AMCI by IDEC CORPORATION

ISMD23E2 Series

Stepper Motor + Drive + Controller

with Integral 2-Port Ethernet Switch

User's Manual

AMCI Motion Control for IDEC FC6A Series MicroSmart

940-0I151

TABLE OF CONTENTS

Notes

About this Manual

Introduction

Read this chapter to learn how to navigate through this manual and familiarize yourself with the conventions used in it. The last section of this chapter highlights the manual's remaining chapters and their target audience.

Audience

This manual explains the installation, and operation of the ISMD23E2 Integrated Stepper Motor + Drive + Controller. These devices are designed by Advanced Micro Controls Inc. for IDEC corporation. This manual is written for the engineer responsible for incorporating these products into a design as well as the engineer or technician responsible for their actual installation.

These devices support network communications using Modbus TCP and are compatible with all IDEC controllers that support this protocol. Each unit has two network connectors that are connected through a two port Ethernet switch. This arrangement simplifies network wiring.

Applicable Units

This manual applies to all of the units in the ISMD23E2 family.

Model Number	Description
ISMD23E2-130-M12	Size 23 motor, 130 oz-in holding torque with sealed M12 connectors, with an IP50 rating.
ISMD23E2-240-M12	Size 23 motor, 240 oz-in holding torque with sealed M12 connectors, with an IP50 rating.
ISMD23E2-130A-M12	Same as the ISMD23E2-130-M12 with an integrated absolute multi-turn encoder.
ISMD23E2-240A-M12	Same as the ISMD23E2-240-M12 with an integrated absolute multi-turn encoder.
ISMD23E2-130E-M12	Same as the ISMD23E2-130-M12 with an integrated incremental encoder.
ISMD23E2-240E-M12	Same as the ISMD23E2-240-M12 with an integrated incremental encoder.
ISMD23E2-130-M12S	Size 23 motor, 130 oz-in holding torque with a shaft seal, sealed M12 connectors, and an IP64 rating.
ISMD23E2-240-M12S	Size 23 motor, 240 oz-in holding torque with a shaft seal, sealed M12 connectors, and an IP64 rating.
ISMD23E2-130A-M12S	Same as the ISMD23E2-130-M12S with an integrated absolute multi-turn encoder.
ISMD23E2-240A-M12S	Same as the ISMD23E2-240-M12S with an integrated absolute multi-turn encoder.
ISMD23E2-130E-M12S	Same as the ISMD23E2-130-M12S with an integrated incremental encoder.
ISMD23E2-240E-M12S	Same as the ISMD23E2-240-M12S with an integrated incremental encoder.

Part Number Description

Navigating this Manual

This manual is designed to be used in both printed and on-line forms. Its on-line form is a PDF document, which requires Adobe Acrobat Reader version 7.0+ to open it. You are allowed to select and copy sections for use in other documents and add notes and annotations. If you decide to print out this manual, all sections contain an even number of pages which allows you to easily print out a single chapter on a duplex (two-sided) printer.

Manual Conventions

Three icons are used to highlight important information in the manual:

Note► Notes highlight important concepts, decisions you must make, or the implications of those decisions.

⚠ Caution Cautions tell you when equipment may be damaged if the procedure is not followed properly.

Warning Warnings tell you when people may be hurt or equipment may be damaged if the procedure is not followed properly.

The following table shows the text formatting conventions:

Format	Description
Normal Font	Font used throughout this manual.
Emphasis Font	Font used for parameter names and the first time a new term is introduced.
Cross Reference	When viewing the PDF version of the manual, clicking on a blue cross reference jumps you to referenced section of the manual.
HTML Reference	When viewing the PDF version of the manual, clicking on a red cross reference opens your default web browser to the referenced section of the AMCI website if you have Internet access.

Trademarks

The AMCI logo is a trademark of Advanced Micro Controls Inc. “FC6A Series MicroSmart” is a trademark of IDEC Corporation. “Adobe” and “Acrobat” are registered trademarks of Adobe Systems Incorporated.

All other trademarks contained herein are the property of their respective holders.

Revision Record

This manual, 940-0I151 is the first release of this manual. It was first released on May 4th, 2017.

Revision History

940-0I151: May 4th, 2017 - Initial Release

Manual Layout

You will most likely read this manual for one of two reasons:

- If you are curious about the Integrated Stepper Motor + Drive + Controller products, this manual contains the information you need to determine if these products are the right products for your application. The first chapter, *ISMD23E2 Specifications* contains all of the information you will need to fully specify the right product for your application.
- If you need to install and use an Integrated Stepper Motor + Drive + Controller product, then the rest of the manual is written for you. To simplify installation and configuration, the rest of the manual is broken down into *references* and *tasks*. Using an Integrated Stepper Motor + Drive + Controller product requires you to complete multiple tasks, and the manual is broken down into sections that explain how to complete each one.

Section Title	Starting Page #	Section Description
ISMD23E2 Specifications	9	Complete specifications for the ISMD23E2 products.
Motion Control	19	Reference information on how the ISMD23E2 can be used to control motion in your application.
Homing an ISMD23E2	29	Reference information on how to set the home position of the ISMD23E2.
Installing the ISMD23E2	33	Task instructions covering how to install an ISMD23E2 on a machine. Includes information on mounting, grounding, and wiring specific to the units.
Set the IP Address	39	Task instructions that covers the options for setting the IP address on an ISMD23E2.
WindLDR 8.5 Project Setup	41	Task instructions for adding an ISMD23E2 to a WindLDR 8.5+ project.
Installing Custom Macros	45	Task instructions for downloading and installing custom macros into a new or existiong WindLDR 8.58.5+ project.
Using a Custom Macro in WindLDR	47	Task instructions for adding a custom macro to a ladder logic program.
ISMD23E2 Input Data Formats	59	Reference information on the format of the input data from the ISMD23E2. Data formats for configuration and command responses are given.
Additional Logic	63	Task guidelines for adding the applications specific logic needed to control an ISMD23E2.
Configure Your Network Interfaces	65	Instructions for the optional task of installing the AMCI by IDEC Ethernet tool software for changing the IP address of an ISMD23E2.
Configure Your Network Interfaces	65	Instructions for the optional task of configuring network interfaces on your computer or laptop.
Configuration Mode Data Format	71	Reference information on the format of configuration data written down to the ISMD23E2. Custom macros simplify the programming of the ISMD23E2, and should be used in most cases. This reference is provided as a troubleshooting aid should the need arise.
Command Mode Data Format	75	Reference information on the format of command data written down to the ISMD23E2. Custom macros simplify the programming of the ISMD23E2, and should be used in most cases. This reference is provided as a troubleshooting aid should the need arise.
Calculating Move Profiles	85	Reference information on calculating detailed move profiles.

Manual Sections

REFERENCE 1: ISMD23E2 SPECIFICATIONS

Introduction

This chapter contains all of the information needed to specify an ISMD23E2 device for your project.

The ISMD23E2 Family

The ISMD23E2 units are part of a growing product line from a partnership between IDEC Corporation and AMCI that brings stepper motor control to the IDEC Corporation industrial control families. Each ISMD23E2 attaches to your Ethernet network and communicates using the Modbus TCP protocol.

Each unit has two Ethernet ports which are internally connected through an onboard, two port, 10/100 Mbps Ethernet switch. These ports allow you to wire your network in a "daisy-chain" fashion, which may lower network wiring costs and complexities.

Each unit can be ordered with an optional incremental or absolute multi-turn encoder. This encoder gives you the additional functionality of position verification and stall detection. The absolute multi-turn encoder allows you to track machine position with power removed, eliminating the need to home the machine after cycling power.

All ISMD23E2 products have sealed M12 connectors for their Ethernet, I/O, and power connections. These units carry an IP50 environmental rating. Units can also be ordered with a shaft seal. These units carry an IP64 rating. These units are protected against liquid splash and condensation, but liquid should not be allowed to pool on the device itself.

Figure R1.1 IP50 Rated SMD23E2

Part Numbering System

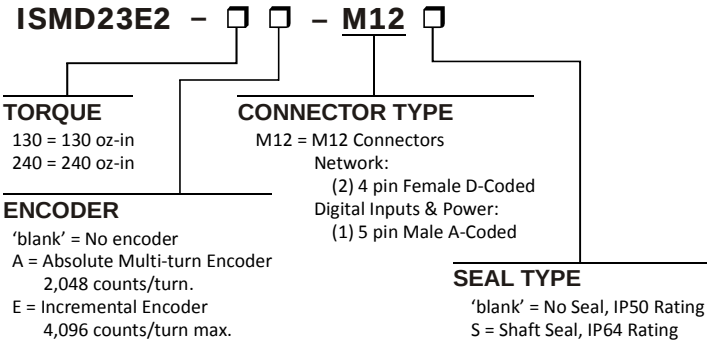


Figure R1.2 Part Numbering System

General Functionality

Each member of the ISMD23E2 family has three integrated parts:

- An indexer that accepts commands over an Ethernet connection using the Modbus TCP protocol
- A 3.4 Arms micro-stepping drive that accepts 24 to 48 Vdc as its input power source
- A high torque size 23 stepper motor (130 or 240 oz-in holding torque).

An incremental or absolute multi-turn encoder is also available for applications that require position feedback or verification.

The availability of user macros for the WindLDR Automation Organizer software makes the ISMD23E2 units easy to integrate into IDEC control systems. This combination of host and drive gives you several advantages:

- Sophisticated I/O processing can be performed in the IDEC controller before sending commands to the ISMD23E2 unit
- All motion logic is programmed in the host, eliminating the need to learn a separate motion control language
- The integral two port Ethernet switch simplifies network cabling
- The elimination of the separate indexer and drive lowers total system cost.

An ISMD23E2 is powered by a nominal 24 to 48 Vdc power source, and can accept surge voltages of up to 60 Vdc without damage. The output motor current is fully programmable from 0.1 Arms to 3.4 Arms which makes the ISMD23E2 suitable to a wide range of applications. In addition to the Motor Current setting, the Motor Steps per Turn, Idle Current Reduction, and Anti-Resonance Circuit features are also programmable.

The ISMD23E2 contains a true RMS motor current control drive. This means that you will always receive the motor's rated torque regardless of the *Motor Steps/Turn* setting. (Drives that control the peak current to the motor experience a 30% decrease in motor torque when microstepping a motor.) The combination of an ultra-low inductance motor and a high-power, true RMS drive gives unprecedented torque vs. speed performance for any DC application.

The ISMD23E2 units have two DC inputs that are used by the indexer. Configuration data from the host sets the function of these inputs. Each input can be individually configured as a:

- CW or CCW Limit Switch
- Home Limit Switch
- Capture Position Input (Will capture encoder position on units with the internal encoder.)
- Stop Jog or Registration Move Input
- Start Indexer Move
- Emergency Stop Input
- General Purpose Input

Encoder Functionality

All ISMD23E2 units can be ordered with an internal incremental or absolute multi-turn encoder. Incremental encoders can be programmed to 1,024, 2,048, or 4,096 counts per turn. Absolute encoders have a fixed resolution of 2,048 counts per turn and encoder a total of 2^{21} turns. (Thirty-two bits total.) Using an encoder gives you the ability to:

- Verify position during or after a move
- Detect motor stall conditions
- Maintain machine position when power is removed if using an absolute encoder.

The motor position can be preset to the encoder position with a single command. ISMD23E2 units with absolute encoders allow you to preset the encoder position and save the resulting offset in Flash memory.

Specifications

Network Interface

10/100baseT. Two switched ports.
Supports Modbus TCP protocol.

Drive Type

Two bipolar MOSFET H-bridges with 20KHz PWM current control.

Physical Dimensions

See page 34

Weight

SMD23E2-130 (All versions) 1.96 lbs. (0.90 kg.)
SMD23E2-240 (All versions) 2.73 lbs. (1.24 kg.)
All weights are without mating connectors

Maximum Shaft Loads

Radial: 19 lbs (85 N) at end of shaft
Axial: 3.37 lbs (15 N)

Maximum Operating Temperature

203°F/95°C (Note that this is the operating temperature of the motor, not maximum ambient temperature. An Over Temperature fault occurs at this point and current is removed from the motor.)

Over Temperature Fault

Over temperature faults are reported in the Network Input Data. Power must be cycled to the unit to clear the fault.

Inputs

Electrical Characteristics:
Single ended sinking.
Accept 3.5 to 27Vdc without the need for an external current limiting resistor.

Motor Current

Programmable from 0.1 to 3.4 Arms in 0.1 Amp steps.

$DCPower_{AUX}$ Current

70 mA @ 24Vdc, 40mA @48Vdc

Motor Counts per Turn

Programmable to any value from 200 to 32,767 steps per revolution.

Internal Encoder (Optional)

Incremental encoder option supplies 1,024, 2,048, or 4,096 counts per turn.

Absolute encoder option supplies 2,048 counts per turn, 32 bit max. counts.

Idle Current Reduction

Programmable from 0% to 100% of programmed motor current in 1% increments. Motor current is reduced to selected level if there is no motion for 1.5 seconds. Current is restored to full value when motion is started.

Environmental Specifications

Input Power 24 to 48 Vdc, surge to 60 Vdc without damage to unit.

Ambient Operating Temperature:
-40° to 122°F (-40° to 50°C)

Storage Temperature:
-40° to 185°F (-40° to 85°C)

Humidity: 0 to 95%, non-condensing

IP Rating:
IP50 without shaft seal
IP64 with shaft seal

Status LED's

See *Status LED's* section starting on page 15.

Connectors and Cables

All mating connectors are available separately under the following part numbers.

Connector	Part #	Wire	Strip Length	Connection Type
Ethernet	IMS-28	18 AWG max.	0.197 inches	Screw Terminals
Power & Digital Inputs	IMS-31	18 AWG max.	0.197 inches	Screw Terminals

Cable	Part #	Length
Ethernet	ICNER-5M	5 meter
Power & Digital Inputs	ICNPL-5M	5 meter

Controller Functionality

The table below lists the functionality offered by the indexer built into the ISMD23E2 units.

Feature	Description
Modbus TCP Protocol	The ISMD23E2 units communicate through the Modbus TCP protocol. This allows easy setup and communication with a wide range of IDEC host controllers.
Programmable Inputs	Each of the inputs can be programmed as a Home Limit, Over Travel Limit, Capture Input, Stop Jog or Registration Move, Start Indexer Move, E-Stop, or a General Purpose Input.
Programmable Parameters	Starting Speed, Running Speed, Acceleration, Deceleration, Accel/Decel Types, and Motor Current are fully programmable.
Homing	Allows you to set the machine to a known position. The motor and encoder positions can be preset with a network command or the ISMD23E2 can home to a discrete input.
Jog Move	Allows you to drive the motor in either direction as long as the command is active.
Relative Move	Allows you to drive the motor a specific number of steps in either direction from the current location.
Absolute Move	Allows you to drive the motor from one known location to another known location.
Registration Move	Allows you to jog the motor in either direction based on a command from your IDEC host controller. When a controlled stop is issued, the move will output a programmable number of steps before coming to a stop.
Blend Move†	Allows you to perform a sequence of relative moves without stopping between them. This feature is used to create highly accurate move profiles that avoid network latency issues.
Dwell Move†	Allows you to perform a sequence of relative moves with a stop between each move that has a programmable length of time. This feature is used to create highly accurate move profiles that avoid network latency issues.
Indexer Move	Allows you to program a move that does not start until one of the programmable inputs makes a transition.
Hold Move	Allows you to suspend a move, and optionally restart it, without losing your position value.
Resume Move	Allows you to restart a previously held move operation.
Immediate Stop	Allows you to immediately stop all motion if an error condition is detected by your host controller.
Stall Detection	When an ISMD23E2 is purchased with an encoder option, the encoder can be used to verify motion when a move command is issued.

† This functionality is not used in many applications, and is not programmable through custom macros supplied by IDEC. A full description of this functionality is available in the *Assembled Moves* section of this manual, starting on page 67.

Table R1.1 Controller Functionality

Drive Functionality

This table summarizes the features of the stepper motor drive portion of the ISMD23E2 units.

Feature	Benefits
RMS Current Control	RMS current control give an ISMD23E2 the ability to drive the motor at its fully rated power regardless of the programmed steps per turn. There is no reduction in power when microstepping that may occur with other drives.
Programmable Motor Current	RMS current supplied to the motor can be programmed from 0.1 to 3.4 amps in 0.1 amp increments. Reducing the motor current to the minimum needed for your application will significantly reduce the motors operating temperature
Programmable Idle Current Reduction	Extends motor life by reducing the motor current when motion is not occurring. This extends the life of the motor by reducing its operating temperature.
Programmable Motor Steps/Turn	Allows you to scale your motor count to a real world value. (counts per inch, counts per degree, etc.)
Anti-Resonance Circuitry	This feedback circuitry and algorithm gives the ISMD23E2 the ability to modify motor current waveforms to compensate for mechanical resonance in your system. This will give you smooth performance over the entire speed range of the motor.
Over Temperature Detection	An ISMD23E2 sets a warning bit in the network data when the internal temperature of the unit approaches its safe operating threshold.
Over Temperature Protection	Protects your ISMD23E2 from damage by removing power from the motor if the internal temperature of the drive exceeds the safe operating threshold of 203°F/95°C.

Table R1.2 Drive Functionality

Idle Current Reduction

Idle Current Reduction allows you to prolong the life of your motor by reducing its idling temperature. Values for this parameter range from 0% (no holding torque when idle) to 100%.

Idle current reduction should be used whenever possible. By reducing the current, you are reducing the I^2R losses in the motor. Therefore, the temperature drop in the motor is exponential, not linear. This means that even a small reduction in the idle current can have a large effect on the temperature of the motor.

Note➤ Note that the reduction values are “to” values, not “by” values. Setting a motor current to 2 Arms and the current reduction to 25% will result in an idle current of 0.5 Apk. (The ISMD23E2 always switches from RMS to peak current control when the motor is idle to prevent motor damage due to excessive heating.)

Optional Encoder

The ISMD23E2 can be ordered with an integral encoder. The encoder is typically used for position verification and stall detection. Additionally, an input can be configured to capture the encoder value when the input makes an inactive to active transition. This captured value is written to the host controller. Two encoder options are available:

Incremental Encoder

The incremental encoder can be programmed to 1,024, 2,048, or 4,096 counts per turn. The ISMD23E2 has an internal thirty-two bit counter associated with the encoder.

Absolute Multi-turn Encoder

The absolute encoder has a fixed resolution of 2,048 counts per turn. The absolute encoder is a multi-turn device that encodes a total of 2^{21} turns, yielding a full thirty-two bits of position resolution. The absolute encoder can be used for position verification and stall detection, but its primary advantage is that it eliminates the need to home the axis after cycling power to the drive.

Like many intelligent absolute encoders on the market today, the absolute encoder in the ISMD23E2 uses a battery backed circuit to count zero crossings while power is removed from the rest of the device. The circuit will accurately track position as long as the shaft acceleration is limited to 160,000 degrees/sec^2, (444.4 rev/sec^2), or less.

Additional Notes on Stall Detection

When Stall Detection is enabled, the ISMD23E2 monitors the encoder for position changes, regardless of whether or not a move is in progress. If the error between the encoder position and the motor position exceeds forty-five degrees, the ISMD23E2 responds in the following manner:

- The stall is reported in the network input data.

- The motor position becomes invalid. (The machine must be homed or the motor position preset before Absolute moves can be run again.
- If a move was in progress, the move is stopped.

Note that a move does not have to be in progress for stall detection to be useful. As described later in this chapter, there is an auxiliary power pin that powers the electronics of an ISMD23E2 but does not power the motor. The primary use of this feature is to keep the unit connected to the network while power is removed from the motor. When using the $DCPower_{AUX}$ pin, the ISMD23E2 cannot sense when power has been removed from the $DCPower_{MAIN}$ pin. By enabling stall detection, the ISMD23E2 can notify the system if the motor shaft moves more than forty-five degrees while power is removed from the motor.

Available Discrete Inputs

The ISMD23E2 has two discrete DC inputs that accept 3.5 to 27 Vdc signals. (5 to 24 Vdc nominal) How the ISMD23E2 uses these inputs is fully programmable. The active state of each input is also programmable. Programming their active states allow them to act as Normally Open (NO) or Normally Closed (NC) contacts. These inputs are open collector sinking and share their common connection with the common of the main and auxilary power supplies.

Home Input

Many applications require that the machine be brought to a known position before normal operation can begin. This is commonly called “homing” the machine or bringing the machine to its “home” position. An ISMD23E2 allows you to define this starting position in two ways. The first is with a Position Preset command. The second is with a sensor mounted on the machine. When you define one of the inputs as the Home Input, you can issue commands to the ISMD23E2 that will cause the unit to seek this sensor. How the ISMD23E2 actually finds the home sensor is described in the reference chapter *Homing an ISMD23E2* starting on page 29.

CW Limit Switch or CCW Limit Switch

Each input can be defined as a CW or CCW Limit Switch. When used this way, the inputs define the limits of mechanical travel. For example, if you are moving in a clockwise direction and the CW Limit Switch activates, all motion will immediately stop. At this point, you will only be able to jog in the counter-clockwise direction until the sensor deactivates.

Start Indexer Move Input

Indexer Moves are programmed through the Network Data like every other move. The only difference is that Indexer Moves are not run until a Start Indexer Move Input makes a inactive-to-active state transition. This allows an ISMD23E2 to run critically timed moves that cannot be reliably started from the network due to data transfer lags.

If the unit was ordered with the encoder option, and one of the discrete DC inputs is programmed as a Start Indexer Move Input, then the encoder position data will also be captured whenever the DC input makes a transition. An inactive-to-active state transition on the DC input will also trigger an Indexer Move if one is pending.

Emergency Stop Input

When an input is defined as an Emergency Stop, or E-Stop Input, motion will immediately stop when this input becomes active. The drive remains enabled and power is supplied to the motor. Any type of move, including a Jog or Registration Move, cannot begin while this input is active.

Stop Jog or Registration Move Input

When an input is configured as a Stop Jog or Registration Move Input, triggering this input during a Jog Move or Registration Move will bring the move to a controlled stop. The controlled stop is triggered on an inactive-to-active state change on the input. Only Jog Moves and Registration Moves can be stopped this way, all other moves ignore this input.

If the unit was ordered with an integral encoder, the encoder position data will be captured when the DC input makes an inactive-to-active transition if it is configured as a Stop Jog or Registration Move Input. The encoder position data is not captured if a Jog or Registration Move is not in progress. If you want to capture encoder position data on every transition of a DC input, configure it as a Start Indexer Move Input.

Capture Encoder Position Input

As described in the *Start Indexer Move Input* and *Stop Jog or Registration Move Input* sections above, an ISMD23E2 can be configured to capture the encoder position value on a transition of a discrete DC input.

General Purpose Input

If your application does not require one or both of the inputs, you can configure the unused inputs as General Purpose Inputs. The inputs are not used by the ISMD23E2, but their on/off state is reported in the network data and is available to your IDEC controller.

Status LED's

Each ISMD23E2 has two status LED's that show module and network status. As shown in figure R1.3, these LED's are located on the rear cover.

Module Status LED

The Module LED is a bi-color red/green LED that show the general status of the unit.

- **Steady Green:** Unit OK
- **Steady Red:** An Over Temperature Fault exists.
- **Blinking Green:** Successful write to flash memory. Power must be cycled to the unit before additional commands can be written to it.
- **Blinking Red:** Failed write to flash memory. You must cycle power to the unit to clear this fault.
- **Alternating Red/Green:** Communications failure. There is a communications error between the main processor and the ethernet co-processor within the unit. You must cycle power to the ISMD23E2 to attempt to clear this fault.

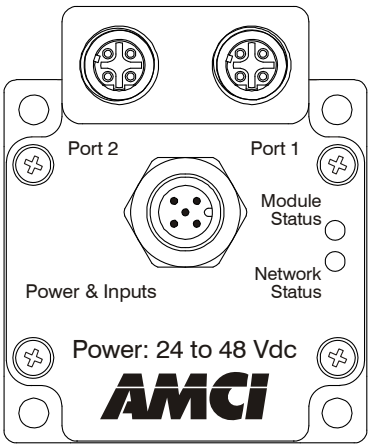


Figure R1.3 ISMD23E2 Rear View

Power Up Behavior

- **Blinking Green:** The unit will blink the Module Status LED green during initialization.
- **Blinking Red:** The unit will blink the Module Status LED red three times if there is an error with the internal absolute encoder.

Network Status (NS) LED

The Network Status LED is a bi-color red/green LED.

LED State	Definition
Off	No power or no Modbus TCP connections
Alternating Red/Green	Power up Self-Test
Flashing Green	Indicates number of concurrent connections with 2 second delay between group. The ISMD23E2 supports up to 3 concurrent connections.
Steady Green	Should not occur. LED should always flash when network is connected.
Steady Red	Duplicate IP address on network.

Table R1.3 Network Status LED States

ISMD23E2 Connectors

Ethernet Connectors

Figure R1.3 above shows the placement of the connectors on an ISMD23E2 unit. Figure R1.4 shows the pinout of the Ethernet connectors when viewed from the back of the ISMD23E2. Each Ethernet port on the ISMD23E2 is an "auto-sense" port that will automatically switch between 10baseT and 100baseT depending on the network equipment it is attached to. Each port also has "auto switch" capability that will switch the direction of the Tx and Rx pairs if necessary. This eliminates the need for cross-over cables in all circumstances. A standard cable can be used when connecting the ISMD23E2 to any device, including a personal computer.

ETHERNET
Ports 1 & 2

Figure R1.4 Ethernet Connector Pinout

The connector is a standard four pin D-coded female M12 connector that is rated to IP67 when the mate is properly installed.

Input Connector

As shown in figure R1.3 on the previous page, the Input Connector is located on the back of the unit near its center. All digital input and power supply connections are made at this connector. The connector is a standard five pin A-coded M12 connector that is rated to IP67 when the mate is properly attached. Figure 1.5 shows the pinout of the connector when viewed from the back of the SMD23E2.

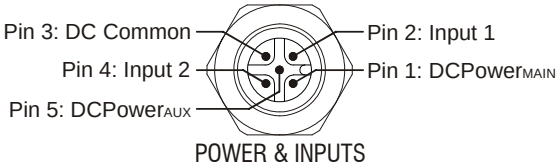


Figure R1.5 M12 Input Connector

The two digital inputs are single ended, sinking inputs and referenced to the DC Common pin.

Both of the inputs accept a nominal 5 Vdc to 24 Vdc signal without the need of a current limiting resistor. Additional information on how the digital inputs can be used can be found in the *Available Discrete Inputs* section of the chapter, starting on page 14.

There are two power pins. $DCPower_{MAIN}$ powers both the control electronics and the motor. $DCPower_{AUX}$ powers only the control electronics. Using the $DCPower_{AUX}$ pin is optional. If your application requires you to cut power to your motor under some conditions, using the $DCPower_{AUX}$ pin allows you to cut power to your motor without losing your network connection.

Note► If the unit was ordered with an encoder, the $DCPower_{AUX}$ pin will also maintain power to the encoder. If the motor shaft is rotated while motor power is removed, the encoder position will update. (The motor position will not update.) Once power is restored to the motor, a Preset Position command can be issued to restore the correct motor position without having to go through a homing sequence. If Stall Detection is enabled on the ISMD23E2, it will also be able to tell the IDEC system if the motor shaft rotated more than forty-five degrees with power removed.

Torque and Power Curves

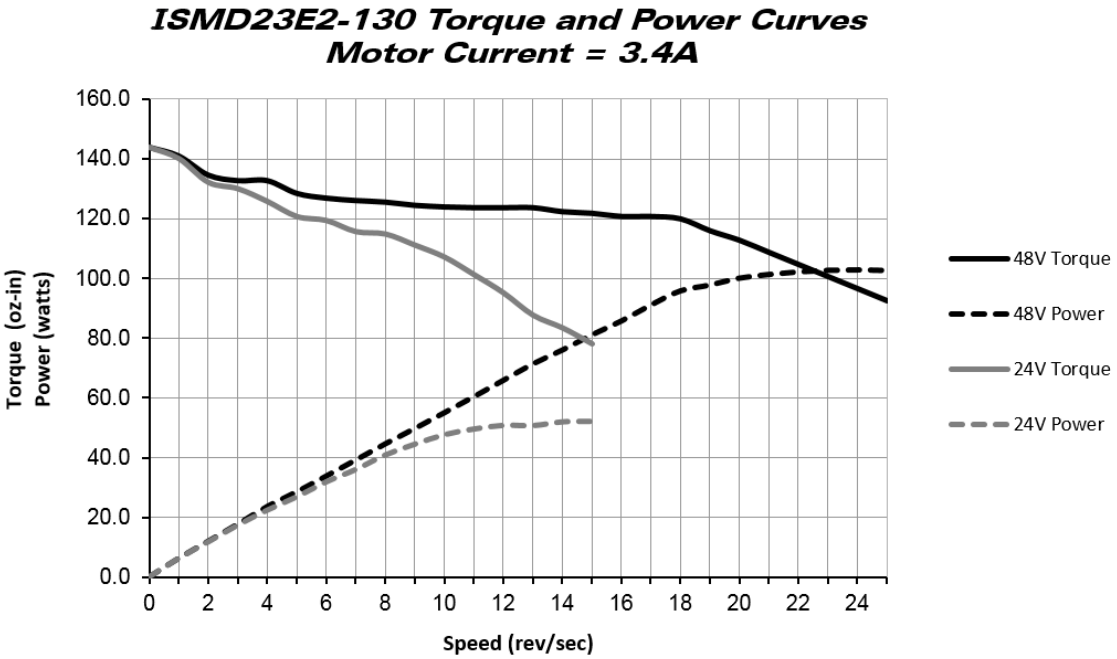


Figure R1.6 ISMD23E-130 Torque and Power Curves

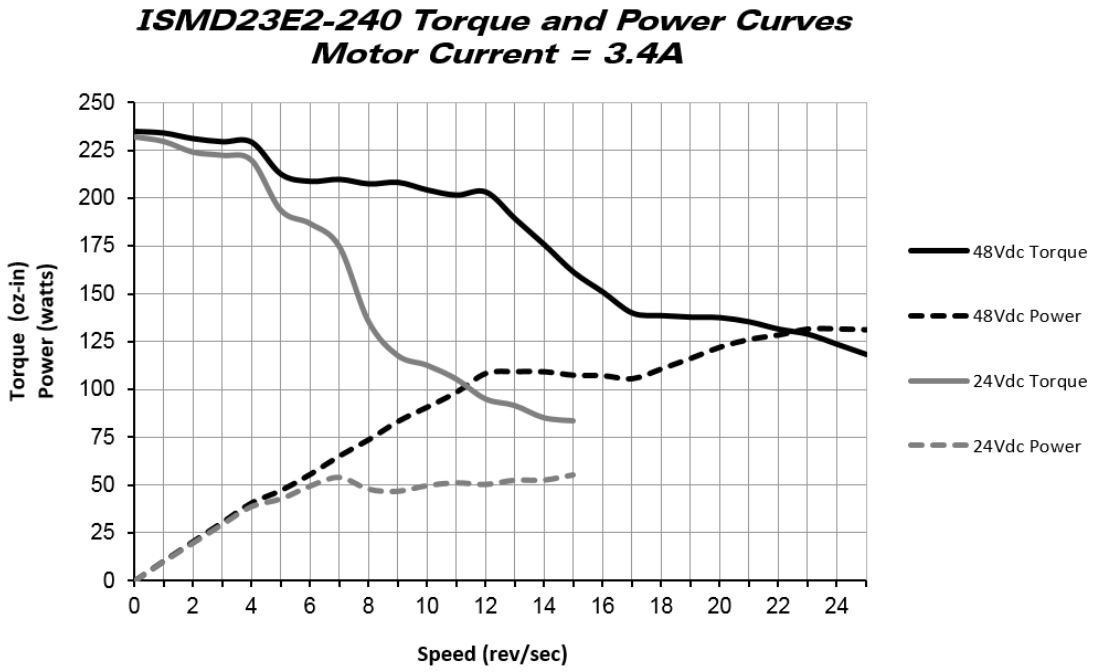


Figure R1.7 ISMD23E-240 Torque and Power Curves

Power Supply Sizing

The power supply can be sized based on the power the motor must generate during its operation. As a general guideline, your supply should be able to produce 150% to 175% of the power the motor can produce. The power and torque curves on the previous page can be used to determine the maximum power the motor can generate over its speed range.

Note that the power value that you should use is the *maximum* power value over the range of speeds that the motor will be operated at. The power generated by the motor may decrease towards the end of its usable speed range. Therefore, the power generated at your machine's operating point may be less than the maximum the motor can generate at a lower speed.

Example 1: An ISMD23E2-130 will be running at a maximum of 20 RPS and a 48 Vdc supply will be used. Based on the power curve in figure R1.6 on the previous page, the power at this speed is 100 watts, which is the maximum power over the entire speed range. Therefore, the 48 Vdc supply should be able to supply 150 to 175 watts of power.

Example 2: An ISMD23E2-240 will be running at a maximum of 9 RPS and a 24 Vdc supply will be used. Based on the power curve in figure R1.7 on this page, the power at this speed is approximately 47 watts, but the maximum power over the entire speed range is 55 watts, which occurs at 7 RPS. Therefore, the 55 watt value should be used, and the 24 Vdc supply should be able to supply 83 to 97 watts of power.

Table below shows the suggested power supply sizes based on the maximum power the motor can generate over its entire speed range.

		ISMD23E2-130			ISMD23E2-240		
		Motor Power	**150% Supply**	**175% Supply**	**Motor Power**	**150% Supply**	**175% Supply**
Supply Voltage	24 Vdc	52 W	78 W	91 W	56 W	84 W	98 W
	48 Vdc	103 W	155 W	180 W	132 W	198 W	231 W

Table R1.4 Suggested Power Supply Ratings

Regeneration (Back EMF) Effects

All motors generate electrical energy when the mechanical speed of the rotor is greater than the speed of the rotating magnetic fields set by the drive. This is known as regeneration, or back EMF. Designers of systems with a large mass moment of inertia or high deceleration rates must take regeneration effects into account when selecting power supply components.

The motors used in the ISMD23E2 products are low inductance motors. Regeneration effects are not an issue in most applications. The one exception is when a gearhead is placed on the motor and the system allows the gearhead shaft to be manually rotated. In these applications, the motor may rotate at high speeds and act as a generator.

The first line of defense against regenerative events is an appropriately sized power supply. The additional capacitance typically found in a larger supplies can be used to absorb the regenerative energy. If your application has high deceleration rates, or can be rotated manually at high speeds, then a supply that can deliver 175% of peak motor power should be used.

⚠ Caution Regeneration events can raise the Vdc supply voltage to an unexpected value. Components not rated for this voltage may be damaged.

- Consider using a separate supply for any sensor attached to the ISMD23E2 to protect it from regeneration voltages.
- Only use power supplies that will not be damaged by regeneration events.

Compatible Connectors and Cordsets

Many different connectors and cordsets are available on the market, all of which will work with the ISMD23E2 provided that the manufacturer follows the connector and Ethernet standards. AMCI and IDEC Corporation have reviewed the following connectors and ethernet cordsets for compatibility with the ISMD23E2.

Ethernet Connector

Part #	Description
IMS-28	Mating connector for Ethernet Connector. Male, 4 pin D-coded. Screw terminal connections. 6 to 8 mm dia. cable. Straight, IP67 rated when properly installed.

Table R1.5 Ethernet Connectors

Ethernet Cordset

Part #	Description
ICNER-5M	4-position, 24 AWG, shielded. EIA/TIA 568B color coded. Connectors: Straight M12, D-coded, Male to RJ45. Shield attached to both connectors. Cable length: 5 meters

Table R1.6 Ethernet Cordset

Power & Digital Input Connector

Part #	Description
IMS-31	Mating connector for Power & Digital Input Connector. Female, 5 pin A-coded. Screw terminal connections. 6 to 8 mm dia. cable. Straight, IP67 rated when properly installed.

Table R1.7 Power and Digital Input Connector

Power & Digital Input Cordset

Part #	Description
ICNPL-5M	5-position, 18 AWG. Connector: Straight M12, A-coded, Female to 2 inch flying leads, 0.28” stripped. Cable length: 5 meters

Table R1.8 Power and Digital Input Cordset

REFERENCE 2: MOTION CONTROL

Introduction

When a move command is sent to an ISMD23E2, the unit calculates the entire profile before starting the move or issuing an error message. This chapter explains how the profiles are calculated and the different available moves.

Definitions

Units of Measure

Distance: Every distance is measured in steps. When you configure the unit, you will specify the number of steps you want to complete one rotation of the motor shaft. It is up to you to determine how many steps are required to travel the appropriate distance in your application.

Speed: All speeds are measured in steps/second. Since the number of steps needed to complete one shaft rotation is determined by your programming, it is up to you to determine how many steps per second is required to rotate the motor shaft at your desired speed.

Acceleration: The typical unit of measure for acceleration and deceleration is steps/second/second, or steps/second^2. However, when programming an ISMD23E2, all acceleration and deceleration values must be programmed in the unit of measure of steps/second/millisecond.

- To convert from steps/second^2 to steps/millisecond/second, divide the value by 1000. This must be done when converting from a value used in the equations to a value programmed into an ISMD23E2.
- To convert from steps/millisecond/second to steps/second^2, multiply the value by 1000. This must be done when converting from the value programmed into an ISMD23E2 to the value used in the equations.

Motor Position

Motor Position is defined in counts, and its limits are -2,147,483,648 to +2,147,483,647 counts. The optional encoder has the same range.

Home Position

The Home Position is any position on your machine that you can sense and stop at. There are two ways to defining the Home Position. The first is using the Preset Position command to set the Motor Position register to a known value. The second method is using one of the *Find Home* commands. If you use the unit's *Find Home* commands, the motor position and encoder position registers will automatically be set to zero once the home position is reached. Defining a Home Position is completely optional. Some applications, such as those that use the ISMD23E2 for speed control, don't require position data at all.

Valid and Invalid Positions

The Motor Position is considered *Valid* once the Home Position has been defined. Until that time, the motor position is considered *Invalid*. Absolute moves cannot be run unless the position is Valid. Uncontrolled, immediate stops will force the position to become Invalid.

Count Direction

Clockwise moves will always increase the motor position register reported back to the host. Some of the moves, such as the Jog Move, have a positive and negative command. A positive command, such as the +Jog Move command, will result in a clockwise rotation of the shaft.

Starting Speed

The Starting Speed is the speed that most moves will begin and end at. This value is set while configuring the unit and it has a valid range of 1 to 999 steps/second. This value is typically used to start the move above the motor's low frequency resonances and, in micro-stepping applications, to limit the amount of time needed for acceleration and deceleration. AMCI does not specify a default value in this manual because it is very dependent on motor size and attached load. While bench testing the ISMD23E2, a starting speed between 0.25 and 0.5 RPS is generally a safe value to begin with.

Target Position

The Target Position is the position that you want the move to end at. There are two ways to define the Target Position, with relative coordinates or absolute coordinates.

Relative Coordinates

Relative coordinates define the Target Position as an offset from the present position of the motor. Most ISMD23E2 moves use relative coordinates.

- The range of values for the Target Position when it is treated as an offset is ±8,388,607 counts. Positive offsets will result in clockwise moves, while negative offsets result in counter-clockwise moves.
- The Motor Position value reported back to the host exceeds ±8,388,607 counts. The only way to move beyond ±8,388,607 counts is with multiple relative moves or jog commands.

Absolute Coordinates

Absolute coordinates treat the Target Position as an actual position on the machine. Note that you must set the Home Position on the machine before you can run an Absolute Move. (See *Home Position* on the previous page.)

- The range of values for the Target Position when it is treated as an actual position on the machine is ±8,388,607 counts. The move will be clockwise if the Target Position is greater than the Current Position and counter-clockwise if the Target Position is less than the Current Position.
- The Motor Position value reported back to the host exceeds ±8,388,607 counts. However, you cannot move beyond ±8,388,607 counts with an Absolute Move. The only way to move beyond ±8,388,607 counts is with multiple relative moves or a jog move.

Definition of Acceleration Types

With the exception of Registration Moves, all move commands, including homing commands, allow you to define the acceleration type used during the move. The ISMD23E2 supports three types of accelerations and decelerations. The type of acceleration used is controlled by the *Acceleration Jerk* parameter.

Detailed move profile calculations, including the effect of the *Acceleration Jerk* parameter, can be found in the reference section, *Calculating Move Profiles*, starting on page 85.

Linear Acceleration

When the Acceleration Jerk parameter equals zero, the axis accelerates (or decelerates) at a constant rate until the programmed speed is reached. This offers the fastest acceleration, but consideration must be given to insure the smoothest transition from rest to the acceleration phase of the move. The smoothest transition occurs when the configured Starting Speed is equal to the square root of the programmed Linear Acceleration. Note that other values will work correctly, but you may notice a quick change in velocity at the beginning of the acceleration phase.

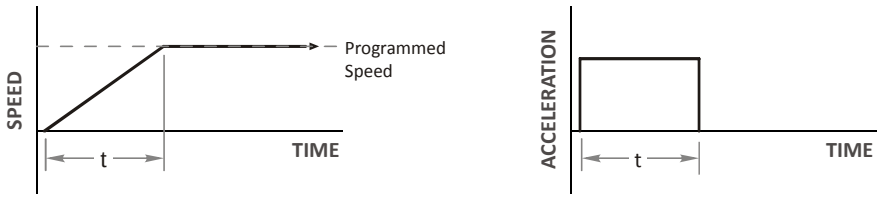


Figure R2.1 Linear Acceleration

Triangular S-Curve Acceleration

When the Acceleration Jerk parameter equals one, the axis accelerates (or decelerates) at a constantly changing rate that is slowest at the beginning and end of the acceleration phase of the move. The Triangular S-Curve type offers the smoothest acceleration, but it takes twice as long as a Linear Acceleration to achieve the same velocity.

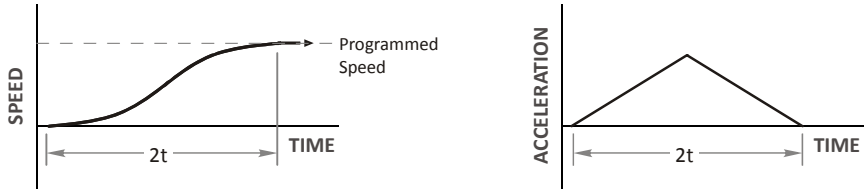


Figure R2.2 Triangular S-Curve Acceleration

Trapezoidal S-Curve Acceleration

When the Acceleration Jerk parameter is in the range of 2 to 5,000, Trapezoidal S-Curve acceleration is used. The Trapezoidal S-Curve acceleration is a good compromise between the speed of Linear acceleration and the smoothness of Triangular S-Curve acceleration. Like the Triangular S-Curve, this acceleration type begins and ends the acceleration phase smoothly, but the middle of the acceleration phase is linear. Figure R2.3 shows a trapezoidal curve when the linear acceleration phase is half of the total acceleration time. With this setting, the Trapezoidal S-Curve acceleration only requires 33% more time to achieve the same velocity as a Linear Acceleration.

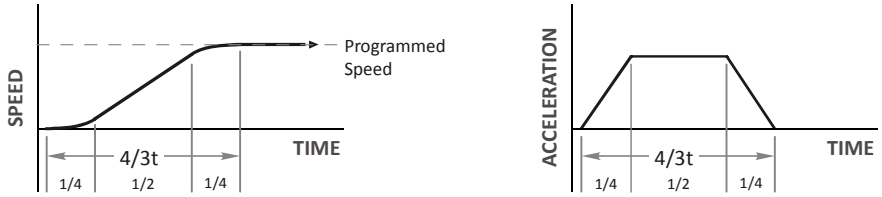


Figure R2.3 Trapezoidal S-Curve Acceleration

An acceleration jerk setting of 2 will result in the smallest amount of constant acceleration during a trapezoidal S-curve acceleration. An acceleration jerk setting of 5000 will result in the largest amount of constant acceleration during a trapezoidal S-curve acceleration. See *S-Curve Acceleration Equations*, which starts on page 87, for a methods of calculating the Acceleration Jerk parameter.

A Simple Move

As shown in the figure below, a move from A (Current Position) to B (Target Position) consists of several parts.

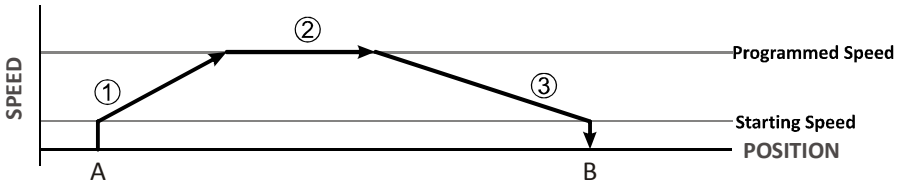


Figure R2.4 A Trapezoidal Profile

1) The move begins at point A, where the motor jumps from rest to the configured *Starting Speed*. The motor then accelerates at the programmed *Acceleration Value* until the speed of the motor reaches the *Programmed Speed*. Both the Acceleration Value and the Programmed Speed are programmed when the move command is sent to the ISMD23E2.
2) The motor continues to run at the Programmed Speed until it reaches the point where it must decelerate before reaching point B.
3) The motor decelerates at the *Deceleration Value*, which is also programmed by the move command, until the speed reaches the Starting Speed, which occurs at the Target Position (B). The motor stops at this point. Note that the acceleration and deceleration values can be different in the move.

Figure R2.4 above shows a Trapezoidal Profile. A Trapezoidal Profile occurs when the Programmed Speed is reached during the move. This occurs when the number of steps needed to accelerate and decelerate are less than the total number of steps in the move.

Figure R2.5 below shows a Triangular Profile. A Triangular Profile occurs when the number of steps needed to accelerate to the Programmed Speed and decelerate from the Programmed Speed are greater than the total number of steps in the move. In this case, the profile will accelerate as far as it can before it has to decelerate to reach the Starting Speed at the Target Position. The Programmed Speed is never reached.

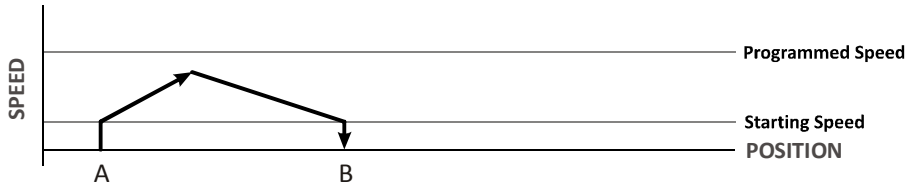


Figure R2.5 A Triangular Profile

Controlled and Immediate Stops

Once a move is started, there are several ways to stop the move before it comes to an end. These stops are broken down into two types:

- **Controlled Stop:** The axis immediately begins decelerating at the move's programmed deceleration value until it reaches the configured Starting Speed. The axis stops at this point. The motor position value is still considered valid after a Controlled Stop and the machine does not need to be homed again before Absolute Moves can be run.
- **Immediate Stop:** The axis immediately stops motion regardless of the speed the motor is running at. Since it is possible for the inertia of the load attached to the motor to pull the motor beyond the stopping point, the motor position value is considered invalid after an Immediate Stop. The machine must be homed or the position must be preset before Absolute Moves can be run again.

Host Control

Hold Move Command: This command can be used with some moves to bring the axis to a Controlled Stop. The move can be resumed and finished, or it can be aborted. Not all moves are affected by this command. The section *Basic Move Types*, starting on page 22, describes each move type in detail, including if the move is affected by this command.

Immediate Stop Command: When this command is issued from the host, the axis will come to an Immediate Stop. The move cannot be restarted and the machine must be homed again before Absolute Moves can be run. Note that power is not removed from the motor.

Hardware Control

Stop Jog or Registration Move Input: Triggering this input type during a Jog Move or Registration Move will bring the move to a controlled stop. The controlled stop is triggered on an inactive-to-active state change on the input. Only Jog Moves and Registration Moves can be stopped this way, all other moves ignore this input.

CW Limit and CCW Limit Inputs: In most cases, activating these inputs during a move will bring the axis to an Immediate Stop. The exceptions are the *CW/CCW Find Home* commands, the *CW/CCW Jog Move* commands, and the *CW/CCW Registration Move* commands. The *Find Home* commands are explained in the reference section, *Homing an ISMD23E2*, which starts on page 29. The *CW/CCW Jog Move* commands are fully explained on page 23, and the *CW/CCW Registration Move* commands are fully explained on page 24.

Emergency Stop Input: It is possible to configure an input as an Emergency Stop Input. When an Emergency Stop Input is activated, the axis will come to an Immediate Stop, regardless of the direction of travel. The move cannot be restarted and the machine must be homed again before Absolute Moves can be run. Note that power is not removed from the motor.

Basic Move Types

Relative Move

Relative Moves move an offset number of steps (n) from the Current Position (A). A trapezoidal profile is shown to the right, but Relative Moves can also generate triangular profiles. The command's Target Position is the move's offset. The offset can be in the range of ±8,388,607 counts. Positive offsets will result in clockwise moves, while negative offsets result in counter-clockwise moves.

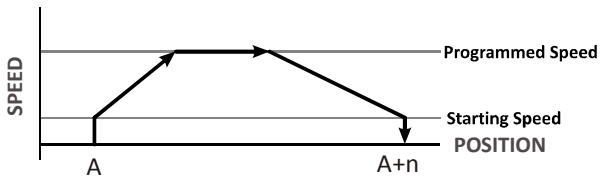


Figure R2.6 Relative Move

Note► 1) You do not have to preset the position or home the machine before you can use Relative Moves. That is, the Position_Invalid status bit can be set.

2) Relative Moves allow you to move your machine without having to calculate absolute positions. If you are indexing a rotary table, you can preform a relative move of 30° multiple times without recalculating new target positions in your controller. If you perform the same action with Absolute Moves, you would have to calculate your 30° position followed by your 60° position, followed by your 90° position, etc.

Relative Moves can be brought to a Controlled Stop by using the Hold Move Command from your host controller. When the command is accepted, the axis will immediately decelerate at the programmed rate and stop. When stopped successfully, the ISMD23E2 will set the *In_Hold_State* bit in the input data table. The Relative Move can be restarted with the Resume Move command from the host controller or the move can be aborted by starting another move. The Resume Move command allows you to change the move's Programmed Speed, Acceleration Value and Type, and the Deceleration Value and Type. The Target Position cannot be changed with the Resume Move Command.

Controlled Stop Conditions

- The move completes without error.
- You toggle the Hold_Move control bit in the Network Output Data. Note that your holding position will most likely not be the final position you commanded. You can resume a held Relative Move by using the Resume Move command. The use of the Hold_Move and Resume_Move bits is further explained in the *Controlling Moves In Progress* section starting on page 26.

Immediate Stop Conditions

- The Immediate Stop bit makes a 0→1 transition in the Network Output Data.
- An inactive-to-active transition on an input configured as an E-Stop Input.
- A CW or CWW Limit Switch is reached. If the limit that is reached is the same as the direction of travel, for example, hitting the CW limit while running a CW move, a *Reset Errors* command must be issued before moves are allowed in that direction again. If the limit that is reached is opposite the direction of travel, a *Reset Errors* command does not have to be issued.

Absolute Move

Absolute Moves move from the Current Position (A) to a given position (B). (The ISMD23E2 calculates the direction and number of steps needed to move to the given position and moves that number of steps.) A trapezoidal profile is shown to the right, but Absolute Moves can also generate triangular profiles. The command's Target Position can be in the range of ±8,388,607 counts. The move will be clockwise if the Target Position is greater than the Current Position and counter-clockwise if the Target Position is less than the Current Position.

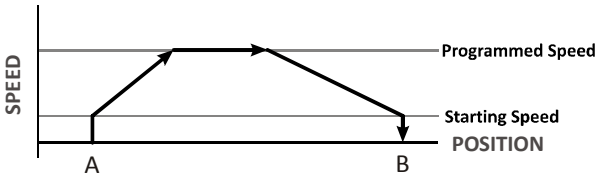


Figure R2.7 Absolute Move

Note► 1) The Motor Position must be valid before you can use an Absolute Move. The Motor Position becomes valid when you preset the position or home the machine. See the reference section, *Homing an ISMD23E2*, which starts on page 29, for information on homing the machine.

2) Absolute Moves allow you to move your machine without having to calculate relative positions. If you are controlling a rotary table, you can drive the table to any angle without having to calculate the distance to travel. For example an Absolute Move to 180° will move the table to the correct position regardless of where the move starts from.

Controlled Stop Conditions

- The move completes without error.
- You toggle the Hold_Move control bit in the Network Output Data. Note that your holding position will most likely not be the final position you commanded. You can resume a held Absolute Move by using the Resume_Move bit or the move can be aborted by starting another move. The use of the Hold_Move and Resume_Move bits is explained in the *Controlling Moves In Progress* section starting on page 26.

Immediate Stop Conditions

- The Immediate Stop bit makes a 0→1 transition in the Network Output Data.
- An inactive-to-active transition on an input configured as an E-Stop Input.
- A CW or CWW Limit Switch is reached. If the limit that is reached is the same as the direction of travel, for example, hitting the CW limit while running a CW move, a *Reset Errors* command must be issued before moves are allowed in that direction again. If the limit that is reached is opposite the direction of travel, a *Reset Errors* command does not have to be issued.

CW/CCW Jog Move

Jog Moves move in the programmed direction as long as the command is active. Two commands are available. The CW Jog Move will increase the motor position count while the CCW Jog Move will decrease the motor position count. These commands are often used to give the operator manual control over the axis.

Jog Moves are also used when you are interested in controlling the speed of the shaft instead of its position. One such application is driving a conveyor belt. To accommodate these applications, the running speed, acceleration, and deceleration of the Jog Move can be changed *while the move is in progress*.

The CW Limit and CCW Limit inputs behave differently for CW/CCW Jog Moves and CW/CCW Registration Moves than all other move types. Like all moves, activating a limit will bring the move to an Immediate Stop. Unlike other moves, a Jog or Registration move can be started when an end limit switch is active provided that the commanded direction is opposite that of the activated switch. For example, a CW Jog Move can be issued while the CCW limit switch is active. This allows you to move off of an activated end limit switch.

As shown below, a Jog Moves begins at the programmed Starting Speed, accelerates at the programmed rate to the Programmed Speed and continues until a stop condition occurs. If it is a *Controlled Stop Condition*, the ISMD23E2 will decelerate the motor to the starting speed and stop without losing position. If it is an *Immediate Stop Condition*, the motion stops immediately and the position becomes invalid.

It is possible to change the speed of a Jog Move without stopping the motion. The Programmed Speed, Acceleration, and Deceleration parameters can be changed during a Jog Move. When the Programmed Speed is changed, the motor will accelerate or decelerate to the new Programmed Speed using the new accelerate/decelerate parameter values. If you write a Programmed Speed to the unit that is less than the starting speed, the Jog Move will continue at the previously programmed speed.

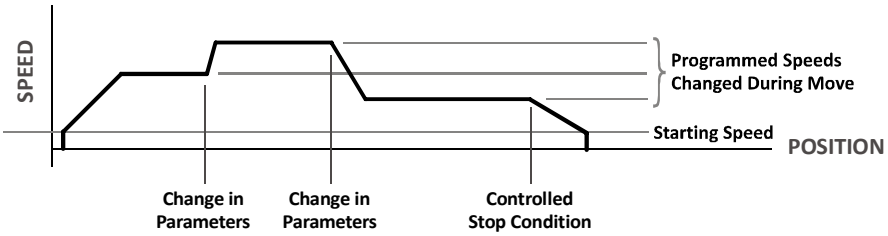


Figure R2.8 Jog Move

Controlled Stop Conditions

- The Jog Move Command bit is reset to "0".
- An inactive-to-active transition on an input configured as a *Stop Jog or Registration Move* Input.
- You toggle the Hold_Move control bit in the Network Output Data. The use of the Hold_Move and Resume_Move bits is explained in the *Controlling Moves In Progress* section starting on page 26.

Immediate Stop Conditions

- The Immediate_Stop bit makes a 0→1 transition in the Network Output Data.
- A inactive-to-active transition on an input configured as an E-Stop Input.
- A CW or CWW Limit Switch is reached. If the limit that is reached is the same as the direction of travel, for example, hitting the CW limit while running a CW move, a *Reset Errors* command must be issued before moves are allowed in that direction again. If the limit that is reached is opposite the direction of travel, a *Reset Errors* command does not have to be issued.

Note► Note that it is possible to *start* a move while a CW or CCW Limit Switch is active as long as the direction of travel is *opposite* that of the activated Limit Switch. For example, it is possible to start a CW Jog Move while the CCW Limit Switch is active

CW/CCW Registration Move

Similar to a Jog Move, a Registration Move will travel in the programmed direction as long as the command is active. CW Registration Moves increase the motor position count while the CCW Registration Moves decrease the motor position count. When the command terminates under Controlled Stop conditions, the ISMD23E2 will output a programmed number of steps as part of bringing the move to a stop. Note that all position values programmed with a Registration Move are relative values, not absolute machine positions.

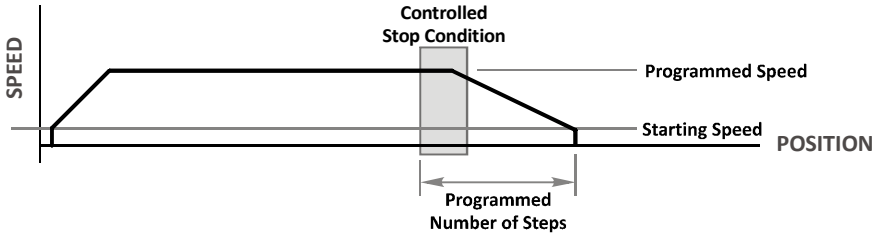


Figure R2.9 Registration Move

Note► If the Programmed Number of Steps are less than the number of steps needed to bring the axis to a stop based on the Programmed Speed and Deceleration values set with the command, the ISMD23E2 will decelerate at the programmed Deceleration value until it has output the Programmed Number of Steps and then stop the move without further deceleration.

An additional feature of the Registration Moves is the ability to program the drive to ignore the Controlled Stop conditions until a minimum number of steps have occurred. This value is programmed through the Minimum Registration Move Distance parameter, which is set when you command the Registration Move. The figure below shows how the Minimum Registration Move Distance parameter affects when the Stop Condition is applied to the move. As shown in the second diagram, Controlled Stop conditions are level triggered, not edge triggered. If a Controlled Stop Condition occurs before the Minimum Registration Move Distance is reached and the condition remains active, the move will begin its controlled stop once the Minimum Registration Move Distance is reached.

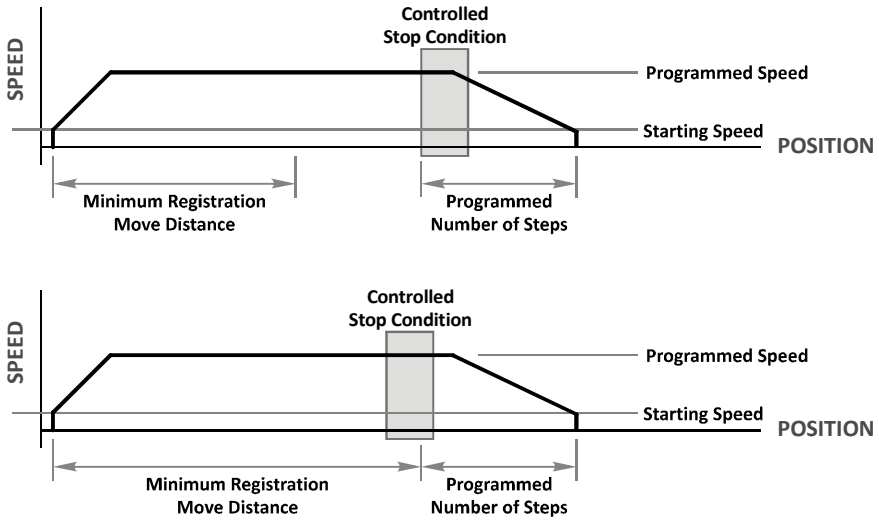


Figure R2.10 Min. Registration Move Distance

Controlled Stop Conditions

- The Registration Move Command bit is reset to "0".
- A positive transition on an input configured as a *Stop Jog or Registration Move* Input.

Note► Starting a Registration Move with a *Stop Jog or Registration Move* Input in its active state will result in a move of (*Minimum Registration Distance* + *Programmed Number of Steps*).

- You toggle the Hold_Move control bit in the Network Output Data. The ISMD23E2 responds by using the programmed Deceleration value to bring the move to a stop, without using the value of the Programmed Number of Steps parameter. A Registration Move does not go into the Hold State if the Hold_Move control bit is used to stop the move and it cannot be restarted with the Resume Move command.

Immediate Stop Conditions

- The Immediate_Stop bit makes a 0→1 transition in the Network Output Data.
- An inactive-to-active transition on an input configured as an E-Stop Input.
- A CW or CWW Limit Switch is reached. If the limit that is reached is the same as the direction of travel, for example, hitting the CW limit while running a CW move, a *Reset Errors* command must be issued before moves are allowed in that direction again. If the limit that is reached is opposite the direction of travel, a *Reset Errors* command does not have to be issued.

Note► Note that it is possible to start a move while a CW or CCW Limit Switch is active as long as the direction of travel is opposite that of the activated Limit Switch. For example, it is possible to start a CW Registration Move while the CCW Limit Switch is active.

Indexed Moves

All of the moves that have been explained in the chapter up to this point can be started by a transition on one of the inputs instead of a command from the network. If the *Indexed Move* bit is set when the command is issued, the ISMD23E2 will not run the move until the configured input makes an inactive-to-active transition. This allows you to run time critical moves that cannot be reliably started from the network because of messaging time delays.

- The input must be configured as a *Start Indexed Move Input.*
- The move begins with an inactive-to-active transition on the input. Note that an active-to-inactive transition on the input will not stop the move.
- The move command must stay in the Network Output Data while performing an Indexed Move. The move will not occur if you reset the command word before the input triggers the move.
- The move can be run multiple times as long as the move command data remains unchanged in the Network Output Data. The move will run on every inactive-to-active transition on the physical input if a move is not currently in progress. Once a move is triggered, the Start Indexed Move Input is ignored by the ISMD23E2 until the triggered move is finished.
- As stated above, a move can be run multiple times as long at the move command data remains unchanged. If you wish to program a second move and run it as an Indexed Move type, then you must have a 0→1 transition on the move command bit before the new parameters are accepted. The easiest way to accomplish this is by writing a value of zero to the command word between issuing move commands.
- A Jog Move that is started as an Indexed Move will come to a controlled stop when the command bit in the Network Output Data is reset to zero.
- It is possible to perform an indexed Registration Move by configuring two inputs for their respective functions. The first input, configured as a *Start Indexed Move Input*, starts the move and the second, configured as a *Stop Jog or Registration Move Input* causes the registration function to occur.
- You cannot issue a Hold Command with the Indexed Bit set and have the Hold Command trigger on the inactive-to-active transition of a physical input. Hold Commands are always acted upon as soon as they are accepted from the Network Output Data.
- You cannot issue an Immediate Stop Command with the Indexed Bit set and have the Immediate Stop Command trigger on the inactive-to-active transition of a physical input. Immediate Stop Commands are always acted upon as soon as they are accepted from the Network Output Data. If you need this functionality, consider programming the physical input as an E-Stop Input.
- You cannot issue a Clear Error Command with the Indexed Bit set and have the Clear Error Command trigger on the inactive-to-active transition of a physical input. Clear Error Commands are always acted upon as soon as they are accepted from the Network Output Data.

Controlling Moves In Progress

Each ISMD23E2 has the ability to place a running move on hold and later resume the move if an error did not occur while the move was in its Hold state. One potential application for this feature is bringing a move to a controlled stop when your controller senses an end-of-stock condition. The move can be put in its Hold state until the stock is replenished and then the move can be resumed.

Note that you do not have to resume a move once it has been placed in its Hold state. You can place a move in its Hold state to prematurely end the move with a controlled stop and issue a new move of any type from the stopped position.

The figure below shows a profile of a move that is placed in its Hold state and later resumed.

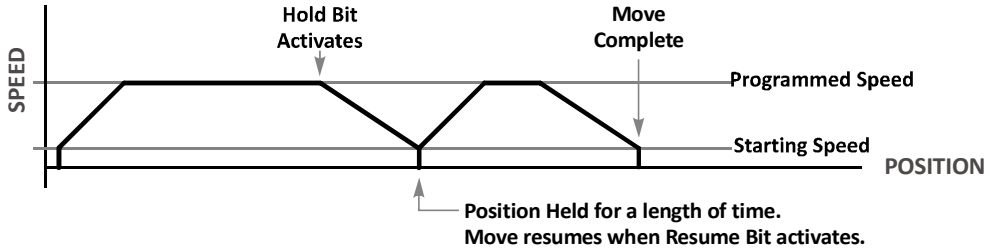


Figure R2.11 Hold/Resume a Move Profile

Jog Moves

Jog Moves can be placed in a Hold state and resumed if error conditions, such as programming errors, have not occurred. New Acceleration, Deceleration, and Programmed Speed parameters can be written to the ISMD23E2 while a Jog Move is in its hold state. If these parameters are accepted without error, the move can be resumed and it will use the new parameter values.

Registration Moves

Registration Moves can be brought to a controlled stop with the Hold bit, but they cannot be restarted.

Absolute and Relative Moves

Absolute and Relative Moves can be placed in a Hold state and resumed if error conditions, such as programming errors, have not occurred. New Acceleration, Deceleration, and Programmed Speed parameters can be written to the ISMD23E2 while these moves are in their hold states. If the parameters are accepted without error, the move can be resumed and it will use the new parameter values. Note that a change to the Target Position is ignored.

Notes:

REFERENCE 3: HOMING AN ISMD23E2

Introduction

This chapter explains the various ways of homing an ISMD23E2. Inputs used to home the unit are introduced and diagrams that show how the unit responds to a homing command are given.

Definition of Home Position

The Home Position is any position on your machine that you can sense and stop at. Once at the Home Position, the motor position register of an ISMD23E2 must be set to an appropriate value. If you use the unit's *CW/CCW Find Home* commands, the motor position register will automatically be set to zero once the home position is reached. *The Encoder Position register will also be reset to zero if the encoder is available and enabled.*

> **Note►** Defining a Home Position is completely optional. Some applications, such as those that use an ISMD23E2 for speed control, don't require position data at all.

With the exception of Absolute Moves, an ISMD23E2 can still perform all of its move commands if the Home Position is not defined.

Position Preset

One of the ways to define the Home Position is to issue the Preset Position command over the network. On units with integral encoders, both the motor position and the encoder position can be preset separately, and the motor position can be preset to the encoder position. The motor and encoder position values can be preset anywhere in the range of –8,388,607 to +8,388,607.

> **Note►** When presetting the motor position to the encoder position, the programmed Steps per Turn and Counts per Turn parameter values are used to scale the encoder position before the motor position is set to it. For example, assume that the Encoder Counts per Turn is programmed to 4,096, and the Motor Steps per Turn is programmed to 2,000. If the encoder position is 4,096 when the preset command is issued, the motor position will be set to 2,000.

CW/CCW Find Home Commands

The other choice is to use the drive's Find Home commands to order the ISMD23E2 to find the Home Position based on sensors brought into the unit. The CW Find Home command begins searching by rotating the motor shaft in the clockwise direction and ends when the home sensor triggers while the ISMD23E2 is rotating in the clockwise direction *at the starting speed*. The CCW Find Home command operates in the same way but starts and ends with motion in the counter-clockwise direction.

Homing Inputs

Either or both of the physical DC inputs can be used when homing the drive. A Backplane Proximity bit is also available in the network output data that can be used to control when the home input is acted upon. This is typically used in applications where the home input is triggered multiple times in a machine cycle, and the system need to control which trigger is acted upon.

Physical Inputs

- **Home Input:** This input is used to define the actual home position of the machine.
- **CW Limit Switch Input:** This input is used to prevent overtravel in the clockwise direction.
- **CCW Limit Switch Input:** This input is used to prevent overtravel in the counter-clockwise direction.

Network Data Input

- **Backplane_Proximity_Bit:** An ISMD23E2 can be configured to ignore changes on the physical homing input until the Backplane_Proximity_Bit makes a 0→1 transition. The ISMD23E2 will home on the next inactive-to-active change on the physical input once this transition occurs. You must program your host to control the state of this bit.

> **Note►** This bit is not used by default, and must be activated when you configure the ISMD23E2 before is can be used. If you decide to activate this bit when you configure the unit and then never set the Backplane_Proximity_Bit to a "1" while homing the ISMD23E2, the unit will never act on the physical Home Input and the homing routine will fail.

Homing Configurations

A ISMD23E2 must have one of its DC inputs configured as the home input before one of the *CW/CCW Find Home* commands can be issued.

Note► 1) You do not have to configure and use CW or CCW Limits. If you choose to configure the unit this way, then the ISMD23E2 has no way to automatically prevent over travel during a homing operation. In linear applications, you must prevent over travel by some external means, or ensure that the homing command is issued in the direction that will result in reaching the homing input directly.

2) You can use a bit in the Network Output Data (the Backplane_Proximity_Bit) as a home proximity input. Using this bit is completely optional and prevents the Home Input from being acted upon until the Backplane_Proximity_Bit makes a 0→1 transition.

Homing Profiles

Note► The CW Find Home command is used in all of these examples. The CCW Find Home command will generate the same profiles in the opposite direction.

Home Input Only Profile

Figure R3.1 below shows the move profile generated by a CW Find Home command when you use the Home Input without the Backplane_Proximity_Bit.

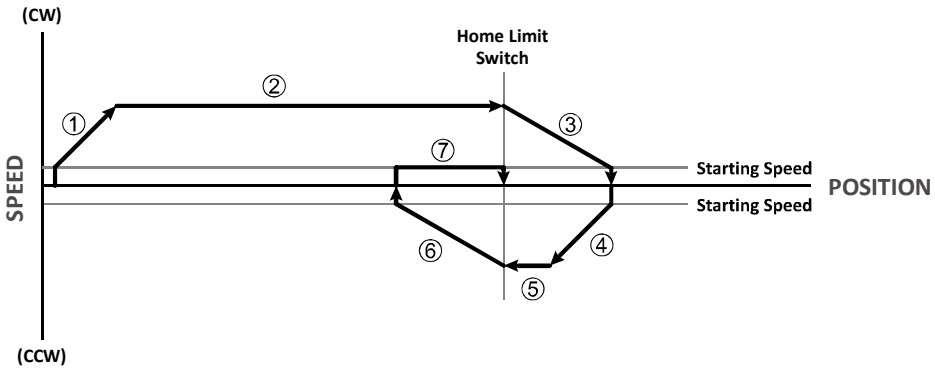


Figure R3.1 Home Input Profile

1) Acceleration from the configured Starting Speed to the Programmed Speed
2) Run at the Programmed Speed until the Home Input activates
3) Deceleration to the Starting Speed and stop, followed by a two second delay.
4) Acceleration to the Programmed Speed opposite to the requested direction.
5) Run opposite the requested direction until the Home Input transitions from Active to Inactive
6) Deceleration to the Starting Speed and stop, followed by a two second delay.
7) Return to the Home Input at the configured Starting Speed. Stop when the Home Input transitions from inactive to active.

Note► If the Home Input is active when the command is issued, the move profile begins at step 5 above.

Profile with Backplane_Proximity_Bit

Figure R3.2 below shows the move profile generated by a CW Find Home command when you use the Home Input with Backplane_Proximity_Bit.

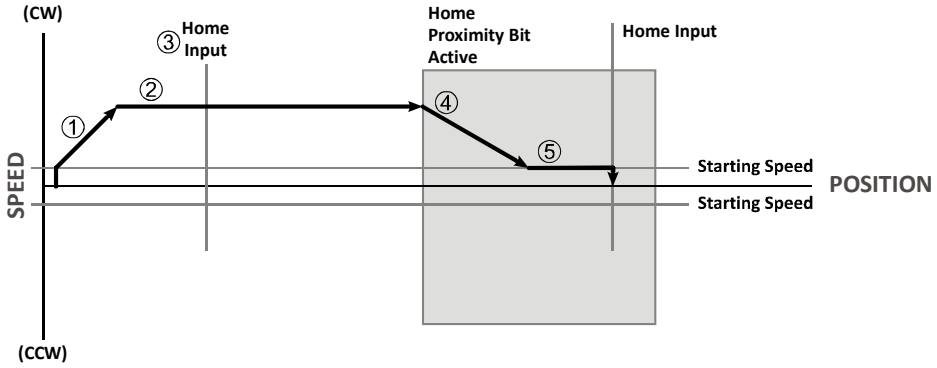


Figure R3.2 Homing with Proximity

1) Acceleration from the configured Starting Speed to the Programmed Speed
2) Run at the Programmed Speed
3) Ignores the Home Input because Backplane_Proximity_Bit has not made a 0→1 transition.
4) Deceleration towards the Starting Speed when the Backplane_Proximity_Bit transitions from 0 to 1. The axis will stop as soon as the Home Input becomes active.
5) The Starting Speed is the minimum speed the profile will run at. If the axis decelerates to the Starting Speed before reaching the Home Input, it will continue at this speed.

Note► Figure R3.2 shows the Backplane_Proximity_Bit staying active until the ISMD23E2 reaches its home position. This is valid, but does not have to occur. As stated in step 4, the ISMD23E2 starts to hunt for the home position as soon as the Backplane_Proximity_Bit makes a 0→1 transition.

Profile with Overtravel Limit

Figure R3.3 below shows the move profile generated by a CW Find Home command when you use:

- CW Overtravel Limit
- Home Input without the Backplane_Proximity_Bit

The profile is generated when you encounter an overtravel limit in the direction of travel. (In this example, hitting the CW limit while traveling in the CW direction.)

Note► The ISMD23E2 will stop and issue a Home Invalid error to your host if you activate the overtravel limit associated with travel in the opposite direction. i.e. Activating the CCW limit during a CW Find Home command. This can occur if the overtravel limits are not wired to the ISMD23E2 correctly, or if both overtravel limits are activated while the unit is trying to find the home position.

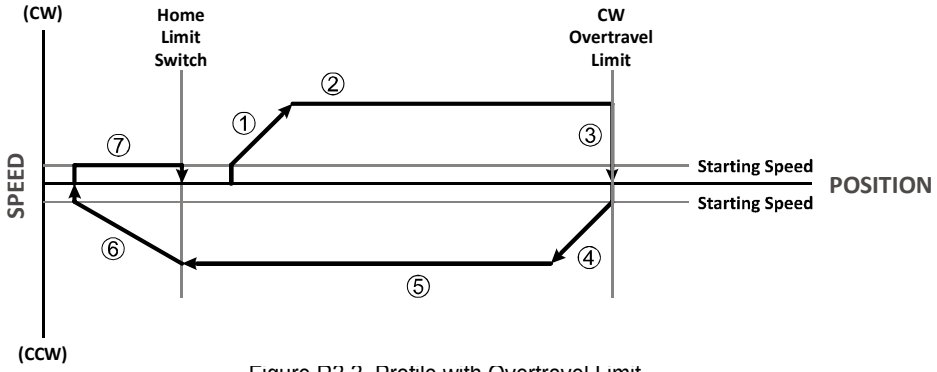


Figure R3.3 Profile with Overtravel Limit

1) Acceleration from the configured Starting Speed to the Programmed Speed
2) Run at the Programmed Speed

3) Hit CW Limit and immediately stop, followed by a two second delay.
4) Acceleration to the Programmed Speed opposite to the requested direction.
5) Run opposite the requested direction until the Home Input transitions from Active to Inactive
6) Deceleration to the Starting Speed and stop, followed by a two second delay.
7) Return to the Home Input at the configured Starting Speed. Stop when the Home Input transitions from Inactive to Active.

Note► If the overtravel limit is active when the Find Home Command is issued, the profile will begin at step 4.

Controlling Find Home Commands In Progress

Controlled Stop Conditions

- The move completes without error.
- You toggle the Hold_Move control bit in the Network Output Data. This will abort the command and the axis will decelerate at the programmed rate until it reaches the Starting Speed. At this point, the motor will stop. Note that Find Home commands cannot be restarted once held.

Immediate Stop Conditions

- The Immediate Stop bit makes a 0→1 transition in the Network Output Data.
- An inactive-to-active transition on an input configured as an E-Stop Input.
- The overtravel limit associated with travel in the opposite direction is activated. i.e. Activating the CCW limit during a CW Find Home command. This can occur if the overtravel limits are not wired to the ISMD23E2 correctly, or if both overtravel limits are activated while the unit is trying to find the home position.

TASK 1: INSTALLING THE ISMD23E2

1.1 Location

1.1.1 IP50 Rated Units (ISMD23E2-M12)

ISMD23E2 units that are IP50 rated are suitable for use in an industrial environment that meet the following criteria:

- Only non-conductive pollutants normally exist in the environment. It is expected that these pollutants will become conductive temporarily if condensation occurs.
- Transient voltages are controlled and do not exceed the impulse voltage capability of the product's insulation.

These criteria are equivalent to the *Pollution Degree 2* and *Over Voltage Category II* designations of the International Electrotechnical Commission (IEC).

1.1.2 IP64 Rated Units (ISMD23E2-M12S)

ISMD23E2 units that are IP64 rated are suitable for use in an industrial environment that meet the following criteria:

- Conductive and non-conductive pollutants occur. It is expected that all pollutants will become conductive temporarily if condensation occurs.
- Transient voltages are controlled and do not exceed the impulse voltage capability of the product's insulation.

These criteria are equivalent to the *Pollution Degree 3* and *Over Voltage Category II* designations of the International Electrotechnical Commission (IEC).

1.2 Safe Handling Guidelines

1.2.1 Prevent Electrostatic Damage

Caution Electrostatic discharge can damage the ISMD23E2. Follow these guidelines when handling the unit.

1) Touch a grounded object to discharge static potential before handling the unit.
2) Work in a static-safe environment whenever possible.
3) Wear an approved wrist-strap grounding device whenever possible.
4) Do not touch the pins of the network connectors or I/O connector.
5) Do not disassemble the unit
6) Store the unit in its shipping box when it is not in use.

1.2.2 Prevent Debris From Entering the Unit

Warning While mounting devices, be sure that all debris (metal chips, wire strands, tapping liquids, etc.) is prevented from falling into the unit, specifically into the M12 connectors. Debris may cause damage to the unit or unintended machine operation with possible personal injury.

1.2.3 Remove Power Before Servicing in a Hazardous Environment

Warning Remove power before removing or installing any ISMD23E2 units in a hazardous environment.

1.3 Operating Temperature Guidelines

Due to the onboard electronics, the maximum operating temperature of the ISMD23E2 is limited to 203°F/95°C. The motor by itself has a maximum operating temperature of 266°F/130°C. Depending on the operating current setting, move profiles, idle time, and the idle current reduction setting, it is possible to exceed these temperatures in a thermally isolated environment. As explained in the mounting section, mounting the ISMD23E2 to a large metal heatsink is the best way to limit the operating temperature of the device. Operating temperature should be monitored during system startup to verify that the maximum motor temperature remains below its 203°F/95°C specification. ISMD23E2 devices have an onboard thermistor and will remove motor current if the operating temperature exceeds this limit. Ovetemperature faults are also reported in the Network Input Data.

1.4 Mounting

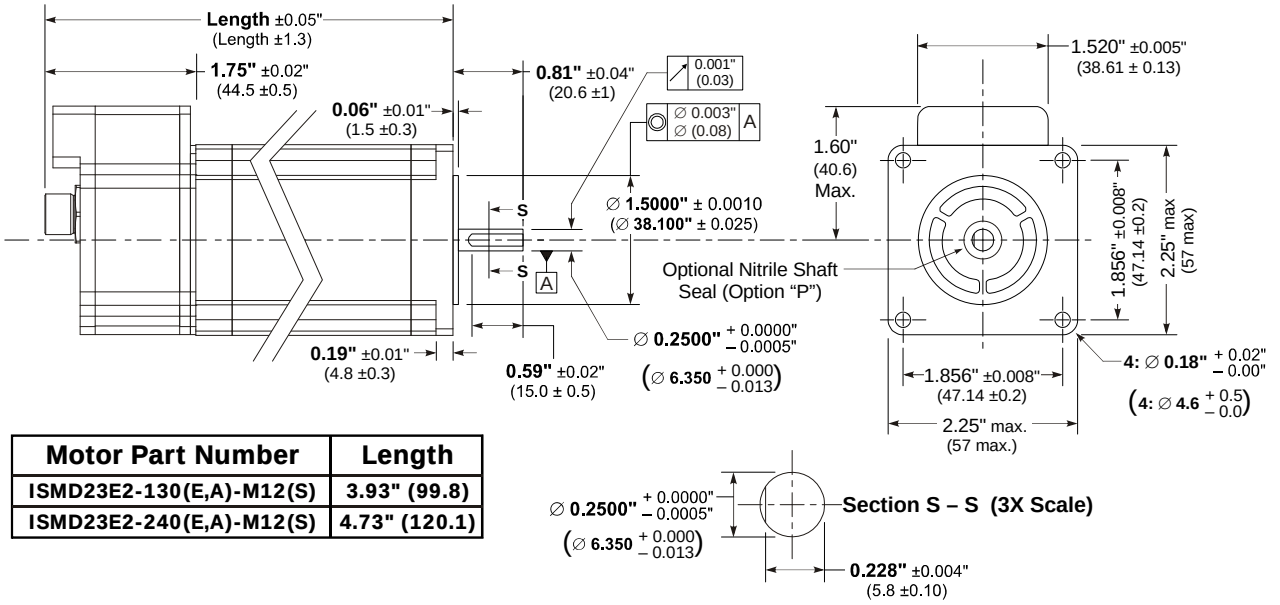
All ISMD23E2 motors have flanges on the front of the motor for mounting. This flange also acts as a heatsink, so motors should be mounted on a large, unpainted metal surface. Mounting a motor in this fashion will allow a significant amount of heat to be dissipated away from the motor, which will increase the unit's life by reducing its operating temperature. If you cannot mount the motor on a large metal surface, you may need to install a fan to force cooling air over the ISMD23E2.

Motors should be mounted using the heaviest hardware possible. These motors can produce high torques and accelerations that may weaken and shear inadequate mounting hardware.

Note► 1) The motor case must be grounded for proper operation. This is usually accomplished through its mounting hardware. If you suspect a problem with your installation, such as mounting the motor to a painted surface, then run a bonding wire from the motor to a solid earth ground point near it. Use a minimum #8 gauge stranded wire or 1/2" wire braid as the grounding wire.

2) Do not disassemble *any* stepper motor. A significant reduction in motor performance will result.

1.4.1 ISMD23E2 Outline Drawing



Motor Part Number	Length
ISMD23E2-130(E,A)-M12(S)	3.93" (99.8)
ISMD23E2-240(E,A)-M12(S)	4.73" (120.1)

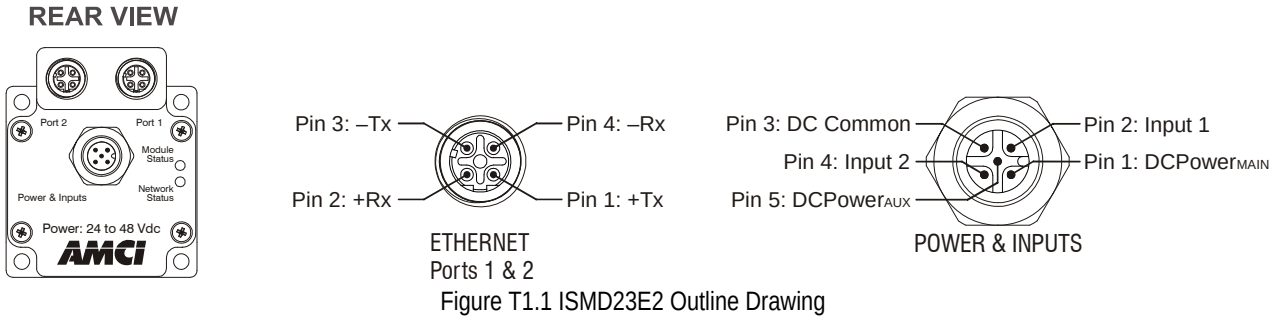


Figure T1.1 ISMD23E2 Outline Drawing

1.4.2 ISMD23E2 Mounting

An ISMD23E2-M12 unit is not water tight. Its IP50 rating makes it acceptable for use in dusty environments with occasional condensation. The ISMD23E2-M12 should be mounted in such a way that condensation will naturally drain off of the unit instead of pooling at the motor shaft or on the motor laminations.

The ISMD23E2-M12S is not water tight. Its IP64 rating makes it acceptable for use in dusty environments, environments with condensation, and environments where the unit may be exposed to splashing water. ISMD23E2-M12S units should be mounted in such a way that condensation and liquids will naturally drain off of the unit instead of pooling on the motor laminations.

1.4.3 Connecting the Load

Care must be exercised when connecting your load to the stepper motor. Even small shaft misalignments can cause large loading effects on the bearings of the motor and load. The use of a flexible coupler is *strongly* recommended whenever possible.

- Maximum radial load is 19 lbs. (85 N) at the end of the shaft.
- Maximum axial load is 3.37 lbs. (15 N)

Note► Internal encoders are mounted on the end of the motor shaft that is internal to the unit. Excessive axial load may cause encoder mis-alignment and damage to the unit. This type of damage is not covered under warranty.

1.5 Network Connectors

1.5.1 Connector Pinout

Figure T1.2 shows the Ethernet connector pinout when viewed from the back of the ISMD23E2. The Ethernet ports on the units are "auto-sense" ports that will automatically switch between 10baseT and 100baseT depending on the network equipment they are attached to. The ports also have "auto switch" capability. This means that a standard cable can be used when connecting the ISMD23E2 to any device, including a personal computer.

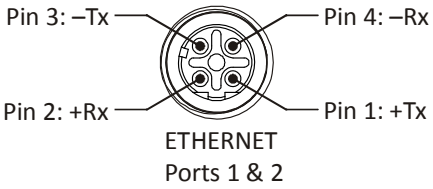


Figure T1.2 M12 Ethernet Connector Pinout

The connector is a standard female four pin D-coded M12 connector that is rated to IP67 when the mate is properly attached.

The ISMD23E2 units have two ethernet ports. Either port can be used to attach the ISMD23E2 unit to the network.

1.5.2 Compatible Connectors and Cordsets

Many different connectors and cordsets are available on the market, all of which will work with the ISMD23E2 provided that the manufacturer follows the connector and Ethernet standards. IDEC offers the following mating connector and cordsets that mate with the Ethernet port connectors.

IDEC #	Description
IMS-28	Mating connector for Ethernet port connector. Screw terminal connections. 6 to 8 mm dia. cable. Straight, IP67 rated when properly installed.
ICNER-5M	Molded cordset for Ethernet connector. 5 meters in length. Straight M12 4 pin D-coded to RJ-45 connector. Wired to TIA/EIA-568B. IP67 rated when properly installed.

Table T1.1 Compatible Ethernet Connectors and Cordsets

1.5.3 TIA/EIA-568 Color Codes

There are two color codes in common use when wiring Ethernet connections with twisted pairs. Either one of these standards is acceptable. The ICNER-5M cable available from IDEC follows the 568B standard. Note that accidently reversing the Tx/Rx pairs will not affect the operation of the ISMD23E2. The ISMD23E2 has an "auto-switch" port that will automatically adjust for swapped pairs.

Signal	568A Color	568B Color
+Transmit (+Tx)	White/Green Tracer	White/Orange tracer
–Transmit (–Tx)	Solid Green	Solid Orange
+Receive (+Rx)	White/Orange Tracer	White/Green Tracer
–Receive (–Rx)	Solid Orange	Solid Green

Table T1.2 TIA/EIA Color Codes

1.6 Power and I/O Connector

Figure T1.3 shows the power connector pinout when viewed from the back of the ISMD23E2. The connector is a male, five pin, A-coded, M12 connector that is IP67 rated when its mate is properly installed.

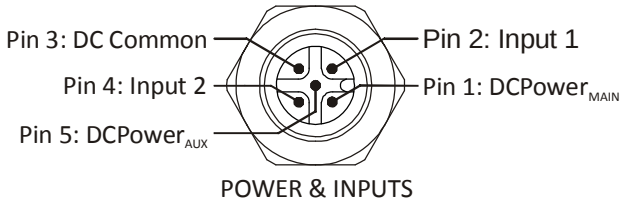


Figure T1.3 M12 Ethernet Connector Pinout

Digital inputs on the ISMD23E2 units are single ended and referenced to the DC Common pin.

There are two power pins.

- $DCPower_{MAIN}$ powers both the control electronics and the motor.
- $DCPower_{AUX}$ powers only the control electronics, which includes the encoder if it is available.

Using the $DCPower_{AUX}$ pin is optional. If your application requires you to cut power to your motor under some conditions, using the $DCPower_{AUX}$ pin allows you to cut power to your motor without losing your network connection.

Note► If the unit was ordered with an encoder, the $DCPower_{AUX}$ pin will also maintain power to the encoder. If the motor shaft is rotated while motor power is removed, the encoder position will update. (The motor position will not update.) Once power is restored to the motor, a Preset Position command can be issued to restore the correct motor position without having to go through a homing sequence. If Stall Detection is enabled on the ISMD23E2, the unit will set the Stall_Detected bit (Status Word 1, bit 14) if the motor shaft rotated more than forty-five degrees with power removed.

1.6.1 Compatible Connectors and Cordsets

Many different connectors and cordsets are available on the market, all of which will work with the ISMD23E2 provided that the manufacturer follows the connector and Ethernet standards. IDEC offers the following mating connector and cordsets that mate with the power connector.

IDEC #	Description
IMS-31	Mating connector for Power Connector. Female, 5 pin A-coded. Screw terminal connections. 6 to 8 mm dia. cable. Straight, IP67 rated when properly installed.
ICNPL-5M	5-position, 18 AWG. Connector: Straight M12, A-coded, Female to 2 inch flying leads, 0.28” stripped. Cable length: 5 m. IP67 rated when properly installed.

Table T1.3 Compatible Power Connectors and Cordsets

Note► Note that the ICNPL-5M cordset is manufactured with 18AWG wires. This is the suggested minimum gauge wire when using the ISMD23E2. If you use a different cordset, verify the gauge of the wire before making your purchase.

1.7 Power Wiring

The ISMD23E2 accepts 24 to 48Vdc as its input power. IDEC strongly suggests using 18 AWG or larger wire for the power connections. The IMS-31 connector will accept up to 18 gauge wire.

> **Caution** Do not apply 120 Vac to any pins of the ISMD23E2. If this occurs, the unit will be damaged and you will void the unit's warranty.

Figure T1.4 below shows how to wire power to the ISMD23E2 units. Note that Pin 5, $DCPower_{AUX}$, is only used when you introduce a circuit for removing power from the motor. Colors in parentheses are the appropriate wire color of the ICNPL-5M cable.

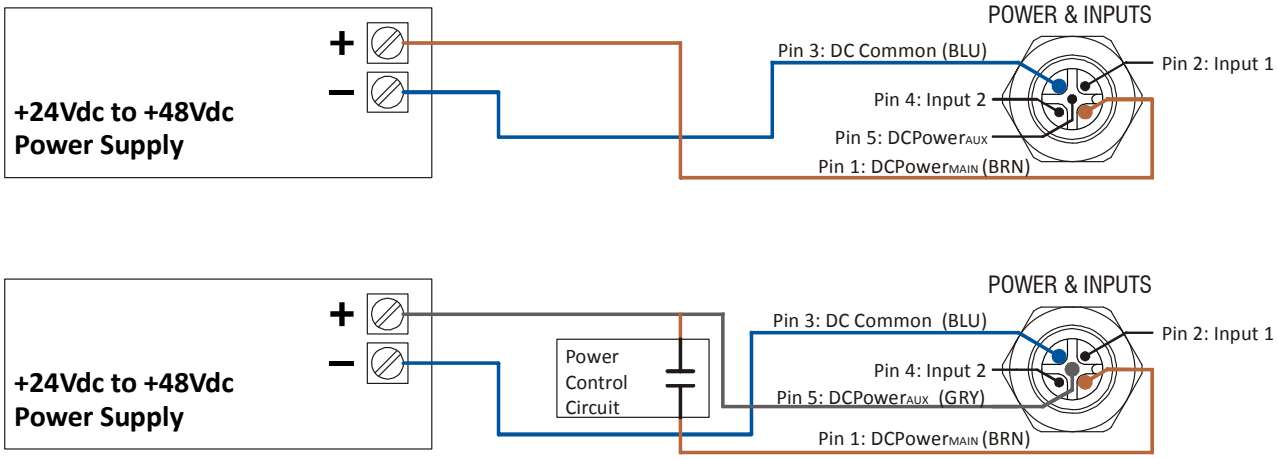


Figure T1.4 M12 Power Wiring

1.8 Input Wiring

Inputs 1 and 2 are single ended inputs that share the DC Common return pin. They accept 3.5 to 27 Vdc without the need for an external current limiting resistor. Figure T1.5 below shows how to wire discrete DC sourcing and sinking sensors to inputs 1 and 2 of the ISMD23E2. Colors in parentheses are the appropriate wire color of the ICNPL-5M cable.

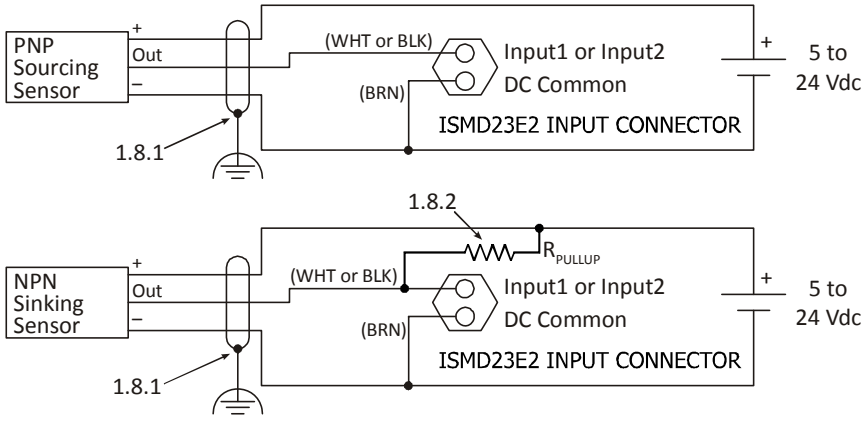


Figure T1.5 Input Wiring

1.8.1 Cable Shields

Because they are low power signals, cabling from the sensor to the ISMD23E2 should be done using a twisted pair cable with an overall shield. The shield should be grounded at the end when the signal is generated, which is the sensor end. If this is not practical, the shield should be grounded to the same ground bus as the ISMD23E2.

1.8.2 Sinking Sensors Require a Pull Up Resistor

Sinking output sensors require an external pull up resistor because inputs 1 and 2 of the ISMD23E2 also sink current. Table T1.1 below shows the values of pull up resistors that will allow the ISMD23E2 input to activate along with the current that the sensor must be able to sink when it is active.

Input Voltage	Pull Up Resistor	Sensor Current When Active
5	300 ohm	16.7 mA
12	1.4 kilohm	8.6 mA
24	3.8 kilohm	6.3 mA

Table T1.1 Pull Up Resistor

The logical states of the sensor and ISMD23E2 input will be reversed. The ISMD23E2 input is off when the sensor is active. You can set the logic state of the ISMD23E2 input when you configure the unit.

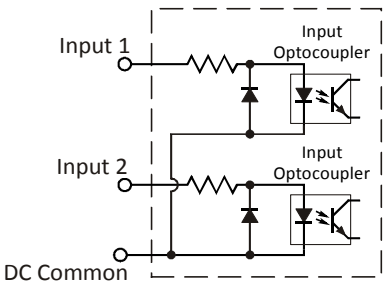


Figure T1.6 Schematic: Inputs 1 & 2

1.9 Ethernet Connections

The ISMD23E2 has two Ethernet ports with a built-in Ethernet switch connecting the two. Either port can be used to attach the unit to the network. The second port allows you to daisy-chain two devices together instead of running cables for both back to your IDEC controller. For example, if two units are located some distance from your controller, then you need only run one cable from your controller to the first unit. The second unit can then be attached to the first with a short cable.

Connections are made through ethernet cables with M12 connectors. Section 1.5.2, *Compatible Connectors and Cordsets* on page 35, lists the IDEC connector and cordset for this unit.

TASK 2: SET THE IP ADDRESS

Introduction

This section is intended for the engineer or technician responsible for setting the IP address of an ISMD23E2.

2.1 Determine the Best Method for Setting the IP Address

There are two methods for setting the IP address on an ISMD23E2. Table T2.1 below outlines the available methods and when you can use them.

Method	Restrictions	Starting page
Use Factory Default Settings	1) The machine must use 192.168.1.xxx subnet. 2) The 192.168.1.50 address must be available.	39
Use the AMCI by IDEC Ethernet Tool	No restrictions on use. The software can be used to set the ISMD23E2 to any IPv4 address. The IP address will be stored in nonvolatile memory and used on subsequent power-ups.	39

Table T2.1 Methods for Setting the IP Address

Note► There is a MAC address label on each ISMD23E2 which has a writable surface. There is room on the label for writing the programmed IP address of the unit. It is a best practice to use this label to document the IP address of the unit in case it is ever repurposed.

2.2a Use Factory Default Settings

The factory default address for the ISMD23E2 is 192.168.1.50 with a subnet mask of 255.255.255.0. The easiest way to verify this address is with the ping command as described in steps A.3 and A.4 of the Optional Task *Configure Your Network Interfaces* which starts on page 65.

If the ISMD23E2 does not respond to this address then it may take some effort to determine the correct address. There is a label on the drive that lists the MAC address of the device. There is space on the label for noting the IP address of the device if it is changed. If the address was not documented, a program called Wireshark (*https://www.wireshark.org/*) can be used to determine the address of the ISMD23E2.

Task Complete

The remainder of this chapter can be skipped, as it contains alternate methods of setting the IP address of an ISMD23E2.

2.2b Use the AMCI by IDEC Ethernet Tool

PREREQUISITE: You must know the present IP address. The factory default address is 192.168.1.50.

PREREQUISITE: Task 1.7: *Power Wiring* found on page 37. You must be able to power the ISMD23E2.

PREREQUISITE: Task 1.9: *Ethernet Connections* found on page 38. You must be able to attach your ISMD23E2 to your computer.

PREREQUISITE: Optional Task A: *Configure Your Network Interfaces*. (page 65) The network interfaces on your computer must be correctly configured before you can communicate with an ISMD23E2.

2.2b.1 Verify that Your IDEC Controller is Disconnected from the ISMD23E2

Successfully changing the IP address of the ISMD23E2 is fundamental to the operation of the unit. While performing this operation, the computer that is running the Ethernet tool should be the only device that can establish communications with the ISMD23E2.

2.2b.2 Apply or Cycle Power to the ISMD23E2

Cycling power to the ISMD23E2 will reset any connections it may have had with the IDEC host controller.

2.2b.3 If need be, download the AMCI by IDEC Ethernet Tool

1) Visit *http://us.idec.com/Home.aspx*
2) Click Products -> Automation -> AMCI Motion Controls
3) Select the *Motor+Drive+Controller* category
4) Click on the *Download* tab
5) Select the *Ethernet Tool* and download it to your computer.
6) Unzip the file to any location on your computer.

2.2b.4 If need be, install the AMCI by IDEC Ethernet Tool

Once downloaded, simply extract the program from the ZIP file and run the program to install the AMCI by IDEC Ethernet software tool on your computer. The software installs as most products do, giving you the option to change the file locations before installing the utility. Once the install is complete, a link to the utility is available on the Start Menu. The install process only copies the utility to the designated location and creates links to the Start Menu. No changes are made to the registry settings of the computer.

2.2b.5 Start the AMCI by IDEC Ethernet Tool

Double click on the utility's icon. A welcome screen similar to the one in figure T2.1 below will appear.

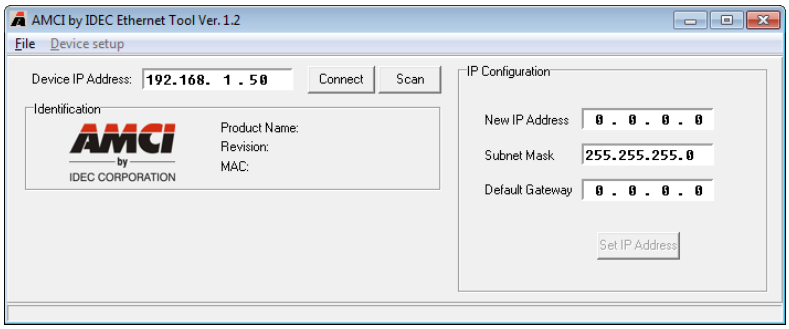


Figure T2.1 Net Configurator Welcome Screen

2.2b.6 Press the [SCAN] button and Connect to the ISMD23E2

Pressing the [Scan] button will open the window shown in figure T2.2. The ISMD23E2 will appear in the scan list after a few seconds only if the unit and your network interface are on the same subnet.

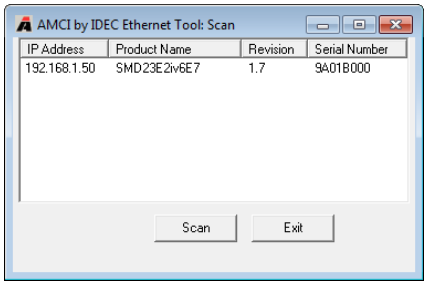


Figure T2.2 Scan for ISMD23E2

Once the ISMD23E2 appears in the list, double click on the IP Address of the ISMD23E2. The Ethernet Tool will connect to the unit.

Optionally, from the main screen, you can enter the IP address of the ISMD23E2 and press the [Connect] button. If the unit is found, the Ethernet Tool will directly connect with the unit.

2.2b.7 Set the IP Address, Subnet Mask, and Default Gateway

Enter your desired values into the IP Address, Subnet Mask, and Default Gateway fields.

> **Note►** The Default Gateway setting is not optional! It must be set to a valid address on the chosen subnet. If you do not have a required value for it, set the Default Gateway to the IP address of your FC6A controller.

2.2b.8 Write the New IP Address to the ISMD23E2

Click on the [Set IP Address] button. If there is an error in the settings, the utility will tell you what is wrong. Once all parameter values are correct, the utility will write the new IP address settings to the unit. These settings are saved to nonvolatile memory.

2.2b.9 Remove Power from the ISMD23E2

The new IP address will not be used until power to the ISMD23E2 has been cycled.

Task Complete

TASK 3: WINDLDR 8.5 PROJECT SETUP

Introduction

The following task adds the ISMD23E2 to a WindLDR 8.5+ project. (WindLDR versions 8.5+ is required.) This involves adding the ISMD23E2 to the Remote Host List and then configure Modbus TCP connection requests.

> **Note►** Before adding an ISMD23E2 to a new project, the controller must be defined. Refer to IDEC manuals and help files for information on accomplishing this task.

3.1 Add the ISMD23E2 to the Remote Host List

3.1.1 Bring up the project window in the Work Space area on the left of the window.

If need be, select the View Tab, and verify that the Project Window icon is enabled. (Property Sheet and Cross Reference are disabled in figure T3.1 below)

Select the Project Window tab in the bottom of the Work Space area.

3.1.2 Double click on the Remote Host List in the Project Window to open the Remote Host List screen.

3.1.3 Click on the [New] button to open the Remote Host screen.

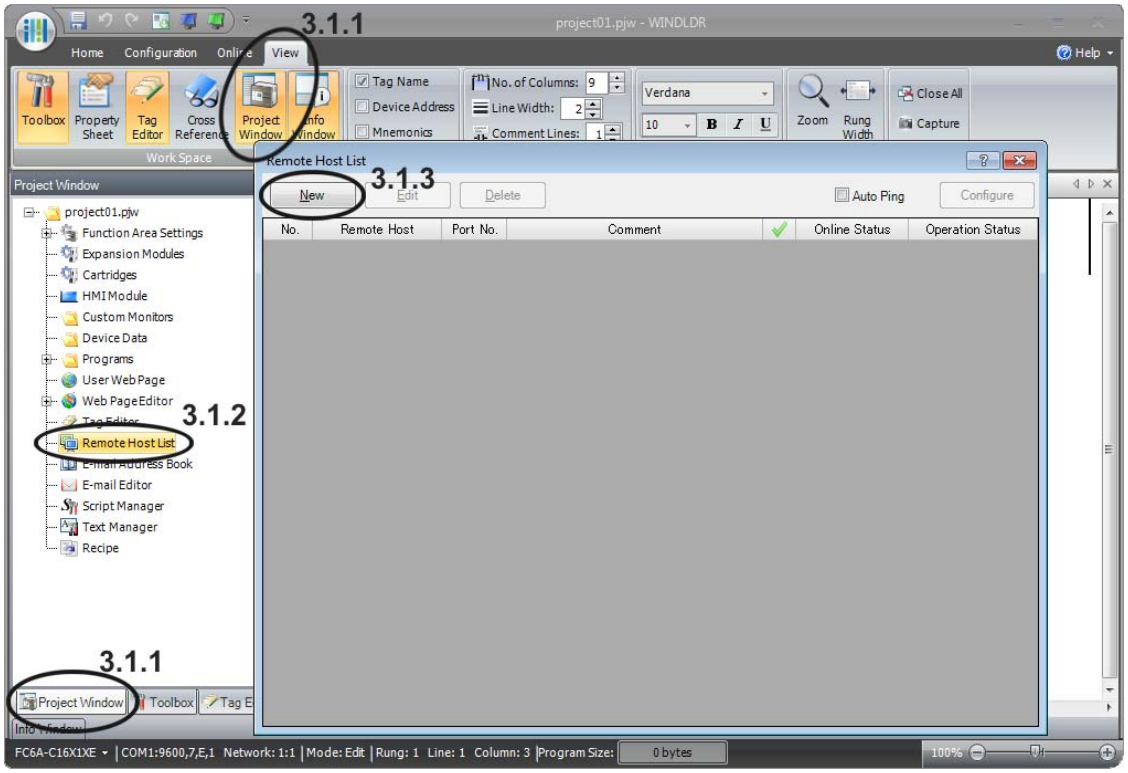


Figure T3.1 WindLDR Remote Host List

3.1.4 Enter the IP Address Information.

Enter the IP Address of the ISMD23E2. The default address is 192.168.1.50. You can enter a Host Name if you have configured a DNS server for the IDEC controller, which is beyond the scope of this manual.

Set the *Port:* field to 502, which is the default port used by the Modbus TCP protocol

You can add a *Comment:*, such as the name of the ISMD23E2 in the project. This step is optional.

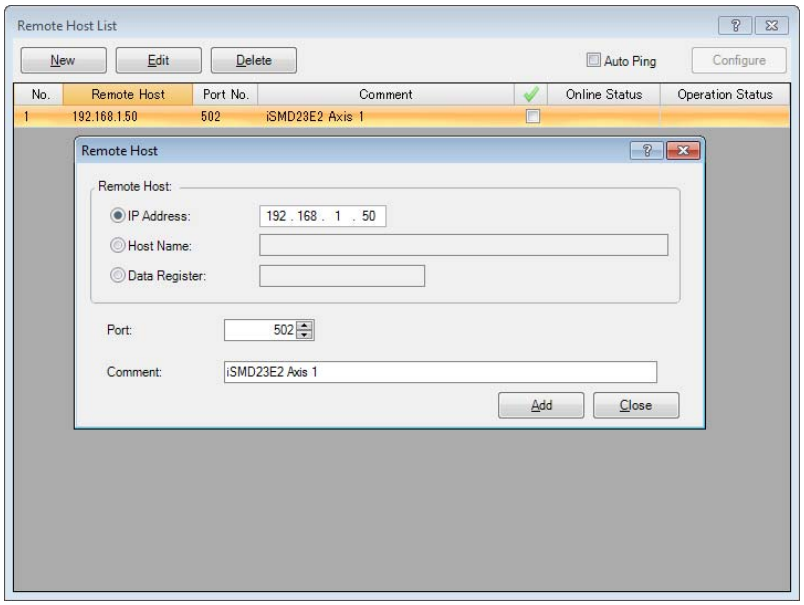


Figure T3.2 Remote Host List Entry Screen

3.1.5 Click on the [Add] button to add the ISMD23E2 to the Remote Host List.

3.1.6 Repeat steps 3.1.3 and 3.1.4 for each ISMD23E2 that must be added to the project.

3.1.7 Click on the [Close] button to close the Remote Host screen.

3.1.8 Close the Remote Host List screen.

3.2 Configure the Connection Settings for the ISMD23E2

FC6A processors support a maximum of eight communication channels on the Ethernet port. At least one of the communications channels must be configured to service “Modbus TCP Client” communications.

3.2.1 From the WindLDR menu, select Configuration > Connection Settings.

The Function Area Settings dialog box will open, with the Connection Settings view selected. The eight communication channels are shown. Note that the default Communication Mode for all channels is “Maintenance Communication Server”.

3.2.2 Set the Communications Mode for the selected channel to "Modbus TCP Client"

Click on the Communications Mode cell of the selected port. A drop down menu will appear. (See figure T3.3 below.) In this menu, select "Modbus TCP Client". The Modbus TCP Client dialog box will appear.

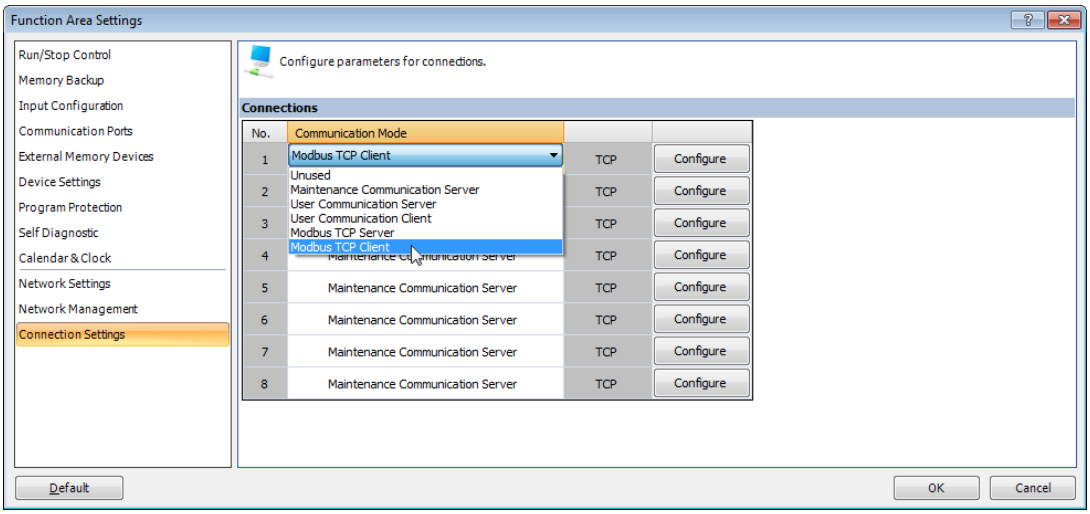


Figure T3.3 Connection Settings

3.2.3 Configure the Modbus TCP Client Requests

Two requests are required to communicate with the ISMD23E2. One request reads data from the ISMD23E2 and the other writes data to the ISMD23E2. In addition to these requests, there are three setting areas that affect the communication channel as a whole.

Request Execution Settings: The "Request Execution Device" checkbox is typically left unchecked so that the FC6A communicates with the ISMD23E2 continuously. If checked, communications is controlled by internal relays or data register bits. If your application requires you to enable this option, refer to the FC6A help file or the FC6A Series Communications Manual for additional information. The "Synchronize with auto ping" checkbox is typically left unchecked. If your application requires you to enable this option, refer to the FC6A help file or the FC6A Series Communications Manual for additional information.

Error Status: Error status information should be collected on each communication request. Select the "Use" radio button and set a data register address in the text box. All IDEC sample programs use register D49240 as the first error status register. The "Use a single DR for all communication requests" checkbox should be left unchecked so that error data on each request is collected individually. The number of data registers consumed will be equal to the number of requests on the channel, starting at the specified address. The "Update error status only when communication fails" checkbox is left unchecked in all IDEC sample programs. This setting allows the sample programs to act when communications is established.

Communication Settings: If needed, click on the [Communication Settings] button. A dialog box will open that allows you to set the Receive Timeout and Transmission Wait Time values. These setting are application specific and depend on network loading. Default values have been tested by AMCI and IDEC. Click on the [OK] button to close the dialog box.

Read Request: Set the following values to read values from the ISMD23E2.

Function Code: Click inside the cell and select "04 Read Input Registers".

Master Device Address: Depends on the axis being controlled. See table T3.1 below to determine the correct address.

Machine Axis	Starting Data Register Address	Machine Axis	Starting Data Register Address
1	D49000	7	D49060
2	D49010	8	D49070
3	D49020	9	D49080
4	D49030	10	D49090
5	D49040	11	D49100
6	D49050	12	D49110

Table T3.1 Master Device Addresses: Read Input Registers

Data Size: 10

Remote Host No.: Click inside the cell to change it into a drop down menu control. Click on the down arrow and select the proper device that was configured in step 3.1, *Add the ISMD23E2 to the Remote Host List* , above.

Slave Number: Not used by the SMD23E2. Set to "1".

Modbus Slave Address: 300001

The remaining fields are filled in automatically

Write Request: Set the following values to write values to the ISMD23E2.

Function Code: Click inside the cell and select "16 Preset Multiple Registers".

Master Device Address: Depends on the axis being controlled. See table T3.1 below to determine the correct address.

Machine Axis	Starting Data Register Address	Machine Axis	Starting Data Register Address
1	D49120	7	D49180
2	D49130	8	D49190
3	D49140	9	D49200
4	D49150	10	D49210
5	D49160	11	D49220
6	D49170	12	D49230

Table T3.2 Master Device Addresses: Read Input Registers

Data Size: 10

Remote Host No.: Click inside the cell to change it into a drop down menu control. Click on the down arrow to open the list and select the proper device that was configured in step 3.1, *Add the ISMD23E2 to the Remote Host List* , above.

Slave Number: Not used by the SMD23E2. Set to "1".

Modbus Slave Address: 401025

The remaining fields are filled in automatically. Figure T3.4 below shows the Modbus TCP Client dialog box once it is fully populated.

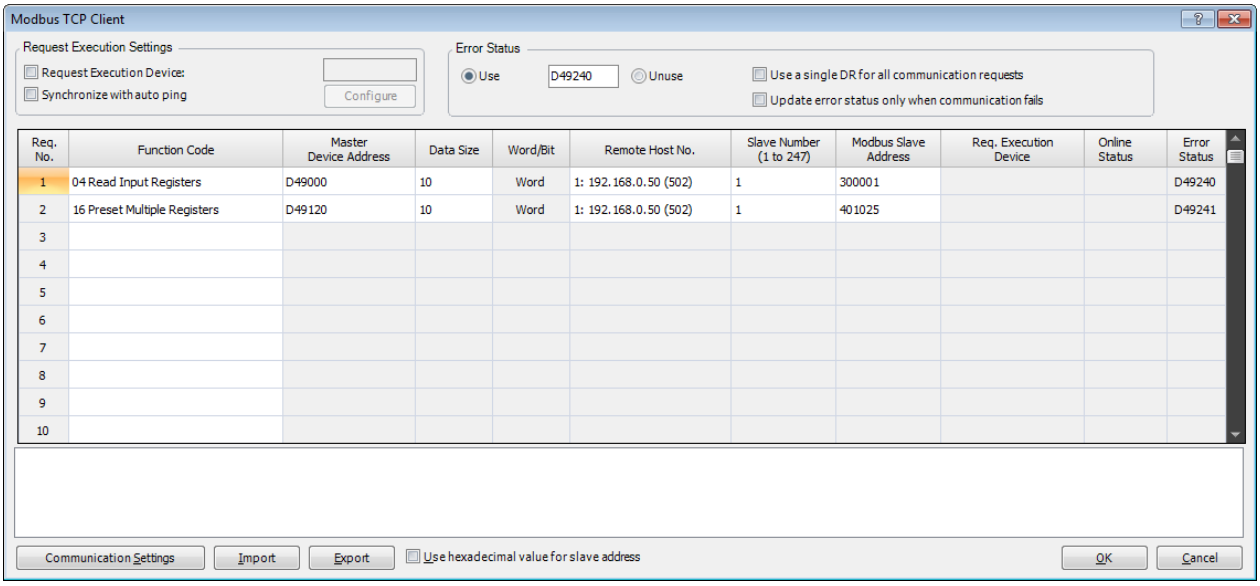


Figure T3.4 Modbus TCP Client Configuration Screen

TASK 4: INSTALLING CUSTOM MACROS

Introduction

IDEC has created User-defined Macros for configuring the ISMD23E2 and most of the commands. These macros simplify the configuration and control of an ISMD23E2. This task covers how to add these macros to a WindLDR 8.5+ project.

4.1 Download the Custom Macro File

1) Visit *http://us.idec.com/Home.aspx*
2) Click Products -> Automation -> AMCI Motion Controls
3) Select the *Motor+Drive+Controller* category
4) Click on the *Download* tab
5) Select the custom macro file and download it to your computer.
6) Unzip the file to any location on your computer.

4.2 Add the Macros to a New or Existing Project

4.2.1 Open the Import User-defined Macro dialog box.

In the Project Window, if necessary, double click on Programs in order to make User-defined Macro visible. Right click on User-defined Macro and select Import from the resulting menu. In the Open dialog box, browse to the folder that contains the file you downloaded and unzipped in step 4.2 above. Double click on the file name to open the Import User-defined Macro dialog box.

4.2.2 Import the macros into your program.

Click on the checkboxes next to the macros required for your project. You can import all of them if you so choose. Once the macros are selected, click on the [OK] button to import them.

> **Note►** You can repeat the steps in 4.2 to import additional macros at any time.

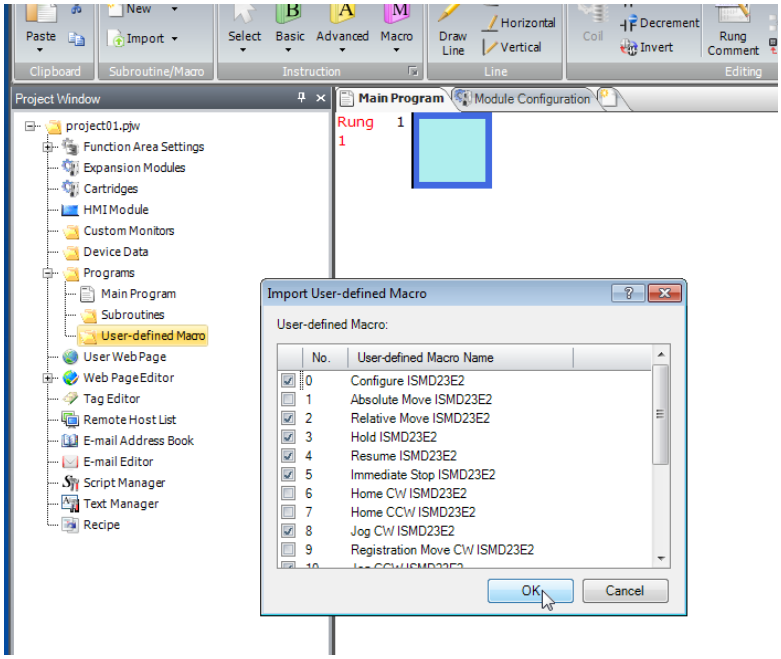


Figure T4.1 Importing User-defined Macros

Notes

5.1 Verify Memory Usage

The following address ranges are used by the IDEC supplied macros. You should check the registers currently being used in your project against this list and make adjustments as necessary. Note that registers D49000 through D49239 are the same addresses listed in the *WindLDR 8.5 Project Setup* Task starting on page 43.

Address Range			Use	Comments
D48000	to	D48999	Internal to Macros	These registers are used internally by the macros. Using registers in this range for any other purpose may result in unexpected operation.
D49000	to	D49009	Axis 1	Read Registers (Input data from the ISMD23E2. Updated through Modbus Function Code 04, Read Registers.) Data registers assigned to unused axes can be used for other purposes in the program.
D49010	to	D49019	Axis 2	
D49020	to	D49029	Axis 3	
D49030	to	D49039	Axis 4	
D49040	to	D49049	Axis 5	
D49050	to	D49059	Axis 6	
D49060	to	D49069	Axis 7	
D49070	to	D49079	Axis 8	
D49080	to	D49089	Axis 9	
D49090	to	D49099	Axis 10	
D49100	to	D49109	Axis 11	
D49110	to	D49119	Axis 12	
D49120	to	D49129	Axis 1	Preset Multiple Registers (Output data to the ISMD23E2. Updated through Modbus Function Code 16, Preset Multiple Registers.) Data registers assigned to unused axes can be used for other purposes in the program.
D49130	to	D49139	Axis 2	
D49140	to	D49149	Axis 3	
D49150	to	D49159	Axis 4	
D49160	to	D49169	Axis 5	
D49170	to	D49179	Axis 6	
D49180	to	D49189	Axis 7	
D49190	to	D49199	Axis 8	
D49200	to	D49209	Axis 9	
D49210	to	D49219	Axis 10	
D49220	to	D49229	Axis 11	
D49230	to	D49239	Axis 12	
D49240	to	D49499	Modbus TCP Communications	These registers used when monitoring and controlling Modbus TCP communications with the ISMD23E2.
D49500	to	D49619	Read Registers Buffer	Memory space used to buffer the data read from the ISMD23E2 controllers. Any registers can be used, but all sample programs use addresses in this range. Note that this area is enough to buffer 12 axes. Unused registers can be used to store macro user data.
D49620	to	D49999	Macro User Data	Used to store data that is sent to an ISMD23E2 through the macros. Any registers can be used, but all sample programs use addresses in this range.
M14500	to	M14747	Macro User Data	Addresses used to store data that is sent to an ISMD23E2 through the macros. Any internal relays can be used, but all sample programs use internal relays in this range.
M14750	to	M14997	Program Control Bits	Internal relays used to store move states and control program flow. Any internal relays can be used, but all sample programs use internal relays in this range.

Table T5.1 Reserved Memory Addresses

5.2 Verify/Alter Internal Relay Keep Settings

By default, the M internal relays are cleared at startup. The M14500 to M14747 internal relays listed above are used to store configuration and command bit settings and must be maintained between power cycles. Use the Configuration > Memory Backup screen to configure the FC6A to keep previous settings at startup.

5.2.1 Open the Function Area Settings Dialog Box.

From the WindLDR menu bar, select **Configuration > Memory Backup**. The **Function Area Settings** dialog box for Configure Keep/Clear Settings appears.

5.2.2 Verify or Alter the Keep Settings.

Under the 'M10000 to M17497' heading of the Internal Relay section, select either the 'Keep All' or 'Keep Specified Range' radio button. If your program does not rely on internal relays being reset on startup, then you can select 'Keep All'. If your program does rely on this behavior, select the 'Keep Specified Range' radio button and specify a range of M14500 to M14747.

Note► You can only specify one range when using the 'Keep Specified Range' feature. If you are already using this feature, adjust this setting as necessary.

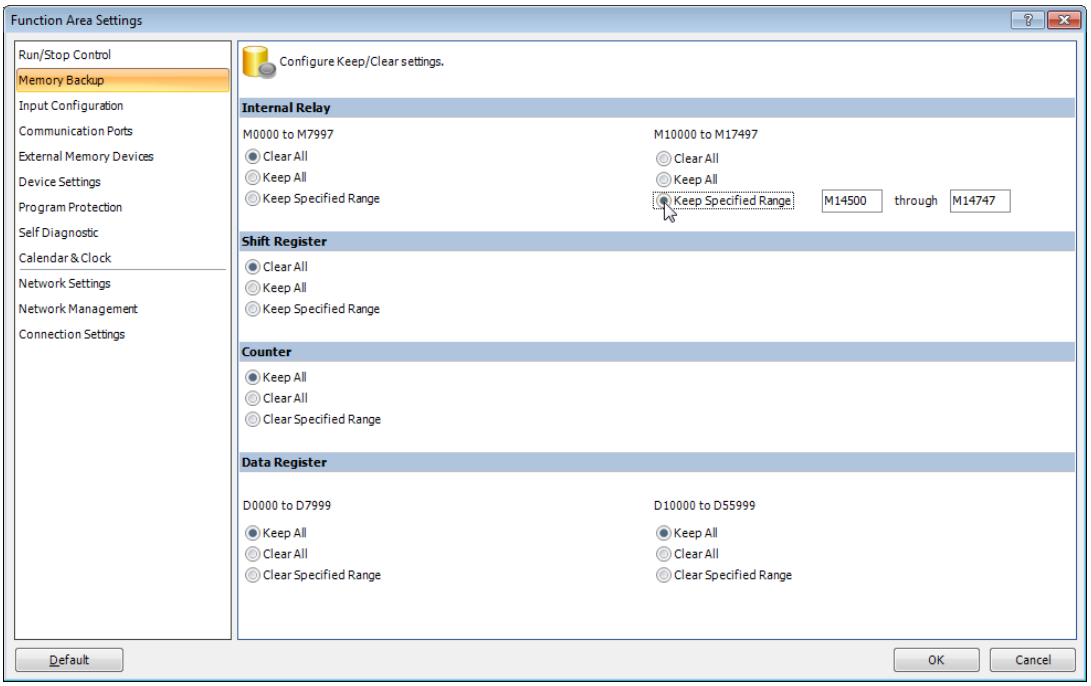


Figure T5.1 Internal Relay Keep Settings

5.3 Define the Memory Map to the Macro's Argument Devices.

Macros contain a list of arguments that are used to pass data into, and out of, the macro. While adding the macro to your ladder logic, the list of arguments must be mapped to actual Device Addresses in the processor's memory. Sample programs available from IDEC use Device Addresses starting at register D49620 and internal relay M14500 for these purposes.

Even though it is possible to map from one register or relay to multiple macros, sample programs from IDEC use separate blocks of registers for each instance of a macro in the ladder logic program. This aids in debugging by preventing a change in value for one macro from inadvertently affecting another.

Before mapping registers and relays to their macros, tag name and comment fields can be added through the Tag Editor dialog box. This simplifies register lookup when adding them to the macro. The Tag Editor in the WindLDR program is used to add tag names and comments to the Device Addresses. Consult IDEC manuals and help files for information on using the Tag Editor.

5.3.1 Arguments to the "Configure ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	D (Data Register)	Word	0 to 6	Input 1 Function. See Table T5.3 below
A3	D (Data Register)	Word	0 to 6	Input 2 Function. See Table T5.3 below
A4	M (Internal Relay)	Bit	0/1	Input 1 Active State 0 = NC Input, 1 = NO Input
A5	M (Internal Relay)	Bit	0/1	Input 2 Active State 0 = NC Input, 1 = NO Input
A6	M (Internal Relay)	Bit	0/1	Encoder Enable (Encoder must be present on device) 0 = Encoder Disabled, 1 = Encoder Enable
A7	M (Internal Relay)	Bit	0/1	Backplane_Proximity_Bit Enable 0 = Backplane Home Proximity Bit is not used 1 = Backplane Home Proximity Bit is used See *Network Data Input* found on page 29 for a description of this bit. This bit is reset in most applications.
A8	M (Internal Relay)	Bit	0/1	Stall_Detect_Enable 0 = Stall detection disabled 1 = Stall detection enabled Stall detection is only available on units with an integral encoder that has been enabled. (Argument A6 = 1)
A9	M (Internal Relay)	Bit	0/1	Antiresonance_Disable 0 = Antiresonance enabled 1 = Antiresonance disabled
A10	D (Data Register)	Word	0 to 999	Starting Speed. All moves begin and end at this speed
A11	D (Data Register)	Word	200 to 32,767	Motor Steps per Turn
A12	D (Data Register)	Word	See Description	Encoder Counts per Turn For Incremental Encoder: 1,024, 2,048, or 4,096 For Absolute Encoder: 2,048
A13	D (Data Register)	Word	0 to 100	Idle Current, as a percentage of Motor Current 0 = Current removed when idle 100 = No current reduction when idle
A14	D (Data Register)	Word	10 to 34	Motor Current in steps of 0.1 amps. 10 = 1.0 amp motor current 34 = 3.4 amp motor current

Table T5.2 Arguments to the "Configure ISMD23E2" Macro

Value	Function	Description
0	General Purpose Input	The input is not used in any of the functions of the ISMD23E2, but it's status is reported in the Network Data. This allows the input to be used as a discrete DC input to the host controller.
1	CW Limit	Input defines the mechanical end point for CW motion.
2	CCW Limit	Input defines the mechanical end point for CCW motion.
3	Start Indexed Move	Starts the move that is currently located in the output registers.
3	Start Indexed Move / Capture Encoder Value	When the encoder is enabled on an ISMD23E2, the encoder position value is captured whenever this input transitions. An inactive-to-active state transition will also trigger an Indexed Move if one is pending in the ISMD23E2.
4	Stop Jog or Registration Move	Brings a Jog or Registration Move to a controlled stop.
4	Stop Jog or Registration Move & Capture Encoder Value	When the encoder is enabled on an ISMD23E2, the encoder position value is captured when the input triggers a controlled stop to a Jog or Registration move.
5	Emergency Stop	All motion is immediately stopped when this input makes an inactive-to-active transition.
6	Home	Used to define the home position of the machine.

Table T5.3 Input Function Definitions

5.3.2 Arguments to the "Absolute Move ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Move Trigger Type 0 = Normal Move. Move begins as soon as data is accepted. 1 = Indexed Move. Move triggered by discrete input. See *Indexed Moves* found on page 26 for additional information.
A3	D (Data Register)	Long†	–8,388,607 to +8,388,607	Target Position
A4	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A5	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A6	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A7	D (Data Register)	Word	0 to 5000	Acceleration/Deceleration Jerk 0 = Linear Acceleration 1 = Triangular Acceleration 2 to 5,000 = Trapezoidal Acceleration

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.4 Arguments to the "Absolute Move ISMD23E2" Macro

5.3.3 Arguments to the "Relative Move ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Move Trigger Type 0 = Normal Move. Move begins as soon as data is accepted. 1 = Indexed Move. Move triggered by discrete input. See *Indexed Moves* found on page 26 for additional information.
A3	D (Data Register)	Long†	–8,388,607 to +8,388,607	Target Distance
A4	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A5	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A6	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A7	D (Data Register)	Word	0 to 5000	Acceleration/Deceleration Jerk 0 = Linear Acceleration 1 = Triangular Acceleration 2 to 5,000 = Trapezoidal Acceleration

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.5 Arguments to the "Relative Move ISMD23E2" Macro

5.3.4 Arguments to the "Hold ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.6 Arguments to the "Hold ISMD23E2" Macro

5.3.5 Arguments to the "Resume ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.7 Arguments to the "Resume ISMD23E2" Macro

5.3.6 Arguments to the "Immediate Stop ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.8 Arguments to the "Immediate Stop ISMD23E2" Macro

5.3.7 Arguments to the "Home CW ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Move Trigger Type 0 = Normal Move. Move begins as soon as data is accepted. 1 = Indexed Move. Move triggered by discrete input. See *Indexed Moves* found on page 26 for additional information.
A3	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A4	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A5	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A6	D (Data Register)	Word	0 to 5000	Acceleration/Deceleration Jerk 0 = Linear Acceleration 1 = Triangular Acceleration 2 to 5,000 = Trapezoidal Acceleration

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.9 Arguments to the "Home CW ISMD23E2" Macro

5.3.8 Arguments to the "Home CCW ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Move Trigger Type 0 = Normal Move. Move begins as soon as data is accepted. 1 = Indexed Move. Move triggered by discrete input. See *Indexed Moves* found on page 26 for additional information.
A3	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A4	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A5	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A6	D (Data Register)	Word	0 to 5000	Acceleration/Deceleration Jerk 0 = Linear Acceleration 1 = Triangular Acceleration 2 to 5,000 = Trapezoidal Acceleration

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.10 Arguments to the "Home CCW ISMD23E2" Macro

5.3.9 Arguments to the "Jog CW ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A3	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A4	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A5	D (Data Register)	Word	0 to 5000	Acceleration/Deceleration Jerk 0 = Linear Acceleration 1 = Triangular Acceleration 2 to 5,000 = Trapezoidal Acceleration

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.11 Arguments to the "Jog CW ISMD23E2" Macro

5.3.10 Arguments to the "Registration Move CW ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Move Trigger Type 0 = Normal Move. Move begins as soon as data is accepted. 1 = Indexed Move. Move triggered by discrete input. See *Indexed Moves* found on page 26 for additional information.
A3	D (Data Register)	Long†	0 to 8,388,607	Stopping Distance.
A4	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A5	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A6	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A7	D (Data Register)	Long†	0 to 8,388,607	Minimum Registration Move Distance.

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.12 Arguments to the "Registration Move CW" Macro

5.3.11 Arguments to the "Jog CCW ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A3	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A4	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A5	D (Data Register)	Word	0 to 5000	Acceleration/Deceleration Jerk 0 = Linear Acceleration 1 = Triangular Acceleration 2 to 5,000 = Trapezoidal Acceleration

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.13 Arguments to the "Jog CCW ISMD23E2" Macro

5.3.12 Arguments to the "Registration Move CCW ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Move Trigger Type 0 = Normal Move. Move begins as soon as data is accepted. 1 = Indexed Move. Move triggered by discrete input. See *Indexed Moves* found on page 26 for additional information.
A3	D (Data Register)	Long†	0 to 8,388,607	Stopping Distance.
A4	D (Data Register)	Long†	Starting Speed to 2,999,999	Programmed Speed. (Starting Speed is one of the *Arguments to the "Configure ISMD23E2" Macro* found on page 49.)
A5	D (Data Register)	Word	1 to 5000	Acceleration in steps/millisecond/second.
A6	D (Data Register)	Word	1 to 5000	Deceleration in steps/millisecond/second.
A7	D (Data Register)	Long†	0 to 8,388,607	Minimum Registration Move Distance.

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.14 Arguments to the "Registration Move CCW" Macro

5.3.13 Arguments to the "Preset Position ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	D (Data Register)	Long†	–8,388,607 to +8,388,607	Desired Motor Position value.

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.15 Arguments to the "Preset Position" Macro

5.3.14 Arguments to the "Clear Errors ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.16 Arguments to the "Clear Errors ISMD23E2" Macro

5.3.15 Arguments to the "Drive Enable ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.17 Arguments to the "Drive Enable ISMD23E2" Macro

5.3.16 Arguments to the "Drive Disable ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.18 Arguments to the "Drive Disable ISMD23E2" Macro

5.3.17 Arguments to the "Preset Encoder Position ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)
A2	M (Internal Relay)	Bit	0/1	Save to Flash 0 = Save calculated offset to flash memory. 1 = Do not save calculated offset in flash memory. This bit is only acted upon when the ISMD23E2 contains an absolute encoder.
A3	D (Data Register)	Long†	0 to 8,388,607	Stopping Distance.

† Long integer values consume two consecutive data registers and the starting address must be even. For example, if argument A3 begins at address D49622, the next available address would be D49624. (A3 would consume both D49622 and D49623.) An address value of D49621, being odd, would be an invalid setting for any argument with a Long device size.

Table T5.19 Arguments to the "Preset Encoder Position" Macro

5.3.18 Arguments to the "Preset to Encoder ISMD23E2" Macro

Argument	Device Type	Device Size	Range of Values	Description
A1	D (Data Register)	Word		Starting Address of the output registers assigned to the ISMD23E2 during Modbus TCP Client configuration. (D49120 to D49230 in the sample programs.)

Table T5.20 Arguments to the "Preset to Encoder ISMD23E2" Macro

5.4 Add a User-defined Macro to Ladder Logic

Adding a User-defined Macro to your ladder logic requires you to set the initial conditions for the rung, select the macro, and map the macro's arguments to the Device Addresses chosen in the previous step.

5.4.1 Set initial conditions

Most macros should be triggered once to initiate a data transfer to the ISMD23E2. Therefore, a **Single Output Up** instruction, (SOTU), should be added to the rung as part of the condition. The exceptions to this guideline are the Jog and Registration Moves.

5.4.2 Choose the desired macro to add to the ladder.

To add a User-defined Macro to the ladder, first, right-click where you wish to add the macro. In the popup menu that appears, hover over the "**Macro Instructions ▶**" item until a second menu appears. From that menu, select **UMACRO (User-defined Macro)**. The UMACRO (User-defined Macro) dialog box will appear.

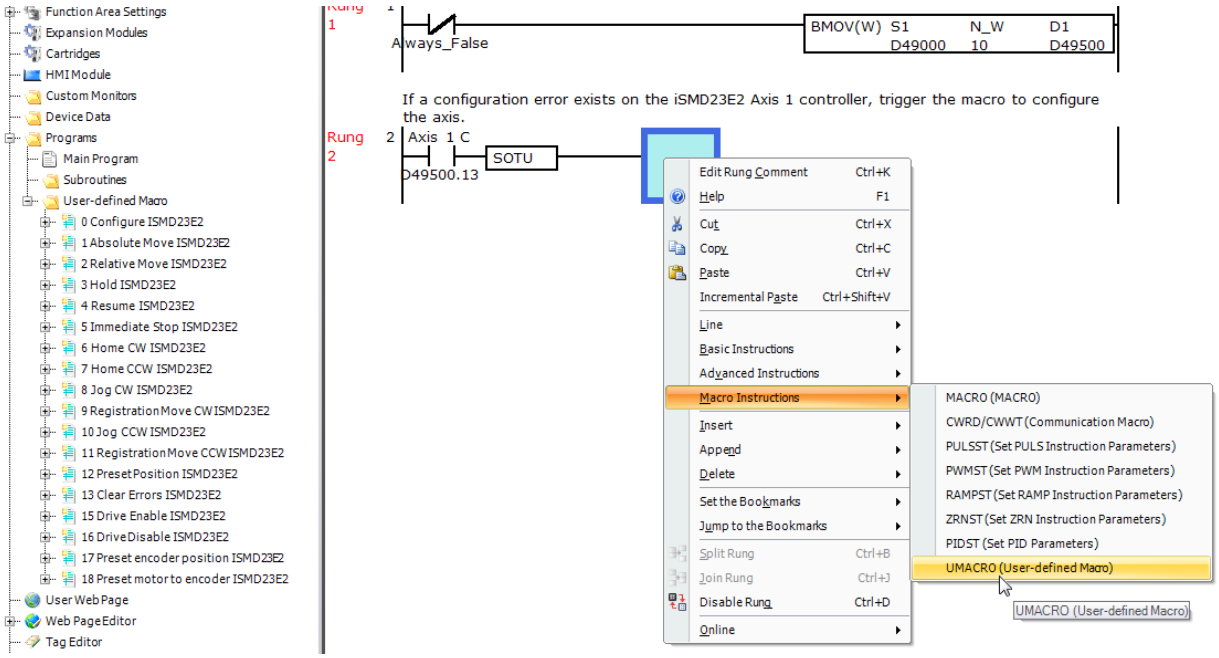


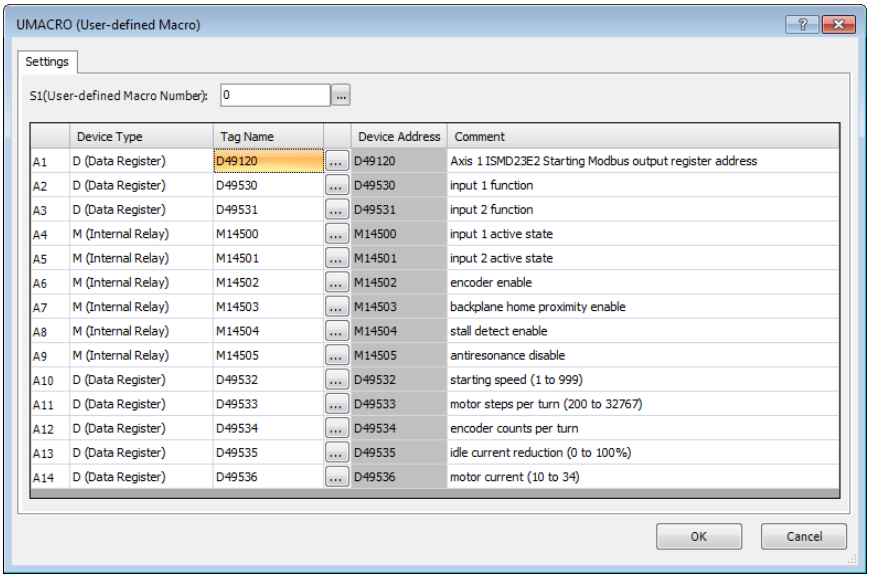
Figure T5.2 Selecting a User-defined Macro

5.4.3 Select the Proper Macro by Macro Number

Each macro is assigned a number between 0 and 255. When user-defined macros are viewed in the Project Window, these numbers are shown before the name of the macro. (See figure T5.2 above for an example.) To select the proper macro, enter the correct number in the "S1 (User-defined Macro Number):" field and press the [Tab] key. The program will open a confirmation window stating that the current settings will be lost and the argument list will be updated. Click on the [Yes] button to confirm the change.

5.4.4 Map Arguments to Device Addresses

(Continuing from the previous step.) The **UMACRO (User-defined Macro)** dialog box now displays a list of the macro's arguments along with their required Device Type. The Tag Name fields are used to map the Device Addresses to the macro's arguments. If a tag name has been defined, this name can be typed directly into the field. Optionally, the actual Device Address can be entered into the field. Finally the ellipsis button [...] can be clicked to open the **Tag Name Editor** dialog box. Once opened, is can be used to chose the desired Device Address. Figure T5.3 below shows a **UMACRO (User-defined Macro)** dialog box for the Configure ISMD23E2 macro after Device Addresses have been mapped to the macro's arguments.



	Device Type	Tag Name		Device Address	Comment
A1	D (Data Register)	D49120	...	D49120	Axis 1 ISMD23E2 Starting Modbus output register address
A2	D (Data Register)	D49530	...	D49530	input 1 function
A3	D (Data Register)	D49531	...	D49531	input 2 function
A4	M (Internal Relay)	M14500	...	M14500	input 1 active state
A5	M (Internal Relay)	M14501	...	M14501	input 2 active state
A6	M (Internal Relay)	M14502	...	M14502	encoder enable
A7	M (Internal Relay)	M14503	...	M14503	backplane home proximity enable
A8	M (Internal Relay)	M14504	...	M14504	stall detect enable
A9	M (Internal Relay)	M14505	...	M14505	antiresonance disable
A10	D (Data Register)	D49532	...	D49532	starting speed (1 to 999)
A11	D (Data Register)	D49533	...	D49533	motor steps per turn (200 to 32767)
A12	D (Data Register)	D49534	...	D49534	encoder counts per turn
A13	D (Data Register)	D49535	...	D49535	idle current reduction (0 to 100%)
A14	D (Data Register)	D49536	...	D49536	motor current (10 to 34)

Figure T5.3 Argument Assignments

Notes

REFERENCE 4: ISMD23E2 INPUT DATA FORMATS

Configuration Mode Input Data Format

The format for the Network Input Data when an ISMD23E2 is in Configuration Mode is shown below.

Modbus TCP Register	Configuration Data
0	Mirror of Output Configuration Word 0
1	Mirror of Output Configuration Word 1
2	0000
3	Mirror of Starting Speed
4	Mirror of Motor Steps/Turn
5	0000
6	Mirror of Encoder_Resolution
7	Mirror of Idle Current Percentage
8	Mirror of Motor Current (X10)
9	0000 or Status message when writing Configuration data to flash memory.

Table R4.1 Network Input Data Format: Configuration Mode

Configuration Word 0 Format (Word 0)

When the Configuration data is valid and accepted, this word mirrors the value of the Configuration Word 0 written to the ISMD23E2. When the configuration data written to it is invalid, the unit remains in Command Mode and sets the Configuration_Error bit. The Configuration_Error bit is bit 13 of the first word written back to the IDEC host controller while in Command Mode. The format of this word when an error exists is explained in the *Status Word 0 Format* section starting on page 60.

Invalid Configurations

The following configurations are invalid:

1) Setting any of the reserved bits in the configuration words.
2) Setting any parameter to a value outside of its valid range. This includes setting the Starting Speed to a value greater than 999.
3) You configure two or more inputs to have the same function, such as two CW Limit Switches. (An error does not occur if both are configured as General Purpose Inputs.)
4) Setting the *Stall Detection Enable Bit* without configuring the ISMD23E2 to use its built in encoder.
5) Setting the Input Configuration bits for any input to "111". See table R6.3 on page 73 for more information. When using the IDEC User-defined Macro for configuration, the input configuration value must be between 0 and 6.

Factory Default Values

The following configuration parameter settings are programmed at the factory.

Configuration Word 0 = 0x8400

- Input 1 Function: General Purpose Input
- Input 2 Function: General Purpose Input
- Encoder: Enabled if an encoder was ordered as part of the unit.
- Backplane Proximity Bit: Disabled
- Stall Detection: Disabled
- Antiresonance: Enabled

Configuration Word 1 = 0x0383

- Input 1 Active Level: High. (NO contacts)
- Input 2 Active Level: High. (NO contacts)
- Data Format: IDEC

Remaining Parameters

- Starting Speed: 1
- Motor Steps/Turn: 2,048
- Encoder Pulses/Turn: 2,048
- Idle Current Percentages: 20%
- Motor Current: 2.8 Arms

Command Mode Input Data Format

The format for the Network Input Data when the ISMD23E2 is in Command Mode is shown below.

Register	Command Mode Input Data
0	Status Word 0
1	Status Word 1
2	32 bit Motor Position: Upper 16 bits
3	32 bit Motor Position: Lower 16 bits
4	32 bit Encoder Position: Upper 16 bits
5	32 bit Encoder Position: Lower 16 bits
6	32 bit Trapped Encoder Position: Upper 16 bits
7	32 bit Trapped Encoder Position: Lower 16 bits
8	Programmed Motor Current (X10)
9	Value of Acceleration Jerk Parameter

Table R4.2 Network Input Data Format: Command Mode

Status Word 0 Format

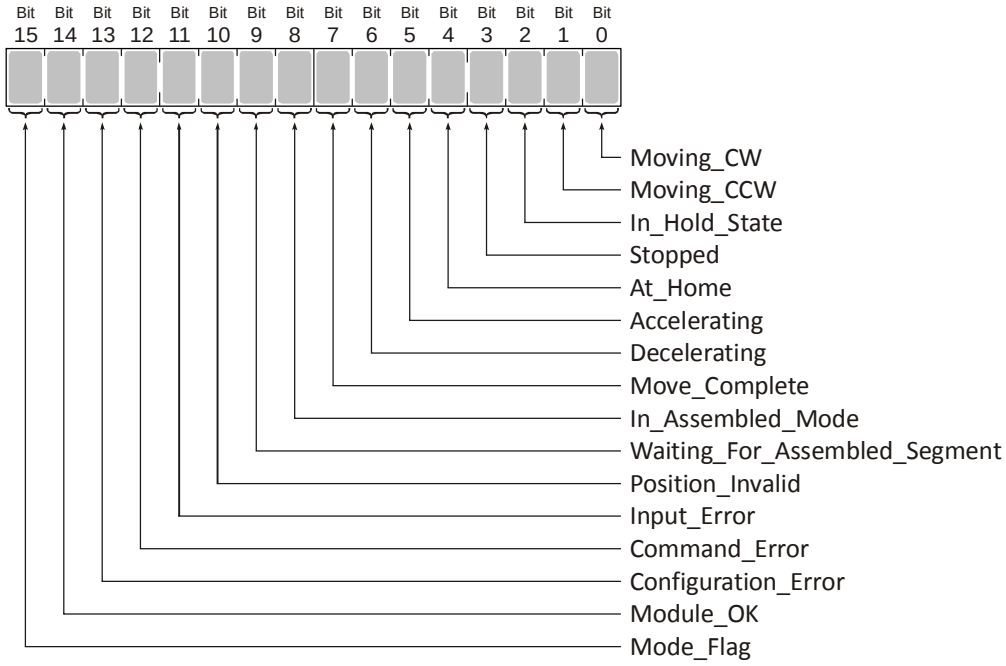


Figure R4.1 Command Mode: Status Word 0 Format

Bit 0: Moving_CW – Set to "1" when the motor is rotating in a clockwise direction.

Bit 1: Moving_CCW – Set to "1" when the motor is rotating in a counter-clockwise direction.

Bit 2: In_Hold_State – Set to "1" when a move command has been successfully brought into a Hold State. Hold States are explained is the Controlling Moves In Progress section starting on page 22.

Bit 3: Stopped – Set to "1" when the motor is not in motion. Note that this is stopped for any reason, not just a completed move. For example, an Immediate Stop command during a move will set this bit to "1", but the Move_Complete Bit, (bit 7 above) will not be set.

Bit 4: At_Home – Set to "1" when a homing command has completed successfully, "0" at all other times.

Bit 5: Accelerating – Set to "1" when the present move is accelerating. Set to "0" at all other times.

Bit 6: Decelerating – Set to "1" when the present move is decelerating. Set to "0" at all other times.

Bit 7: Move_Complete – Set to "1" when the present move command completes without error. This bit is reset to "0" when the next move command is written to the ISMD23E2, when the position is preset, or a Reset Errors command is issued to the unit. This bit is also set along with the Command_Error bit (Bit 12 of this word), when any Jog Move or Registration Move parameters are outside of their valid ranges. This bit is not set on a command error for any other type of command. Finally, this bit is not set at the end of a homing operation.

Bit 8: In_Assembled_Mode – The ISMD23E2 sets this bit to signal the host that it is ready to accept assembled move profile programming data. Assembled Moves do not have User-defined Macros at this point in time. The use of this bit is explained in the *Assembled Move Programming* section of this manual starting on page 69.

Bit 9: Waiting_For_Assembled_Segment – The ISMD23E2 sets this bit to tell the host that it is ready to accept the data for the next segment of your assembled move profile. Assembled Moves do not have User-defined Macros at this point in time. The use of this bit is explained in the *Assembled Move Programming* section of this manual starting on page 69.

Bit 10: Position_Invalid – "1" when:

- A configuration is written to the ISMD23E2
- The motor position has not been preset or the machine has not been homed
- The network connection has been lost and re-established
- An Immediate or Emergency Stop has occurred
- An End Limit Switch has been reached
- A motor stall has been detected.

Absolute moves cannot be performed while the position is invalid.

Bit 11: Input_Error – "1" when:

- Emergency Stop input has been activated
- Either of the End Limit Switches activates during any move operation except for homing
- Starting a Jog Move in the same direction as an active End Limit Switch
- If the opposite End Limit Switch is reached during a homing operation.

This bit is reset by a *Reset Errors* command. The format of the command is given on page 83.

Bit 12: Command_Error – "1" when an invalid command has been written to the ISMD23E2. This bit can only be reset by the *Reset_Errors* bit, Command Word 0, Bit 10.

Bit 13: Configuration_Error – "1" on power up before a valid configuration has been written to the ISMD23E2 or after any invalid configuration has been written to the unit. "0" when the ISMD23E2 has a valid configuration in memory.

Note➤ When in Command Mode, bit 13 of word 0 is set to "1" when there is a configuration error. When in Configuration Mode, bit 13 of word 0 is set to "1" when stall detection is enabled. When using the state of bit 13 of word 0 in your logic, always include the state of bit 15 of word 0 to assure that you are only acting on the bit when in the proper mode.

Bit 14: Module_OK – "1" when the ISMD23E2 is operating without a fault, "0" when an internal fault condition exists.

Bit 15: Mode_Flag – Set to "1" if in Configuration Mode, and set to "0" if in Command Mode.

Status Word 1 Format

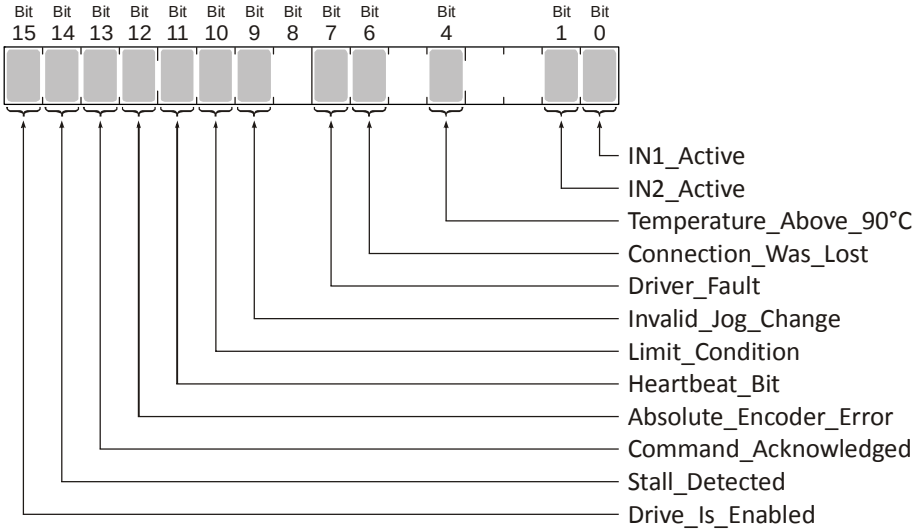


Figure R4.2 Command Mode: Status Word 1 Format

Bit 0: IN1_Active – "1" when Input 1 is in its active state. The active state of the input is programmed as explained in the *Configuration Word 1 Format* section starting on page 73.

Bit 1: IN2_Active – "1" when Input 2 is in its active state. The active state of the input is programmed as explained in the *Configuration Word 1 Format* section starting on page 73.

Bits 2-3: Reserved – Will always equal zero.

Bit 4: Temperature_Above_90°C – This bit is set to "1" when the processor internal temperature exceeds 90°C. At this point, the heatsink temperature is typically near 83°C. If this bit trips often and you want to lower the operating temperature of the unit, consider changing how the device is mounted, or installing a fan to force additional airflow through the device.

Bit 5: Reserved – Will always equal zero.

Bit 6: Connection_Was_Lost – If the *physical* network connection is lost at any time, this bit will be set when the connection is re-established. The Input_Error bit will also be set. Note that this bit is not set if the communication loss is on the protocol level, not the physical level.

Bit 7: Drive_Fault – If the drive section of the ISMD23E2 is enabled, this bit will be a "1" during a Over Temperature Fault. This fault can only be cleared by cycling power to the ISMD23E2.

Bit 8: Reserved – Will always equal zero.

Bit 9: Invalid_Jog_Change – Set during a Jog Move if parameters are changed to invalid values. Parameters that can be changed during a Jog Move are Programmed Speed, Acceleration, and Deceleration.

Bit 10: Limit_Condition – This bit is set if an End Limit Switch is reached during a move. This bit will be reset when the Limit Switch changes from its active to inactive state, or when a Reset Errors Command is issued.

Bit 11: Heartbeat_Bit – This bit will change state approximately every 500 milliseconds. Monitor this bit to verify that the unit and network connection are operating correctly. Note that this bit is only available while the Networked Drive is in Command Mode.

Bit 12: Absolute Encoder Error – Only available on units with the absolute encoder, this bit is set to "1" under the following conditions:

- The shaft was subject to acceleration in excess of 160,000°/sec^2 (444.4 rev/sec^2) while power was removed from the unit
- The internal battery is fully discharged or damaged
- The unit itself is damaged

If this bit is set, cycle power to the unit. If the bit remains set, contact IDEC technical support for assistance.

Bit 13: Command_Acknowledge – Normally "0". This bit is set to "1" when one of the following commands completes successfully:

- Preset Position
- Preset Encoder Position
- Reset Errors

This bit resets to "0" when the command bit is reset to "0" by the host controller.

Bit 14: Stall_Detected – Set to "1" when a motor stall has been detected.

Bit 15: Drive_Is_Enabled – Set to "1" when the motor drive section of the Networked Drive is enabled and current is available to the motor. Set to "0" when the motor drive section is disabled. If this bit is set to "1", the motor current remains present when an E-Stop input is active. Motor current is removed if there is a Drive_Fault (Bit 7 above) regardless of the state of this bit. Motor current is also removed if the motor is idle and Idle Current Reduction is programmed to its *To 0%* setting.

Notes on Clearing a Drive Fault

A Drive Fault occurs when there is an over temperature condition. When a Drive Fault occurs, the ISMD23E2 sets bit 7 of the Status Word 1 word in the Network Input Data. The only way to clear this fault is to lower the temperature of the motor and cycle power to the ISMD23E2.

TASK 6: ADDITIONAL LOGIC

Introduction

The IDEC User-defined Macros simplify issuing commands to the ISMD23E2. Monitoring the state of the ISMD23E2 once the command is issued is application specific and must be written. The ISMD23E2 sample program available on the IDEC website shows one method of monitoring the ISMD23E2 and controlling when commands are issued.

The following sections give guidelines for writing the additional logic needed to control an ISMD23E2.

6.1 Buffer Input Data

It is common practice to buffer the input data from the ISMD23E2 to guarantee that it does not change during a program scan. IDEC sample programs use a BMOV instruction to copy data from the Modbus TCP registers to a buffer in the D49500 to D49999 range. This instruction is conditioned by an NC relay tied to a bit defined as Always_False. This buffers the data on every program scan and must occur before the data is used in the program.

6.2 Command Bits Must Transition

> **Note►** Commands are only accepted when the command bit makes a 0→1 transition. The easiest way to meet this requirement is to write a value of zero into the first of the output registers assigned to the ISMD23E2 before writing the next command. These registers are in the range of D49120 to D49239 in the IDEC sample program. See T5.1, *Reserved Memory Addresses* on page 47 for the memory layout used by the IDEC sample program.

This condition also applies when switching from Configuration Mode to Command Mode. Assume a bit is set in the first of the registers assigned to the ISMD23E2 while in Configuration Mode. If a switch is made to Command Mode with the same bit set, the command will not occur because the bit must transition between writes to the unit.

6.2.1 Commands that do not cause motion: Clear Errors, preset commands, etc.

In the unlikely event that a command in this group is issued successively, its command bit must be reset and held for at least one Modbus TCP update. The time it must be held is dependent on the amount of TCP traffic in your application.

6.2.2 Jog and Registration Moves

Unlike other motion commands, the Jog and Registration Moves only occur while the command bit is set. Your program must monitor machine conditions to determine when to reset the command bit to zero. Once the command bit is off, your program should monitor the Moving_CW, Moving_CCW, or Stopped status bits to verify that the axis has stopped moving before issuing another command.

6.2.3 Absolute Move, Relative Move, and Find Home

Once issued, these commands will run to their completion regardless of the state of their command bit. Once the command has been issued, your program can monitor the Moving_CW, Moving_CCW, or Stopped status bits to verify the axis is in motion and reset the command word at that point. Unless the commanded move is very short, a network update will occur while motion is occurring and the ISMD23E2 will see the reset command bit. As with the Jog and Registration Moves, the program should monitor the Moving_CW, Moving_CCW, or Stopped status bits to verify that the axis has stopped moving before issuing another command.

Notes

OPTIONAL TASK A: CONFIGURE YOUR NETWORK INTERFACES

A.1 Firewall Settings

Firewalls are hardware devices or software that prevent unwanted network connections from occurring. Firewall software is present in Windows XP and above and it may prevent your computer for communicating with the ISMD23E2. Configuring your firewall to allow communication with the ISMD23E2 is beyond the scope of this manual.

It may be necessary to temporarily disable any firewall software while using the Ethernet Tool. This is typically done from the Control Panel. You should re-enable the firewall once you have finished using the tool.

A.2 Disable All Unused Network Interfaces

Routing and default gateway setting on your computer can interfere with the proper operation of the Ethernet Tool. The software uses broadcast packets to locate devices on the network, and sometimes these packets are sent out through the default gateway instead of the interface attached to the ISMD23E2. The easiest way to avoid this problem is to temporarily disable all network interfaces that are not attached to the stepper drive.

Note► This includes all wireless interfaces as well as all Bluetooth interfaces.

A.3 Configure Your Network Interface

Before you can communicate with the ISMD23E2, your network interface must be on the same subnet as the drive.

Note► The rest of this procedure assumes you are using the 192.168.1.xxx subnet. If you are not, you will have to adjust the given network addresses accordingly.

The easiest way to check the current settings for your NIC is with the 'ipconfig' command.

- For Vista and Windows 7, click on the [Start] button, and type "cmd" in the "*Search programs and files*" text box. Press [Enter] on the keyboard.
- For Windows 8 and 10, press the [Win+X] keys and select "Command Prompt" from the resulting popup. There is no need to run the command prompt as the administrator, so do not select "Command Prompt (Admin)".

A DOS like terminal will open. Type in 'ipconfig', press [Enter] on the keyboard and the computer will return the present Address, Subnet Mask, and Default Gateway for all of your network interfaces. If your present address is 192.168.1.xxx, where 'xxx' does not equal 50, and your subnet mask is 255.255.255.0, then you are ready to configure your ISMD23E2. Figure 0.1 shows the output of an ipconfig command that shows the "Local Area Connection 2" interface on the 192.168.1.xxx subnet.

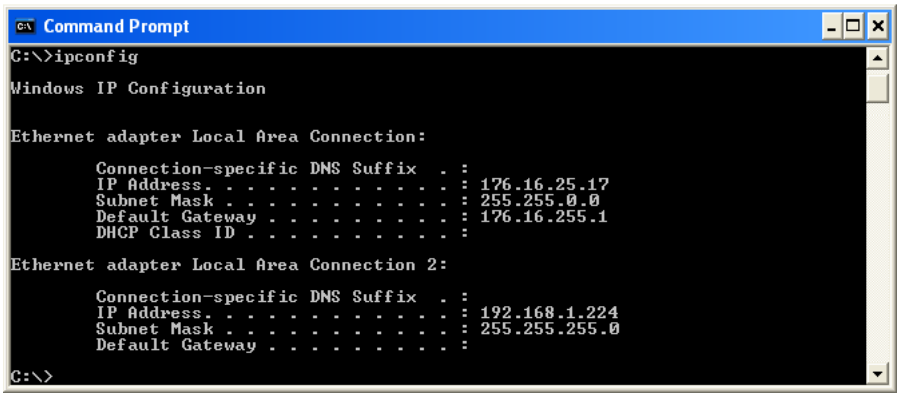


Figure 0.1 ipconfig Command

If your present address in not in the 192.168.1.xxx range, type in 'ncpa.cpl' at the command prompt and hit [Enter] on the keyboard.

- For Vista and Windows 7, this open the *Network Connections* window. Double click on the appropriate interface. In the window that opens, select "Internet Protocol Version 4 (TCP/IP v4)" from the list and then click on the [Properties] button.
- For Windows 8 and 10, this open the *Network Connections* window. Double click on the appropriate interface. In the window that opens, select "Internet Protocol Version 4 (TCP/IP v4)" from the list and then click on the [Properties] button.

Set the address and subnet mask to appropriate values. (192.168.1.1 and 255.255.255.0 will work for an ISMD23E2 that has factory default settings.) The default gateway and DNS server settings can be ignored.

A.4 Test Your Network Interface

Going back to the terminal you opened in the last step, type in 'ping aaa.bbb.ccc.ddd' where 'aaa.bbb.ccc.ddd' in the IP address of the ISMD23E2. The computer will ping the unit and the message "Reply from aaa.bbb.ccc.ddd: bytes=32 time<10ms TTL=255" should appear four times.

If the message "Request timed out." or "Destination host unreachable" appears, then one of four things has occurred:

- You set a new IP address, but have not yet cycled power to the ISMD23E2
- You did not enter the correct address in the ping command.
- The IP address of the ISMD23E2 is not set correctly.
- The ISMD23E2 and the computer are not on the same subnet.

REFERENCE 5: ASSEMBLED MOVES

Introduction

The ISMD23E2 contains functionality that is not programmable through custom macros at this time. This reference section describes this additional functionality.

Assembled Moves

All of the moves explained so far must be run individually to their completion or must be stopped before another move can begin. The ISMD23E2 also gives you the ability to pre-assemble more complex profiles from a series of relative moves that are then run with a single command. Each Assembled Move consists of two to sixteen segments. Assembled Moves are programmed through a hand shaking protocol that uses the input and output registers assigned to the unit. The protocol is fully described in the *Assembled Move Programming* section on page 69.

Two types of Assembled Moves exist in an ISMD23E2:

- **Blend Move -** A Blend Move gives you the ability to string multiple relative moves together and run all of them sequentially without stopping the shaft between moves. A Blend Move can be run in either direction, and the direction is set when the command is issued. The direction of motion cannot be reversed within a Blend Move.
- **Dwell Move -** A Dwell Move gives you the ability to string multiple relative moves together, and the ISMD23E2 will stop between each move for a programed *Dwell Time*. Because motion stops between each segment, a Dwell Move allows you to reverse direction during the move.

Blend Move

Each Relative Move defines a *segment* of the Blend Move. The following restrictions apply when programming Blend Moves.

1) Each segment of the Blend Move must be written to the ISMD23E2 before the move can be initiated.
   - The ISMD23E2 supports Blend Moves with up to sixteen segments.
2) Each segment is programmed as a relative move. Blend Moves cannot be programmed with absolute coordinates.
3) All segments run in the same direction. The sign of the target position is ignored and only the magnitude of the target position is used. The move's direction is controlled by the bit pattern used to start the move. If you want to reverse direction during your move, consider using the *Dwell Move* which is explained starting on page 68.
4) The Programmed Speed of each segment must be greater than or equal to the Starting Speed.
5) The Programmed Speed can be the same between segments. This allows you to chain two segments together.
6) For all segments except for the last one, the programmed position defines the end of the segment. For the last segment, the programmed position defines the end of the move.
7) Once you enter a segment, that segment's programmed acceleration and deceleration values are used to change the speed of the motor.
8) The blend segment must be long enough for the acceleration or deceleration portions of the segment to occur. If the segment is not long enough, the motor speed will jump to the speed of the next segment without acceleration or deceleration.

The figure below shows a three segment Blend Move that is run twice. It is first run in the clockwise direction, and then in the counter-clockwise direction.

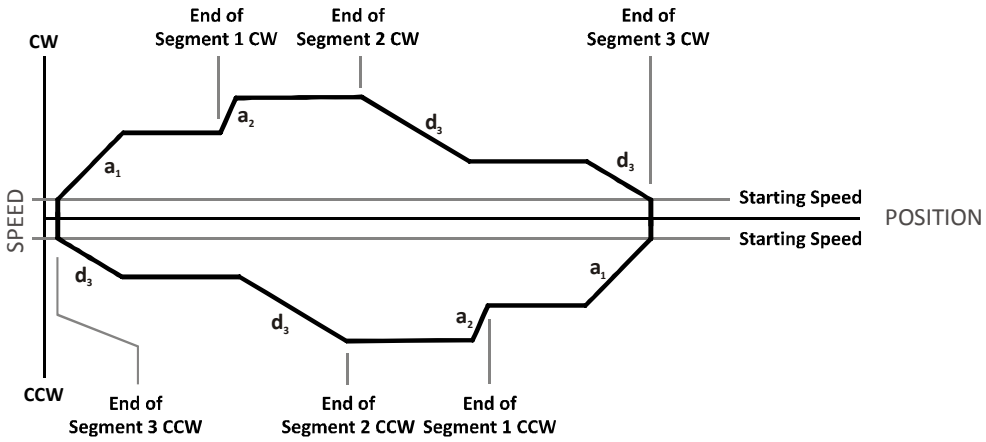


Figure R5.1 Blend Move

Note► 1) The deceleration value programmed with segment 3 is used twice in the segment. Once to decelerate from the Programmed Speed of segment 2 and once again to decelerate at the end of the move.

2) You do not have to preset the position or home the machine before you can use a Blend Move. Because the Blend Move is based on Relative Moves, it can be run from any location.

3) The Blend Move is stored in the internal memory of the ISMD23E2 and can be run multiple times once it is written to the unit. The Blend Move data stays in memory until power is removed, the unit is sent new Configuration Data, or a new Blend or Dwell Move is written to the unit. As described in *Saving an Assembled Move in Flash* on page 70, it is also possible to save a Blend Move to flash memory. This move is restored on power up and can be run as soon as you can issue the command.

4) There are two control bits used to specify which direction the Blend Move is run in. This gives you the ability to run the Blend Move in either direction.

Controlled Stop Conditions

- The move completes without error.
- You toggle the Hold_Move control bit in the Network Output Data. When this occurs, the ISMD23E2 decelerates the move at the deceleration rate of the present segment to the Starting Speed and ends the move. Note that your final position will most likely not be the one you commanded. A Blend Move that is brought to a controlled stop with the Hold_Move bit cannot be restarted. The use of the Hold_Move bit is explained in the *Controlling Moves In Progress* section starting on page 26.

Immediate Stop Conditions

- The Immediate_Stop bit makes a 0→1 transition in the Network Output Data.
- A positive transition on an input configured as an E-Stop Input.
- A CW or CWW Limit Switch is reached. If the limit that is reached is the same as the direction of travel, for example, hitting the CW limit while running a CW move, a *Reset Errors* command must be issued before moves are allowed in that direction again. If the limit that is reached is opposite the direction of travel, a *Reset Errors* command does not have to be issued.

Dwell Move

A Dwell Move gives you the ability to string multiple relative moves together and run all of them sequentially with a single start condition. Like a Blend Move, a Dwell Move is programmed into an ISMD23E2 as a series of relative moves before the move is started.

Unlike a Blend Move, the motor is stopped between each segment of the Dwell Move for a programed *Dwell Time*. The Dwell Time is programmed as part of the command that starts the move. The Dwell Time is the same for all segments. Because the motor is stopped between segments, the motor direction can be reversed during the move. The sign of the target position for the segment determines the direction of motion for that segment. Positive segments will result in clockwise shaft rotation while a negative segment will result in a counter-clockwise shaft rotation.

The following figure shows a drilling profile that enters the part in stages and reverses direction during the drilling operation so chips can be relieved from the bit. You *could* accomplish this Dwell Move with a series of six relative moves that are sent down to the ISMD23E2 sequentially. The two advantages of a Dwell Move in this case are that the ISMD23E2 will be more accurate with the Dwell Time then you can be in your control program, and Dwell Moves simplify your program's logic.

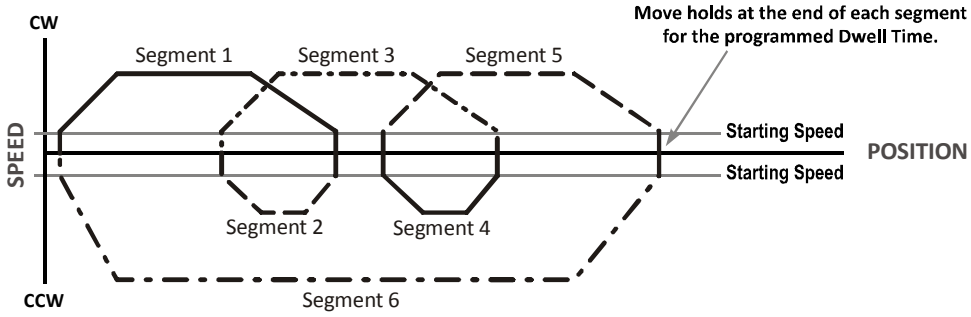


Figure R5.2 Dwell Move

Note► 1) You do not have to preset the position or home the machine before you using a Dwell Move. The Dwell Move is based on Relative Moves, and can be run from any location.

2) The Dwell Move is stored in the internal memory of the ISMD23E2 and can be run multiple times once it is written to the unit. The Dwell Move data stays in memory until power is removed, the unit is sent new Configuration Data, or a new Blend or Dwell Move is written to the iSDM23E2. As described in *Saving an Assembled Move in Flash* on page 70, it is also possible to save a Dwell Move to flash memory. This move is restored on power up and can be run as soon as you can issue a command.

Controlled Stop Conditions

- The move completes without error.
- You toggle the Hold_Move control bit in the Network Output Data. When this occurs, the ISMD23E2 decelerates the move at the deceleration rate of the present segment to the Starting Speed and ends the move. Note that your final position will most likely not be the one you commanded. A Dwell Move that is brought to a controlled stop with the Hold_Move bit cannot be restarted.

Immediate Stop Conditions

- The Immediate_Stop bit makes a 0→1 transition in the Network Output Data.
- A positive transition on an input configured as an E-Stop Input.
- A CW or CWW Limit Switch is reached. If the limit that is reached is the same as the direction of travel, for example, hitting the CW limit while running a CW move, a *Reset Errors* command must be issued before moves are allowed in that direction again. If the limit that is reached is opposite the direction of travel, a *Reset Errors* command does not have to be issued.

Controlling an Assembled Move In Progress

A Blend or Dwell Move can be placed in a Hold state but cannot be resumed. This give you the ability to prematurely end an Assembled Move with a controlled stop. The Assembled Move is not erased from memory and can be run again without having to reprogram it.

Assembled Move Programming

All of the segments in a Blend or Dwell Move must be written to the ISMD23E2 before the move can be run. Segment programming is controlled with two bits in the Network Output Data and two bits in the Network Input Data. Blend and Dwell Moves are programmed in exactly the same way. When you start the move, a bit in the command data determines which type of Assembled Move is run. In the case of a Blend Move, the sign of each segment's Target Position is ignored and all segments are run in the same direction. In the case of a Dwell Move, the sign of each segment's Target Position determines the direction of the segment. For Dwell Moves, the Dwell Time is sent to the ISMD23E2 as part of the command.

Control Bits – Output Data

- **Program_Assembled bit (Command Word 0, bit 12) –** A 0→1 transition on this bit tells the ISMD23E2 that you want to program a Blend or Dwell Move Profile. The ISMD23E2 will respond by setting the *In_Assembled_Mode* bit in the Network Input Data. At the beginning of the programming cycle, the ISMD23E2 will also set the *Waiting_For_Assembled_Segment* bit to signify that it is ready for the first segment.

- **Read_Assembled_Data bit (Command Word 0, bit 11) –** A 0→1 transition on this bit tells the ISMD23E2 that the data for the next segment is available in the remaining data words.

Control Bits – Input Data

- **In_Assembled_Mode bit (Status Word 0, bit 8) –** The ISMD23E2 sets this bit to signal that it is ready to accept segment programming data in the remaining output data words. The actual transfer of segment data is controlled by the *Waiting_For_Assembled_Segment* and *Read_Assembled_Data* bits.
- **Waiting_For_Assembled_Segment bit (Status Word 0, bit 9) –** A 0→1 transition on this bit from the ISMD23E2 is the signal to the host that the ISMD23E2 is ready to accept the data for the next segment.

Programming Routine

1) The host sets the *Program_Assembled* bit in the Network Output Data.
2) The ISMD23E2 responds by setting both the *In_Assembled_Mode* and *Waiting_For_Assembled_Segment* bits in the Network Input Data.
3) When the host detects that the *Waiting_For_Assembled_Segment* bit is set, it writes the data for the first segment in the Network Output Data and sets the *Read_Assembled_Data* bit.
4) The ISMD23E2 checks the data, and when finished, resets the *Waiting_For_Assembled_Segment* bit. If an error is detected, it also sets the *Command_Error* bit.
5) When the host detects that the *Waiting_For_Assembled_Segment* bit is reset, it resets the *Read_Assembled_Data* bit.
6) The ISMD23E2 detects that the *Read_Assembled_Data* bit is reset, and sets the *Waiting_For_Assembled_Segment* bit to signal that it is ready to accept data for the next segment.
7) Steps 3 to 6 are repeated for the remaining segments until the entire move profile has been entered. The maximum number of segments per profile is sixteen.
8) After the last segment has been transferred, the host exits Assembled Move Programming Mode by resetting the *Program_Assembled* bit.
9) The ISMD23E2 resets the *In_Assembled_Mode* and the *Waiting_For_Assembled_Segment* bits.

Saving an Assembled Move in Flash

The ISMD23E2 also contains the *Save_Assembled_to_Flash* bit that allows you to store the Assembled Move in flash memory. This allows you to run the Assembled Move right after power up, without having to go through a programming sequence first. To use this bit, you follow the above programming routine with the *Save_Assembled_to_Flash* bit set. When you reach step 9 in the sequence, the ISMD23E2 responds by resetting the *In_Assembled_Mode* and *Transmit Blend Move Segments* bits as usual and then it will flash the Status LED. If the LED is flashing green, the write to flash memory was successful. If it flashes red, then there was an error in writing the data. In either case, power must be cycled to the ISMD23E2 before you can continue. With a limit of 10,000 write cycles, the design decision that requires you to cycle power to the ISMD23E2 was made to prevent an application from damaging the module by continuously writing to it.

Commanding an Assembled Move

Three bits in the output data are used to start an Assembled Move.

- **Run_Assembled_Move bit (Command Word 0, bit 13) –** A 0→1 transition on this bit commands an Assembled move to begin. Bits in Command Word 1 control the type of move and its direction.
- **Assembled_Move_Type bit (Command Word 1, bit 9) –** When this bit equals “0”, a Blend Move is started on the 0→1 transitionwhen of the Run_Assembled_Move bit. When this bit equals “1”, a Dwell Move is started on the transition. The direction of a Blend Move is controlled by the Blend Move Direction bit, (Command Bits LSW, Bit 4). In a Dwell Move, the Dwell Time between segments is programmed in Word 9 of the command data.
- **Blend_Move_Direction bit (Command Word 0, bit 4) –** This bit determines the direction of rotation of a Blend Move. Set to “0” for a clockwise Blend Move, ‘1’ for a counter-clockwise Blend Move.

REFERENCE 6: CONFIGURATION MODE DATA FORMAT

Introduction

This chapter covers the formats of the Network Output Data used to configure the ISMD23E2. An ISMD23E2 requires twenty bytes of Output Data as well as 20 bytes for Input Data. The format of this data is a mix of sixteen bit and thirty-two bit integers.

> **Note►** The custom macros supplied by IDEC simplify the configuration and programming of the ISMD23E2 units. The data in this reference is presented as background information to aid in troubleshooting.

Modes of Operation

An ISMD3E2 has two operating modes, Configuration Mode and Command Mode. You switch between these modes by changing the state of a single bit in the Network Output Data.

Configuration Mode

Configuration Mode gives you the ability to select the proper configuration for your application without having to set any switches. The ladder logic needed to configure a unit is included in the sample programs available on the IDEC website. This method simplifies change over if the unit ever needs to be replaced.

A valid configuration can be saved to the unit's flash memory and the ISMD23E2 will use this as a default configuration on every power up. If you use this method, you can still write down a new configuration to the unit at any time. The new configuration is stored in RAM and is lost on power down unless you issue a command to store the new configuration in flash.

Command Mode

This mode gives you the ability to program and execute stepper moves, and reset errors when they occur. An ISMD23E2 will always power up in this mode. The command data formats are described in the following chapter.

Power Up Behavior

An ISMD23E2 will always power up in Command Mode. The ISMD23E2 will use its stored configuration data to configure the unit. The ISMD23E2 will then check for valid network command data and will only enable the motor drive section if the Enable_Drive bit is set.

Data Format

An ISMD23E2 requires ten words of Output Data as well as ten words of Input Data. These words are broken down into a combination of sixteen and thirty-two bit integers.

Thirty-two bit values are transmitted most significant word first (big endian). Table R6.1 below shows the format of the data as it is stored and transmitted.

	32 bit Integer Format	
Value	**First Word**	**Second Word**
12	0x0000	0x000C
-12	0xFFFF	0xFFF4
1,234,567	0x0012	0xD687
-7,654,321	0xFF8B	0x344F

Table R6.1 Position Data Format Examples

Output Data Format

The correct format for the Network Output Data when the ISMD23E2 is in Configuration Mode is shown below.

Register	Configuration Data	Range
1024	Configuration Word 0	See below
1025	Configuration Word 1	See below
1026	Reserved	Set to zero
1027	Starting Speed (steps/sec)	1 to 999
1028	Motor Steps/Turn	200 to 32,767
1029	Reserved	Set to zero
1030	Encoder_Resolution	Set to 1,024, 2,048, or 4,096 for incremental encoder. Set to 2,048 for absolute encoder. If no encoder, set to 2,048.
1031	Idle Current Percentage	0 to 100%
1032	Motor Current (X10)	1 to 34, Represents 0.1 to 3.4 Arms
1033	Reserved	Set to zero

Table R6.2 Network Output Data Format: Configuration Mode

Configuration Word 0 Format

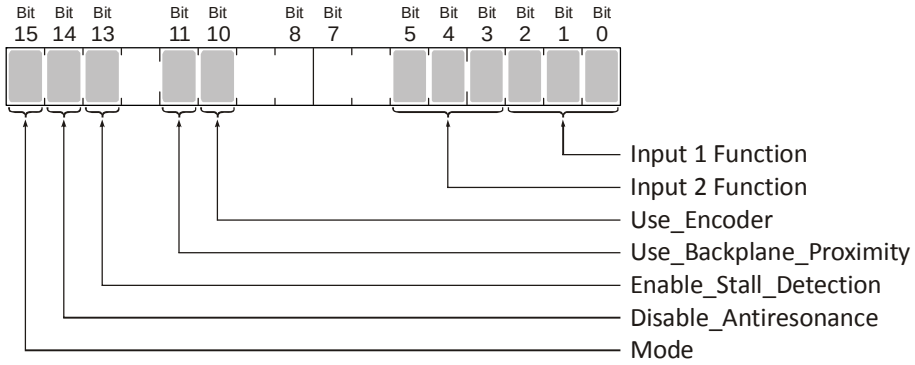


Figure R6.1 Configuration Mode: Configuration Word 0 Format

Bits 2-0: Input 1 Function – See the table on the following page.

Bits 5-3: Input 2 Function – See the table on the following page.

Bits 9-6: Reserved – Must equal zero.

Bit 10: Use_Encoder – "0" when the built-in encoder is not used. "1" to enable the built-in absolute or incremental encoder. Only valid with the ISMD23E2 units that have the built-in encoder. You must also program the Encoder_Resolution parameter in configuration word 6.

Bit 11: Use_Backplane_Proximity – "0" when the Backplane_Proximity_Bit is not used when homing the ISMD23E2. "1" when the Backplane_Proximity_Bit is used when homing the ISMD23E2. Note that this bit is not the Backplane_Proximity_Bit, but enables or disables its operation. Do not use the Backplane_Proximity_Bit if you only want to home to a Home Limit Switch. (Leave this bit equal to "0".) If you enable this bit and then never turn on the Backplane_Proximity_Bit, the ISMD23E2 will ignore all transitions of the home limit switch and you will not be able to home the ISMD23E2.

Bit 12: Reserved – Must equal zero.

Bit 13: Enable_Stall_Detection – "0" disables motor stall detection. "1" enables motor stall detection. Only valid on ISMD23E2 units with built in encoders. The Use_Encoder bit, which is bit 10 of this word, must be also be set to "1". You must also program the Encoder_Resolution parameter in configuration word 6.

Bit 14: Disable_Antiresonance – "1" disables the antiresonance feature. "0" enables the anti-resonance feature of the ISMD23E2. The Anti-resonance feature will provide smoother operation in most cases. If you are still experiencing resonance problems with this feature enabled, disable this feature and test the machine again.

Bit 15: Mode – "1" for Configuration Mode Programming, "0" for Command Mode Programming.

Bits				
5	4	3		
2	1	0	Function	Description
0	0	0	General Purpose Input	The input is not used in any of the functions of the ISMD23E2, but it's status is reported in the Network Data. This allows the input to be used as a discrete DC input to the host controller.
0	0	1	CW Limit	Input defines the mechanical end point for CW motion.
0	1	0	CCW Limit	Input defines the mechanical end point for CCW motion.
0	1	1	Start Indexed Move	Starts the move that is currently located in the output registers.
0	1	1	Start Indexed Move / Capture Encoder Value	When the encoder is enabled on an ISMD23E2, the encoder position value is captured whenever this input transitions. An inactive-to-active state transition will also trigger an Indexed Move if one is pending in the ISMD23E2.
1	0	0	Stop Jog or Registration Move	Brings a Jog or Registration Move to a controlled stop.
1	0	0	Stop Jog or Registration Move & Capture Encoder Value	When the encoder is enabled on an ISMD23E2, the encoder position value is captured when the input triggers a controlled stop to a Jog or Registration move.
1	0	1	Emergency Stop	All motion is immediately stopped when this input makes an inactive-to-active transition.
1	1	0	Home	Used to define the home position of the machine.
1	1	1	Invalid Combination	This bit combination is reserved.

Table R6.3 Configuration Data: Input Function Selections

Note➤ When using an encoder, you must set bit 10, the Use_Encoder bit. You must also program the Encoder_Resolution parameter in configuration word 6.

Configuration Word 1 Format

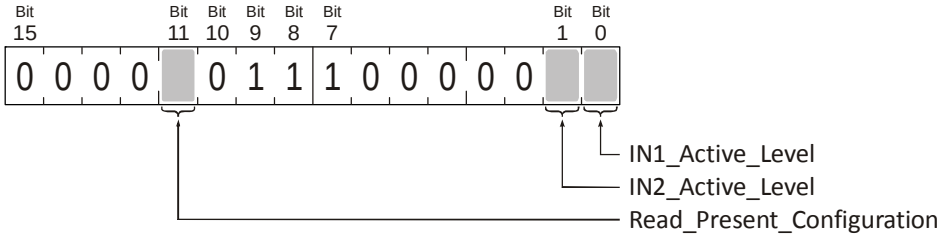


Figure R6.2 Configuration Mode: Configuration Word 1 Format

Bit 0: IN1_Active_Level – Determines the active state of Input 1. Set to "0" if your sensor has Normally Closed (NC) contacts and the input is active when there is no current flow through it. Set to "1" if your sensor has Normally Open (NO) contacts and current flows through the input when it is active.

Bit 1: IN2_Active_Level – Determines the active state of Input 2. Set to "0" if your sensor has Normally Closed (NC) contacts and the input is active when there is no current flow through it. Set to "1" if your sensor has Normally Open (NO) contacts and current flows through the input when it is active.

Note➤ If you are not using the input, sets its Active_Level bit to "1". The input will then always report as inactive in the network data.

Bits 2 - 6: Reserved – Must be reset to zero.

Bits 7 - 9: Reserved – Must be set to one. These bits control the format of data accepted and transmitted by the ISMD23E2. Then must be set to "111" in order to be compatible with the programming macros published by IDEC.

Bit 10: Reserved – Must be reset to zero.

Bit 11: Read_Present_Configuration – If this bit is set when you enter Configuration Mode, the ISMD23E2 responds by placing the present configuration data in the Network Input Data. You cannot write new configuration data to the unit while this bit is set. The format of the Configuration Data is given in the *Command Mode Data Format* section of this chapter, starting on page 75.

Bits 15 - 12: Reserved – Must be reset to zero.

Notes on Other Configuration Words

- Changes to the Idle Current only take effect at the *end of the first move after re-configuration*.

REFERENCE 7: COMMAND MODE DATA FORMAT

Introduction

This chapter covers the formats of the Network Output Data used to command the ISMD23E2. An ISMD23E2 requires twenty bytes of Output Data as well as 20 bytes for Input Data. The format of this data is a mix of sixteen bit and thirty-two bit integers.

Note► The custom macros supplied by IDEC simplify the configuration and programming of the ISMD23E2 units. The data in this reference is presented as background information to aid in troubleshooting.

Data Format

Thirty-two bit values are transmitted most significant word first (big endian). This is the data format specified by the Modbus specification and used by IDEC controllers. Table R6.1 below shows the format of the data as it is stored and transmitted.

	32 bit Integer Format	
Value	**First Word**	**Second Word**
12	0x0000	0x000C
-12	0xFFFF	0xFFF4
1,234,567	0x0012	0xD687
-7,654,321	0xFF8B	0x344F

Table R7.1 Position Data Format Examples

Command Bits Must Transition

Note► Commands are only accepted when the command bit makes a 0→1 transition. The easiest way to do this is to write a value of zero into the Command Word 0 before writing the next command.

This condition also applies when switching from Configuration Mode to Command Mode. If a bit is set in Configuration Word 0 while in Configuration Mode and you switch to Command Mode with the same bit set, the command will not occur because the bit must transition between writes to the unit.

The command bits are split between 2 sixteen bit words, Command Word 0 and Command Word 1. Only one bit in Command Word 0 can make a 0→1 transition at a time.

Output Data Format

The following table shows the format of the output network data words when writing command data to the ISMD23E2.

Register	Function
1024	Command Word 0
1025	Command Word 1
1026	Command Parameters: Word meaning depends on the command set to the ISMD23E2
1027	
1028	
1029	
1030	
1031	
1032	
1033	

Figure R7.1 Command Data Format

Command Word 0

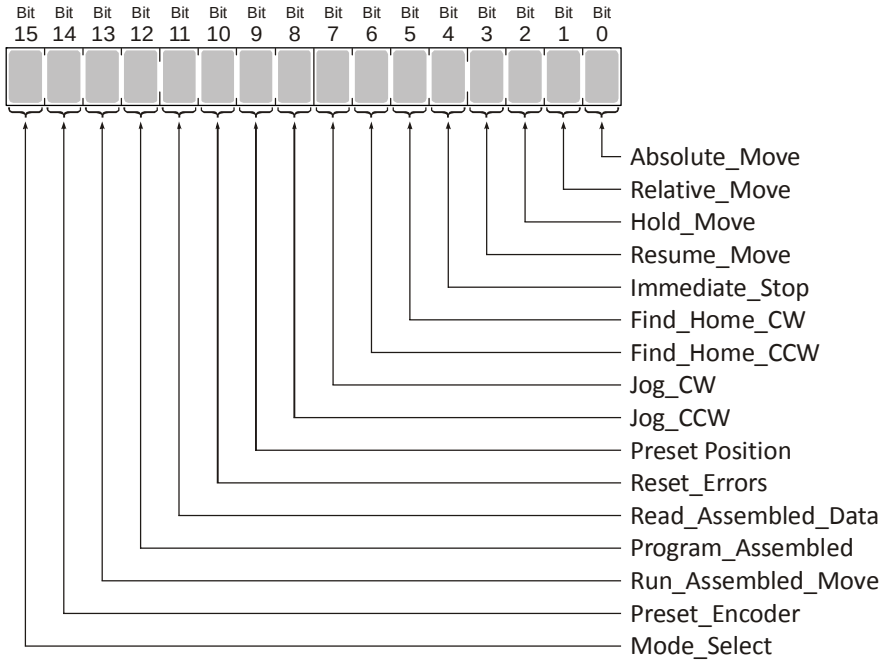


Figure R7.2 Command Word 0 Format

Bit 0: Absolute_Move – When this bit makes a 0→1 transition, the ISMD23E2 will perform an Absolute Move using the values in the rest of the command data. The full explanation of an *Absolute Move* can be found starting on page 23.

Bit 1: Relative_Move – When this bit makes a 0→1 transition, the ISMD23E2 will perform a Relative Move using the values in the rest of the command data. The full explanation of a *Relative Move* can be found starting on page 22.

Bit 2: Hold_Move – When this bit makes a 0→1 transition, the ISMD23E2 will place a move in its hold state. The move will decelerate to its programmed Starting Speed and stop. The move can be completed by using the Resume_Move bit. Use of the Hold_Move and Resume_Move bits can be found in the *Controlling Moves In Progress* section of this manual starting on page 26.

Bit 3: Resume_Move – When this bit makes a 0→1 transition, the ISMD23E2 will resume a move that you previously placed in a hold state. Use of the Resume_Move and Hold_Move bits can be found in the *Controlling Moves In Progress* section of this manual starting on page 26. Note that a move in its hold state need not be resumed. The move is automatically cancelled if another move is started in its place.

Bit 4: Immediate_Stop – When this bit makes a 0→1 transition, the ISMD23E2 will stop all motion without deceleration. The Motor Position value will become invalid if this bit is set during a move. Setting this bit when a move is not in progress will not cause the Motor Position to become invalid.

Bit 5: Find_Home_CW – When this bit makes a 0→1 transition, the ISMD23E2 will attempt to move to the Home Limit Switch in the clockwise direction. A full explanation of homing can be found in the *Homing an ISMD23E2* reference chapter starting on page 29.

Bit 6: Find_Home_CCW – When this bit makes a 0→1 transition, the ISMD23E2 will attempt to move to the Home Limit Switch in the counter-clockwise direction. A full explanation of homing can be found in the *Homing an ISMD23E2* reference chapter starting on page 29.

Bit 7: Jog_CW – When this bit makes a 0→1 transition, the ISMD23E2 will run a Jog Move in the clockwise direction. The full explanation of a *CW/CCW Jog Move* can be found starting on page 23.

- **Registration_Move – Command Word 1, Bit 7:** When this bit equals "0", and a Jog Move command is issued, it will run as a standard Jog Move. When this bit equals "1" and a Jog Move command is issued, the move will run as a Registration Move. The state of this bit cannot be changed while a Jog Move is in progress.

Bit 8: Jog_CCW – When this bit makes a 0→1 transition, the ISMD23E2 will run a Jog Move in the counter-clockwise direction. The full explanation of a *CW/CCW Jog Move* can be found starting on page 23.

- **Registration_Move – Command Word 1, Bit 7:** When this bit equals "0", and a Jog Move command is issued, it will run as a standard Jog Move. When this bit equals "1" and a Jog Move command is issued, the move will run as a Registration Move. The state of this bit cannot be changed while a Jog Move is in progress.

Bit 9: Preset_Position – When this bit makes a 0→1 transition, the ISMD23E2 will preset the Motor Position. The value depends on the state of the *Preset_To_Encoder* bit (Command Word 1, bit 13). If the *Preset_To_Encoder* bit equals "0", the Motor Position is preset to the value stored in Output Words 2 and 3. If the *Preset_To_Encoder* bit equals "1", the Motor Position is set to:

$$\text{Motor Position} = \text{Encoder Position} \times \frac{\text{Motor Programmed Steps per Turn}}{\text{Encoder Programmed Pulses per Turn}}$$

In either case, the *Move_Complete* bit in the Network Input Data is reset to "0".

Bit 10: Reset_Errors – When this bit makes a 0→1 transition, the ISMD23E2 will clear all existing command errors and reset the *Move_Complete* bit in the Network Input Data. This bit does not clear a configuration error or the *Position_Invalid* status bit.

Bits 11 & 12: Read_Assembled_Data & Program_Assembled – These bits are used to program the segments of an Assembled Move before the move can be run. Their use is explained in the *Assembled Move Programming* section of this manual starting on page 69.

Bit 13: Run_Assembled_Move – When this bit makes a 0→1 transition, the ISMD23E2 will run the Assembled Move already stored in memory.

- **Assembled_Move_Type – Command Word 1, Bit 9:** This bit determines the type of move that is run. When this bit equals "0", a Blend Move is run. When this bit equals "1", a Dwell Move is run. When starting a Dwell Move, the Dwell Time is programmed in word 9 of the Command Data. The value is programmed in milliseconds and can range from 0 to 65,536.
- **Reverse_Blend_Direction – Command Word 1, Bit 4:** This bit is used to determine the direction that the Blend Move will be run in. When this bit equals "0", the Blend Move runs in the clockwise direction. When this bit equals "1", the Blend Move is run in the counter-clockwise direction. This bit is ignored if a Dwell Move is being run.

Bit 14: Preset_Encoder – When this bit makes a 0→1 transition, the ISMD23E2 will preset the Encoder Position to the value stored in Output Words 2 and 3.

Bit 15: Mode_Select – "1" for Configuration Mode Programming "0" for Command Mode Programming.

Command Word 1

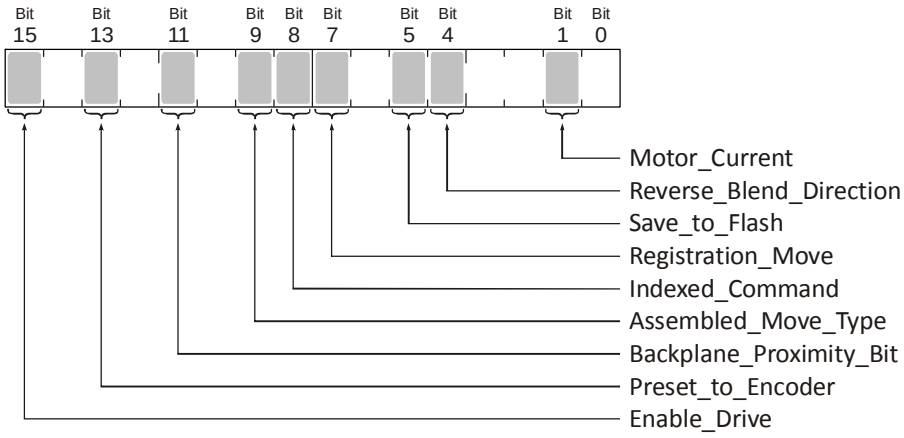


Figure R7.3 Command Word 1 Format

Bit 0: Reserved – Must equal "0".

Bit 1: Motor_Current – If reset to "0" when a move command is issued, the motor current will be the value specified when the ISMD23E2 was configured. Set to "1" to program the motor current to the value in word 8 of the command block when a move command is issued. Motor current can be changed during a move as long as this bit is set to "1" during the move.

Bits 3-2: Reserved – Must equal "0".

Bit 4: Reverse_Blend_Direction – When you command a Blend Move to run, this bit determines the direction of rotation. Set to "0" for a clockwise Blend Move, '1' for a counter-clockwise Blend Move. This bit is ignored when running a Dwell Move.

Bit 5: Save_to_Flash - This bit can be used to save a programmed Assembled Move to flash memory or to store the absolute encoder position offset to flash. (The absolute encoder position offset is generated by the Encoder Preset command.)

- When using this bit to save the programmed Assembled Move to flash memory, this bit must be set when the Program_Assembled bit (Command Word 0, bit 12) makes a 1 → 0 transition at the end of the programming cycle. The ISMD23E2 responds by flashing the Status LED when the writing is complete. If the LED is flashing green, the write to flash memory was successful. If it flashes red, then there was an error in writing the data. In either case, power must be cycled to the ISMD23E2 before you can continue. This design decision is to protect the flash memory from constant write commands. The flash memory has a minimum of 10,000 write cycles.
- When using this bit to save the calculated absolute encoder offset value to flash memory, this bit must be set when the Preset Encoder command is issued. (Bit 14 of *Command Word 0* is set to "1", see page 76.) If the offset is stored without error, the unit will respond by setting the Acknowledge bit. (Bit 13 of *Status Word 1 Format*, see page page 61.)

Bit 6: Reserved – Must equal "0".

Bit 7: Registration_Move – When this bit equals "0", and a Jog Move command is issued, it will run as a standard Jog Move. When this bit equals "1" and a Jog Move command is issued, the move will run as a Registration Move.

Bit 8: Indexed_Command – If this bit is set when a move command is issued, the ISMD23E2 will not run the move immediately, but will instead wait for an inactive-to-active transition on an input configured as a *Start Indexer Move* input. The move command data, including this bit, must remain in the Network Output Registers while performing an Indexed Move.

Bit 9: Assembled_Move_Type – When this bit equals "0", a Blend Move is started when the Run Assembled Move bit, (Command Word 1, Bit 13) makes a 0→1 transition. When this bit equals "1", a Dwell Move is started on the transition. The direction of a Blend Move is controlled by the Reverse_Blend_Direction bit, (Command Word 1, Bit 4). In a Dwell Move, the Dwell Time between segments is programmed in Word 9 of the command data.

Bit 10: Reserved – Must equal "0".

Bit 11: Backplane_Proximity_Bit – When the ISMD23E2 is configured to use the Backplace_Proximity_Bit, the unit will ignore the state of the Home Input as long as this bit equals "0". This bit must equal "1" before a transition on the Home Input can be used to home the machine. Further information on using the Backplace_Proximity_Bit can be found in the *Profile with Backplane_Proximity_Bit* section found on page 31.

Bit 12: Reserved – Must equal "0".

Bit 13: Preset_to_Encoder – Only used when the Preset Motor Position bit (Command Word 0, Bit 9) is set to "1". If this bit equals "0" when the Preset Motor Position bit equals "1", the Motor Position is set to the value contained in words 2 and 3 of the command block. If this bit equals "1" when the Preset Motor Position bit equals "1", the Motor Position is set to:

$$\text{Motor Position} = \text{Encoder Position} \times \frac{\text{Motor Programmed Steps per Turn}}{\text{Encoder Programmed Pulses per Turn}}$$

In either case, the *Move_Complete* bit in the Network Input Data is reset to "0".

Bit 14: Reserved – Must equal "0".

Bit 15: Enable_Drive – "0" to disable the motor current, "1" to enable motor current. A valid configuration must be written to the ISMD23E2 before the drive can be enabled.

Command Blocks

The following section lists the output data format for the sixteen different commands.

Absolute Move

Register	Function	Units	Range
1024	*Command Word 0*		0x0001
1025	*Command Word 1*		See pg. 77
1026	Abs. Target Position: Upper 16 bits	Steps	Combined value between –8,388,607 and +8,388,607
1027	Abs. Target Position: Lower 16 bits		
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to '1'.
1033	Acceleration Jerk		0 to 5000

Table R7.2 Absolute Move Command Block

Relative Move

Register	Function	Units	Range
1024	*Command Word 0*		0x0002
1025	*Command Word 1*		See pg. 77
1026	Rel. Target Position: Upper 16 bits	Steps	Combined value between –8,388,607 and +8,388,607
1027	Rel. Target Position: Lower 16 bits		
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to '1'.
1033	Acceleration Jerk		0 to 5000

Table R7.3 Relative Move Command Block

Hold Move

Register	Function	Units	Range
1024	*Command Word 0*		0x0004
1025	*Command Word 1*		See pg. 77
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused		See Note Below

Table R7.4 Hold Move Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Resume Move

Register	Function	Units	Range
1024	*Command Word 0*		0x0008
1025	*Command Word 1*		See pg. 77
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to '1'.
1033	Acceleration Jerk		0 to 5000

Table R7.5 Resume Move Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command. This is typically the case when resuming a move, the words are listed as "Unused" to highlight that the target position of a held move cannot be changed when the move is resumed.

Immediate Stop

Register	Function	Units	Range
1024	*Command Word 0*		0x0010
1025	*Command Word 1*		See pg. 77
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused		See Note Below

Table R7.6 Immediate Stop Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Find Home CW

Register	Function	Units	Range
1024	*Command Word 0*		0x0020
1025	*Command Word 1*		See pg. 77
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to '1'.
1033	Acceleration Jerk		0 to 5000

Table R7.7 Find Home CW Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Find Home CCW

Register	Function	Units	Range
1024	*Command Word 0*		0x0040
1025	*Command Word 1*		See pg. 77
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to '1'.
1033	Acceleration Jerk		0 to 5000

Table R7.8 Find Home CCW Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Jog CW

Register	Function	Units	Range
1024	*Command Word 0*		0x0080
1025	*Command Word 1*		See pg. 77 Bit 7 must equal "0"
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to '1'.
1033	Acceleration Jerk		0 to 5000

Table R7.9 Jog Move CW Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Registration Move CW

Register	Function	Units	Range
1024	*Command Word 0*		0x0080
1025	*Command Word 1*		See pg. 77 Bit 7 must equal "1"
1026	Stopping Distance: Upper 16 bits	Steps	Combined value between 0 and +8,388,607
1027	Stopping Distance: Lower 16 bits		
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Min. Reg. Move Distance: Upper 16 bits	Steps	Combined value between 0 and +8,388,607
1033	Min. Reg. Move Distance: Lower 16 bits		

Table R7.10 Registration Move CW Command Block

Jog CCW

Register	Function	Units	Range
1024	*Command Word 0*		0x0100
1025	*Command Word 1*		See pg. 77 Bits 7 must equal “0”
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the Configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Motor Current	0.1 amps	1 to 34. Ignored if bit 1 of Command Word 1 is not set to ‘1’.
1033	Acceleration Jerk		0 to 5000

Table R7.11 Jog CCW Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Registration Move CCW

Register	Function	Units	Range
1024	*Command Word 0*		0x0100
1025	*Command Word 1*		See pg. 77 Bits 7 must equal “1”
1026	Stopping Distance: Upper 16 bits	Steps	Combined value between 0 and +8,388,607
1027	Stopping Distance: Lower 16 bits		
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	Min. Reg. Move Distance: Upper 16 bits	Steps	Combined value between 0 and +8,388,607
1033	Min. Reg. Move Distance: Lower 16 bits		

Table R7.12 Registration Move CCW Command Block

Preset Position

Register	Function	Units	Range
1024	*Command Word 0*		0x0200
1025	*Command Word 1*		See pg. 77
1026	Position Preset Value: Upper 16 bits	Steps	Combined value between –8,388,607 and +8,388,607
1027	Position Preset Value: Lower 16 bits		
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused		See Note Below

Table R7.13 Preset Position Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Presetting the position resets the *Position_Invalid* and *Move_Complete* status bits in the Network Input Data.

Reset Errors

Register	Function	Units	Range
1024	*Command Word 0*		0x0400
1025	*Command Word 1*		See pg. 77 Set bit 10 to clear motor faults
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused		See Note Below

Table R7.14 Reset Errors Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Resetting errors will also reset the *Move_Complete* status bit in the Network Input Data. Resetting errors will not reset the *Position_Invalid* or *Configuration_Error* bits.

Run Assembled Move

Register	Function	Units	Range
1024	*Command Word 0*		0x2000
1025	*Command Word 1*		See pg. 77 Blend Move: Bit 9 = “0”, Dwell Move: Bit 9 = “1” If Blend Move, CW move when Bit 4 = “0”, CCW move when Bit 4 = “1”.
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused with Blend Move Dwell Time with Dwell Move	milliseconds	0 to 65,535

Table R7.15 Run Assembled Move Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Preset Encoder Position

Register	Function	Units	Range
1024	*Command Word 0*		0x4000
1025	*Command Word 1*		See pg. 77
1026	Encoder Preset Value: Upper 16 bits	Steps	Combined value between –8,388,607 and +8,388,607
1027	Encoder Preset Value: Lower 16 bits		
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused		See Note Below

Table R7.16 Preset Encoder Position Command Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Programming Blocks

The following blocks are used to program an Assembled Move. Both of the move types, Blend Move, and Dwell Move, are programmed exactly the same way. The bit configuration used when starting the move determines which type of Assembled Move is run.

First Block

Register	Function	Units	Range
1024	*Command Word 0*		0x0800
1025	*Command Word 1*		See pg. 77
1026	Unused		See Note Below
1027	Unused		See Note Below
1028	Unused		See Note Below
1029	Unused		See Note Below
1030	Unused		See Note Below
1031	Unused		See Note Below
1032	Unused		See Note Below
1033	Unused		See Note Below

Table R7.17 Assembled Move First Programming Block

Unused words are ignored by the ISMD23E2 and can be any value, including parameter values from the previous command.

Once the first block is transmitted, the ISMD23E2 responds by setting bits 8 and 9 in Status Word 0. (See *Status Word 0 Format* starting on page 60.) Once these are set, you can then start transmitting Segment Blocks.

Segment Block

Register	Function	Units	Range
1024	*Command Word 0*		0x1800
1025	*Command Word 1*		See pg. 77
1026	Rel. Target Position: Upper 16 bits	Steps	Combined value between –8,388,607 and +8,388,607
1027	Rel. Target Position: Lower 16 bits		
1028	Programmed Speed: Upper 16 bits	Steps/Second	Combined value between the configured Starting Speed and 2,999,999
1029	Programmed Speed: Lower 16 bits		
1030	Acceleration	Steps/ms/sec	1 to 5000
1031	Deceleration	Steps/ms/sec	1 to 5000
1032	*Reserved*		Must equal zero for compatibility with future releases.
1033	Acceleration Jerk		0 to 5000

Table R7.18 Assembled Move Segment Programming Block

Note that each Segment Block starts with bits 11 and 12 in Command Word 0 set to “1” (0x1800). When the ISMD23E2 sees bit 12 of Command Word 0 set, it will accept the block and reset bit 9 in Status Word 0. When your program sees this bit reset, it must respond by resetting bit 12 of Command Word 0. The ISMD23E2 will respond to this by setting bit 9 in Status Word 0 and the next Segment Block can be written to the Networked Drive. You can write a maximum of sixteen Segment Blocks for each Assembled Move.

REFERENCE 8: CALCULATING MOVE PROFILES

Introduction

This appendix was added for customers that must program very precise profiles. Understanding this section is not necessary before programming the ISMD23E2 and it can be considered optional. Two different approaches are presented here. The constant acceleration example takes given parameters and calculates the resulting profile. The variable acceleration example starts with a desired speed profile and calculates the required parameters.

Units of Measure

The equations in this appendix use a unit of measure of steps/second/second (steps/second^2) for acceleration and deceleration. However, when programming the ISMD23E2, all acceleration and deceleration values must be programmed in the unit of measure of steps/second/millisecond.

- To convert from steps/second^2 to steps/millisecond/second, divide the value by 1000. This must be done when converting from a value used in the equations to a value programmed into the ISMD23E2.
- To convert from steps/millisecond/second to steps/second^2, multiply the value by 1000. This must be done when converting from the value programmed into the ISMD23E2 to the value used in the equations.

Constant Acceleration Equations

When you choose to use constant accelerations, the speed of the move will increase linearly towards the Programmed Speed. This is the fastest form of acceleration, resulting in the fastest move between two points at its programmed speed. For the smoothest transition from the starting speed, the starting speed should be equal to the square root of the acceleration in steps/sec^2. For example, if the choose acceleration is 20,000 steps/sec^2, the smoothest transition occurs when the starting speed is 141. ($141^2 \approx 20{,}000$)

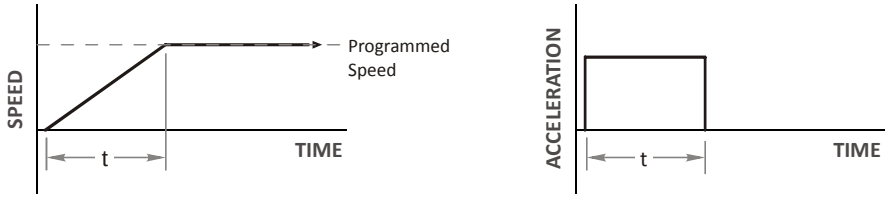


Figure R8.1 Constant Acceleration Curves

Variable Definitions

The following variables are used in these equations:

- **V_S** = Configured Starting Speed of the move
- **V_P** = Programmed Speed of the move
- **a** = Acceleration value. Must be in the units of steps/second^2
- **d** = Deceleration value. Must be in the units of steps/second^2
- **T_A or T_D** = Time needed to complete the acceleration or deceleration phase of the move
- **D_A or D_D** = Number of Steps needed to complete the acceleration or deceleration phase of the move

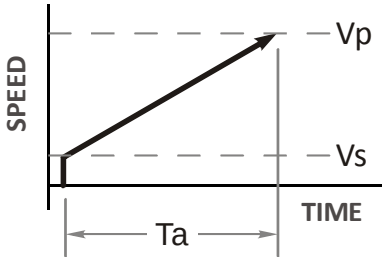


Figure R8.1 gives the equations to calculate Time, Distance, and Acceleration values for a constant acceleration move.

Acceleration Type	T_A or T_D (Time to Accelerate or Decelerate)	D_A or D_D (Distance to Accelerate or Decelerate)	a (Average Acceleration)
Linear	$T_A = (V_P - V_S)/a$	$D_A = T_A*(V_P + V_S)/2$	$a = (V_P^2 - V_S^2)/2D_A$

Table R8.1 Acceleration Equations

If the sum of the D_A and D_D values of the move is *less than* the total number of steps in the move, your move will have a Trapezoidal profile.

If the sum of the D_A and D_D values of the move is *equal to* the total number of steps in the move, your move will have a Triangular profile and your move will reach the Programmed Speed before it begins to decelerate.

If the sum of the D_A and D_D values of the move is *greater than* the total number of steps in the move, your move will have a Triangular profile and it *will not* reach the Programmed Speed before it begins to decelerate.

As an example, lets assume the values in table R8.2 for a move profile.

Name	Value	ISMD23E2 Parameter Values
Acceleration (a)	20,000 steps/sec^2	20
Deceleration (d)	25,000 steps/sec^2	25
Starting Speed (V_S)	141 steps/sec	141
Programmed Speed (V_P)	100,000 steps/sec	100,000

Table R8.2 Sample Values

From figure R8.1:

Time to accelerate: $T_A = (V_P - V_S)/a = (100,000 - 141)/20,000 = 4.993$ seconds
Time to decelerate: $T_D = (V_P - V_S)/d = (100,000 - 141)/25,000 = 3.994$ seconds
Distance to Accelerate: $D_A = T_A*(V_P + V_S)/2 = 4.993 * (100,000 + 141)/2 = 250,002$ steps
Distance to Decelerate: $D_D = T_D*(V_P + V_S)/2 = 3.994 * (100,000 + 141)/2 = 199,982$ steps

Total Distance needed to accelerate and decelerate: 250,002 + 199,982 = 449,984 steps

If a move with the above acceleration, deceleration, starting speed, and programmed speed has a length greater than 449,984 steps, the ISMD23E2 will generate a Trapezoidal profile. If the move is equal to 449,984 steps, the ISMD23E2 will generate a Triangular profile and the unit will output one pulse at the programmed speed. If the move is less than 449,984 steps, the ISMD23E2 will generate a Triangular profile and the programmed speed will not be reached.

In the case of a Triangular profile where the programmed speed is not reached, it is fairly easy to calculate the maximum speed (V_M) attained during the move. Because the move is always accelerating or decelerating, the total distance traveled is equal to the sum of D_A and D_D.

$D_A = T_A*(V_M + V_S)/2$ and $T_A = (V_M - V_S)/a$. By substitution:
$D_A = (V_M - V_S)/a * (V_M + V_S)/2 = (V_M^2 - V_S^2)/2a$. By the same method,
$D_D = (V_M^2 - V_S^2)/2d$.

Therefore, total distance traveled =

$D_A + D_D = (V_M^2 - V_S^2)/2a + (V_M^2 - V_S^2)/2d$.

In the case where the acceleration and deceleration values are equal, this formula reduces to:

$D_A + D_D = (V_M^2 - V_S^2)/a$

Continuing the example from table R8.2, assume a total travel distance of 300,000 steps.

$$D_A + D_D = \frac{V_M^2 - V_S^2}{2a} + \frac{V_M^2 - V_S^2}{2d}$$

$$300,000 \text{ steps} = \frac{V_M^2 - 141^2}{2(20,000)} + \frac{V_M^2 - 141^2}{2(25,000)}$$

$$300,000 \text{ steps} = \frac{V_M^2 - 20,000}{40,000} + \frac{V_M^2 - 20,000}{50,000}$$

$$300,000 \text{ steps} = \frac{5}{5}\left(\frac{V_M^2 - 20,000}{40,000}\right) + \frac{4}{4}\left(\frac{V_M^2 - 20,000}{50,000}\right)$$

$$300,000 \text{ steps} = \frac{5V_M^2 - 100,000}{200,000} + \frac{4V_M^2 - 80,000}{200,000}$$

$$300,000\ (200,000) = 9V_M^2 - 180,000$$

$$\frac{60,000.18 \times 10^6}{9} = V_M^2$$

$$V_M = 81,650 \text{ steps/sec}$$

Once the maximum speed has been calculated, substitute this value into the time and distance formulas in table R8.1 to calculate time spent and distance traveled while accelerating and decelerating.

Total Time Equations

For Trapezoidal Profiles you must first determine the number of counts that you are running at the Programmed Speed. This value, (D_P below), is equal to your D_A and D_D values subtracted from your total travel. You can then calculate your total profile time, (T_T below), from the second equation.

D_P = (Total Number of Steps) – (D_A + D_D)

$T_T = T_A + T_D + D_P/V_P$

For Triangular Profiles, the total time of travel is simply:

$T_T = T_A + T_D$

S-Curve Acceleration Equations

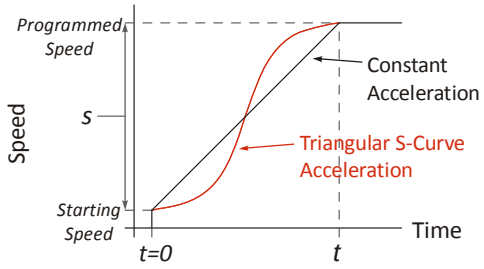
When the Acceleration Jerk parameter value is in the range of 1 to 5,000, the ISMD23E2 uses this value to smoothly change the acceleration value applied during the move. In this case, the speed of the move does not increase linearly, but exponentially, resulting in an "S" shaped curve. This limits mechanical shocks to the system as the load accelerates. Just as constant acceleration will result in a trapezoidal or triangular speed profile, the Acceleration Jerk parameter will result in a trapezoidal or triangular acceleration phase.

In order to keep the Acceleration Jerk parameter value that is programmed into the ISMD23E2 below sixteen bits, the Acceleration Jerk parameter programmed into the drive does not have units of steps/sec^3. The Acceleration Jerk parameter equals ({100 * jerk in steps/sec^3} / acceleration in steps/sec^2). This translates to the jerk property in steps/sec^3 equalling ({Acceleration Jerk parameter/ 100} * acceleration in steps/sec^2). With the range of values for the Acceleration Jerk parameter being 1 to 5,000, the jerk value ranges from 0.01*a* to 50*a* where "*a*" is the acceleration value in steps/sec^2. For example, if the acceleration is programmed to 20,000 steps/sec^2, then the value of the jerk property used by the unit can be programmed to be between 200 steps/sec^3 (0.01*20,000) and 1,000,000 steps/sec^3 (50*20,000). This statement applies to the Deceleration Parameter as well. If the Acceleration and Deceleration parameters are different, the calculated jerk values will also differ.

When using variable accelerations, the starting speed does not have to be equal to the square root of the programmed acceleration value for the smoothest transition from stopped to accelerating. Variable acceleration provides smooth transitions at the beginning and end of the acceleration phase.

Triangular S-Curve Acceleration

Figure R8.2 shows the speed profile of a move during its acceleration phase. The figure shows the desired triangular S-curve acceleration in red along with the equivalent constant acceleration in black. The equivalent constant acceleration is equal to the change in speed divided by the time it takes to achieve this change in speed. This is the value that would have to be used if the Jerk parameter was left at zero. This information is used to calculate the S-curve acceleration and the value of the Jerk Parameter.



s = Programmed Speed – Starting Speed

$$\text{Acceleration} = \frac{\text{speed}}{\text{time}} \qquad a = \frac{s}{t} \qquad at = s$$

$$\text{jerk} = \frac{\text{acceleration}}{\text{time}} \qquad j = \frac{a}{t} \qquad jt = a$$

$$\text{ISMD23E2 Acceleration Jerk Parameter (J)} = \frac{100j}{a} \qquad j = \frac{Ja}{100}$$

Figure R8.2 Move Profile Example

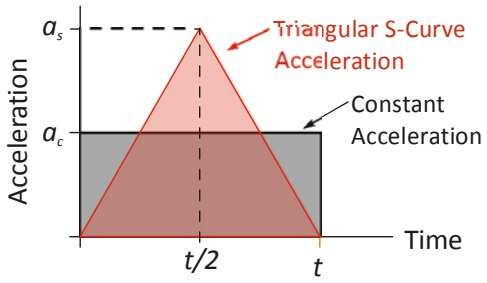


Figure R8.3 Triangular Acceleration

Speed is equal to acceleration multiplied by the time it is applied. This is shown graphically in figure R8.3 as the area of the gray rectangle. In order for the Triangular S-curve acceleration to reach the same speed in the same amount of time, the area of the triangle must equal the area of the square. Area of a triangle is one half of the base length multiplied by the height. Therefore:

$$a_c t = \frac{a_s t}{2} \quad \text{Area of rectangle = Area of triangle}$$

$$a_s = 2a_c$$

This means that a triangular S-curve acceleration profile requires twice the programmed maximum acceleration as a constant acceleration profile to achieve the same speed in the same amount of time.

The value of the Acceleration Jerk parameter can now be easily calculated.

$$j = \frac{a_s}{t/2} \qquad (j = a/t)$$

$$j = \frac{2a_s}{t}$$

$$\frac{Ja_s}{100} = \frac{2a_s}{t} \qquad \left(j = \frac{Ja}{100}\right)$$

$$Ja_s t = 200a_s$$

$$J = \frac{200}{t} \qquad \text{Acceleration Jerk parameter = 200 / acceleration time}$$

This value represents the ideal Acceleration Jerk parameter value for a triangular S-curve acceleration. Setting the value lower than this will result in a longer acceleration period, while setting the value above this will result in a trapezoidal S-curve acceleration.

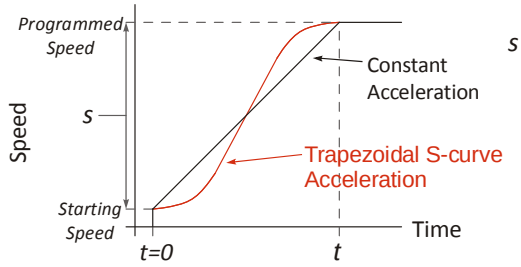
When $a_s = a_c$

The above examples assume that you can increase the programmed acceleration value to keep the acceleration time the same. If your constant acceleration value is the maximum your system will allow, then using
S-curve accelerations will lengthen the time needed to accelerate to your desired speed.

In the case of Triangular S-curve accelerations where the Acceleration Jerk parameter is optimized at 200/t, the value of "t" must be twice that of the acceleration period when constant acceleration is used. For example, assume a equivalent constant acceleration of 20,000 steps/sec^2 that is applied for 2.0 seconds. If the acceleration value must remain at 20,000 steps/sec^2, then the acceleration phase will take 4.0 seconds and the Acceleration Jerk parameter should be set to 50 (200/4.0)

Trapezoidal S-Curve Acceleration

Figure R8.4 shows the speed profile of a move during its acceleration phase. The figure shows the desired trapezoidal S-curve acceleration in red along with the equivalent constant acceleration in blue. The equivalent constant acceleration is equal to the change in speed divided by the time it takes to achieve the change in speed. This is the value that would have to be used if the Acceleration Jerk parameter was left at zero and we will use this information to calculate the S-curve acceleration and the value of the Acceleration Jerk parameter.



s = Programmed Speed – Starting Speed

$$\text{Acceleration} = \frac{\text{speed}}{\text{time}} \qquad a = \frac{s}{t} \qquad at = s$$

$$\text{jerk} = \frac{\text{acceleration}}{\text{time}} \qquad j = \frac{a}{t} \qquad jt = a$$

$$\text{ISMD23E2 Acceleration Jerk Parameter (J)} = \frac{100j}{a} \qquad j = \frac{Ja}{100}$$

Figure R8.4 Move Profile Example

In this example, the period of constant acceleration is 50% of the acceleration phase.

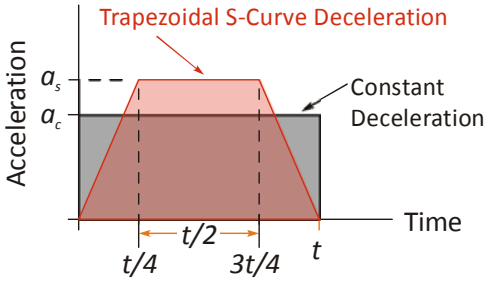


Figure R8.5 Trapezoidal Acceleration

Speed is equal to acceleration multiplied by the time it is applied. This is shown graphically in figure R8.5 as the area of the blue rectangle. In order for the Trapezoidal S-curve acceleration to reach the same speed in the same amount of time, the area of the polygon must equal the area of the rectangle.

$$\frac{a_s t}{2} + \frac{a_s t}{4} = a_c t \quad \text{Area of polygon = Area of rectangle}$$

$$\frac{2a_s t}{4} + \frac{a_s t}{4} = a_c t$$

$$\frac{3a_s t}{4} = a_c t$$

$$a_s = \frac{4}{3} a_c$$

This means that a trapezoidal S-curve acceleration profile that is has a period of constant acceleration equal to half of the total phase time, requires its programmed acceleration value to be 4/3 that of the constant acceleration value used to achieve the same speed in the same amount of time.

The value of the Acceleration Jerk parameter can now be easily calculated.

$$j = \frac{a_s}{t/4} \qquad (j = a/t)$$

$$j = \frac{4a_s}{t}$$

$$\frac{Ja_s}{100} = \frac{4a_s}{t} \qquad \left(j = \frac{Ja}{100}\right)$$

$$Ja_s t = 400 a_s$$

$$J = \frac{400}{t} \qquad \text{Acceleration Jerk Parameter = 400 / acceleration time}$$

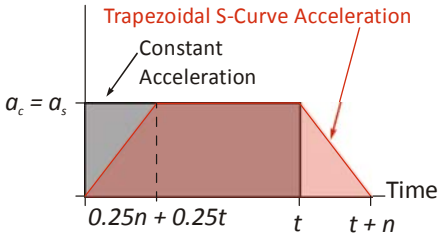
This value represents the ideal Acceleration Jerk parameter value for a trapezoidal S-curve acceleration with a constant acceleration for half of the phase. Setting the value lower than this will result in a shorter constant period, while setting the value greater than this will result in a longer constant period.

Another example of a trapezoidal S-curve acceleration is when the linear acceleration occurs for one third of the time. In this case, the programmed acceleration must be the constant acceleration value multiplied by 3/2 and the Acceleration Jerk parameter must be set to 300/t.

When $a_s = a_c$

The above examples assume that you can increase the programmed acceleration value to keep the time of the acceleration phase the same. If your constant acceleration value is the maximum your system will allow, then using S-curve accelerations will lengthen the time needed to accelerate to your desired speed.

In the case of trapezoidal S-curve accelerations, calculating the percentage increase in time is shown in figure R8.6. The time added to the acceleration phase is equal to the time spent increasing the acceleration during the phase. As shown in the figure, when the Trapezoidal S-curve is programmed to spend 50% of its time at the programmed acceleration value, the time spent in the acceleration phase will be 133.33% of the time spent if a constant acceleration were used.



$a_c(t) = a_c(.5n + .5t) + a_c(.25n + .25t)$

$a_c(t) = a_c((.5n + .5t) + (.25n + .25t))$

$t = .75n + .75t$

$0.25t = .75n$

$t = 3n$

$t/3 = n \Rightarrow t+n = t + t/3 = 4/3t = 1.3333t$

Figure R8.6 Trapezoidal S-curve Time Increase Example

In this case the value of the Acceleration Jerk parameter should be based on the new, longer time. For example, assume an equivalent constant acceleration of 15,000 steps/sec^2 that is applied for 2.0 seconds. If the acceleration value must remain at 15,000 steps/sec^2, then the acceleration phase will take 2.667 seconds (2.0×1.333) and the Acceleration Jerk parameter should be set to 150 (400/2.667).

Similarly, if the Trapezoidal S-curve acceleration is to spend 33.3% of its time at constant acceleration, and the programmed acceleration value cannot be increased, the time spent accelerating will increase by 50% and the Acceleration Jerk parameter should be adjusted accordingly.

Determining Waveforms by Values

If your programmed acceleration and deceleration values are the same, then your move's acceleration and decelerations will be identical. If these two programmed values are different, use the above methods to determine the Acceleration Jerk parameter for either the move's acceleration or deceleration phases and use the following calculations to determine the shape of the other phase.

Two examples are given below. Both assume a change in speed between the Starting Speed and Programmed Speed of 30,000 steps/sec and an acceleration of 58,000 steps/sec^2. The first example uses an Acceleration Jerk parameter value of 20 and the second a value of 400.

Triangular or Trapezoidal S-curve accelerations are always symmetrical. We'll use this fact to calculate the profile up to one-half of the change in speed. At that point, doubling the time and distance will yield the total time and distance traveled.

Example 1, Jerk = 20

$$S_m = \frac{30{,}000 \text{ steps/sec}}{2} = 15{,}000 \text{ steps/sec}$$

S_m = midpoint of change in speed

J = Acceleration Jerk parameter

j = physical jerk property

a_f = calculated final acceleration

$$J = \frac{100j}{a} \Rightarrow j = \frac{Ja}{100}$$

$$j = \frac{20(58{,}000 \text{ steps/sec}^2)}{100}$$

$$j = 11{,}600 \text{ steps/sec}^3$$

Just as displacement = $\frac{1}{2}at^2$, Speed = $\frac{1}{2}jt^2$

$$15{,}000 \text{ steps/sec} = \frac{11{,}600 \text{ steps/sec}^3(t^2)}{2}$$

$$t^2 = \frac{15{,}000 \text{ steps/sec}}{5{,}800 \text{ stesp/sec}^3}$$

$$t = 1.608 \text{ seconds}$$

Just as speed = at, acceleration = jt

$$a_f = 11{,}600 \text{ steps/sec}^3(1.608 \text{ sec})$$

$$a_f = 18{,}655 \text{ steps/sec}^2$$

Because a_f is less than or equal to the programmed acceleration of 58,000 steps/sec^2, the resulting acceleration is a Triangular S-curve. Total time to accelerate is twice the value calculated above, or 3.216 seconds.

Example 2, Jerk = 400

$$S_m = \frac{30{,}000 \text{ steps/sec}}{2} = 15{,}000 \text{ steps/sec}$$

S_m = midpoint of change in speed
J = Acceleration Jerk parameter
j = physical jerk property
a_f = calculated final acceleration

$$J = \frac{100j}{a} \Rightarrow j = \frac{Ja}{100}$$

$$j = \frac{400(58{,}000 \text{ steps/sec}^2)}{100}$$

$$j = 232{,}000 \text{ steps/sec}^3$$

Just as displacement = $\frac{1}{2}at^2$, speed = $\frac{1}{2}jt^2$

$$15{,}000 \text{ steps/sec} = \frac{232{,}000 \text{ steps/sec}^3(t^2)}{2}$$

$$t^2 = \frac{15{,}000 \text{ steps/sec}}{116{,}000 \text{ steps/sec}^3}$$

$$t = 0.3596 \text{ seconds}$$

Just as speed = at, acceleration = jt

$$a_f = 232{,}000 \text{ steps/sec}^3(0.3596 \text{ sec})$$

$$a_f = 83{,}427 \text{ steps/sec}^2$$

Because a_f is greater than the programmed acceleration of 58,000 steps/sec^2, the resulting acceleration is a trapezoidal S-curve. As shown in figure R8.7, two additional calculations must be made. The first is the time (t_1) it takes to jerk to the programmed acceleration value. The second is the time (t_2) it takes to accelerate to half of the required change in speed (S_m).

$$232{,}000 \text{ steps/sec}^3(t_1) = 58{,}000 \text{ steps/sec}^2 \qquad jt = a$$

$$t_1 = 0.25 \text{ seconds}$$

Determine speed at t_1: Speed = $\frac{1}{2}jt^2$

$$S_1 = \frac{232{,}000 \text{ steps/sec}^3(0.25)^2}{2}$$

$$S_1 = 7{,}250 \text{ steps/sec}$$

Determine remaining change in speed and required time based on programmed acceleration

$$S_2 = S_m - S_1 = (15{,}000 - 7{,}250) \text{ steps/sec}$$

$$S_2 = 7{,}750 \text{ steps/sec}$$

$$S_2 = a_c(t_2) \Rightarrow t_2 = S_2/a_c$$

$$t_2 = \frac{7{,}750 \text{ steps/sec}}{58{,}000 \text{ steps/sec}^2}$$

$$t_2 = 0.1336 \text{ seconds}$$

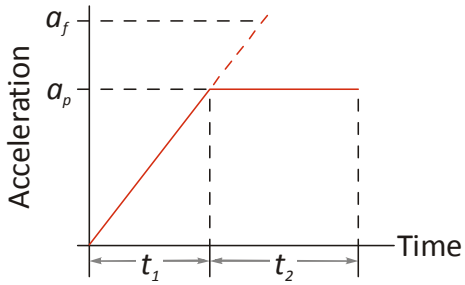


Figure R8.7 Calculating Trapezoidal S-Curve

The time for this acceleration phase is 2(t1 + t2), which equals 2(0.2500 sec + 0.1336 sec) or 0.7672 seconds. Time spent in the constant acceleration period is (2(0.1336))/0.7672) or 34.8% of the entire phase.

Notes


AMCi
by
IDEC CORPORATION