The smaller of the values of Register 10 and the selected Register are stored to Register 10
SN n/a SD 05	28	Sub (Acc = Reg - Acc) Note: Sames as code 2 but with parameters swapped.
SN n/a SD 05	29	Negative (Reg = -Reg)
SN n/a SD 06	30	Add (Acc = HI(Reg) + Acc) Take the high word of the selected register, sign extend it and add it to Register 10
SN n/a SD 06	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.

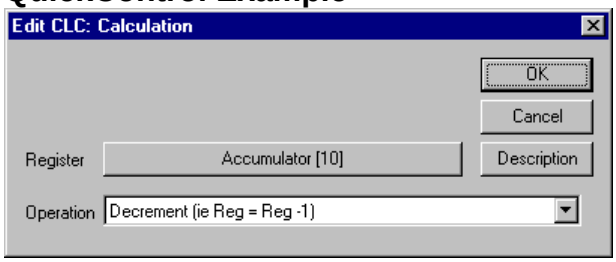
Example
Decrement Accumulator

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 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.

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

SD 30	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
SD 30	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
SD 30	27	Queue-Cmd Err Pop Head (Param 1:Head => Result) Pops Head of Param 1 queue to Result Command Error occurs if queue underflows
SD 30	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
SD 30	29	Queue-Cmd Err Pop Tail (Param 1:Tail => Result) Pops Tail of Param 1 queue to Result Command Error occurs if queue underflows
SD 30	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
SD 30	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
SD 30	32	Queue Pop Head (Param 1:Head => Result) Pops Head of Param 1 queue to Result Zero Flag=Success, Neg Flag=Underflow
SD 30	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
SD 30	34	Queue Pop Tail (Param 1:Tail => Result) Pops Tail of Param 1 queue to Result Zero Flag=Success, Neg Flag=Underflow
SD 30	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
SD 34	100	Mult - Unsigned 64Bit (Result(U64)= Param 1 * Param 2) 64 bit result stored in two consecutive registers.
SD 34	101	Dev - U64/U32 (Result = Param 1(U64)/Param 2) 64 bit numerator stored in two consecutive registers.

Command Info

Command	Command Type/Num	Parameters	Param Type	Parameter Range
CLD SN n/a SD 05	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

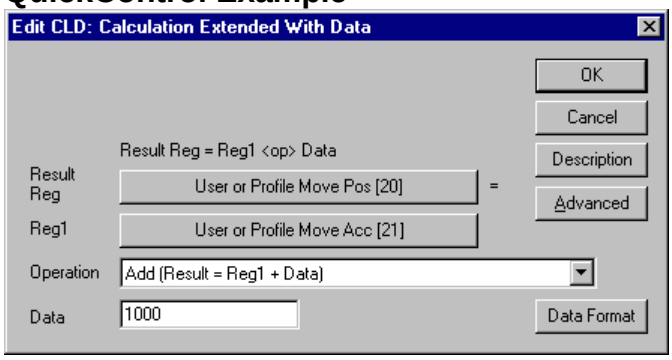
*Thread 2 maintains its own copy of the Accumulator (Register 10) and the zero/positive/negative bits. See Multi-Thread in User Manual for more details.

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

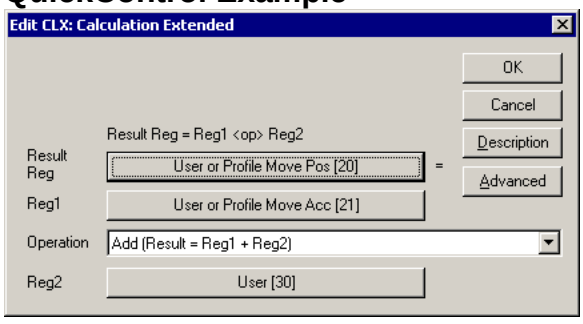
Register 20=Register 21 + Register 30

@16 158 21 30 0 20 (CR)

Response

ACK only

QuickControl Example



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

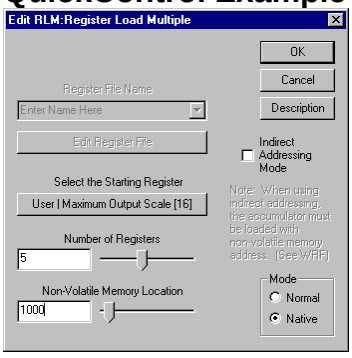
Sequentially Load 5 data registers starting at #16 with the data from NV memory address 1000

@16 197 5 16 1000 (CR)

Response

ACK only

QuickControl Example



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.

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

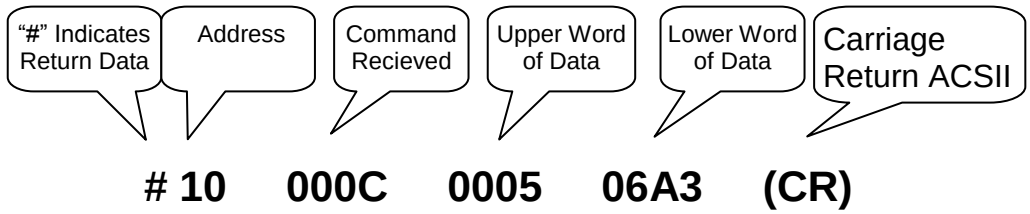
Response
Data Register data

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 SD 05	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
		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 32 3 12 1 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

Assuming register 12 was originally contained zero:

10 0020 0000 0000 (CR)

Following the operation, register 12 would contain a 1.

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

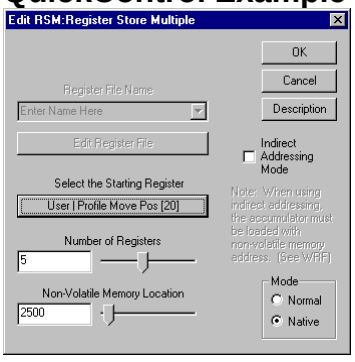
Store 5 data registers starting at #20 to NV memory starting at address 2500

@16 196 5 20 2500 (CR)

Response

ACK only

QuickControl Example



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.

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

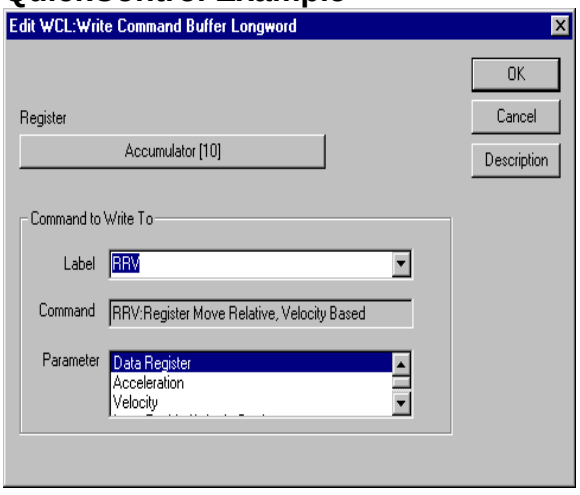
Overwrites the Command Buffer contents at locations 122 and 123 with the contents of Register #10.

@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.

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

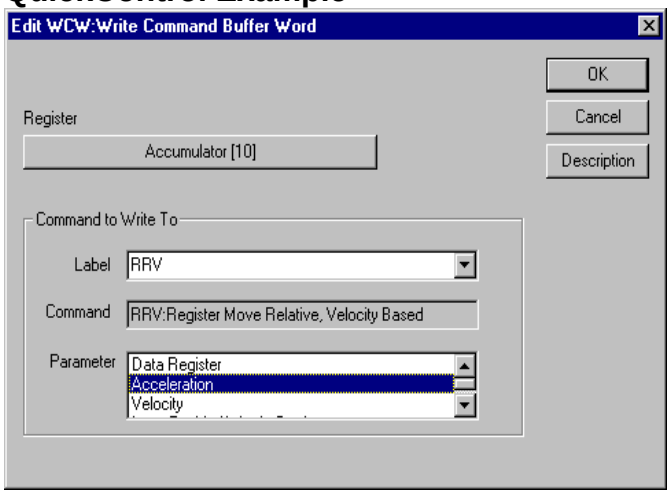
Overwrites the program buffer contents at locations 122 with the contents of the lower word of Register #10.

@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

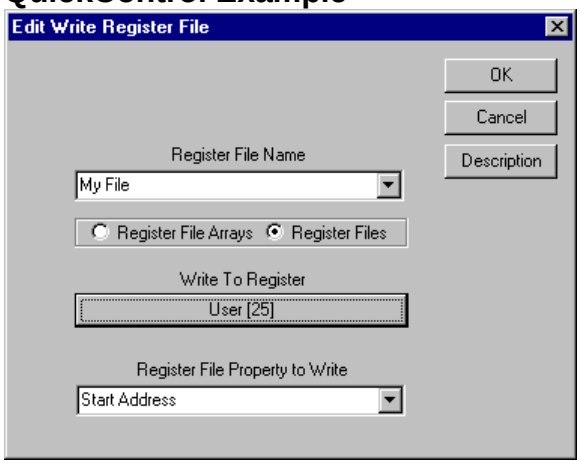
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
WRF	Program Class D 154 (0x9A) 4 words Thread 1&2	Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

Example
Write the number “1” to data register #10.

@16 154 10 1 (CR)

Response
ACK only

QuickControl Example



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

WRP:Write Register, Program Mode

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

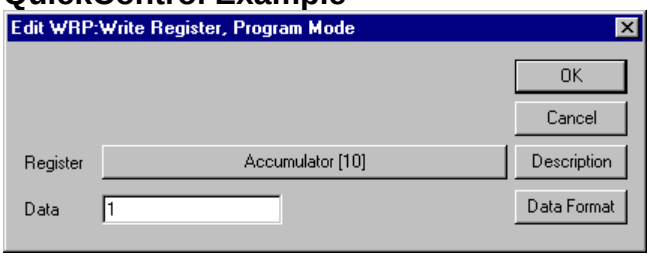
Command Name	Command Type/Num	Parameters	Param Type	Parameter Range
WRP	Program Class D 154 (0x9A) 4 words Thread 1&2	Data Register	U16	Standard Register Range
		Data	S32/U32	0 to 4,294,967,295 or -2,147,483,648 to +2,147,483,647

Example
Write the number “1” to data register #10.

@16 154 10 1 (CR)

Response
ACK only

QuickControl Example



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 SD 05	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
		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 32 3 12 1 (CR)

QuickControl Example

Immediate (Host) Command Only

Response

ACK only

Misc. Commands

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) in the SilverDust units. (See User Manual for more details.)

Command Info

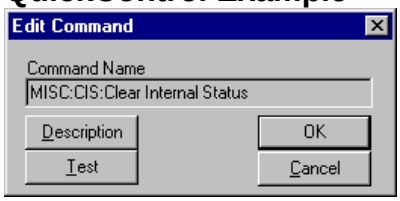
Command	Command Type/Num	Parameters	Param Type	Parameter Range
CIS	Program Class D 163 (0xA3) 1 word Thread 1&2	NONE	NONE	NONE

Example
Clear the Internal Status Word.

@16 163 (CR)

Response
ACK only

QuickControl Example



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

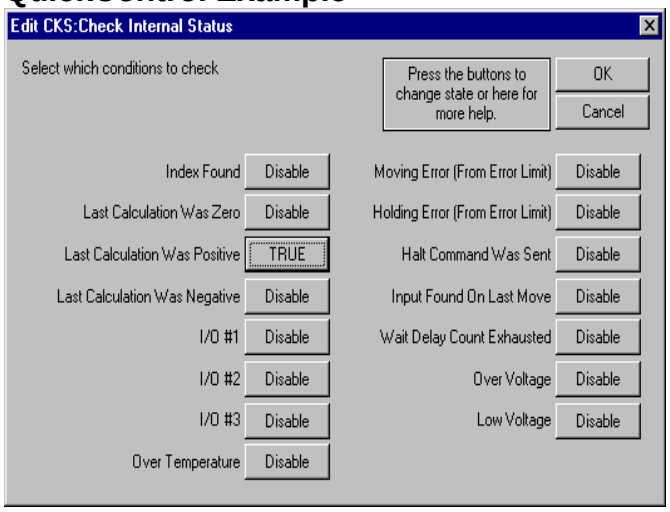
Command	Command Type/Num	Parameters	Param Type	Parameter Range
CKS	Program Class D 164 (0xA4) 3 words Thread 1&2	Condition Enable	U16	0 to 65535
		Condition State	U16	0 to 65535

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

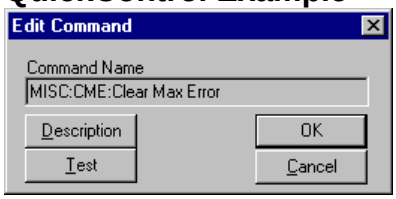
Command	Command Type/Num	Parameters	Param Type	Parameter Range
CME	Program Class D 147 (0x93) 1 word Thread 1&2	NONE	NONE	NONE

Example
Clear the Maximum Error value.

@16 147 (CR)

Response
ACK only

QuickControl Example



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

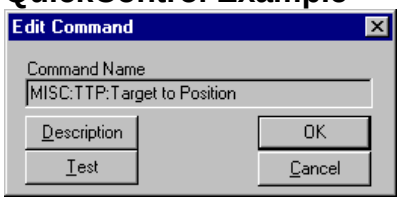
Sets the Target to the current position.

@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 zeros the Target register. Before doing this, the current Position Error (Target – Position) is copied into the Position Register. 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 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

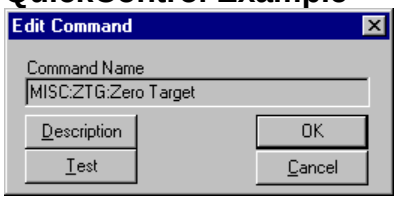
Sets the Target to zero (“0”) does not clear the position error.

@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 the Target register and the Position register. This command zeros out both registers and 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

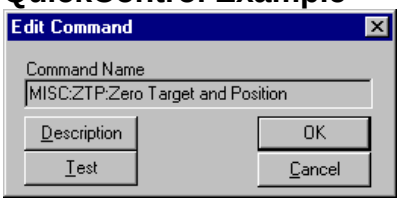
Sets the Target & Position to zero (“0”).

@16 145 (CR)

Response

ACK only

QuickControl Example



CANopen® Commands

CANopen® Commands are for use with devices, both QuickSilver and 3rd party, that support the CANopen standard. The description and use of these commands is beyond the scope of this document. Please see the SilverLode CANopen User Manual for use of these commands.

CANopen® and CiA® are registered community trade marks of CAN in Automation e.V.

CBD:CAN Baud Rate

CCTR:CAN Connect to Remote

CDL:CAN Dictionary Access, Local

CDR:CAN Dictionary Access, Remote

CID:CAN Identity

CRML:CAN Register Map, Local

CRMR:CAN Register Map, Remote

CNL:CAN Set NMT State, Local

CNR:CAN Set NMT State, Remote

CTRL:CAN Transmit Register, Local

CTRR:CAN Transmit Register, Remote

See "SilverLode CANopen User Manual" for details on these commands.

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	(1)
+	CRML	CAN Register Map, Local	(1)
+	CRMR	CAN Register Map, Remote	(1)
+	CTRL	CAN Transmit Register, Local	(1)
+	CTRR	CAN Transmit Register, Remote	(1)
+	DMRM	DMX Register Map	42
0	POL	Poll	20
1	CPL	Clear Poll	19
2	HLT	Halt	85
3	STP	Stop	105
4	RST	Restart	143
5	RVN	Revision	25
6	RPB	Read Program Buffer	24
8	CLP	Clear Program	115
9	SDL	Start Download	145
10	RUN	Run Program	144
11	WRI	Write Register, Immediate Type	186
12	RRG	Read Register	180
13	SPR	Store Program	146
14	LPR	Load Program	134
15	VMI	Velocity Mode, Immediate Mode	108
20	RIS	Read Internal Status Word	23
21	RIO	Read I/O States	22
25	IMW	Interpolated Mode Write Queue	QCI-TD044
27	POR	Poll With Response	21
30	WRX	Write Register Extended	188
31	CII	Configure I/O, Immediate Mode	155
32	RRW	Read Register Write	181
33	PUP	Protect User Program	QCI-TD061
64	ADX	ACK Delay Extended	QCI-TD053
65	CER	Command Error Recovery	33
66	ETP	End Of Travel, Positive	50
67	ETN	End Of Travel, Negative	49
68	FL2	Filter Constants 2	52
69	VLL	Velocity Limits	82
70	CT2	Control Constants 2	36

Sorted By TLA

Cmd Num	Acronym (TLA)	Command Name	Reference: Page Number or Tech Doc
207	ACR	Analog Continuous Read	153
173	ADL	ACK Delay	27
64	ADX	ACK Delay Extended	QCI-TD053
150	AHC	Anti-Hunt Constants	29
230	AHD	Anti-Hunt Delay	30
219	AHM	Anti-Hunt Mode	31
193	ARI	Analog Read Input	154
248	ATR	Add To Register	QCI-TD019
174	BRT	Baud Rate	32
71	CBD	CAN Baud Rate	(1)
+	CCTR	CAN Connect to Remote	(1)
72	CDL	CAN Dictionary Access, Local	(1)
80	CDR	CAN Dictionary Access, Remote	(1)
65	CER	Command Error Recovery	33
73	CID	CAN Identity	(1)
31	CII	Configure I/O, Immediate Mode	155
188	CIO	Configure I/O	156
163	CIS	Clear Internal Status	190
164	CKS	Check Internal Status	191
165	CLC	Calculation	170
200	CLD	Calculation Extended with Data	174
166	CLM	Control Loop Mode	34
8	CLP	Clear Program	115
158	CLX	Calculation Extended	178
147	CME	Clear Max Error	192
74	CNL	CAN Set NMT State, Local	(1)
81	CNR	CAN Set NMT State, Remote	(1)
206	COB	Clear Output Bit	157
1	CPL	Clear Poll	19
+	CRML	CAN Register Map, Local	(1)
+	CRMR	CAN Register Map, Remote	(1)
70	CT2	Control Constants 2	36
148	CTC	Control Constants	35
+	CTRL	CAN Transmit Register, Local	(1)
+	CTRR	CAN Transmit Register, Remote	(1)
215	CTW	Calculation Two Word	QCI-TD019
171	DDB	Disable Done Bit	37
171	DEM	Disable Encoder Monitor	158
252	DIF	Digital Input Filter	38
184	DIR	Direction	39
243	DLC	Dual Loop Control	40
140	DLT	Delay In Ticks	117
140	DLY	Delay	116

Index