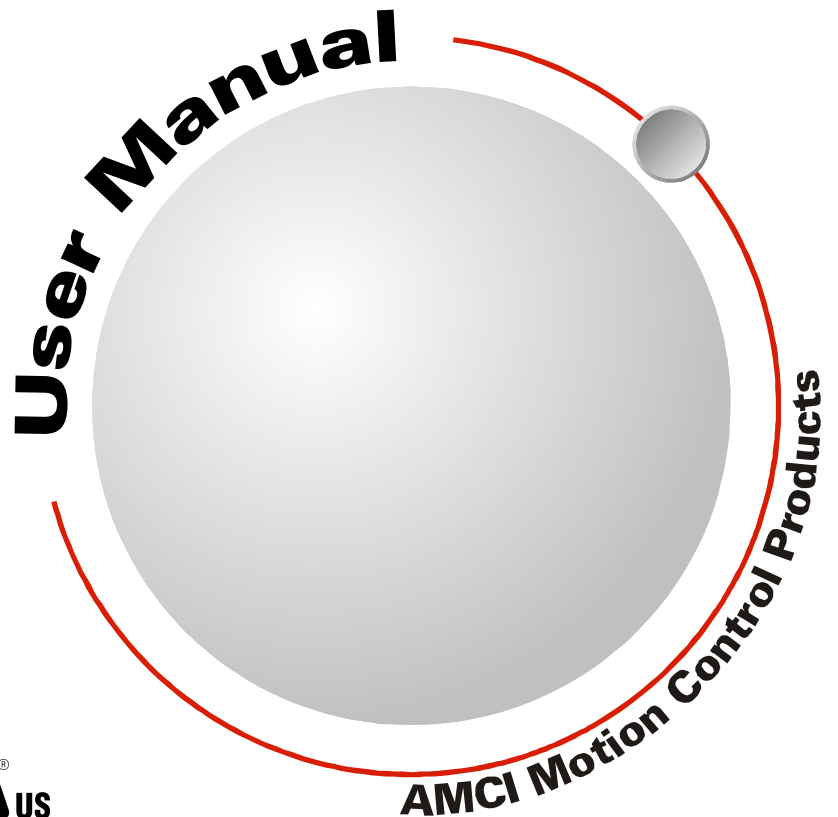


SD4840EK

Networked Stepper Indexer/Driver

with EtherCAT Interface



GENERAL INFORMATION

Important User Information

The products and application data described in this manual are useful in a wide variety of different applications. Therefore, the user and others responsible for applying these products described herein are responsible for determining the acceptability for each application. While efforts have been made to provide accurate information within this manual, AMCI assumes no responsibility for the application or the completeness of the information contained herein.

UNDER NO CIRCUMSTANCES WILL ADVANCED MICRO CONTROLS, INC. BE RESPONSIBLE OR LIABLE FOR ANY DAMAGES OR LOSSES, INCLUDING INDIRECT OR CONSEQUENTIAL DAMAGES OR LOSSES, ARISING FROM THE USE OF ANY INFORMATION CONTAINED WITHIN THIS MANUAL, OR THE USE OF ANY PRODUCTS OR SERVICES REFERENCED HEREIN.

No patent liability is assumed by AMCI, with respect to use of information, circuits, equipment, or software described in this manual.

The information contained within this manual is subject to change without notice.

This manual is copyright 2020 by Advanced Micro Controls Inc. You may reproduce this manual, in whole or in part, for your personal use, provided that this copyright notice is included. You may distribute copies of this complete manual in electronic format provided that they are unaltered from the version posted by Advanced Micro Controls Inc. on our official website: www.amci.com. You may incorporate portions of this documents in other literature for your own personal use provided that you include the notice "Portions of this document copyright 2020 by Advanced Micro Controls Inc." You may not alter the contents of this document or charge a fee for reproducing or distributing it.

Standard Warranty

ADVANCED MICRO CONTROLS, INC. warrants that all equipment manufactured by it will be free from defects, under normal use, in materials and workmanship for a period of [18] months. Within this warranty period, AMCI shall, at its option, repair or replace, free of charge, any equipment covered by this warranty which is returned, shipping charges prepaid, within eighteen months from date of invoice, and which upon examination proves to be defective in material or workmanship and not caused by accident, misuse, neglect, alteration, improper installation or improper testing.

The provisions of the "STANDARD WARRANTY" are the sole obligations of AMCI and excludes all other warranties expressed or implied. In no event shall AMCI be liable for incidental or consequential damages or for delay in performance of this warranty.

Returns Policy

All equipment being returned to AMCI for repair or replacement, regardless of warranty status, must have a Return Merchandise Authorization number issued by AMCI. Call (860) 585-1254 with the model number and serial number (if applicable) along with a description of the problem during regular business hours, Monday through Friday, 8AM - 5PM Eastern. An "RMA" number will be issued. Equipment must be shipped to AMCI with transportation charges prepaid. Title and risk of loss or damage remains with the customer until shipment is received by AMCI.

24 Hour Technical Support Number

24 Hour technical support is available on this product. If you have internet access, start at www.amci.com. Product documentation and FAQ's are available on the site that answer most common questions.

If you require additional technical support, call (860) 583-1254. Your call will be answered by the factory during regular business hours, Monday through Friday, 8AM - 5PM Eastern. During non-business hours an automated system will ask you to enter the telephone number you can be reached at. Please remember to include your area code. The system will page an engineer on call. Please have your product model number and a description of the problem ready before you call.

Waste Electrical and Electronic Equipment (WEEE)



At the end of life, this equipment should be collected separately from any unsorted municipal waste.

TABLE OF CONTENTS

General Information

Important User Information	2
Standard Warranty	2
Returns Policy	2
24 Hour Technical Support Number	2
WEEE Statement	2

About this Manual

Audience	7
Applicable Units	7
Navigating this Manual	7
Manual Conventions	7
Trademark Notices	7
Revision Record	8
Revision History	8
Where To Go From Here	8

Reference: Introduction to the SD4840EK

The SD4840EK	9
Conformance Markings	10
UL	10
CE	10
RoHS	10
Specifications	11
Indexer Functionality	12
Synchronizing Moves	13
Encoder Functionality	13
Stall Detection	14
Encoder Registration	
Move	14
Electronic Gearing	14
Driver Functionality	15
Idle Current Reduction	15
Current Loop Gain	15
I/O Connector	16
Available Discrete Inputs	16
Home Input	16
CW Limit Switch or	
CCW Limit Switch	16
Start Indexer Move Input	16
Emergency Stop Input	16
Stop Jog or	
Registration Move Input	17
Capture Encoder Position Input	17
General Purpose Input	17
Available Discrete Output	17

Reference: Introduction to the SD4840EK (continued)

Ethernet Ports	17
Motor Connector	18
Front Panel	18
Status LED	18
EtherCAT Run LED	18
EtherCAT ERR LED	18

Reference: Move Profiles

Definitions	19
Units of Measure	19
Distance	19
Speed	19
Acceleration	19
Motor Position	19
Home Position	19
Count Direction	19
Starting Speed	20
Target Position	20
Relative Coordinates	20
Absolute Coordinates	20
Definition of Acceleration Types	20
What is jerk?	20
Constant Acceleration	21
S-Curve Accelerations	21
Trapezoidal S-Curve	
Acceleration	21
Triangular S-Curve	
Acceleration	22
A Simple Move	22
Controlled and Immediate Stops	23
Host Control	23
Hardware Control	23
Basic Move Types	24
Relative Move	24
Controlled Stops	24
Immediate Stops	24
Absolute Move	25
Controlled Stops	25
Immediate Stops	25
CW/CCW Jog Move	26
Controlled Stops	26
Immediate Stops	27
CW/CCW Registration Move	27
Controlled Stops	28
Immediate Stops	28

**Reference: Move Profiles
(continued)**

Encoder Registration Moves	29
Controlled Stops	29
Immediate Stops	29
Assembled Moves	29
Blend Move	30
Controlled Stops	31
Immediate Stops	31
Dwell Move	31
Controlled Stops	32
Immediate Stops	32
Assembled Move Programming	33
Control Bits – Output Data	33
Control Bits – Input Data	33
Programming Routine	33
Saving an Assembled Move in Flash	33
Indexed Moves	34
Controlling Moves In Progress	35
Jog Moves	35
Registration Moves	35
Absolute, Relative and Encoder Registration Moves	35
Assembled Moves	35
Electronic Gearing	36
Motor Steps/Turn	36
ELGearing Multiplier and Divisor	36
How It Works	36
Controlled Stops	36
Immediate Stops	36
Advanced Ratio Control	37

**Reference: Calculating Move
Profiles**

Constant Acceleration Equations	39
Variable Definitions	39
Total Time Equations	41
S-Curve Acceleration Equations	42
Triangular S-Curve Acceleration	42
Trapezoidal S-Curve Acceleration	44
Determining Waveforms by Values	46

Chapter 4: Homing the SD4840EK

Definition of Home Position	49
Position Preset	49
CW/CCW Find Home Commands	49
Homing Inputs	49
Physical Inputs	49
Backplane Inputs	49
Homing Configurations	50
Homing Profiles	50
Home Input Only Profile	50
Profile with Proximity Input	51
Profile with Overtravel Limit	52

**Reference: Configuration Data
Format**

CoE Registers	53
Output Data Format	53
CFG_Word_0 Format	53
CFG_Word_1 Format	56
Current Loop Gain	57
Determining Current Loop Gain ...	58
Notes on Other Configuration Words	58
Invalid Configurations	58

Reference: Command Data Format

Command Bits Must Transition	59
Output Data Format	59
CMD_word0	60
CMD_word1	62
Command Blocks	64
Absolute Move	64
Relative Move	64
Hold Move	65
Resume Move	65
Immediate Stop	66
Find Home CW	66
Find Home CCW	67
Jog CW	67
Registration Move CW	68
Jog CCW	68
Registration Move CCW	69
Encoder Follower Move	70
Preset Position	70
Reset Errors	71
Run Assembled Move	71
Preset Encoder Position	72

Reference: Command Data Format

Programming Blocks	73
First Block	73
Segment Block	73
Input Data Format	74
STATUS_word0 Format	74
STATUS_word1 Format	76
Notes on Clearing a Driver Fault	77
Reset Driver Fault	77

Task 1: Motor and Power Supply Sizing

Sizing Your Motor	79
Determining Your Motor	
Current Setting	79
A Note on Microstepping	79
Torque and Power Curves	80
Power Supply Sizing	82
Regeneration Effects	82

Task 2: Installing the SD4840EK

Safe Handling Guidelines	83
Prevent Electrostatic Damage	83
Prevent Debris From	
Entering the Module	83
Remove Power Before	
Servicing	83
Mounting	83
Dimensions	83
Minimum Spacing	84
Mounting the SD4840EK	
Module	84
I/O Connector Pin Out	85
Power Wiring	85
Input Wiring	87
Cable Shields	87
Output Wiring	87
Encoder Wiring	88
Differential Wiring	88
Single Ended Wiring	89
Installing the Stepper Motor	89
Outline Drawings	89
Mounting the Motor	90
Connecting the Load	90
Extending the Motor Cable	90
Installing the Motor Cable	90
Connecting the Motor	91
Motor Connector	91
Motor Wiring	91
Ethernet Connections	93

Task 3: EtherCAT System Configuration

Install the ESI file	95
Obtain the ESI file	95
Install the ESI file	95
Restart the Programming	
System If Needed	95
Add the SD4840EK to the Project	95
Scan for the SD4840EK	95
Rename the Device	95
Configure the SD4840EK	96
Create a DUT	97
Create Variables Based on the DUT	97
Link Variable Names to I/O Words	97

Task 4: Distributed Clock - SYNC0 Setup

Verify Main PLC Task Timing	99
Create New PLC Task	100
Set Operational Mode	102
Set CFG_word1 Value to	
Enable SYNC0	103

Notes

ABOUT THIS MANUAL

Read this chapter to learn how to navigate through this manual and familiarize yourself with the conventions used in it.

Audience

This manual explains the set-up, installation, and operation of the SD4840EK Stepper Motor Indexer / Driver from AMCI. It is written for the engineer responsible for incorporating these modules into a design, as well as the engineer or technician responsible for its actual installation.

Applicable Units

This manual applies to the SD4840EK Networked Stepper Indexer/Driver with the EtherCAT[®] network interface.

Navigating this Manual

This manual is designed to be used in both printed and on-line formats. Its on-line form is a PDF document, which requires Adobe Acrobat Reader version 7.0+ to open it. The manual is laid out with an even number of pages in each chapter. This makes it easier to print a chapter to a duplex (double sided) printer.

Bookmarks of all the chapter names, section headings, and sub-headings were created in the PDF file to help navigate it. The bookmarks should have appeared when you opened the file. If they didn't, press the F5 key on Windows platforms to bring them up.

The PDF file is password protected to prevent changes to the document. You are allowed to select and copy sections for use in other documents and you are allowed to add notes and annotations.

Manual Conventions

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



NOTES highlight important concepts, decisions you must make, or the implications of those decisions.



CAUTIONS tell you when equipment may be damaged if the procedure is not followed properly.



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.
<i>Emphasis Font</i>	Font used the first time a new term is introduced.
Cross Reference	When viewing the PDF version of the manual, clicking on the cross reference text jumps you to referenced section.
HTML Reference	When viewing the PDF version of the manual, clicking on the HTML reference text will open your default web browser to the referenced web page.

Trademark Notices

The AMCI logo is a trademark of Advanced Micro Controls Inc. EtherCAT[®] is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany. "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, 940-0S310, is the first revision of this manual. It was first released December 14th, 2020.

Revision History

940-0S310 Initial Release.

Where To Go From Here

This manual contains information that is of interest to everyone from engineers to operators. The table below gives a brief description of each chapter’s contents to help you find the information you need to do your job.

Section Title	Page #	Section Description
<i>Introduction to the SD4840EK</i>	9	Complete specifications for the SD4840EK product.
<i>Move Profiles</i>	19	Reference information on how the SD4840EK can be used to control motion in your application.
<i>Calculating Move Profiles</i>	39	Reference information on calculating detailed move profiles.
<i>Homing the SD4840EK</i>	49	Reference information on how to set the home position of the SD4840EK.
<i>Configuration Data Format</i>	53	Reference information on the format of the CANopen over EtherCAT (CoE) data that is used to configure the units.
<i>Command Data Format</i>	59	Reference information on the format of the network data to and from the SD4840EK that is used to command it.
<i>Motor and Power Supply Sizing</i>	79	Reference information and performance data on compatible AMCI stepper motors and guidelines for using a foreign stepper motor.
<i>Installing the SD4840EK</i>	83	Task instructions covering how to install an SMD23K or SMD24K on a machine. Includes information on mounting, grounding, and wiring specific to the units.
<i>EtherCAT System Configuration</i>	95	Task instructions that covers how to add an SD4840EK to an EtherCAT network.
<i>Distributed Clock - SYNC0 Setup</i>	99	Task instructions that covers the optional task of using the Distributed Clock functionality of the EtherCAT protocol with the SD4840EK.

INTRODUCTION TO THE SD4840EK

This manual is designed to get you quickly up and running with the SD4840EK stepper indexer / driver. As such, it assumes you have some basic knowledge of stepper systems. If stepper systems are new to you, we're here to help. AMCI has a great deal of information on our website and we are adding more all the time. If you can't find what you're looking for at <http://www.amci.com>, send us an e-mail or call us. We're here to support you with all of our knowledge and experience.

The SD4840EK

The AMCI SD4840EK is a DC powered, 4.0 Arms, micro-stepping driver with an internal bus voltage equal to the supply voltage. The drive's built-in indexer accepts configuration and command data from a host system. The SD4840EK driver attaches to your Ethernet based network and communicates using the EtherCAT protocol.

EtherCAT uses standard Ethernet cabling, but forgoes the full TCP/IP stack typically associated with Ethernet communications. The EtherCAT protocol transfers data to and from multiple slaves with a single packet of information. Data for each slave is located at a known position within the packet. EtherCAT slave devices use a hardware only solution to read and write data to the packet before transmitting the packet to the next slave. This solution leads to a delay between nodes that is typically 4 microseconds or less. This results in a very fast and deterministic network.

An EtherCAT Slave Information (ESI) file is available for the SD4840EK. The ESI file is required to add the device to the network. Configuration of the SD4840EK is accomplished using the CANopen PDO and SDO objects. EtherCAT supports CANopen through its CANopen over Ethernet (CoE) interface.

Motion commands and status information is transmitted using the output and input variables assigned to the SD4840EK when the EtherCAT system is configured.

Each unit also supports the Distributed Clock (DC) functionality of the EtherCAT system. This allows you to synchronize the start of moves across devices using the SYNC0 signal instead of the SyncManager 2 event.

The combination of host and driver gives you several advantages:

- Sophisticated I/O processing can be performed in the host (PLC or other controller) before sending commands to the Networked Indexer/Driver.
- All motion logic is programmed in the host, eliminating the need to learn a separate motion control language
- Eliminating the separate indexer lowers Total System Cost

The SD4840EK is powered by a nominal 24 to 48 Vdc power source, and can accept surge voltages of up to 60Vdc without damage. The output motor current is fully programmable from 0.1 Arms to 4.0 Arms which makes the SD4840EK compatible with the complete line of size 23 and size 34 stepper motors that are available from AMCI. In addition to the Motor Current setting, the Motor Steps per Turn, Idle Current Reduction, and Anti-Resonance Circuit features are also fully programmable. If you have used other stepper indexer products from AMCI you will find programming to be very similar to these products.

The SD4840EK (continued)

The SD4840EK is a true RMS motor current control driver. This means that you will always receive the motor's rated torque regardless of the *Motor Steps/Turn* setting. (Drivers that control the peak current to the motor experience a 30% decrease in motor torque when microstepping a motor.) The SD4840EK automatically switches from RMS to peak current control when the motor is idle to prevent overheating the motor.

In addition to power and motor hookups, the SD4840EK has three DC inputs and one DC output that are used by the indexer. Configuration data from the host sets the function of these points. The output can be configured to be a Fault Output or a general purpose output. Inputs accept 5 to 24 Vdc signals and they can be individually configured as a:

- CW or CCW Limit Switch
- Home Limit Switch
- Capture Encoder Position Input
- Stop Jog or Registration Move Input
- Start Indexer Move
- Emergency Stop Input
- General Purpose Input

Conformance Markings

The SD4840EK meets the requirements for the following conformance markings when installed and operated in accordance with the instructions contained in their product documentation.

UL

Recognized component in the USA and Canada. UL file number is E231137.

- US: UL 61800-5-1 Standard For Safety For Adjustable Speed Electrical Power Drive Systems - Part 5-1: Safety Requirements - Electrical, Thermal and Energy
- Canada: CSA C22.2 No. 274 - Adjustable Speed Drives

CE

- Directive 2014/30/EU of the European Parliament and of the Council (EMC), of 26 February 2014 on the harmonisation of the laws of the Member States relating to electromagnetic compatibility; per EN 61800-3:2004/A1:2012.
- Directive 2014/35/EU of the European Parliament and of the Council (Low Voltage), of 26 February 2014 on the harmonisation of the laws of the Member States relating to the making available on the market of electrical equipment designed for use within certain voltage limits; per EN 61010-1:2010.

RoHS

Directive (EU) 2015/863 (RoHS 3) and Directive 2011/65/EU (RoHS 2)

Specifications

Driver Type

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

Physical Dimensions

Width: 0.9 inches max.
Depth: 4.5 inches max.
Height: 3.9 inches
5.0 inches min. with mating connectors

Weight

0.38 lb. (0.17 kg) with mating connectors

Inputs

Electrical Characteristics:
Differential. 560 Vac/dc opto-isolated. Can be wired as single ended inputs.
DC Inputs accept 3.5 to 27 Vdc without the need for an external current limiting resistor.
Encoder Inputs are designed for 5 Vdc differential (Operational range of 3 to 6 Vdc.) An external current limiting resistor is required for 12 to 24 Vdc operation.

Output

Electrical Characteristics:
Open Collector/Emitter. 560 Vac/dc opto-isolated. 30 Vdc, 20 mA max.
The Output can be programmed to be a general purpose output or a Fault Output.
The Fault Output is normally on. Turns off under the following conditions:
Reset The driver initialization is not yet complete on power up.
Short Circuit Motor Phase to Phase or Phase to Case
Over Temp Heat Sink temperature exceeds 90° C (195° F)
Faults are reported in the Network Input Data and can be cleared through the Network Output Data.

Motor Current

Programmable from 0.1 to 4.0 Arms in 0.1 Amp steps.

+Vdc Auxiliary Current

70 mA @ 24 Vdc, 40 mA @ 48 Vdc

Motor Resolution

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

Idle Current Reduction

Programmable from 0% to 100% 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.

Internal Power Fuse

7 Amp Fast Blow. Fuse is not user replaceable.

Environmental Specifications

Input Power 24 to 48 Vdc, surge to 60 Vdc without damage to module.
Ambient Operating Temperature
..... -4° to 122° F (-20° to 50° C)
Storage Temperature
..... -40° to 185° F (-40° to 85° C)
Humidity 0 to 95%, non-condensing

Motor Specifications

Type 2 phase hybrid. 4, 6, or 8 lead motor
Inductance 0.3 mH minimum. 2.5 to 45 mH recommended

Status LED's

See *Front Panel* on page 18 for a complete description.

Connectors

Mating connectors are supplied with the module and are also available separately under the following AMCI part numbers.

Connector	AMCI Part #	Wire	Strip Length	Min. Tightening Torque
I/O	MS-2x11	28 - 16 AWG	0.275 inches	Spring Cage Connector
Motor	MS-4M	28 - 12 AWG	0.394 inches	4.43lb-in (0.5 Nm)

Indexer Functionality

The table below lists the functionality offered by the indexer built into the SD4840EK.

Feature	Description
Programmable Inputs	Each of the three inputs can be programmed as a Home Limit, Over Travel Limit, Capture Input, Stop Jog or Registration Move, E-Stop Input, or a General Purpose Input.
Programmable Output	The single output on the SD4840EK can be programmed as a Fault Output or as a general purpose DC output point.
Encoder Inputs	Allows the SD4840EK to used a quadrature encoder for position verification, stall detection, or Electronic Gearing.
Programmable Parameters	Starting Speed, Running Speed, Acceleration, Deceleration, Distance to Move, and Accel/Decel Types are fully programmable.
Homing	Allows you to set the machine to a known position. The SD4840EK can home to a discrete input or to an encoder marker pulse.
Synchronous Moves	Using the Distributed Clock functionality of the EtherCAT system, multiple devices can synchronize the start of their moves to the SYNC0 signal. Moves will begin within ± 25 microseconds of each other.
Jog Move	Allows you to jog the motor in either direction based on an input bit from your host controller.
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 an input bit from your host controller. When a controlled stop is received, 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.
Dwell Move	Allows you to perform a sequence of relative moves with a stop between each move that has a programmable length of time. 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 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.

Table R1.1 Indexer Functionality

Indexer Functionality (continued)

Synchronizing Moves

By default, the SD4840EK uses the SyncManager 2 event to control the transfer of data from the EtherCAT Slave Controller (ECS) to the microprocessor that controls motion. This allows the SD4840EK to execute commands as soon as new data arrives. When more than one axis is updated with a single EtherCAT packet, the time difference between axis updates is very short. Updating multiple axes with a single packet allows the EtherCAT network to out perform the update times of other industrial network protocols.

On very fast machines, or large machines that require more than one transfer to update all axes, the EtherCAT Distributed Clock (DC) functionality can be used to closely synchronize motion over multiple axes if using the SyncMaster2 event proves to be ineffective.

The EtherCAT Distributed Clock functionality is built into the ECS used by the SD4840EK product. The SD4840EK can act as the reference clock for the system if it is the first device in the EtherCAT network. The SYNC0 signal, which is based off of the Distributed Clock, can be used to synchronize the start of moves over multiple devices. The time between when the SYNC0 signal is received by the main processor of the SD4840EK and when the driver begins to cause motion is 520 ± 25 microseconds. The 520 microseconds is the time required to read the data from the ECS once the SYNC0 signal becomes active. The ± 25 microseconds is caused by the 20 kHz update frequency of the PWM drivers.

For the SD4840EK, the minimum update time on the SYNC0 signal is two milliseconds. If the task time is less than two milliseconds, the SYNC0 time must be a multiple of the task time.

Encoder Functionality

In addition to the discrete I/O points, the SD4840EK has three inputs for a 5Vdc differential quadrature encoder. The inputs will also accept 12 to 24Vdc single ended encoder inputs with current limiting resistors. These signals are always decoded using X4 decoding. When the incremental encoder is used, the indexer section of the SD4840EK has the following additional functionality.

Feature	Description
Position Verification	When mounted on the motor that is controlled by the SD4840EK, the encoder can be used to verify motor position motion when a move command is issued. The encoder position value can be preset to any value and it can be captured during a move.
Home to Z pulse	The Z pulse of the encoder can be used to home the machine. In this mode, any discrete input that is configured as a home input will act as a home proximity sensor.
Stall Detection	The encoder can be used to verify motion when a move command is issued. An error is issued if the difference in motor and encoder positions is greater than forty-five degrees.
Encoder Registration Move	Uses the count from a quadrature encoder to determine when a move begins to decelerate and stop. Commonly used in spooling/despooling applications.
Electronic Gearing	The SD4840EK can be configured to control the position of a motor based on feedback from an external encoder. The ratio of encoder pulses to motor pulses is full programmable and can be changed on-the-fly.

Table R1.2 Indexer Encoder Functionality

Indexer Functionality (continued)**Encoder Functionality (continued)*****Stall Detection***

When Stall Detection is enabled, the SD4840EK 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 SD4840EK 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. By enabling stall detection, the SD4840EK can notify the system if the motor shaft moves more than forty-five degrees while the motor should be holding the load. For example, if the motor should hold a load against gravity, but the Idle Current Reduction parameter is set to high, the SD4840EK will issue a stall detection error if the load begins to fall.

Encoder Registration Move

The encoder position acts as a registration mark in an Encoder Registration Move. Typically, the encoder is mounted separately from the motor. With this mode, the SD4840EK will drive the motor until the desired encoder count is reached, at which time the motor begins to decelerate and stop. This functionality is useful in spool winding or unwinding applications, where the motor drives the spool and the encoder measures the material length as it enters or exits the spool. As the spool diameter increases or decreases, the programmed distance remains the same because you are measuring material, not the number of turns.

Electronic Gearing

In this mode, the stepper motor follows the rotation of an external encoder. This encoder is typically attached to another motor. The ratio of encoder pulses to stepper pulses is programmable over a wide range. This mode electronically couples the two motors together through a programmable gear ratio.

Driver Functionality

This table summarizes the features of the stepper motor driver portion of the SD4840EK.


Feature	Benefits
RMS Current Control	RMS current control give the SD4840EK the ability to drive the motor at its fully rated power when microstepping. Peak current controllers typically experience a 30% drop in power when microstepping a motor.
Programmable Motor Current	RMS current supplied to the motor can be programmed from 0.1 to 4.0 amps in 0.1 amp increments. This allows you to use the driver with the full line of AMCI stepper motors.
Programmable Idle Current Reduction	Extends motor life by reducing the motor current when not running. This extends the life of the motor by reducing its operating temperature.
Programmable Current Loop Gain	Allows you to tailor the driver circuitry to the motor's impedance, thereby maximizing your motor's performance.
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 circuitry gives the SD4840EK 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.
Wiring Short Detection	Safety feature that removes power from the motor if a short is detected in one of its windings.
Over Temperature Detection	The SD4840EK sets a warning bit in the network data when the temperature of the module approaches its safe operating threshold.
Over Temperature Protection	Protects the SD4840EK from damage by removing power from the motor if the internal temperature of the driver exceed a safe operating threshold.

Table R1.3 Driver 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 4 Arms and the current reduction to 25% will result in an idle current of 1Apk. (The SD4840EK always switch from RMS to peak current control when the motor is idle.)

Current Loop Gain

This feature gives you the ability to adjust the gain of the power amplifiers in the SD4840EK to match the electrical characteristics of your motor. The value of this parameter can range from 1 to 80 with 80 representing the largest gain increase. In general, using a larger gain will increase high speed torque but the motor will run louder. A lower gain will offer quieter low speed operation at the cost of some high speed torque.

The use of this feature is completely optional and you can leave the Current Loop Gain at its default setting of “1” for standard motor performance.

NOTE  For AMCI motors, use a value of “5” as a starting point.

NOTE  High induction motors, often referred to as “tin can” steppers, need a higher Current Loop Gain setting for normal operation. For these motors, start with a Current Loop Gain of “20”.

I/O Connector

As shown in figure R1.1, the I/O connector is located on the top of the module. All digital I/O connections are made at this connector as well as the power supply connections. The mating connector is supplied with the SD4840EK and is also available from AMCI under the part number MS-2X11. It is also available from Phoenix Contact under their part number 173 88 98.

Available Discrete Inputs

The SD4840EK has three discrete DC inputs that accept 3.5 to 27 Vdc signals. (5 to 24 Vdc nominal) They can be wired as differential, sinking, or sourcing inputs. How the SD4840EK uses these inputs is fully programmable as is their active states. (Inputs can be programmed as Normally Open (NO) or Normally Closed (NC) inputs.)

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. The SD4840EK allows you to define this starting position in three 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 SD4840EK that will cause the unit to seek this sensor. The third option is homing to the Z pulse of a quadrature encoder. When using the Z pulse, you can use one of the inputs or a network data bit as a home proximity sensor. How the SD4840EK actually finds the Home sensor is described in chapter 4, *Homing the SD4840EK*, starting on page 49.

CW Limit Switch or CCW Limit Switch

Each input can be defined as a CW or CCW Limit Switch. When configured this way, the inputs are used to 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 move in the counter-clockwise direction.

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 an input that is configured as a Start Indexer Move Input makes an inactive-to-active state transition. This allows the SD4840EK to run critically timed moves that cannot be reliably started from the network due to data transfer lags.

If the quadrature encoder is enabled and one of the discrete DC inputs is programmed as a Start Indexer Move Input, then the quadrature encoder position data will 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 driver remains enabled and power is supplied to the motor. No move can begin while this input is active.

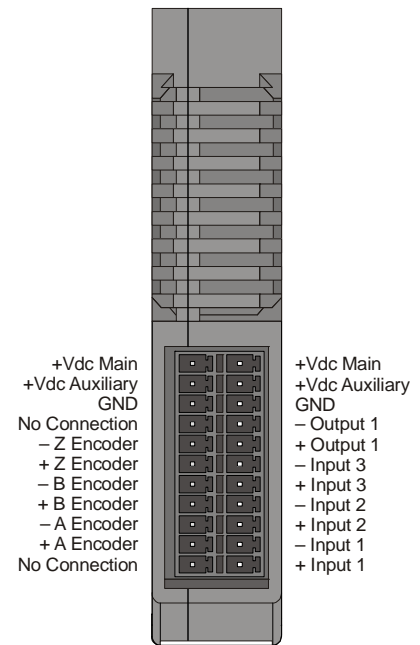


Figure R1.1 I/O Connector

Available Discrete Inputs (continued)

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 quadrature encoder is enabled, the quadrature 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, the SD4840EK 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 all three inputs, you can configure the unused inputs as General Purpose Inputs. The inputs are not used by the SD4840EK, but the input state is reported in the network data.

Available Discrete Output

The SD4840EK has a single DC output that has a maximum rating of 30 Vdc at 20 mA. The output can be configured to be a general purpose output or a Fault Output. When configured as a Fault Output, the output will conduct under normal conditions and will switch off when a fault occurs. The following faults affect the Fault Output:

- Reset The driver initialization is not yet complete on power up.
- Short Circuit ... Motor Phase to Phase or Phase to Earth Ground
- Over Temp Heat Sink temperature exceeds 90° C (195° F)

Faults are reported in the Network Input Data and can be cleared through the Network Output Data.

Ethernet Ports

The Ethernet Interface on the SD4840EK driver has two standard RJ-45 jacks that accept any standard 100Base-TX cable. Both ports are also auto MDI-X capable. This means that a standard cable can be used when connecting the Ethernet Driver to any device. Crossover cables are never needed.

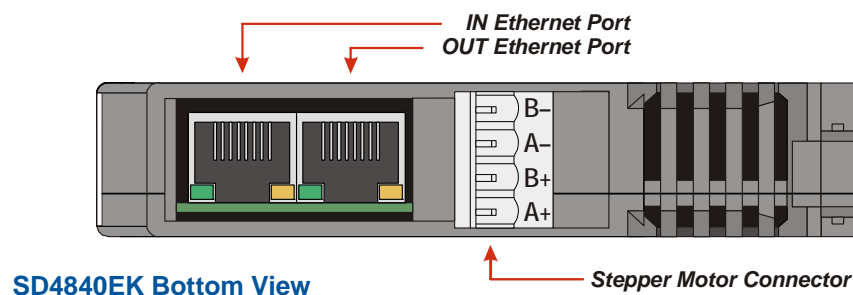


Figure R1.2 Location of Ethernet Ports and Motor Connector

The “IN” network jack in the diagram above must be attached to the upstream device in the EtherCAT network. (Closer to the network master.) The “OUT” jack must be attached to the next (downstream) device in the network of the Networked Driver is not the last device in the network.

There are two LED’s on each of the RJ-45 jacks. The amber LED is not used. The green LED shows the link status of the Ethernet connection. They are on when there is a physical connection between the SD4840EK and the previous or next device in the network. They blink when data is being transmitted over the network.

Motor Connector

Figure R1.2 also shows the location of the Stepper Motor Connector. The mate to this connector is included with the SD4840EK and is also available from AMCI under the part number MS-4M. It is also available from Phoenix Contact under their part number 187 80 37.

Front Panel

The front panel of the SD4840EK contains the drive status LED as well as the two EtherCAT status LED's.

Status LED

The Status LED is a bi-color red/green LED shows the general status of the module.

- **Steady Green:** Module OK
- **Steady Red:** An Overtemperature Fault or Motor Short Circuit Fault exists. Note that the SD4840EK will only detect short circuit faults when the motor current is enabled.
- **Blinking Green:** Successful write to flash memory. Power must be cycled to the module before additional commands can be written to it.
- **Blinking Red:** Failed write to flash memory. You must cycle power to the module to clear this fault.
- **Alternating Red/Green:** There is a communications error between the main processor and the Ethernet co-processor within the drive. You must cycle power to the module to attempt to clear this fault.

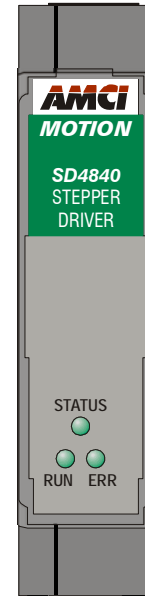


Figure R1.3 SD4840EK Front Panel

EtherCAT Run LED

The green RUN LED indicates the logical state of the device.

LED State	Description
Off	Device in the EtherCAT Init state
Fast Blink (4 Hz)	Device in the EtherCAT Pre-Operational (Pre-Op) state
Slow Blink (1 Hz)	Device in the EtherCAT Safe-Operational (Safe-Op) state
Steady Green	Device in the EtherCAT Operational (Op) state.

Table R1.4 Module RUN LED States

EtherCAT ERR LED

The red ERR LED indicates an error state in the EtherCAT protocol.

Red LED State	Description
Off	No errors in device operation
Single blink 200 ms ON / 1 s OFF	Problems with synchronization such as with the Distributed Clock (DC) PLL.
Double Blink 200 ms Pulses / 1 s between	SYNC manager watchdog timeout. The master has not updated the output data in the configured time interval. The Networked Driver has switched to the Safe-OP state.
Slow Blink (1 Hz)	All other issues, such as cable disconnect. The Networked Driver has switched to the Safe-OP state.

Table R1.5 Red ERR LED States

REFERENCE 2

MOVE PROFILES

When a move command is sent to the SD4840EK, the module 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/second². However, when programming an SD4840EK, all acceleration and deceleration values must be programmed in the unit of measure of steps/second/millisecond.

- ▶ To convert from steps/second² to steps/second/millisecond, divide the value by 1000. This must be done when converting from a value used in the equations to a value programmed into the unit.
- ▶ To convert from steps/second/millisecond to steps/second², multiply the value by 1000. This must be done when converting from the value programmed into a unit to the value used in the equations.

Motor Position

Motor Position is defined in counts, and its limits are based on the data format you choose when configuring the unit. The default multi-word format limits the Motor Position range from -32,768,000 to +32,767,999. If you choose the thirty-two bit double integer format, the range is -2,147,483,648 to +2,147,483,647. In continuous rotation applications, you should choose the double integer format.

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 SD4840EK for speed control, don't require position data at all.

Count Direction

Relative moves with a positive offset will 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 CW Jog Move command, will result in an increase in the motor position value. If the motor is wired as shown in this manual, these moves will result in clockwise rotation when looking at the motor shaft.

Definitions (continued)

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 1,999,999 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. With that said, a starting speed between 0.25 and 0.5 RPS is usually a good starting point.

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 SD4840EK moves use relative coordinates.

- ▶ The range of values for the Target Position when it is treated as an offset is $\pm 8,388,607$ counts. Assuming that the motor is wired to the SD4840EK as shown in this manual, positive offsets will result in clockwise moves, while negative offsets result in counter-clockwise moves.
- ▶ The Motor Position value reported back to the host can exceed $\pm 8,388,607$ counts. The only way to move beyond $\pm 8,388,607$ 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 $\pm 8,388,607$ counts. The move will be positive if the Target Position is greater than the Current Position and negative if the Target Position is less than the Current Position.
- ▶ The Motor Position value reported back to the host can exceed $\pm 8,388,607$ counts. However, you cannot move beyond $\pm 8,388,607$ counts with an Absolute Move. The only way to move beyond $\pm 8,388,607$ counts is with multiple relative moves or jog commands.

Definition of Acceleration Types

Most of the move commands allow you to define the acceleration type used during the move. The SD4840EK supports three types of accelerations and decelerations. The type of acceleration used is controlled by the *Acceleration Jerk* parameter.

What is jerk?

Just as speed is a measurement of change in position per unit time and acceleration is a measurement of change in speed per unit time, jerk is a measurement of change in acceleration per unit time. Likewise, just as a change in position equals speed * time, $\Delta p = s(t)$, and a change in speed equals acceleration * time, $\Delta s = a(t)$, a change in acceleration equals jerk * time, $\Delta a = j(t)$. Jerk has units of steps/sec³.

The SD4840EK uses the jerk property to smoothly change the acceleration 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.

Definition of Acceleration Types (continued)

What is jerk? (continued)

In order to keep the Acceleration Jerk parameter value that is programmed into the SD4840EK below sixteen bits, the SD4840EK's Acceleration Jerk parameter does not have units of $\text{steps}/\text{sec}^3$. The Acceleration Jerk parameter equals $(\{100 * \text{jerk in steps}/\text{sec}^3\} / \text{acceleration in steps}/\text{sec}^2)$. This translates to the jerk property in $\text{steps}/\text{sec}^3$ equalling $(\{\text{Acceleration Jerk parameter}/100\} * \text{acceleration in steps}/\text{sec}^2)$. With the range of values for the Acceleration Jerk parameter being 1 to 5,000, the jerk value ranges from $0.01a$ to $50a$ where "a" is the acceleration value in $\text{steps}/\text{sec}^2$. For example, if the acceleration is programmed to 20,000 $\text{steps}/\text{sec}^2$, then the value of the jerk property used by the module can be programmed to be between 200 $\text{steps}/\text{sec}^3$ ($0.01 * 20,000$) and 1,000,000 $\text{steps}/\text{sec}^3$ ($50 * 20,000$).

Constant 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 Acceleration value. Note that other values will work correctly, but you may notice a quick change in velocity at the beginning or end of the acceleration phase.

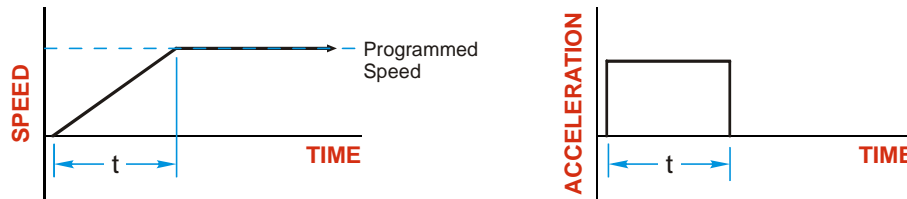


Figure R2.1 Constant Acceleration

Additional information, including example move calculations, can be found in section 3, *Calculating Move Profiles* starting on page 39.

S-Curve Accelerations

When the Acceleration Jerk parameter value is in the range of 1 to 5,000, the SD4840EK uses this value to accelerate and decelerate the rate of acceleration. This is known as S-Curve acceleration because of the shape of the speed curve that results from the variable acceleration.

When using S-Curve accelerations, the starting speed does not have to be equal to the square root of the programmed acceleration value. The S-Curve acceleration will provide smooth transitions at the beginning and end of the acceleration phase.

Trapezoidal S-Curve Acceleration

When the Acceleration Jerk parameter is set high, Trapezoidal S-Curve acceleration usually results. The acceleration value quickly increases (accelerates) until it reaches the value of the Acceleration Parameter. At this point, the acceleration remains constant until the SD4840EK begins to apply the jerk property value to decrease the acceleration value until it equals zero when the programmed maximum speed is reached. Figure R2.2 shows a trapezoidal curve when the Acceleration Jerk setting results in the acceleration being constant for half of the acceleration time. With this setting, the Trapezoidal S-Curve acceleration only requires 33% more time to achieve the same velocity as a Constant Acceleration move.

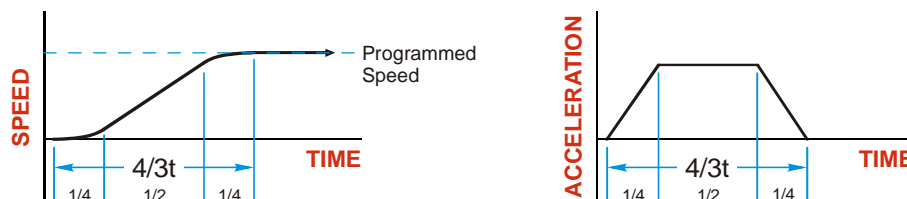


Figure R2.2 Trapezoidal S-Curve Acceleration

Definition of Acceleration Types (continued)

S-Curve Accelerations (continued)

Triangular S-Curve Acceleration

When the Acceleration Jerk parameter is set low, Triangular S-Curve acceleration usually results. This occurs because the programmed maximum acceleration value is not reached before the SD4840EK must start decreasing the acceleration value as the move’s speed approaches its programmed maximum value. Triangular S-Curve is the smoothest form of acceleration, but the time needed to reach the move’s programmed speed is increased. An example is shown in figure R2.3 where the acceleration and jerk settings results in a move that takes twice as long as a Constant Acceleration move to achieve the same velocity.

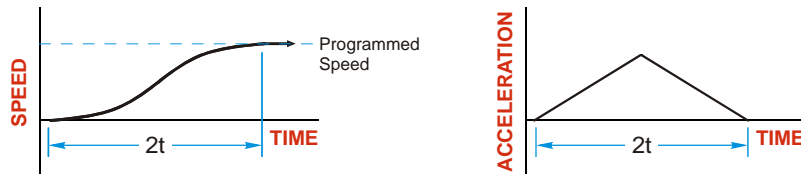


Figure R2.3 Triangular S-Curve Acceleration

Additional information, including example move calculations, can be found in section 3, *Calculating Move Profiles* starting on page 39.

A Simple Move

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

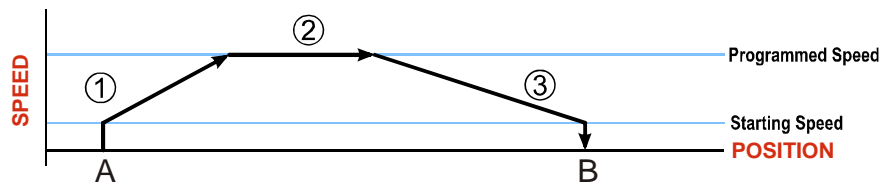


Figure R2.4 A Trapezoidal Profile

- 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 SD4840EK.
- 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 decelerating and the *Programmed Speed* is never reached.

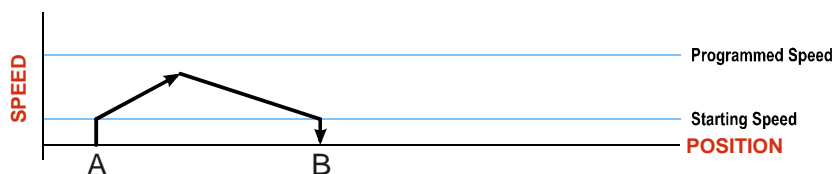


Figure R2.5 A Triangular Profile

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 outputting pulses regardless of the speed the motor is running at. Because 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 and the machine must be homed again before Absolute Moves can be run.

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 24, 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.

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 *CW/CCW Find Home* commands are explained in chapter 3, *Homing the SD4840EK*, which starts on page 49. The *CW/CCW Jog Move* commands are fully explained on page 26, and the *CW/CCW Registration Move* commands are fully explained on page 27.

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.

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 $\pm 8,388,607$ counts. Positive offsets will result in clockwise moves, while negative offsets result in counter-clockwise moves.

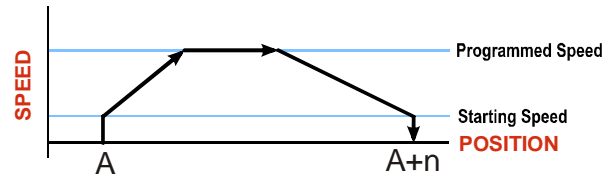


Figure R2.6 Relative Move

NOTE

- 1) You do not have to preset the position or home the machine before you can use a 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 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 the network data. When the command is accepted, the axis will immediately decelerate at the programmed rate and stop. When stopped successfully, the SD4840EK will set a *Hold State* bit in the input data table. The Relative Move can be restarted with the Resume Move command from the network data or the move can be aborted. 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 Stops

- 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 or abandon the held move by starting a new one. The use of the Hold Move and Resume Move bits is further explained in the *Controlling Moves In Progress* section starting on page 35.

Immediate Stops

- The Immediate Stop bit makes a $0 \rightarrow 1$ transition in the Network Output Data.
- A positive transition on an input configured as an E-Stop Input.
- A CW/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.

Basic Move Types (continued)

Absolute Move

Absolute Moves move from the Current Position (A) to a given position (B). (The SD4840EK calculates the 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 $\pm 8,388,607$ 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.

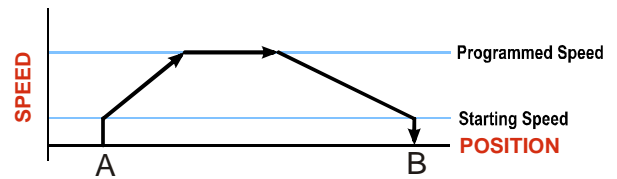


Figure R2.7 Absolute Move

NOTE

- 1) The *Home Position* of the machine must be set before running an Absolute Move. See reference 4, [Homing the SD4840EK](#), which starts on page 49, for information on homing the machine.
- 2) 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.
- 3) 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 Stops

- 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 abandon the held move by starting a new one. The use of the Hold Move and Resume Move bits is explained in the [Controlling Moves In Progress](#) section starting on page 35.

Immediate Stops

- 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/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.

Basic Move Types (continued)

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 cause clockwise motion when the motor is wired as shown in this manual, while the CCW Jog Move will cause counter-clockwise motion. These commands are often used to give the operator manual control over the axis.

Jog Moves are typically used to drive the machine under direct operator control, but they can also be 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 Move 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 SD4840EK 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 SD4840EK that is less than the configured Starting Speed, the Jog Move will continue at the present speed and the *Invalid_Jog_Change* status bit will be set to “1”.

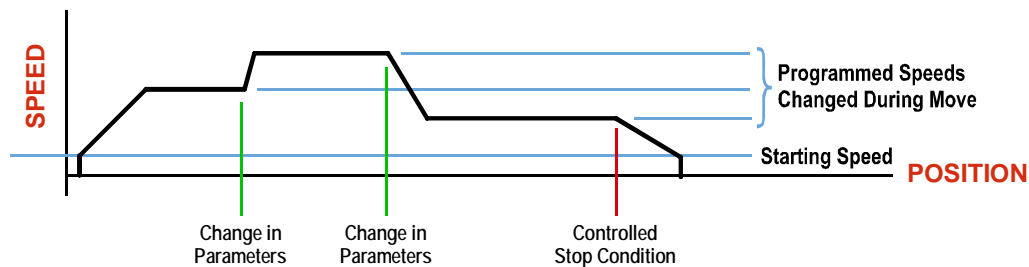


Figure R2.8 Jog Move

Controlled Stops

- 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 35.

Basic Move Types (continued)**CW/CCW Jog Move (continued)****Immediate Stops**

- 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/CCW 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 clockwise 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 will cause clockwise motion when the motor is wired as shown in this manual, while CCW Registration Moves will cause counter-clockwise motion. When the command terminates under Controlled Stop conditions, the SD4840EK 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.

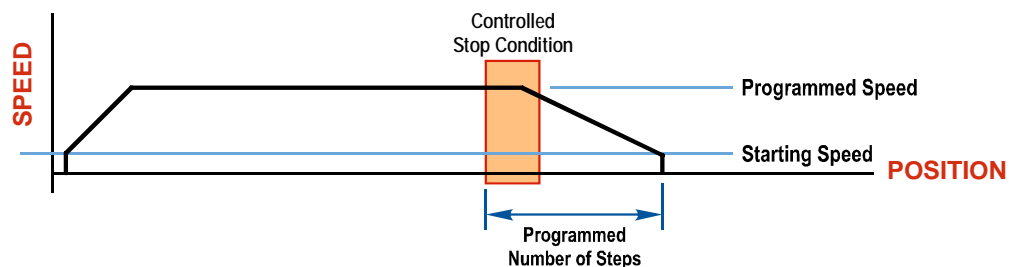


Figure R2.9 Registration Move

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 SD4840EK will decelerate at the programmed Deceleration value until it has output the Programmed Number of Steps and then stop the move without further deceleration.

Basic Move Types (continued)**CW/CCW Registration Move (continued)**

An additional feature of the SD4840EK is the ability to program the driver 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 stays active, the move will begin its controlled stop once the Minimum Registration Move Distance is reached.

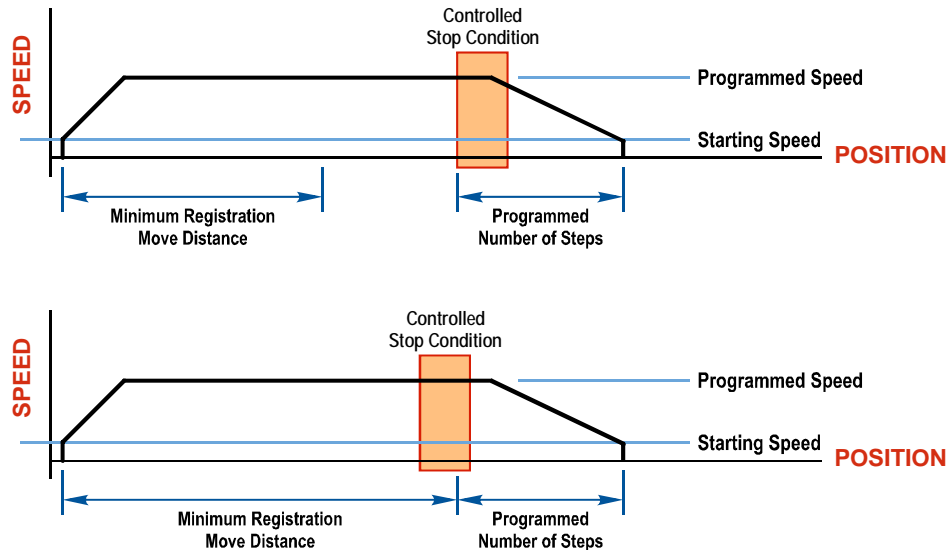


Figure R2.10 Min. Registration Move Distance

Controlled Stops

- 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 SD4840EK responds by using the programmed Deceleration value to bring the move to a stop, without using the programmed Programmed Number of Steps. 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.

Immediate Stops

- 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/CCW 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 clockwise Registration Move while the CCW Limit Switch is active.

Encoder Registration Moves

When the SD4840EK is configured to use a quadrature encoder, the position value from the encoder can be used as a registration mark to control when the move begins to decelerate. Absolute and relative type moves are both supported.

NOTE You do not have to preset the position or home the machine before you can use a relative Encoder Registration Move.

The figure below represents either a relative Encoder Registration Move of 11,000 counts or an absolute Encoder Registration Move to position 16,000. The figure shows that the encoder position you program in the move defines the point at which the motor begins to decelerate and stop. *It does not define the stopping position as it does in other move types.* The endpoint of the move depends on the speed of the motor when the programmed encoder position is reached and the deceleration values. This behavior is different from Absolute and Relative Moves where the position you program into the move is the end point of the move.

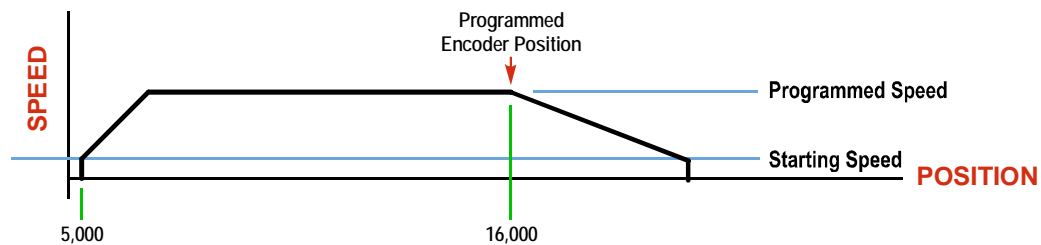


Figure R2.11 Encoder Registration Move

Controlled Stops

- 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 Encoder Registration Move by using the Resume Move bit or abandon the held move by starting a new one. The use of the Hold Move and Resume Move bits is explained in the *Controlling Moves In Progress* section starting on page 35.

Immediate Stops

- 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/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.

Assembled Moves

All of the moves explained so far must be run individually to their completion or be stopped before another move can begin. The SD4840EK gives you the ability to assemble more complex profiles from a series of relative moves that are stored in memory and then run with a single command. Each Assembled Move can consist of 2 to 16 segments. Two types of Assembled Moves exist in the SD4840EK:

- **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. All of the moves are in the same direction.
- **Dwell Move** - A Dwell Move gives you the ability to string multiple relative moves together, and the SD4840EK will stop between each move for a programmed *Dwell Time*. Because motion stops between each segment, a Dwell Move allows you to reverse direction during the move.

Assembled Moves (continued)

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 SD4840EK before the move can be initiated.
 - ▶ The SD4840EK 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 31.
- 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.

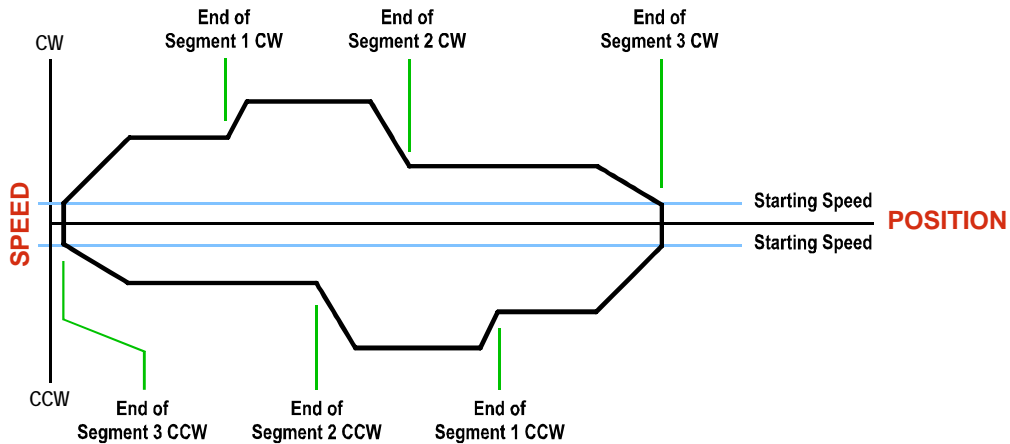


Figure R2.12 Blend Move

NOTE

- 1) 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.
- 2) The Blend Move is stored in the internal memory of the SD4840EK 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 SD4840EK. As described in *Assembled Move Programming* on page 33, 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 configure the SD4840EK and enter Command Mode.
- 3) 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.

Assembled Moves (continued)

Blend Moves (continued)

Controlled Stops

- The move completes without error.
- You toggle the Hold Move control bit in the Network Output Data. When this occurs, the SD4840EK 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. The use of the Hold Move bit is explained in the *Controlling Moves In Progress* section starting on page 35.

Immediate Stops

- 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/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 the SD4840EK 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 programmed *Dwell Time*. The Dwell Time is programmed with the command that starts the Dwell Time. The same Dwell Time is used between 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 moves while a negative segment will result in a counter-clockwise move. 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 can accomplish this Dwell Move with a series of six relative moves that are sent down to the SD4840EK sequentially. The two advantages of a Dwell Move in this case is that the SD4840EK will be more accurate with the Dwell Time than you could accomplish in the PLC, and Dwell Moves simplifies your program's logic.

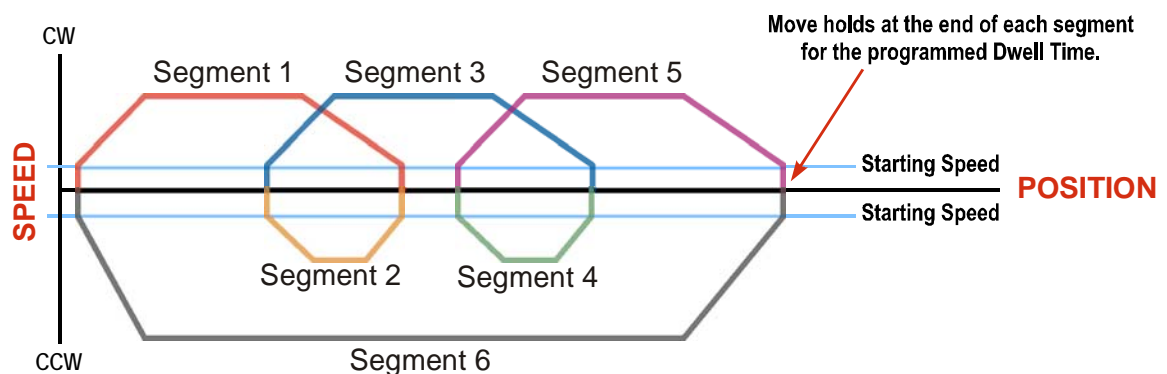


Figure R2.13 Dwell Move

Assembled Moves (continued)**Dwell Move (continued)****NOTE** 

- 1) You do not have to preset the position or home the machine before you can use a Dwell Move. Because the Dwell Move is based on Relative Moves, it can be run from any location.
- 2) The Dwell Move is stored in the internal memory of the SD4840EK 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 SD4840EK. As described in *Assembled Move Programming* on page 33, 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 configure the SD4840EK and enter Command Mode.

Controlled Stops

- The move completes without error.
- You toggle the Hold Move control bit in the Network Output Data. When this occurs, the SD4840EK 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. The use of the Hold Move bit is explained in the *Controlling Moves In Progress* section starting on page 35.

Immediate Stops

- 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/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.

Assembled Move Programming

All of the segments in a Blend or Dwell Move must be written to the SD4840EK 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 SD4840EK as part of the command.

Control Bits – Output Data

- **Program_Assembled bit** – A 0→1 transition on this bit tells the SD4840EK that you want to program a Blend or Dwell Move Profile. The SD4840EK will respond by setting the *In_Assembled_Mode* bit in the Network Input Data. At the beginning of the programming cycle, the SD4840EK will also set the *Waiting_For_Assembled_Segment* bit to signify that it is ready for the first segment.
- **Read_Assembled_Data bit** – A 0→1 transition on this bit tells the SD4840EK that the data for the next segment is available in the remaining data words.

Control Bits – Input Data

- **In_Assembled_Mode bit** – The SD4840EK sets this bit to tell you 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** – A 0→1 transition on this bit from the SD4840EK is the signal to the host that the SD4840EK 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 SD4840EK 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 SD4840EK 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 SD4840EK 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 SD4840EK resets the *In_Assembled_Mode* and the *Waiting_For_Assembled_Segment* bits.

Saving an Assembled Move in Flash

The SD4840EK also contains the *Save_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_to_Flash* bit set. When you reach step 9 in the sequence, the SD4840EK 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 SD4840EK before you can continue. With a limit of 10,000 write cycles, the design decision that requires you to cycle power to the SD4840EK was made to prevent an application from damaging the module by continuously writing to it.

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 three inputs instead of a command from the network. If the *Indexed Move* bit is set when the command is issued, the SD4840EK 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 SD4840EK until the triggered move is finished.
- ▶ As stated above, a move can be run multiple times as long as 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 0x0000 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

The SD4840EK 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 any type of new move from the stopped position.

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

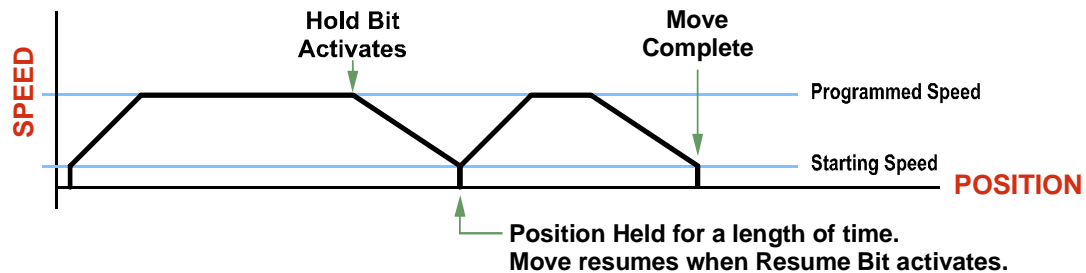


Figure R2.14 Hold/Resume a Move Profile

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 SD4840EK 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, Relative and Encoder Registration Moves

Absolute, Relative and Encoder Registration 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 SD4840EK 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.

Assembled Moves

A Blend or Dwell Move can be placed in a Hold state but cannot be resumed. This gives 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.

Electronic Gearing

The final form of motion control available with the SD4840EK is Electronic Gearing. A quadrature encoder is required but it is not mounted on the motor controlled by the SD4840EK. The encoder is typically mounted on a second motor, but it can be mounted anywhere, including on something as simple as a hand crank.

This mode is sometimes referred to as *encoder following*, because the motor will change position in response to a change in position of the encoder. AMCI refers to it as Electronic Gearing because the SD4840EK has three parameters that allow you to set any turns ratio you want between the encoder and the motor.

Motor Steps/Turn

This is the same parameter explained at the beginning of this chapter. In Electronic Gearing mode, this parameter sets the number of encoder counts required to complete one rotation of the shaft of the motor driven by the SD4840EK. It has a range of 200 to 32,767. This parameter is programmed when you configure the module and cannot be adjusted while a move is in progress.

ELGearing Multiplier and Divisor

The ratio of these two parameters is applied to the number of encoder pulses read by the SD4840EK before it determines the number of motor steps to move. Each parameter has a range of 1 to 255. These two parameters can be adjusted while a move is in progress which allows you to adjust the tracking speed and position of the motor.

How It Works

The SD4840EK always uses 4X decoding when counting pulses from the encoder. If you set both of your ELGearing Multiplier and Divisors to 1 and set the Motor Steps/Turn to four times the number of encoder lines, then the motor will complete one rotation for every rotation of the encoder's shaft.

Once placed in Electronic Gearing mode, the SD4840EK monitors the Jog Move command bits in the Network Output Registers. When either of these bits is set, the encoder inputs are monitored for a change in position. When a change is sensed, the SD4840EK will begin to turn the motor within 50 microseconds. An increase in encoder counts will result in clockwise rotation. A decrease in encoder counts will result in counter-clockwise rotation.

The values of the ELGearing Multiplier and Divisor can be changed while electronic gearing motion is occurring. The SD4840EK will accelerate or decelerate the motor to match the new ratio.

Encoder position data can be trapped while in Electronic Gearing mode by configuring one of the discrete DC input as a Capture Encoder Position input.

Controlled Stops

- The encoder stops moving.
- Both of the Jog Move command bits equal zero.
- Electronic Gearing moves cannot be brought to a controlled stop by using the Hold Move control bit in the Network Output Registers.

Immediate Stops

- 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.

Electronic Gearing (continued)

Advanced Ratio Control

The ELGearing Multiplier and Divisor values give you a great deal of control over the ratio of motor turns per encoder turn, but you can achieve even finer control by adjusting the Motor Steps/Turn parameter.

The Z pulse is not used to correct the encoder position once per turn, so you can actually program the Motor Steps/Turn to any value you want within its valid range. For example, if your encoder outputs 4,096 pulse per turn (a 1,024 line encoder) and you set the Motor Steps/Turn parameter to 8,192, you will have built a 2:1 gear down into your system before applying the ELGearing Multiplier and Divisors. (Two rotations of the encoder = 8,192 counts = 1 motor rotation.)

This technique allows you to set a median gear ratio in your system that you can adjust on-the-fly by using the ELGearing Multiplier and Divisor parameters.

Notes

REFERENCE 3

CALCULATING MOVE PROFILES

This reference was added for customers that must program very precise profiles. Understanding this section is not necessary before programming the SD4840EK and therefore 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.

The equations in this reference use a unit of measure of steps/second/second (steps/second^2) for acceleration and deceleration. However, when programming the SD4840EK, all acceleration and deceleration values must be programmed in the unit of measure of steps/second/millisecond.

- To convert from steps/second^2 to $\text{steps/second/millisecond}$, divide the value by 1000. This must be done when converting from a value used in the equations to a value programmed into the SD4840EK.
- To convert from $\text{steps/second/millisecond}$ to steps/second^2 , multiply the value by 1000. This must be done when converting from the value programmed into the SD4840EK 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 $20,000 \text{ steps/sec}^2$, the smoothest transition occurs when the starting speed is 141. ($141^2 \approx 20,000$)

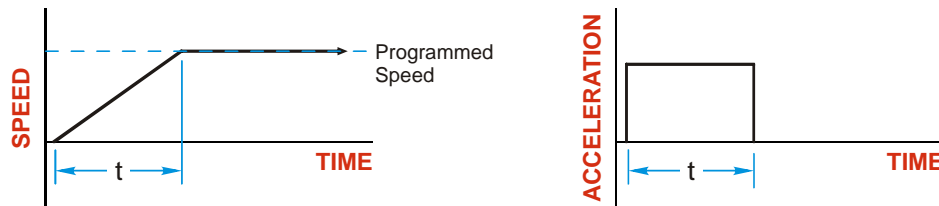
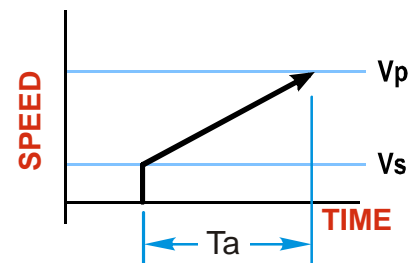


Figure R3.1 Constant Acceleration Curves

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



Constant Acceleration Equations (continued)

Figure R3.1 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 R3.1 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 R3.2 for a move profile.

Name	Value	SD4840EK 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 R3.2 Sample Values

From figure R3.1:

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 SD4840EK will generate a Trapezoidal profile. If the move is equal to 449,984 steps, the unit will generate a Triangular profile and the it will output one pulse at the programmed speed. If the move is less than 449,984 steps, the unit 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:

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

Constant Acceleration Equations (continued)

Continuing the example from table R3.2, assume a total travel distance of 300,000 steps.

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

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

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

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

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

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

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

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

Once you have calculated the maximum speed, you can substitute this value into the time and distance formulas in table R3.1 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_P below), from the second equation.

$$D_P = (\text{Total Number of Steps}) - (D_A + D_D)$$

$$T_P = T_A + T_D + D_P/V_P$$

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

$$T_P = T_A + T_D$$

S-Curve Acceleration Equations

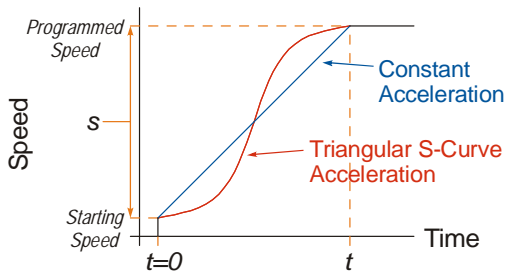
When the Acceleration Jerk parameter value is in the range of 1 to 5,000, the SD4840EK uses this value to smoothly change the acceleration value applied during the move. In this case, the speed of the move does not increase linearly, but exponentially, resulting in an “S” shaped curve. This limits mechanical 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 SD4840EK below sixteen bits, the Acceleration Jerk parameter programmed into the driver does not have units of steps/sec³. The Acceleration Jerk parameter equals $(\{100 * \text{jerk in steps/sec}^3\} / \text{acceleration in steps/sec}^2)$. This translates to the jerk property in steps/sec³ equalling $(\{\text{Acceleration Jerk parameter}/100\} * \text{acceleration in steps/sec}^2)$. With the range of values for the Acceleration Jerk parameter being 1 to 5,000, the jerk value ranges from 0.01a to 50a where “a” is the acceleration value in steps/sec². For example, if the acceleration is programmed to 20,000 steps/sec², then the value of the jerk property used by the unit can be programmed to be between 200 steps/sec³ (0.01*20,000) and 1,000,000 steps/sec³ (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. Variable acceleration provides smooth transitions at the beginning and end of the acceleration phase.

Triangular S-Curve Acceleration

Figure R3.2 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 blue. 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 and we will use this information to calculate the S-curve acceleration and the value of the Jerk Parameter.



$$s = \text{Programmed Speed} - \text{Starting Speed}$$

$$\text{Acceleration} = \frac{\text{speed}}{\text{time}}$$

$$a = \frac{s}{t}$$

$$at = s$$

$$\text{jerk} = \frac{\text{acceleration}}{\text{time}}$$

$$j = \frac{a}{t}$$

$$jt = a$$

$$\text{Unit's Acceleration Jerk Parameter}(J) = \frac{100j}{a}$$

$$j = \frac{Ja}{100}$$

Figure R3.2 Move Profile Example

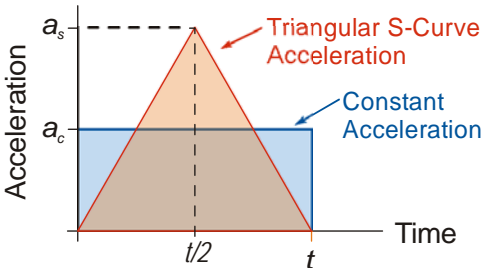


Figure R3.3 Triangular Acceleration

Speed is equal to acceleration multiplied by the time it is applied. This is shown graphically in figure R3.3 as the area of the blue 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:

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

$$a_s = 2a_c$$

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.

S-Curve Acceleration Equations (continued)**Triangular S-Curve Acceleration (continued)**

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

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

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

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

$$Ja_s t = 200a_s$$

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

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 $a_s = a_c$

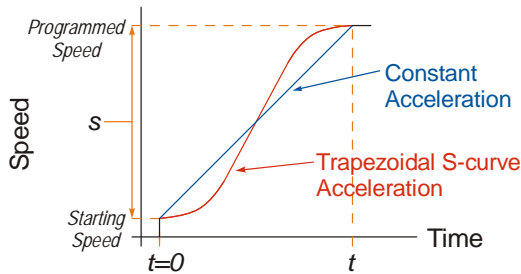
The above examples assume that you can increase the programmed acceleration value to keep the acceleration time the same. If your constant acceleration value is the maximum your system will allow, then using S-curve accelerations will lengthen the time needed to accelerate to your desired speed.

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 $20,000 \text{ steps/sec}^2$ that is applied for 2.0 seconds. If the acceleration value must remain at $20,000 \text{ steps/sec}^2$, then the acceleration phase will take 4.0 seconds and the Acceleration Jerk parameter should be set to 50 ($200/4.0$)

S-Curve Acceleration Equations (continued)

Trapezoidal S-Curve Acceleration

Figure R3.4 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.



$$S = \text{Programmed Speed} - \text{Starting Speed}$$

$$\text{Acceleration} = \frac{\text{speed}}{\text{time}}$$

$$a = \frac{S}{t}$$

$$at = S$$

$$\text{jerk} = \frac{\text{acceleration}}{\text{time}}$$

$$j = \frac{a}{t}$$

$$jt = a$$

$$\text{Unit's Acceleration Jerk Parameter (J)} = \frac{100j}{a}$$

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

Figure R3.4 Move Profile Example

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

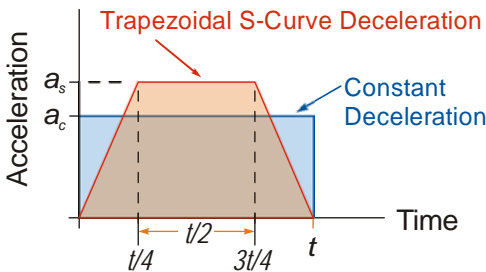


Figure R3.5 Trapezoidal Acceleration

Speed is equal to acceleration multiplied by the time it is applied. This is shown graphically in figure R3.5 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.

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

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

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

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

This means that a trapezoidal S-curve acceleration profile that 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.

S-Curve Acceleration Equations (continued)

Trapezoidal S-Curve Acceleration (continued)

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

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

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

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

$$Ja_s t = 400a_s$$

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

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 R3.6. 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.

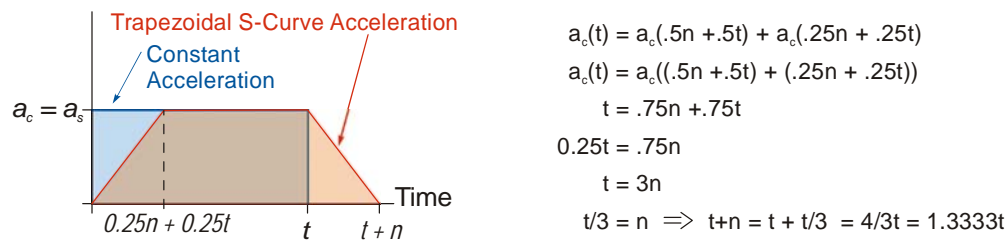


Figure R3.6 Trapezoidal S-curve Time Increase Example

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/sec² that is applied for 2.0 seconds. If the acceleration value must remain at 15,000 steps/sec², 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.

S-Curve Acceleration Equations (continued)

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/sec². 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

$$S_m = \frac{30,000 \text{ steps/sec}}{2} = 15,000 \text{ steps/sec} \quad S_m = \text{midpoint of change in speed}$$

$$J = \frac{100j}{a} \Rightarrow j = \frac{Ja}{100} \quad J = \text{Acceleration Jerk parameter}$$

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

$$j = 11,600 \text{ steps/sec}^3 \quad a_f = \text{calculated final acceleration}$$

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

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

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

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

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/sec², the resulting acceleration is a Triangular S-curve. Total time to accelerate is twice the value calculated above, or 3.216 seconds.

S-Curve Acceleration Equations (continued)**Determining Waveforms by Values (continued)**

Example 2, Jerk = 400

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

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

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

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

S_m = midpoint of change in speed

J = Acceleration Jerk parameter

j = physical jerk property

a_f = calculated final acceleration

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

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

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

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

$$\text{Just as speed} = at, \text{ acceleration} = jt$$

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

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

Because a_f is greater than the programmed acceleration of 58,000 steps/sec², the resulting acceleration is a trapezoidal S-curve. As shown in figure R3.7, 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).

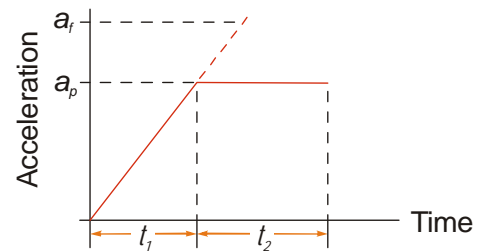
$$232,000 \text{ steps/sec}^3(t_1) = 58,000 \text{ steps/sec}^2 \quad jt = a$$

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

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

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

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



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

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

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

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

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

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

Figure R3.7 Calculating Trapezoidal S-Curve

The time for this acceleration phase is $2(t_1 + t_2)$, which equals $2(0.2500 \text{ sec} + 0.1336 \text{ sec})$ 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

HOMING THE SD4840EK

This chapter explains the various ways of homing an axis on the SD4840EK. Inputs used to home the module are introduced and diagrams that show how the module 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 the SD4840EK must be set to an appropriate value. If you use the module's *CW/CCW Find Home* commands, the motor position register will be set to zero once the home position is reached. The Encoder Position register will also be reset to zero if the quadrature encoder is enabled for the axis.



NOTE Defining a Home Position is completely optional. Some applications, such as those that use a stepper for speed control, don't require position data at all.

With the exception of Absolute Moves, the SD4840EK 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 to the SD4840EK. Before doing this, you will need a way of sensing position outside the SD4840EK module. The machine position data must be brought into the host, the correct preset value calculated, and this value written to the SD4840EK axis with the Position Preset command. The motor and encoder position values can be preset anywhere in the range of -8,388,608 to +8,388,607.

CW/CCW Find Home Commands

The other choice is to use the module's Find Home commands to order the SD4840EK 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, (when the motor is wired as shown in this manual), and ends when the home sensor triggers while the SD4840EK is rotating the motor in a clockwise direction at a low rate. The CCW Find Home command operates in the same way but starts and ends with CCW rotation.

Homing Inputs

Five inputs can be used when homing the module. These inputs are either physical inputs attached to the module or bits in the PLC output data words.

Physical Inputs

- **Home Input:** This input is used in one of two ways: 1) This input is used to define the actual home position of the machine. 2) The input is used as a home proximity input when using the encoder marker pulse to home the machine.
- **Encoder Marker (Z) Pulse:** If you configure the SD4840EK to use an encoder, you have the option of using the encoder's marker pulse to home 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.


Backplane Inputs

- **Backplane_Home_Proximity Bit:** The SD4840EK can be configured to ignore changes on the physical homing input until the Backplane_Home_Proximity Bit makes a 0→1 transition. The SD4840EK 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. Do not use the Backplane_Home_Proximity bit if you only want to home to the Home Limit Switch.


Homing Configurations

The SD4840EK axis must be correctly configured before one of the homing commands will be accepted. One of the following must be part of the module configuration before you can run the homing commands.

- 1) Configure one of the DC inputs as a Home Input
- 2) Configure the SD4840EK to use an encoder and home to the encoder Z-pulse

- NOTE** 
- 1) You do not have to configure and use CW or CCW Limits. If you choose to configure the module this way, then the SD4840EK has no way to automatically prevent overtravel during a homing operation. You must prevent overtravel 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) When using one of the DC inputs as a Home Input, you can use a bit in the network data as a home proximity input. Using this bit is completely optional.
 - 3) When using an encoder's Z-pulse as the homing sensor, any DC input you configure as a Home Input will function as a hardware home proximity sensor. Using this feature is completely optional.

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 R4.1 below shows the move profile generated by a CW Find Home command when you use the Home Input without the Backplane_Home_Proximity bit.

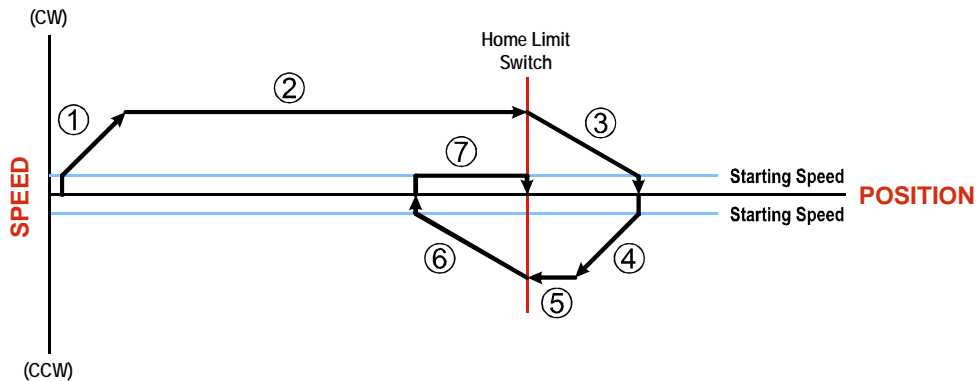



Figure R4.1 Home Input Profile

- 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.

Homing Profiles (continued)

Profile with Proximity Input

Figure R4.2 below shows the move profile generated by a CW Find Home command when you use:

- Home Input with the Backplane_Home_Proximity bit
- Marker Pulse home with Home Input as proximity sensor
- Marker Pulse home with Backplane_Home_Proximity bit

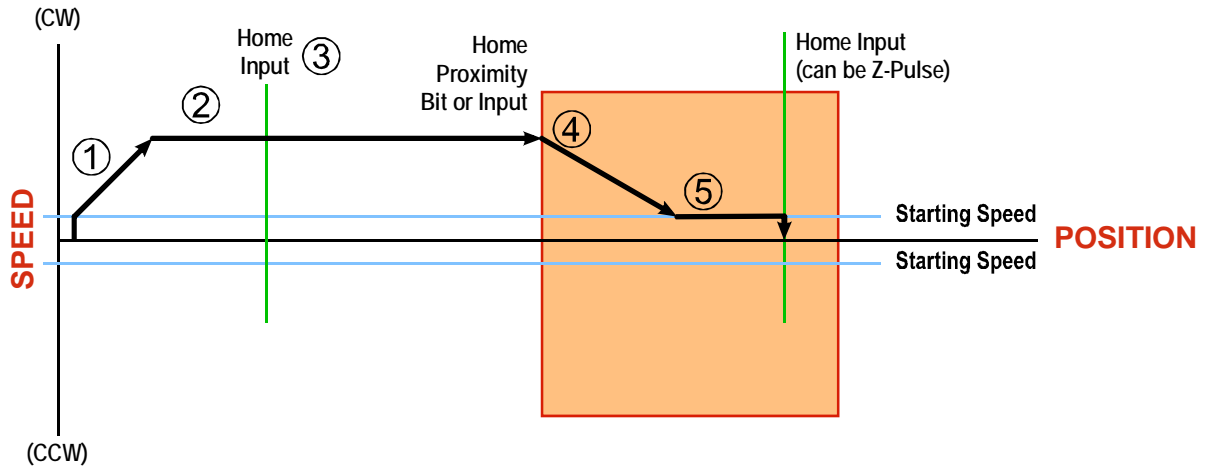



Figure R4.2 Homing with Proximity

- 1) Acceleration from the configured Starting Speed to the Programmed Speed
- 2) Run at the Programmed Speed
- 3) Ignores homing input, (Home Input or Marker Pulse) because the proximity input has not made a 0→1 transition.
- 4) Deceleration towards the Starting Speed when the proximity input (Backplane bit or Home Input) transitions from its inactive to active state. 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 R4.2 shows the Proximity Input, which is either the Backplane_Home_Proximity bit or the Home Input, staying active until the SD4840EK reaches its home position. This is valid, but does not have to occur. As stated in step 4, the SD4840EK starts to hunt for the home position as soon and the Proximity Input makes a 0→1 transition.

Homing Profiles (continued)

Profile with Overtravel Limit

Figure R4.3 below shows the move profile generated by a CW Find Home command when you use:

- CW Overtravel Limit
- Home Input without the Backplane_Home_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.) Hitting the overtravel limit associated with travel in the opposite direction is an Immediate Stop condition. The axis will stop all motion and issue a *Home Invalid* error to your host.

The SD4840EK will stop the axis with an error if both overtravel limits are activated while the unit is trying to find the home position.

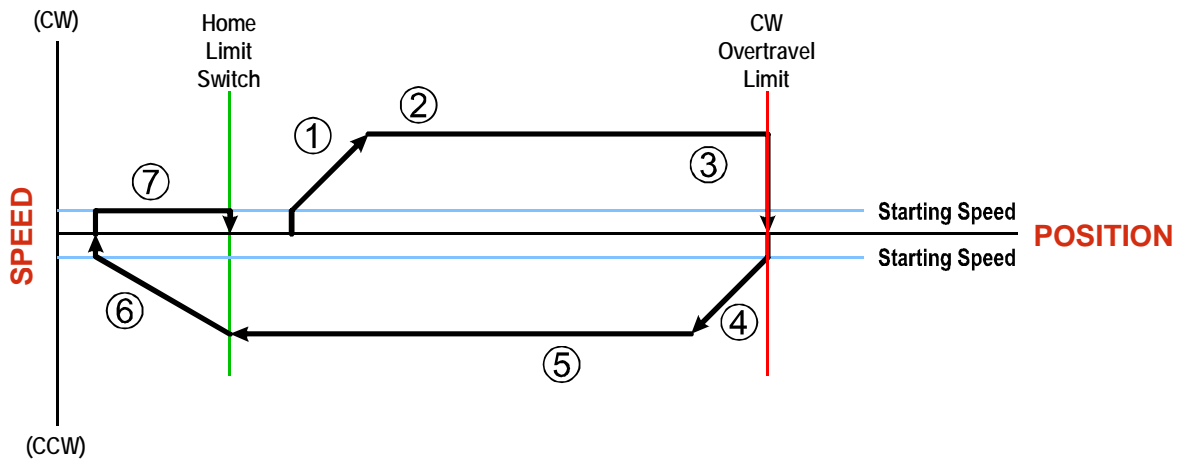


Figure R4.3 Profile with Overtravel Limit

- 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 active, the profile will begin at step 4.

REFERENCE 5

CONFIGURATION DATA FORMAT

This chapter covers the format of the configuration data for an AMCI SD4840EK. Configuration data is stored in registers available through the CANopen over Ethernet (CoE) interface.

CoE Registers

The SD4840EK uses the Service Data Objects (SDO) of the CANopen protocol to configure the unit. Typically, this configuration is specified in the system software and is written down to the SD4840EK when communications is established. The TwinCAT system also allows you to programmatically read or change these values using the FB_EcCoeSdoRead and FB_EcCoeSdoWrite instructions.

Output Data Format

The correct format for the Network Output Data is shown below.

CANopen Index:SubIndex	Data Size	Configuration Data	Range
8010:01	UINT	CFG_Word_0	See below
8010:02	UINT	CFG_Word_1	See below
8010:03	UDINT	Starting_Speed	1 to 1,999,999 steps/sec.
8010:04	UINT	Motor_Resolution	200 to 32,767
8010:05	UINT	Reserved	0
8010:06	UINT	Encoder_Resolution	0 to 32,767
8010:07	UINT	Idle_current_reduction	0 to 100%
8010:08	UINT	Motor_Current (X10)	10 to 40. Represents 1.0 to 4.0 Arms
8010:09	UINT	Current Loop Gain	1 to 80

Table R5.1 Network Output Data Format: Configuration Data

CFG_Word_0 Format

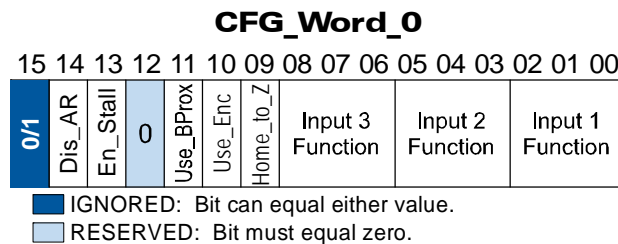


Figure R5.1 Configuration Word 0 Format

Bit 15: Reserved – State ignored.

Bit 14: Disable_Antiresonance – “1” disables the antiresonance feature. “0” enables the anti-resonance feature of the SD4840EK. 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 13: Enable_Stall_Detection – “0” disables motor stall detection. “1” enables motor stall detection. Only valid when an encoder is used and attached to the motor controlled by the SD4840EK. The Encoder_Resolution parameter must also be programmed and must be four times the line count of your encoder.

CFG_Word_0 Format (continued)

Bit 11: Use_Backplane_Proximity – “0” when the Backplane_Proximity_Bit is not used when homing the SD4840EK. “1” when the Backplane_Proximity_Bit is used when homing the SD4840EK. 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 the Home Limit Switch. (Leave this bit equal to “0”.)

Bit 10: Use_Encoder – “0” when Quadrature Encoder is not used. “1” to enable a Quadrature Encoder. You must also enter a pulses per turn value into the Encoder_Resolution register (8010:06) in addition to setting this bit.

Bit 9: Home to Encoder Z Pulse – Set to “1” to home the machine to the encoder’s Z pulse. The Quadrature Encoder Enable Bit, bit 10, must also be set. You must also program the Encoder_Resolution parameter in 8010:06. If a Discrete DC Input is configured as a Home Input, it will act as a Home Proximity Input.

Bits 8-6: Input 3 Function – See Table Below

Bits 5-3: Input 2 Function – See Table Below

Bits 2-0: Input 1 Function – See Table Below

Bits			Function	Available On
8	7	6		
5	4	3		
2	1	0	Function	Available On
0	0	0	General Purpose Input	The input is not used in any of the functions of the SD4840EK, 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 a move that is held in the output registers.
0	1	1	Start Indexed Move / Capture Encoder Value	When the encoder is enabled, 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 SD4840EK.
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, 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. When homing to the Z-pulse of the encoder, (Bit 9 of this word set to “1”), this input will act as a Home Proximity input.
1	1	1	Invalid Combination	This bit combination is reserved.

Table R5.2 Configuration Data: Input Function Selections

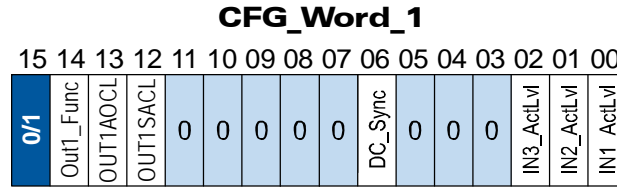
CFG_Word_0 Format (continued)

To Make These Settings...		...Set These Bits To																
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Input 1 ^①	General Purpose				Bit 12 is Reserved. It must always be set to zero.										0	0	0	
	CW Limit															0	0	1
	CCW Limit															0	1	0
	Start Indexed Move [†]															0	1	1
	Stop Jog/Reg. Move [†]															1	0	0
	E-Stop															1	0	1
	Home															1	1	0
Input 2 ^①	General Purpose											0	0	0				
	CW Limit											0	0	1				
	CCW Limit											0	1	0				
	Start Indexed Move ^②											0	1	1				
	Stop Jog/Reg. Move ^②											1	0	0				
	E-Stop											1	0	1				
	Home											1	1	0				
Input 3 ^①	General Purpose								0	0	0							
	CW Limit								0	0	1							
	CCW Limit								0	1	0							
	Start Indexed Move ^②							0	1	1								
	Stop Jog/Reg. Move ^②							1	0	0								
	E-Stop							1	0	1								
	Home							1	1	0								
Home_to_Z ^③	Disabled						0											
	Enabled						1											
Use_Encoder	Disabled					0												
	Enabled					1												
Use_Backplane_Proximity	Disabled				0													
	Enabled				1													
Stall_Detection ^④	Disabled			0														
	Enabled			1														
Antiresonance	To Enable		0															
	To Disable		1															
Resulting Bit Pattern:		0			0													
Hexadecimal Digits for CoE Configuration Setting:																		

- ① Configuring two or more inputs to have the same function, such as two CW Limit Switches, will cause a configuration error. (An error does not occur if two or more inputs are configured as General Purpose Inputs.)
- ② If the encoder is enabled, the encoder position will be trapped and reported when the input makes an inactive-to-active transition.
- ③ Setting the Home_to_Z bit to “1”, without also setting the Use_Encoder bit to “1” will cause an error. If an input is configured as a Home Input when this bit is set, the input will act as a home proximity input.
- ④ Enabling Stall_Detection will cause a error if the encoder is not enabled. (The “Use_Encoder” must equal “1”.)

Table R5.3 Configuration Word 0 Bits

CFG_Word_1 Format



■ IGNORED: Bit can equal either value.
 ■ RESERVED: Bit must equal zero.

Figure R5.2 Configuration Mode: Config Word Format

- Bit 15: Reserved** – State ignored.
- Bit 14: OUT1_Function** – “0” configures Output 1 to be a Fault Output. The output will conduct current until a fault occurs. “1” configures Output 1 to be a general purpose output whose state is determined by a bit in the Command Mode Network Output Data. This output is in an ON state when power is applied to the SD4840EK and it has not yet been configured.
- Bit 13: OUT1_Action_on_Connection_Lost** – This bit is only acted upon if Output 1 is configured as a general purpose output. (Bit 14 above set to “1”.) A “0” on this bit will keep Output 1 at its last value if the network connection is lost. A “1” on this bit will set the state of Output 1 to the value specified in Bit 12 of this word.
- Bit 12: OUT1_State_at_Connection_Lost** – This bit is only acted upon if Output 1 is configured as a general purpose output. (Bit 14 above set to “1”.) When bit 13 of this word is set, Output 1 will be set to the state of this bit if the network connection is lost.
- Bits 11 - 7: Reserved** – Must equal zero.
- Bit 6: DC_Sync** – When set to “0”, the SD4840EK begins a move as soon as data is read from the EtherCAT Slave Controller (ESC). When set to “1”, the SD4840EK will synchronize the move to the SYNC0 output of the system’s Distributed Clock. This allows you to synchronize moves over multiple devices.
- Bits 5 - 3: Reserved** – Must equal zero.
- Bit 2: IN3_Active_Level** – Determines the active state of Input 3. 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.
- 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.

NOTE If you are not using an input, sets its Active_Level bit to “1”. The input will always report as inactive in the network data.

CFG_Word_1 Format (continued)

To Make These Settings...		...Set These Bits To															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input 1	Active Low†																0
	Active High†																1
Input 2	Active Low†															0	
	Active High†															1	
Input 3	Active Low†														0		
	Active High†														1		
DC_Sync	Disabled										0						
	Enabled										1						
OUT1_State_at_Conn_Lost	Output Off				0												
	Output Conducting				1												
OUT1_Action_On_Conn_Lost	Retain Last Value			0													
	Use state of bit 12			1													
OUT1_Function	Fault Output		0														
	GP Output		1														
Resulting Bit Pattern:		0				0	0	0	0	0		0	0	0			
Hexadecimal Digits for CoE Configuration Setting:																	


† Active Low: The SD4840EK will report that the input is active when it is not conducting current.
 Active High: The SD4840EK will report that the input is active when it is conducting current.


Table R5.4 Configuration Word 1 Bits


Current Loop Gain

This feature gives you the ability to adjust the gain of the power amplifiers in the SD4840EK to match the electrical characteristics of your motor. The value of this parameter can range from 1 to 80 with 80 representing the largest gain increase. In general, using a larger gain will increase high speed torque but the motor will run louder. A lower gain will offer quieter low speed operation at the cost of some high speed torque.

The use of this feature is completely optional and you can leave the Current Loop Gain at its default setting of “1” for standard motor performance.

NOTE  AMCI motors should use a Current Loop Gain value of “5”.

NOTE  High induction motors, often referred to as “tin can” steppers, need a higher Current Loop Gain setting for normal operation. For these motors, start with a Current Loop Gain of “20”.

NOTE  If your motor stalls at high speeds, use the motor's torque curve to verify that it is able to generate sufficient torque at the required speed. If it should, increase the Current Loop Gain value to see if its setting has any effect.


Current Loop Gain (continued)

Determining Current Loop Gain (optional)

Determining the current loop gain setting require the following system setup:

- Stable line voltage at the correct level. (Don't determine the setting at 24 Vdc when the drive will operate at 48 Vdc.)
- Motor current set to the operating level. (Don't determine the setting with a motor current of 2 amps if the motor will operate at 3 amps in the production environment.)
- Idle current reduction set to 100%, which means no reduction. (This setting should be used regardless of the setting used in the production environment.)
- An acoustically quiet environment. (You will be listening for quiet changes in the sound of the motor when it is idle.)

- 1) Write a Current Loop Gain value of 5 to the SD4840EK.
- 2) Enable the SD4840EK and listen for a sound from the motor. It will subjectively sound like an AM radio static.
- 3) If you don't hear the radio static sound from the motor, repeat steps 1 and 2 with an increased Current Loop Gain.
- 4) Once you hear the radio static sound from the motor, decrease the Current Loop Gain setting by 10% to 20%. The goal is to eliminate the sound from the motor.

NOTE  Refer to the documentation for your programming software to learn how to set the Current Loop Gain using the CoE registers and how to make the changes permanent. You can refer to step 3.2.3, *Configure the SD4840EK on page 96* of this manual for brief instructions when using the TwinCAT software.

Notes on Other Configuration Words

- Changes to the Idle Current only take effect at *the end of the first move after re-configuration.*
- The motor current can be set to any value within its range.

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.
- 3) Configuring two or more inputs to have the same function, such as two CW Limit Switches. (An error does not occur if two or more are configured as General Purpose Inputs.)
- 4) Setting the Input Configuration bits for any input to "111". See table R5.2 on page 54 for more information.
- 5) Using an encoder and not setting the Encoder_Resolution register (8010:06) to a valid value.
- 6) Setting the Stall Detection Enable Bit without configuring the SD4840EK to use the encoder.
- 7) Setting the Home to Encoder Z-Pulse Bit without configuring the SD4840EK to use the encoder.

REFERENCE 6

COMMAND DATA FORMAT

This chapter covers the format of the Network Output Data used to command the SD4840EK and the format of the Network Input Data that contains the responses from the device. The parameter names of the input and output data are defined by the device ESI file. The size of each data element is also defined by the ESI file.

Command Bits Must Transition

Commands are only accepted when a command bit makes a 0→1 transition. The easiest way to do this is to write a value of zero into the CMD_word0 before writing the next command.

The command bits are split between two, 16 bit words, CMD_word0 and CMD_word1. Only one bit in CMD_word0 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 SD4840EK.

ESI File Name	Data Size	Function
CMD_word0	UINT	Command Word 0
CMD_word1	UINT	Command Word 1
Position	DINT	Command Parameters Word meaning depends on the command set to the SD4840EK
Velocity	UDINT	
Acceleration	UINT	
Deceleration	UINT	
Motor_Current	UINT	
Jerk	UINT	

Figure R6.1 Command Data Format

CMD_word0

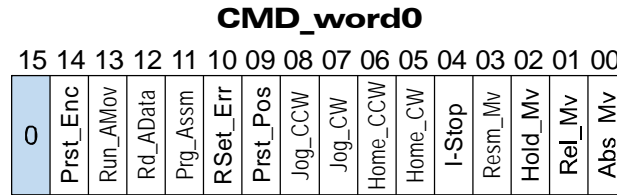



Figure R6.2 CMD_word0 Format

Bit 15: Mode_Select – “0” for Command Mode programming.

NOTE  It is possible to write configuration data to the unit using the output words, but the use of this method is discouraged. Configuration data should be written to the SD4840EK using the CoE interface as described in the previous chapter.

Bit 14: Prset_Encoder – When set to “1” the SD4840EK will preset the Encoder Position to the value stored in the DINT Position output register.

Bit 13: Run_Assembled_Move – When set to “1” the SD4840EK will run the Assembled Move already stored in memory.

- ▶ **Assembled_Move_Type – CMD_word1, 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 with the UINT Jerk output register of the Command Data. The value is programmed in milliseconds and can range from 0 to 65,536.
- ▶ **Reverse_Blend_Direction – CMD_word1, 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.

Bits 11 & 12: Program_Assembled & Read_Assembled_Data – 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 33.

Bit 10: Reset_Errors – When set to “1” the SD4840EK will clear all existing errors and reset the *Move_Complete* bit. A driver fault will be cleared if the Clear_Driver_Fault bit, (CMD_word1, bit 10), is set to “1” when the Reset Errors command is issued.

Bit 9: Prset_Position – When set to “1” the SD4840EK will preset the Motor Position to the value stored in the DINT Position output register and reset the Move_Complete bit in the Network Input Data.

Bit 8: Jog_CCW – When set to “1” the SD4840EK 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 26.

- ▶ **Registration_Move – CMD_word1, 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.
- ▶ **Enable_Electronic_Gearing – CMD_word1, Bit 6:** When this bit equals “1” the SD4840EK will switch its operation to *Electronic Gearing* mode as described on page 36. While in this mode, the two Jog Move bits are used to enable motor motion. One of these two bits must equal “1” before the motor will follow a change in encoder position.

CMD_word0 (continued)

- Bit 7: Jog_CW** – When set to “1” the SD4840EK will run a Jog Move in the clockwise direction. The full explanation of a *CW/CCW Jog Move* can be found starting on page 26.
- **Registration_Move – CMD_word1, 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.
 - **Enable_Electronic_Gearing – CMD_word1, Bit 6:** When this bit equals “1” the SD4840EK will switch its operation to *Electronic Gearing* mode as described on page 36. While in this mode, the two Jog Move bits are used to enable motor motion. One of these two bits must equal “1” before the motor will follow a change in encoder position.
- Bit 6: Find_Home_CCW** – When set to “1” the SD4840EK 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 the SD4840EK* reference chapter starting on page 49.
- Bit 5: Find_Home_CW** – When set to “1” the SD4840EK will attempt to move to the Home Limit Switch in the clockwise direction. A full explanation of homing can be found in the *Homing the SD4840EK* reference chapter starting on page 49.
- Bit 4: Immediate_Stop** – When set to “1” the SD4840EK 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 3: Resume_Move** – Set to “1” to 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 35. Note that a move in its hold state does not need to be resumed. The move is automatically cancelled if another move is started in its place.
- Bit 2: Hold_Move** – Set to “1” to hold a move. 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 35.
- Bit 1: Relative_Move** – Set to “1” to perform a Relative Move using the data in the rest of the Command Data. The full explanation of a *Relative Move* can be found starting on page 24.
- Bit 0: Absolute_Move** – Set to “1” to perform an Absolute Move using the data in the rest of the Command Data. The full explanation of an *Absolute Move* can be found starting on page 25.

CMD_word1

CMD_word1

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
En_Driver	0	GPOut_State	0	BP_Prox	Clr_Drv_Flt	AsMv_Type	Index_Cmd	Reg_Move	En_ElGear	Save_Asmv	Rev_BlendDir	0	Enc_Reg_Mv	Current_Key1	Current_Key0

Figure R6.3 CMD_word1 Format

- Bit 15: Enable_Driver** – “0” to disable the motor current, “1” to enable motor current. A valid configuration must be written to the SD4840EK before the driver can be enabled.
- Bit 14: Reserved** – Must be set to “0”.
- Bit 13: OUT1_Set_State** – When the output is configured as a general purpose output point instead of the Fault Output, this bit controls the state of the output. When this bit equals a “1”, the output is on and conducts current.
- Bit 12: Reserved** – Must be set to “0”.
- Bit 11: Backplane_Proximity_Bit** – When the SD4840EK is configured to use the Backplane_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 Backplane_Proximity_Bit can be found in the *Profile with Proximity Input* section found on page 51.
- Bit 10: Clear_Driver_Fault** – If this bit is set when a Reset Errors Command is issued, (CMD_word0, Bit 10) the SD4840EK will attempt to clear driver errors. Note that the driver must be disabled (CMD_word1, Bit 15 = 0), when using this command.
- Bit 9: Assembled_Move_Type** – When this bit equals “0”, a Blend Move is started when the Run Assembled Move bit, (CMD_word1, 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, (CMD_word1, Bit 4). In a Dwell Move, the Dwell Time between segments is programmed with the UINT Jerk register of the command data.
- Bit 8: Indexed_Command** – If this bit is set when a move command is issued, the SD4840EK 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.
- 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 6: Enable_Electronic_Gearing** – Set to “1” to put the SD4840EK in Electronic Gearing mode. Set to “0” for normal operation. A full description of Electronic Gearing mode starts on page 23.
- Bit 5: Save_Assembled_to_Flash** - Set this bit to save the data of a programmed Assembled Move. This bit is only acted upon this way when the Program_Assembled bit (CMD_word0, Bit 11) makes a 1→0 transition as explained in the *Assembled Move Programming* section of this manual starting on page 33.

CMD_word1 (continued)

- 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.
- Bit 3: Reserved** – Must be set to “0”.
- Bit 2: Encoder_Registration_Move** – This bit is used with the Relative_Move (CMD_word0, bit 1) or Absolute_Move (CMD_word0, bit 0) bits. Set this bit to “1” to command an Encoder Registration Move. This move will run until the encoder count reaches the programmed relative or absolute value and then the move will begin to decelerate and stop. Must equal “0” for normal relative or absolute moves. A full description of an *Encoder Registration Move* can be found starting on page 14.
- Bits 1&0: Current_Key1 and Current_Key0** – These bits can be used to set the motor current “on-the-fly” by setting the bits to a value of “10”. When these bits are “10”, the base motor current will be set to the value contained in the Motor_Current word of the command data.
- ▶ When these bits are set to “10”, changes in the Motor_Current word are acted upon immediately. The range of values for this word are 0 to 40. (0.0 to 4.0 amps) Values outside of this range are ignored and the last valid motor current setting will continue to be used.
 - ▶ When these bits are set to “00”, “01”, or “11” the motor current is left at the last accepted value.
 - ▶ Note that this procedure sets the *base* motor current. Any idle current reduction value set in the Configuration data will still affect the actual current delivered to the motor when the motor is idle.

Command Blocks

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

Absolute Move

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0001
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Absolute Target Position	Steps	-8,388,608 to +8,388,607
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.1 Absolute Move Command Block

Relative Move

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0002
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Relative Target Position	Steps	-8,388,608 to +8,388,607
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.2 Relative Move Command Block

Command Blocks (continued)**Hold Move**

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0004
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Unused		See Note Below
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.3 Hold Move Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Resume Move

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0008
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Unused		See Note Below
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.4 Resume Move Command Block

Unused words are ignored by the SD4840EK 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.

Command Blocks (continued)

Immediate Stop

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0010
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Unused		See Note Below
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.5 Immediate Stop Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Find Home CW

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0020
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Unused		See Note Below
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.6 Find Home CW Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Command Blocks (continued)**Find Home CCW**

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0040
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Unused		See Note Below
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.7 Find Home CCW Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Jog CW

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0080
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Bits 7 & 6 must equal "00"
Position	DINT	Unused		See Note Below
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.8 Jog Move CW Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Command Blocks (continued)

Registration Move CW

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0080
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Bits 7 & 6 must equal “10”
Position	DINT	Stopping Distance	Steps	0 to +8,388,607
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current†	UINT	Minimum Registration Move Distance	steps	UDINT value between 0 and +8,388,607
Jerk†	UINT			

Table R6.9 Registration Move CW Command Block

† Motor Current and Jerk parameters cannot be programmed as part of a Registration Move. These two words are used to store the UDINT value of the Minimum Registration Move Distance. The structured text example below splits the thirty-two bit Minimum Registration Move Distance (nMRMDist) into the two sixteen bit registers. This code was written with TwinCAT software.

```
Motor_Current := UDINT_TO_UINT(nMRMDist); //Motor_Current stores the lower 16 bits.
Jerk := UDINT_TO_UINT(SHR(nMRMDist, 16)); //Jerk stores the upper 16 bits.
```

Jog CCW

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0100
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Bits 7 & 6 must equal “00”
Position	DINT	Unused		See Note Below
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Motor Current	0.1 amps	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.10 Jog CCW Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Command Blocks (continued)**Registration Move CCW**

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0100
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Bits 7 & 6 must equal "10"
Position	DINT	Stopping Distance	Steps	0 to +8,388,607
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current [†]	UINT	Minimum Registration Move Distance	steps	UDINT value between 0 and +8,388,607
Jerk [†]	UINT			

Table R6.11 Registration Move CCW Command Block

[†] Motor Current and Jerk parameters cannot be programmed as part of a Registration Move. These two words are used to store the UDINT value of the Minimum Registration Move Distance. The structured text example below splits the thirty-two bit Minimum Registration Move Distance (nMRMDist) into the two sixteen bit registers. This code was written with TwinCAT software.

```
Motor_Current := UDINT_TO_UINT(nMRMDist); //Motor_Current stores the lower 16 bits.
Jerk := UDINT_TO_UINT(SHR(nMRMDist, 16)); //Jerk stores the upper 16 bits.
```

Command Blocks (continued)

Encoder Follower Move

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0080 or 16#0100
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Bit 6 must equal “1”
Position†	DINT	Upper 16 bits: EG Numerator		Value between 1 and 255
		Lower 16 bits: EG Denominator		Value between 1 and 255
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Motor Current	0.1 A	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Unused		See Note Below

Table R6.12 Encoder Follower Move Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values from the previous command.

† The 32 bit position register is used to store the Electronic Gearing Numerator and Denominator values. Both of these values are USINT values. The structured text example below combines the two values into the one 32 bit register. This code was written with TwinCAT software.

```
//Denominator stored in lower 16 bits.
Position := USINT_TO_DINT(nEGDenominator);
//Numerator converted, shifted, and ANDed into the upper 16 bits.
Position := Position AND SHL(USINT_TO_DINT(nEGNumerator), 16);
```

Preset Position

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0200
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Position Preset Value	Steps	-8,388,608 to +8,388,607
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.13 Preset Position Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Presetting the position will also reset the *Move_Complete* status bit in the Network Input Data.

Command Blocks (continued)**Reset Errors**

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0400
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Set bit 10 to clear driver faults
Position	DINT	Unused		See Note Below
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.14 Reset Errors Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

- Issuing a Reset Errors command will reset the *Move_Complete* status bit in the Network Input Data.
- Issuing a Reset Errors command will not reset the *Position_Invalid* or *Configuration_Error* bits.

Run Assembled Move

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#2000
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62 Blend Move: Bit 9 = "0" Dwell Move: Bit 9 = "1" The Blend Move direction is controlled by Bit 4.
Position	DINT	Unused		See Note Below
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Motor Current	0.1 A	0 to 40 (0.0 to 4.0 amps)
Jerk	UINT	Unused with Blend Move Dwell Time with Dwell Move	milliseconds	0 to 65,535

Table R6.15 Run Assembled Move Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Command Blocks (continued)

Preset Encoder Position

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#4000
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Encoder Preset Value	Counts	-8,388,608 to +8,388,607
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.16 Preset Encoder Position Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Programming Blocks

The following blocks are used to program an Assembled Move. Both of the moves, 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

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0800
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Unused		See Note Below
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.17 Assembled Move First Programming Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values from the previous command.

Once the first block is transmitted, the SD4840EK responds by setting bits 8 and 9 in STATUS_word0. (See *STATUS_word0 Format* starting on page 74.) Once these are set, you can then start transmitting Segment Blocks.

Segment Block

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#1800
CMD_Word1	UINT	<i>CMD_word1</i>		See pg. 62
Position	DINT	Relative Target Position	Steps	-8,388,608 to +8,388,607
Velocity	UDINT	Programmed Speed	Steps/Second	Value between the configured Starting Speed and 2,999,999
Acceleration	UINT	Acceleration	Steps/sec/ms	1 to 5000
Deceleration	UINT	Deceleration	Steps/sec/ms	1 to 5000
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Acceleration Jerk		0 to 5000

Table R6.18 Assembled Move Segment Programming Block

Note that each Segment Block starts with bits 11 and 12 set in the CMD_word0 word (16#1800). When the SD4840EK sees bit 12 of CMD_word0 set, it will accept the block and reset bit 9 in STATUS_word0. When your program sees this bit reset, it must respond by resetting bit 12 of CMD_word0. The SD4840EK will respond to this by setting bit 9 in STATUS_word0 and the next Segment Block can be written to the SD4840EK. You can write a maximum of sixteen Segment Blocks for each Assembled Move.

Input Data Format

The format of the Network Input Data when the SD4840EK is in Command Mode is shown below.

ESI File Name	Data Size	Function
STATUS_word0	UINT	Status Word 0
STATUS_word1	UINT	Status Word 1
Motor_Position	DINT	Current commanded motor position
Encoder_Position	DINT	Current encoder position. (Zero if encoder is not available or disabled.)
Trapped_Encoder_Position	DINT	Last captured encoder position
Motor_Current	UINT	Motor current of last move. †
Jerk	UINT	Jerk value of last move.

† Note that this is the value of the most recently programmed motor current. It is not the actual motor current being sent to the motor when the input data was read.

Table R6.19 Network Input Data Format: Command Mode

STATUS_word0 Format

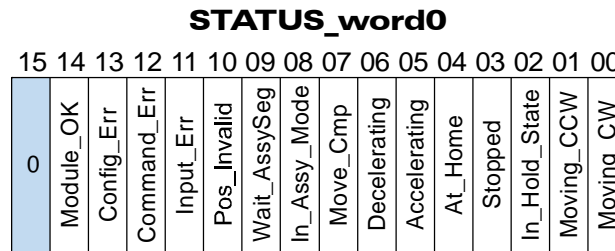


Figure R6.4 Command Mode: Status_word0 Format

- Bit 15: Mode_Flag** – Set to “0” in Command Mode.
- Bit 14: Module_OK** – “1” when the SD4840EK is operating without a fault, “0” when an internal fault condition exists.
- Bit 13: Configuration_Error** – “1” on power up before a valid configuration has been written to the SD4840EK or after any invalid configuration has been written to the driver. “0” when the SD4840EK has valid configuration data.
- Bit 12: Command_Error** – “1” when an invalid command has been written to the SD4840EK. This bit can only be reset by the Reset_Errors bit, CMD_word0, Bit 10. Note that setting the motor current to a value outside the range of 0.0 to 4.0 amps does not force a command error. The driver simply ignores the new value and uses the last accepted value for the motor current.

Input Data Format (continued)**STATUS_word0 Format (continued)**

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.
- A SyncManager 2 time out event has occurred. This is indicative of a networking error that occurred during the EtherCAT Pre-OP or OP states.

This bit is reset by a *Reset Errors* command. The format of the command is given on page 71.

Bit 10: Position_Invalid – “1” when:

- A configuration change is written to the unit through the C0E SDO interface.
- The motor position has not been preset
- The machine has not been homed
- 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 9: Waiting_For_Assembled_Segment – The SD4840EK sets this bit to tell the host that it is ready to accept the data for the next segment of your assembled move profile. Its use is explained in the *Assembled Move Programming* section of this manual starting on page 33.

Bit 8: In_Assembled_Mode – The SD4840EK sets this bit to signal the host that it is ready to accept assembled move profile programming data. Its use is explained in the *Assembled Move Programming* section of this manual starting on page 33.

Bit 7: Move_Complete – Set to “1” when the present Absolute, Relative, Jog, Registration, or Assembled Move command completes without error. This bit is reset to “0” when the next move command is written to the SD4840EK, 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 6: Decelerating – Set to “1” when the present move is decelerating. Set to “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 4: At_Home – Set to “1” when a homing command has completed successfully, “0” at all other times.

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

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

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

Input Data Format (continued)

STATUS_word1 Format

STATUS_word1															
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Drive_Enabled	Stall_Detected	Cmd_Ack	0	Heartbeat_Bit	Limit_Condition	Inv_Jog_Cng	0	Driver_Fault	0	SafeOp	Temp_90°C	0	IN3_Active	IN2_Active	IN1_Active

Figure R6.5 Command Mode: STATUS_word1 Format

- Bit 15: Drive_Is_Enabled** – Set to “1” when the motor driver section of the SD4840EK is enabled and current is available to the motor. Set to “0” when the motor driver section is disabled. If this bit is set to “1”, the motor current remains present when an E-Stop input is active. This bit will remain set if the motor current has been removed because motion is not occurring and the Idle Current Reduction is programmed to its 0% setting. Motor current is removed if there is a Driver_Fault (Bit 7 below) regardless of the state of this bit.
- Bit 14: Stall_Detected** – Set to “1” when a motor stall has been detected.
- Bit 13: OUT1_State** – Present actual state of Output 1. When this bit is set to “1”, the output is in its on state and conducts current.
- Bit 12: Reserved Bit** – Will always equal zero.
- 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.
- 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 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. While in Electronic Gearing mode, this bit is also set if the Numerator or Denominator are set outside their ranges of 1 to 255.
- Bit 8: Reserved** – Will always equal zero.
- Bit 7: Driver_Fault** – If the driver section of the SD4840EK is enabled, this bit will be a “1” during a Over temperature Fault, a Short Circuit Fault, or when the Interlock Jumper is missing. This fault can be cleared by issuing a *Reset Errors* programming block with the Clear_Driver_Fault bit, (CMD_word1, bit 10) set to “1”. For additional information, see *Notes on Clearing a Driver Fault* on page 77. Note that the driver fault bit is set at powerup if the interlock is missing and the driver is immediately enabled.
- Bit 6: Reserved** – Will always equal zero.
- Bit 5: SafeOp** – Set to “1” when the network is in Safe_Op mode. Moves cannot be commanded while in this mode. Reset to “0” when the network is in Operational mode. Moves can be commanded when in Op mode.
- Bit 4: Temperature_Above_90C** – 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 module, consider changing installing a fan to force additional airflow through the SD4840EK.
- Bit 3: Reserved** – Will always equal zero.

Input Data Format (continued)**STATUS_word1 Format (continued)**

- Bit 2:** **IN3_Active** – “1” when Input 3 is in its active state. The active state of the input is programmed as explained in the *CFG_Word_1 Format* section starting on page 56.
- 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 *CFG_Word_1 Format* section starting on page 56.
- 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 *CFG_Word_1 Format* section starting on page 56.

Notes on Clearing a Driver Fault

A Driver Fault occurs when there is an over temperature condition, a short circuit condition in the motor.

When a Driver Fault occurs, the driver sets bit 7 of STATUS_word1 in the Network Input Data. (See *STATUS_word1 Format* on page 76 for a full description of STATUS_word1.) *Once you have cleared the fault condition, you can reset the Driver Fault with the following Command Block:*


Reset Driver Fault

ESI File Name	Data Size	Function	Units	Range
CMD_Word0	UINT	<i>CMD_word0</i>		16#0400
CMD_Word1	UINT	<i>CMD_word1</i>		16#0400
Position	DINT	Unused		See Note Below
Velocity	UDINT	Unused		See Note Below
Acceleration	UINT	Unused		See Note Below
Deceleration	UINT	Unused		See Note Below
Motor_Current	UINT	Unused		See Note Below
Jerk	UINT	Unused		See Note Below

Table R6.20 Reset Driver Fault Command Block

Unused words are ignored by the SD4840EK and can be any value, including parameter values in the previous command.

Once the command block is accepted by the SD4840EK, it will respond by resetting bit 7 in STATUS_word1 of the Network Input Data.

NOTE  After this procedure, there will still be no current to the motor. This is because the Enabled_Driver Bit, (bit 15 of CMD_Word1 in the Network Output Data), must be reset when writing down the Reset Driver Fault Command Block. Setting this bit in the next command block will re-enable the motor.

Notes

MOTOR AND POWER SUPPLY SIZING

This section is intended for anyone that need performance data on compatible AMCI stepper motors or guidelines for using a foreign stepper motor.

1.1 Sizing Your Motor

Your motor choice is based on the output torque you need, the mounting space you have, and your budgetary constraints. Torque curves for all of the motors available from AMCI are presented on the following pages. Torque curves show the performance of the motors at the specified current.

There are a few things to remember when choosing your motor based on torque curves.

- 1) The torque curves in this manual are for the SD4840EK. You cannot use these curves to accurately determine the amount of torque from an AMCI motor when it is attached to a different drive. Nor can you accurately determine the amount of torque from a motor when attached to an SD4840EK if its torque curves were generated using a different drive. In general, if an output bus of the foreign drive is not the same as the voltage supplied to the SD4840EK, then the torque curves will be less accurate at higher speeds.
- 2) Make sure that the motor can provide the needed torque over the entire speed range of your application. Available torque drops as speed increases, so evaluate the motor's torque at its highest operating speed.
- 3) All of the torque curves show when the motor's windings are attached to the SD4840EK in parallel. Parallel attached motors have the advantage of more torque at high speeds when compared to series attached motors.

A simple guideline is to use the largest motor your mounting space and budgetary constraints allow. Because the I^2R losses in the motor's windings manifest themselves as heat, the maximum allowable motor temperature limits the motor's current. Using the largest motor possible may allow you to use a lower current setting on the SD4840EK drive. This lowers the I^2R losses, and the operating temperature of the motor, which increases the motor's life.

1.2 Determining Your Motor Current Setting

Your motor current setting is based on the amount of torque needed from the motor. If you decide to use a lower current setting than the value listed in the curve, be aware that a reduction in current proportionally reduces the holding torque. However, a reduction in current may not lead to a proportionally reduction in torque at high speeds, especially if the motor is series connected. At high speeds, motor torque is limited by the voltage bus of the drive and the inductance of the motor. (The simplest explanation is that the drive does not have enough time to establish the full current through the motor before it must switch the current to the winding.)

Because of this, its difficult to calculate the exact amount of high speed torque a motor will give you when you reduce its current setting. Its often easier to determine your optimum current setting by testing your machine at various current settings and then deciding which setting gives you the best performance.

1.2.1 A Note on Microstepping

Many microstepping drives control the peak current through the motor. At low speeds, this type of current control drops the available torque of a micro-stepped motor to approximately 70.7% of that available when the motor is full stepped.

However, the SD4840EK controls the RMS current through the motor. Therefore, the current supplied by the SD4840EK when microstepping is always the power equivalent of the full step current. This means that the motors' full torque is always available. At very low speeds, the SD4840EK automatically switches to peak current control to prevent motor damage.

1.3 Torque and Power Curves

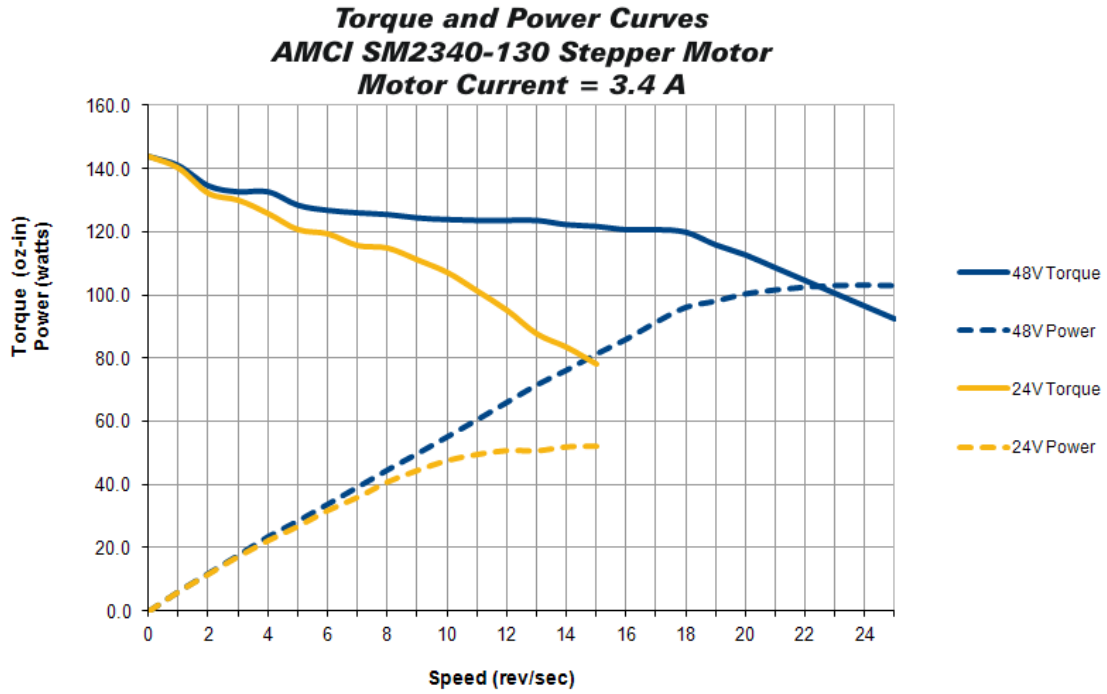


Figure T1.1 SM2340-130 Torque and Power Curves

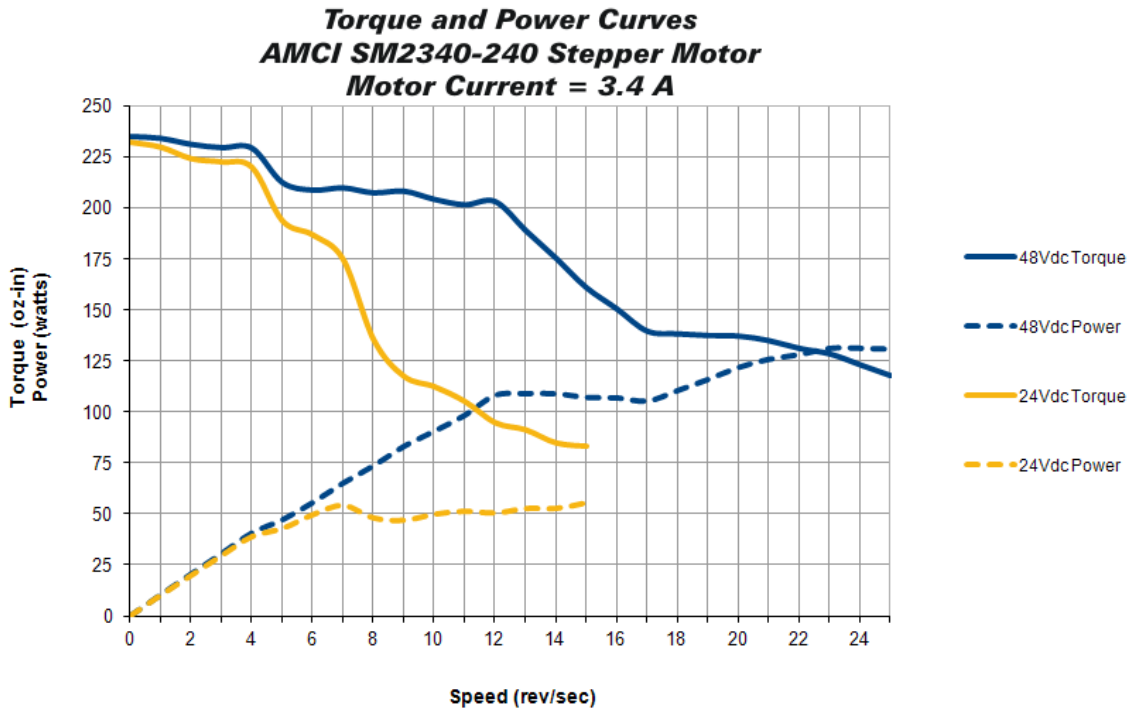


Figure T1.2 SM2340-240 Torque and Power Curves

1.3 Torque and Power Curves (continued)

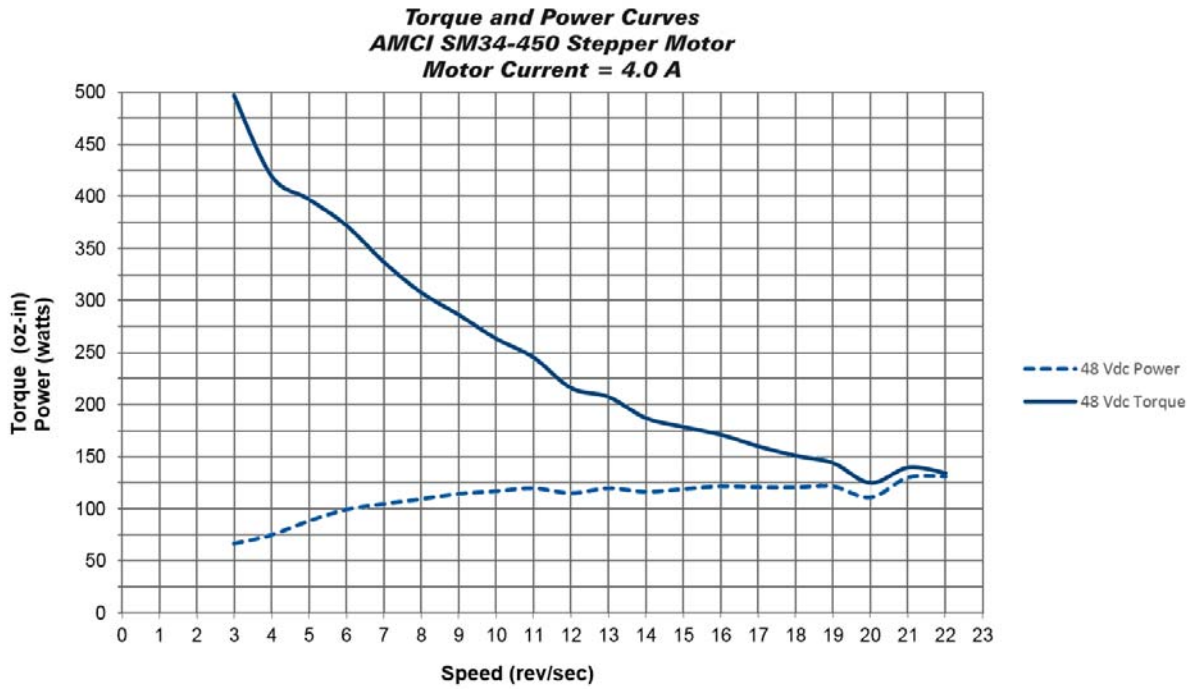


Figure T1.3 SM34-450 Torque and Power Curves

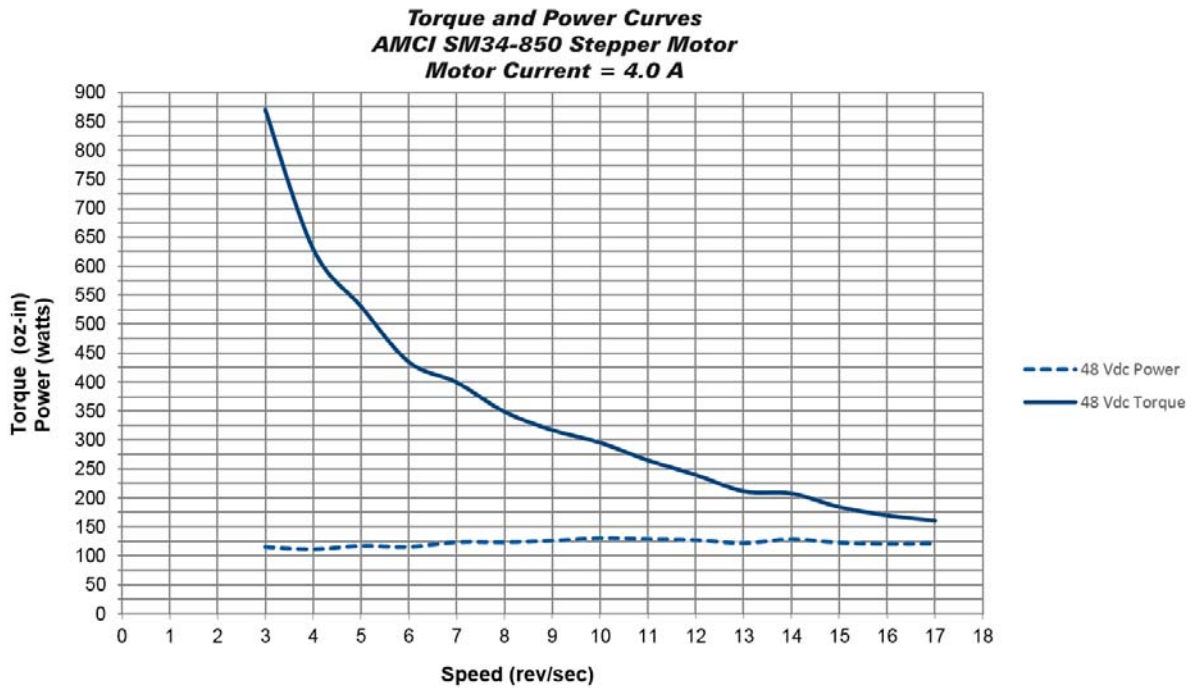


Figure T1.4 SM34-850 Torque and Power Curves

1.4 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 pages 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 decreases towards the end of its usable speed. 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 SM2340-130 will be running at a maximum of 7 RPS and a 48Vdc supply will be used. Based on the power curve in figure T1.1 on page 80, the combinations will generate a maximum of 40 Watts. Therefore a 48 Vdc supply with a power range of 60 W to 70 W can be used in the application.

Example 2: An SM2340-240 will be running at a maximum of 9 RPS and a 24 Vdc supply will be used. Based on the power curve in figure T1.2 on page 80, the power at this speed is 45 W, but the maximum power over the entire speed range is 55 W, which occurs at 7 RPS. Therefore, the 55 Watt value should be used, and the 24 Vdc supply should be able to generate 83 W to 97 W.

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

		SM2340-130			SM2340-240			SM34-450 & SM34-850		
		Motor Power	150% Supply	175% Supply	Motor Power	150% Supply	175% Supply	Motor Power	150% Supply	175% Supply
Supply Voltage	24 Vdc	53W	80 W	93W	57 W	86 W	100 W	Not Applicable		
	48 Vdc	105 W	158 W	184 W	135 W	203 W	237 W	135 W	203 W	237 W

Table T1.1 SM23 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.

All AMCI stepper motors are low inductance motors. Back EMF is typically not an issue unless there is a gearhead attached to the motor and it is driven by hand. In these instances, the motor acts as a generator. With the speed of the motor multiplied by the ratio of the gearhead, this can lead to large enough voltage spikes to damage the attached power supply.

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, then a supply that can deliver 175% of peak motor power should be used.

The second line of defense is a regeneration resistor, also known as a braking resistor. Braking resistors, and their control circuitry, are built into AMCI AC powered drives. They are not included in the SD4840EK products because of the limited ability to dissipate the heat generated by the resistor. An external braking resistor and control circuitry can be added to the system if needed.

TASK 2

INSTALLING THE SD4840EK

This chapter gives detailed information on installing the SD4840EK. The SD4840EK is designed to be DIN rail mounted. The EtherCAT connection allows the drive to be mounted near the motor, which minimized power loss in the motor cable.

2.1 Safe Handling Guidelines

2.1.1 Prevent Electrostatic Damage

⚠ CAUTION

Electrostatic discharge can damage the SD4840EK. Follow these guidelines when handling the module.

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

2.1.2 Prevent Debris From Entering the Module

⚠ WARNING

During DIN rail mounting of all devices, be sure that all debris (metal chips, wire strands, tapping liquids, etc.) is prevented from falling into the module. Debris may cause damage to the module or unintended machine operation with possible personal injury. The DIN rail for the modules should be securely installed and grounded before the modules are mounted on it.

2.1.3 Remove Power Before Servicing

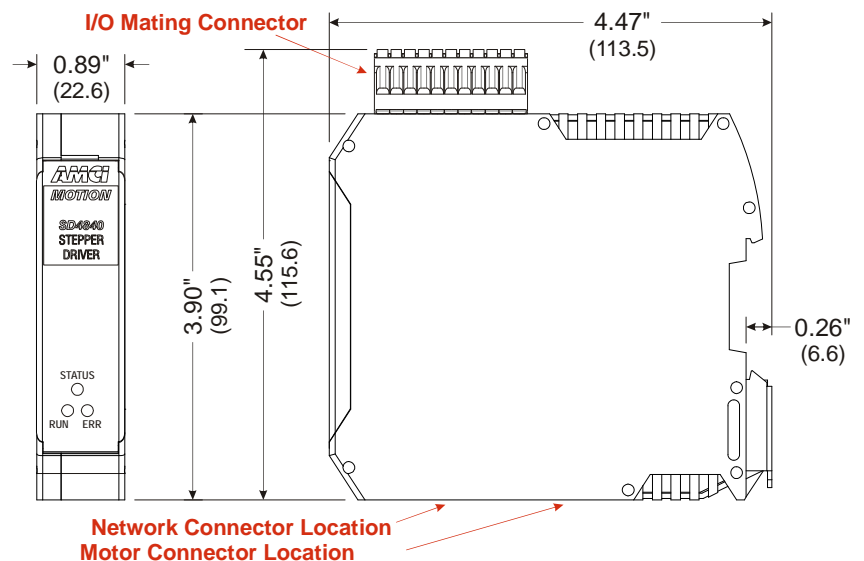
⚠ WARNING

Remove power before removing or installing any modules.

2.2 Mounting

2.2.1 Dimensions

Figure T2.1 shows the dimensions of an SD4840EK.



2.2 Mounting (continued)

2.2.2 Minimum Spacing

As shown in figure T2.2, you must maintain a minimum spacing of 2 inches (50.8 millimeters) from enclosure walls, wireways, adjacent equipment, etc. for adequate system ventilation.

Also note that the SD4840EK drives must be mounted in the orientation shown in the figure. Mounting the system in any other orientation will decrease the efficiency of the ventilation slots in the top and bottom of each module which may lead to system overheating and malfunction.

When installing multiple SD4840EK drives that are all running at the full 4.0 A motorcurrent, it is possible for the drives to overheat. It is difficult to calculate when this can occur because it is based on not only the current and duty cycle of the motor, but also such variables as enclosure size and ambient temperature. If overheating does occur, you have two choices. You can install a cooling fan beneath the drives to force additional air up through the modules or you can increase the space between the drives to limit mutual heating.

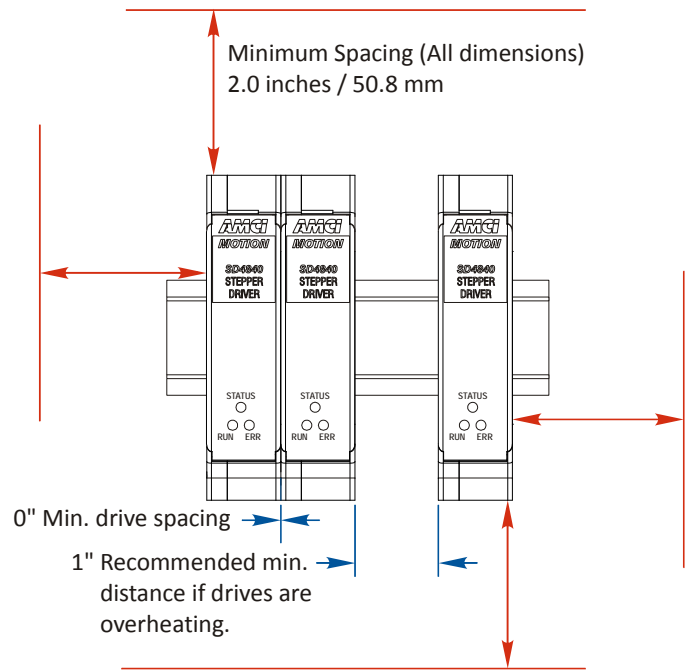


Figure T2.2 Ventilation Spacing

The SD4840EK has a bit in the network data that signals when the module is close to overheating. This bit is the Temperature_Above_90C bit, and is available in Status Word 1 of the network input data while in Command Mode. This bit is further explained in the *STATUS_word1 Format* section starting on page 76.

2.2.3 Mounting the SD4840EK Module

Mounting an SD4840EK is a very simple process thanks to the design of the enclosure.

- 1) Engage the top of the clip on the back of the enclosure with the top of the DIN rail
- 2) Rotate the module down until the metal bracket snaps on to the DIN Rail.

Once all of your drives are installed, it is strongly suggested to use the end caps from Phoenix Contact with the part number of 271 37 80 to secure the modules on the DIN Rail. These end caps prevent the drives from sliding along the DIN rail if it is subjected to shock or vibration during machine operation.

2.3 I/O Connector Pin Out

The I/O Connector is located on the top of the module. The mate for this connector is included with the SD4840EK. It is available from AMCI under the part number MS-2x11 and is also available from Phoenix Contact under their part number 173 88 98. Figure T2.3 shows the pin out for the I/O connector.

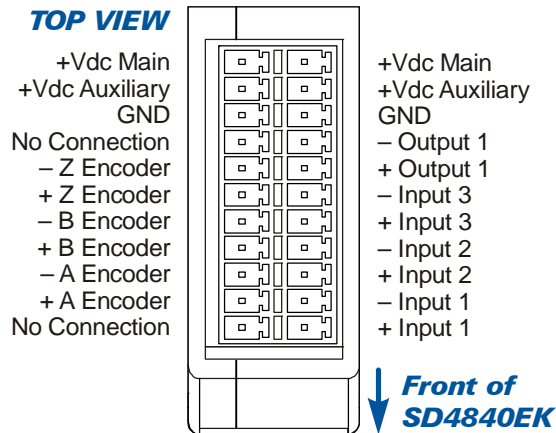


Figure T2.3 I/O Connector

2.4 Power Wiring

The SD4840EK accepts 24 to 48Vdc for its input power.

- ▶ The +Vdc Main pin powers the entire module. This includes the control electronics and the motor drive circuitry. AMCI strongly suggests using 18 AWG or larger wire for the power connections. The MS-2x11 connector will accept up to 16 gauge wire.
- ▶ The +Vdc Auxiliary pin powers the control electronics only. Using this pin is optional. By applying power to this pin, you can remove power from +Vdc Main to remove power from the motor only. The control electronics will remain powered and the drive will not lose its network connection. This feature was incorporated into the design of the SD4840EK for customers that must remove power from the motor due to safety considerations. Maximum current requirements on the +Vdc Auxiliary pin is 70 mA at 24Vdc.

The +Vdc Main, +Vdc Auxiliary, and GND connections each have two pins on the MS-2x11 I/O connector that are internally connected together. You can wire to either or both pins for each connection.

CAUTION

The I/O connector is rated for a maximum current of 8 amps per pin. It is possible to daisy chain the +Vdc Auxiliary connections, but AMCI strongly suggests avoiding this practice for the +Vdc Main connections. It is strongly suggested to run separate wires from your power supply to each drive for the +Vdc Main connections.

CAUTION

Do not apply 120 Vac to any pins of the SD4840EK. If this occurs, the unit will be damaged and you will void the unit's warranty.

2.4 Power Wiring (continued)

! WARNING

The SD4840EK units do not have a circuit to limit inrush current when power is applied. If the power supply voltage is applied through the switching of contacts, damage to the contacts or contact welding may occur.

- ▶ Use a power supply that limits the peak output current to a value below the contact switching limit.
- ▶ The best practice is to remove switching contact from the power supply to the SD4840EK entirely. Remove power from the SD4840EK by switching off its power supply.

! WARNING

Remove power from the SD4840EK before servicing. Do not connect or disconnect the MS-2x11 connector while power is applied.

Figure T1.4 below shows how to wire power to the SD4840EK units. Note that +Vdc Auxiliary is only used when you introduce a circuit for removing power from the motor.

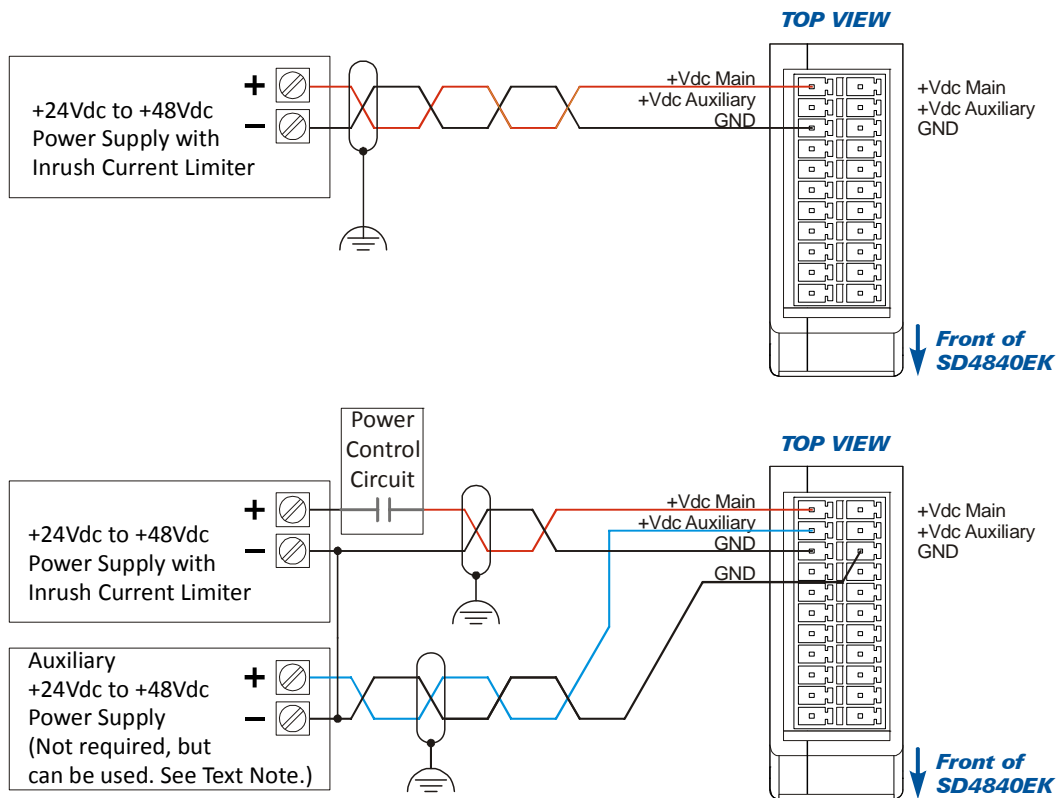


Figure T2.4 SD4840EK Power Wiring

NOTE

Twisted pair cable with an overall shield should be used for all power supply wiring to limit EMI. The shield must be connected to the earth ground point used by the power supply. These cables must be routed away from all of the module's I/O to further limit the possibility of injecting noise into the I/O signals.

NOTE

As shown in the figure above, it is possible, *but not required*, to use a second power supply when powering the +Vdc Auxiliary power input. The +Vdc Auxiliary pin can also be powered by the +Vdc Main supply. If a separate supply is used, it must have a nominal output voltage of 24 to 48 Vdc. The output voltage does not have to match the voltage applied to the +Vdc Main power input. The commons of the two supplies must be electrically tied together.

2.5 Input Wiring

Figure T2.5 below shows how to wire discrete DC sourcing or sinking sensors to Inputs 1, 2, and 3 of the SD4840EK.

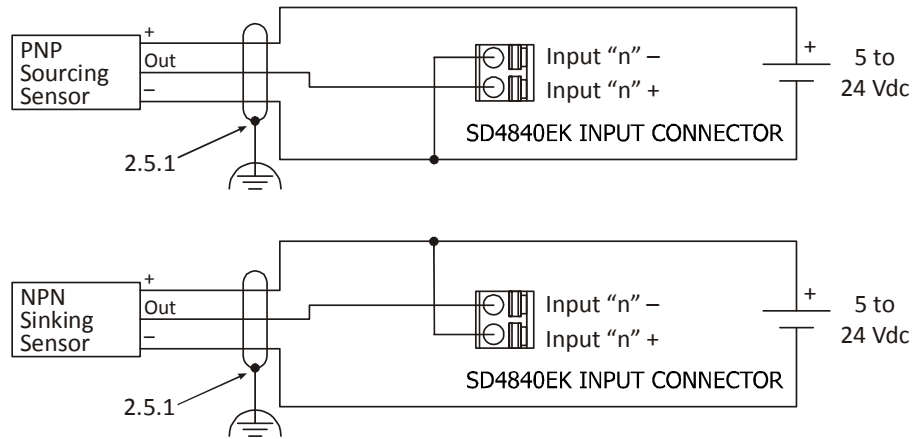


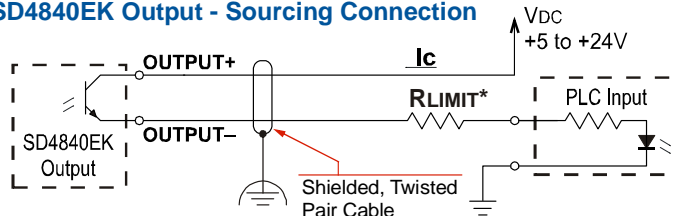
Figure T2.5 Input Wiring

2.5.1 Cable Shields

Because they are low power signals, cabling from the sensor to the SD4840EK 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 SD4840EK.

2.6 Output Wiring

SD4840EK Output - Sourcing Connection

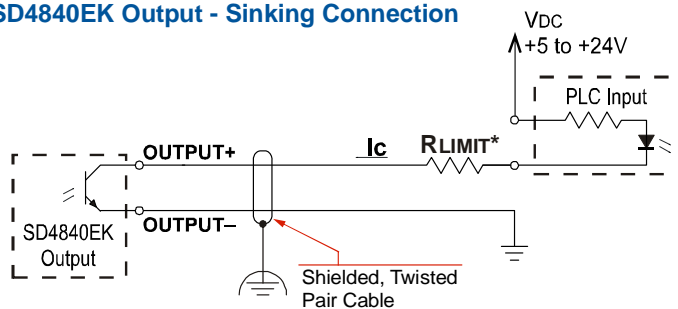


The SD4840EK output is an optically isolated transistor that is capable of driving a typical PLC input. Both ends are uncommitted, so it can be wired as a sinking or sourcing output.

Electrical Specifications

VDC max: 30Vdc	VCE SAT: 1Vdc @ 20 mA
Ic max: 20 mA	Power Dissipation: 20 mW max.

SD4840EK Output - Sinking Connection



RLIMIT

A resistor may be needed to limit the current through the output. The value, and power rating of the resistor is dependent on the value of Vdc, the voltage drop across the input, and the current requirements of the input.

Figure T2.6 Output Wiring

2.7 Encoder Wiring

2.7.1 Differential Wiring

The figure below shows how to wire a 5 Vdc differential encoder to the SD4840EK. There is no standard when it comes to the color code of the encoder’s wires. A document named ‘encoder specs’, is available on the AMCI website (www.amci.com) that lists the color codes of encoders used by AMCI. The direct link to the document is: <https://www.amci.com/files/6714/5495/8474/encoder-specifications-rev.pdf>

Because they are low power signals, cabling from the encoder to the SD4840EK 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 encoder end. If this is not practical, the shield should be grounded to the same ground bus as the SD4840EK.

As shown in the following diagram, the No Connection pins on the SD4840EK can be used to make the connections between the encoder and its external power supply. This may eliminate the need for an external terminal block when wiring the encoder to the unit.

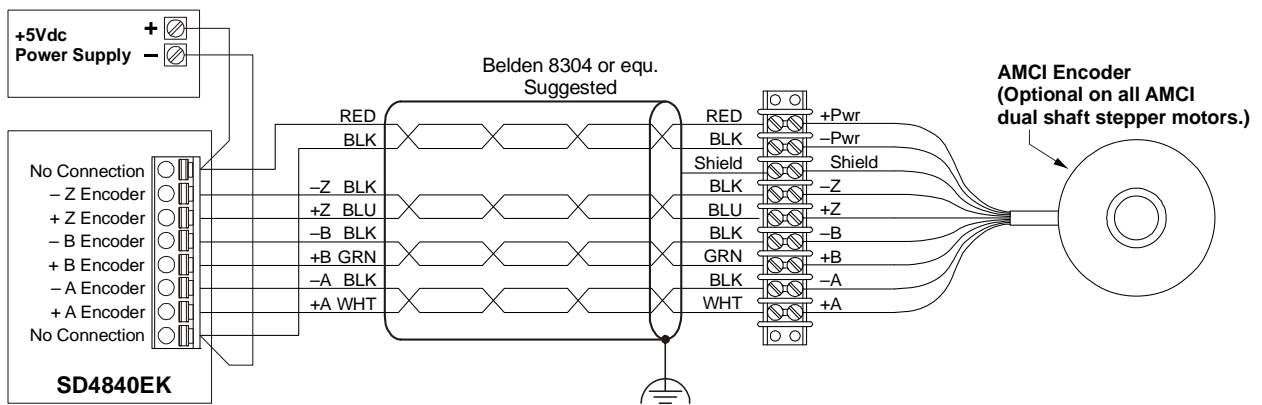


Figure T2.7 Sample Differential Encoder Wiring

2.7 Encoder Wiring (continued)

2.7.2 Single Ended Wiring

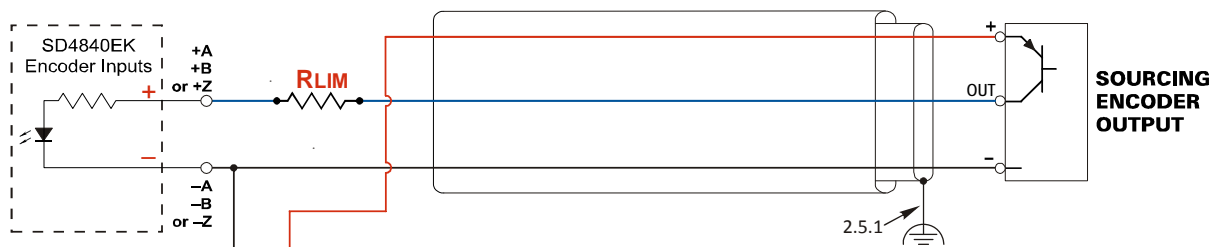
Figure T2.8 below shows how to wire the encoder inputs to both a single ended sourcing and single ended sinking encoder outputs.



The encoder inputs on the SD4840EK are rated for 5 Vdc only. You must use a current limiting resistor on each input if the outputs of your encoder are greater than 5 Vdc. Appropriate current limiting resistors are shown in the figure below.

Because they are low power signals, cabling from the encoder to the SD4840EK 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 encoder end. If this is not practical, the shield should be grounded to the same ground bus as the SD4840EK.

SD4840EK Encoder Input Connection to Sourcing Encoder Output



RLIM: The inputs are designed to accept 5Vdc but can use any voltage up to 24Vdc with the appropriate current limiting resistor. See the table above for the required resistor based on the supply voltage.

Supply	RLIM
5 Vdc	None
12 Vdc	2.0 Kohm
15 Vdc	2.4 Kohm
24 Vdc	4.7 Kohm

SD4840EK Encoder Input Connection to Sinking Encoder Output

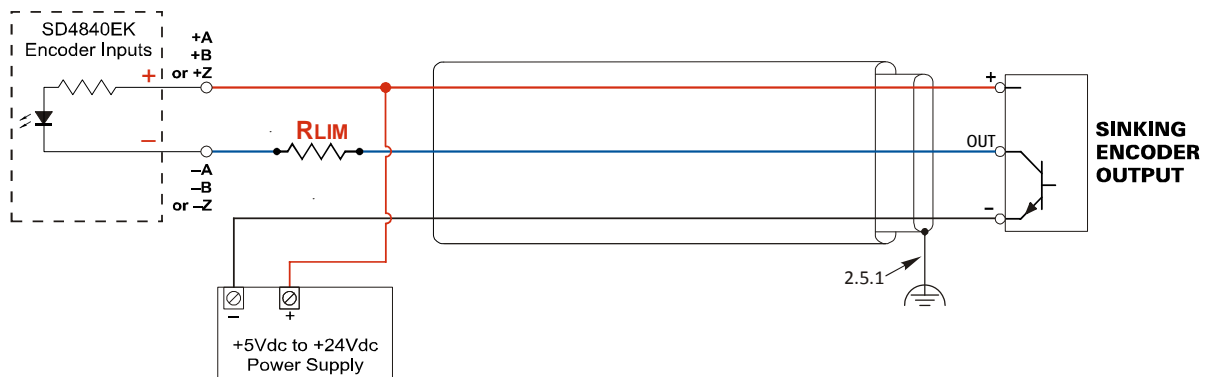


Figure T2.8 Single Ended Encoder Wiring

2.8 Installing the Stepper Motor

2.8.1 Outline Drawings

Outline drawings for all of our motors can be found on our website, www.amci.com, in the *Support -> User Manuals* section. A direct link to the section is: <https://www.amci.com/industrial-automation-support/user-manuals>. The drawings are available as Adobe Acrobat pdf files. A document that is simply called *wiring* lists all of the wiring color codes for all AMCI motors. If you do not have internet access contact AMCI and we will fax the information to you.

2.8 Installing the Stepper Motor (continued)

2.8.2 Mounting the Motor

All AMCI motor 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 motor'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 motor.

Motors should be mounted using the heaviest hardware possible. AMCI 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.

2.8.3 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.

2.8.4 Extending the Motor Cable

Even though it is possible to extend the cable length an additional forty feet, AMCI recommends installing the SD4840EK as close to the motor as possible. This will decrease the chances of forming a ground loop, and has the added benefit of limiting the amount of power loss in the motor cable. If you must extend the cable, you should use a cable with twisted pairs 18 AWG or larger and an overall shield. The motor connector will accept up to 12 AWG wire. The exact gauge that you use will depend on the length of the run and expected temperature rise in your application. Belden 9552 is a suggested 18 AWG cable.

2.8.5 Installing the Motor Cable

NOTE 

- 1) All of the motor connections are high power, high voltage signals. Cable from the motor can be installed in conduit along with ac/dc power lines or high power ac/dc I/O. It cannot be installed in conduit with low power cabling such as I/O cabling or Ethernet cabling attached to the SD4840EK.
- 2) If you decide to extend the motor cable, treat the shield as a signal carrying conductor when installing the motor cable. Do not connect the shield to earth ground at any junction box.

2.9 Connecting the Motor

2.9.1 Motor Connector

The motor connector is included with the SD4840EK. Spares are available from AMCI under the part number MS-4M as well as directly from Phoenix Contact under their part number 187 80 37. Motor connections should be tight, as loose connections may lead to arcing which will heat the connector. Phoenix Contact specifies a tightening torque of 4.4 to 5.4 lb-in (0.5 to 0.6 Nm)

NOTE ⚡ When powered, the motor connector may represent a shock hazard because the full DC input voltage may be present on its terminals.

WARNING ⚠ Always remove power from the SD4840EK before connecting or disconnecting the motor.

NOTE ⚡

- 1) Never connect the motor leads to ground or to a power supply.
- 2) Always connect the cable shield from your motor's cable to Earth Ground. It is best to connect the cable shields to the ground bus of the system. Do not connect the shields to the DIN rail. If you connect the motor shields to the DIN Rail and the grounding connection from the DIN rail to the Grounding Bus fails over time, then you will eventually have a condition where electrical noise is injected into the SD4840EK, which may result in future system errors.

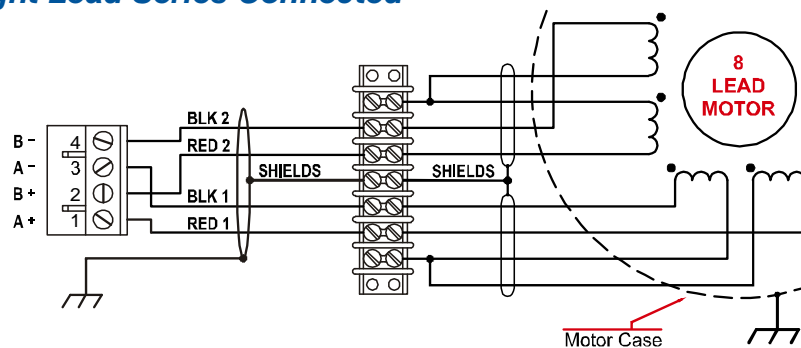
2.9.2 Motor Wiring

The SD4840 will work with many different motors, including those not sold by AMCI. This section assumes that you have already chosen your motor and you are looking for wiring information. No wire colors are given in the figures below because there is no single industry wide color coding scheme for stepper motors. You must refer to your motor data sheets for this information.

A wiring document for all of the motors ever sold by AMCI is available on our website. This single document contains all of the information necessary to connect any AMCI motor to any AMCI driver. It can be found in the *Support -> User Manuals* section. A direct link to the section is: <https://www.amci.com/industrial-automation-support/user-manuals>. The document is simply called "wiring".

Figure T2.9, which is continued on the following page, shows how to wire a motor to the SD4840EK in series, parallel, or center-tap configurations.

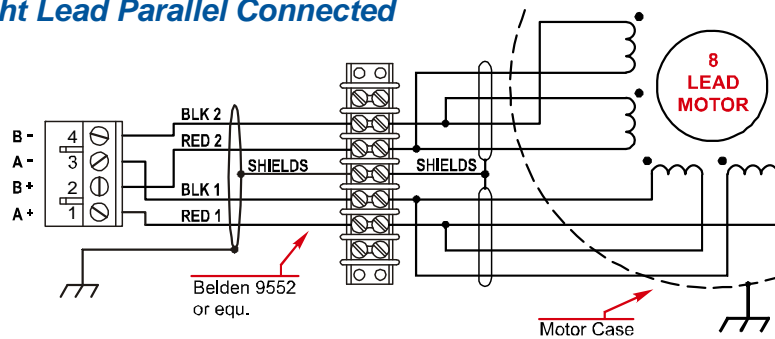
Eight Lead Series Connected



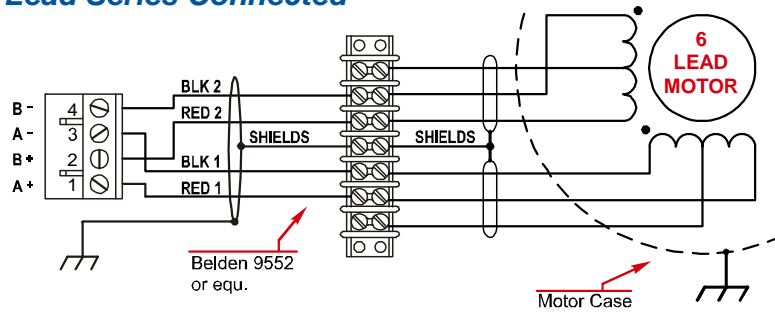
2.9 Connecting the Motor (continued)

2.9.2 Motor Wiring (continued)

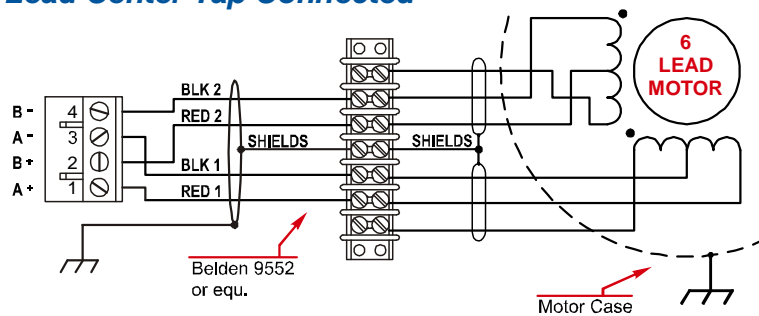
Eight Lead Parallel Connected



Six Lead Series Connected



Six Lead Center Tap Connected



Four Lead Connected

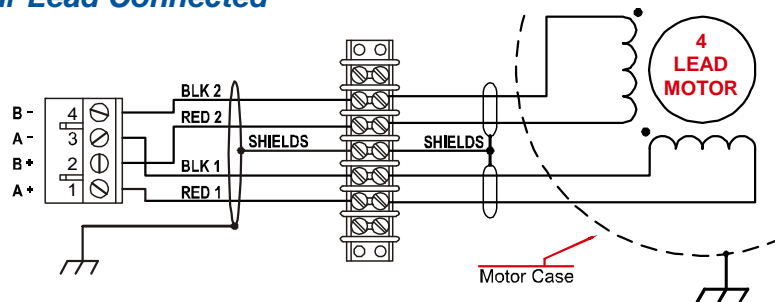
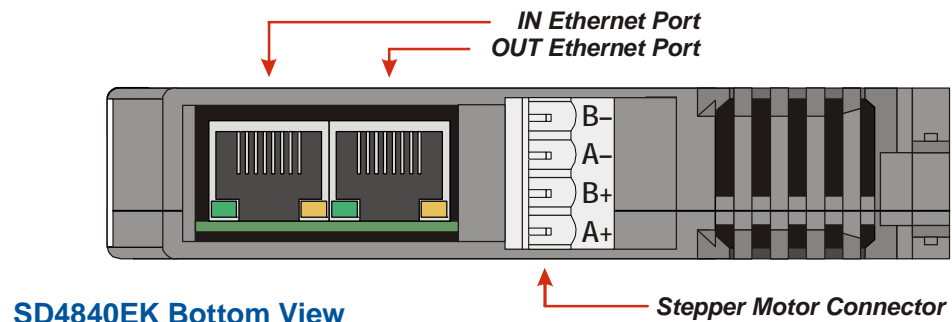


Figure T2.9 Motor Wiring

2.10 Ethernet Connections

The Ethernet connectors are located on the bottom of the unit. The connectors are standard RJ-45 jacks that will accept any standard 100baseT cable. Because the port can run at 100 Mbit speeds, Category 5, 5e, or 6 cable should be used.

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 to connect the SD4840EK to any device, including a personal computer.



SD4840EK Bottom View

Figure R6.6 Ethernet Port Locations

The Network Status LED's are fully described on page 18 in the *Status LED* section of the Specifications reference.

Notes

TASK 3

ETHERCAT SYSTEM CONFIGURATION

This chapter outlines how to add an SD4840EK unit to an EtherCAT system. The TwinCAT version 3 software is used as an example.

3.1 Install the ESI file

3.1.1 Obtain the ESI file

All AMCI ESI files are located on our website at the following address:

➤ <http://www.amci.com/industrial-automation-support/configuration-files/>

Simply download the ZIP file and extract it.

3.1.2 Install the ESI file

Once extracted, the xml file must be copied or moved to the appropriate system directory. For version 3 of TwinCAT, the default directory is:

➤ C:\TwinCAT3.1\Config\Io\EtherCAT\

3.1.3 Restart the Programming System If Needed

If the TwinCAT program was running when the ESI file was copied to the appropriate system directory, you may have to restart the TwinCAT program before it will recognize the new ESI file.

3.2 Add the SD4840EK to the Project



NOTE This section assumes that the TwinCAT software is in Config Mode.

3.2.1 Scan for the SD4840EK

- 1) Attach the SD4840EK to the network and power up the device.
- 2) At this point, the device is in its INIT state. The status LED on the front of the device will be on. Both of the EtherCAT status LED's on the front of the device will be off.
- 3) Right click on the EtherCAT adapter that the SD4840EK is attached to. In the drop down menu that opens, select the "Scan" option. (If the "Scan" option is not available, the TwinCAT software is not in Config Mode.)
- 4) The SD4840EK will appear in the device tree and the name will typically begin with "Box".

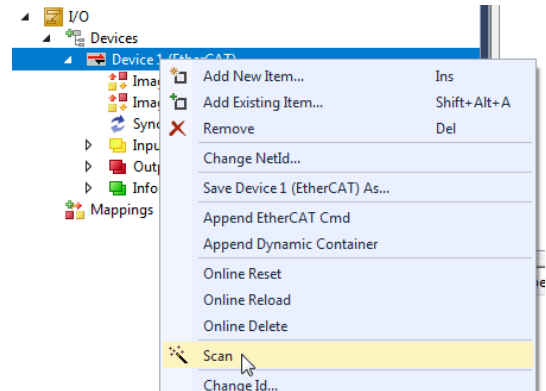


Figure T3.1 Scan for the SD4840EK

3.2.2 Rename the Device

- 1) Click on the unit in the device tree.
- 2) If needed, click on the "General" tab in the window that opens.
- 3) The device name for the SD4840EK can be changed in the *Name:* field.

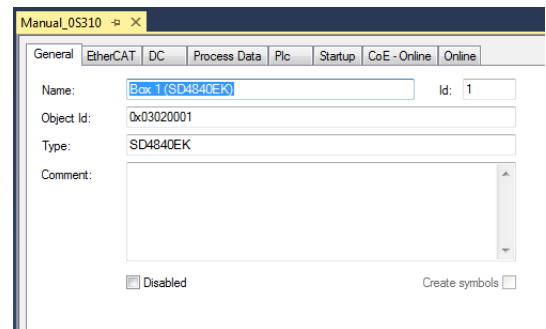


Figure T3.2 Rename the SD4840EK

3.2 Add the SD4840EK to the Project (continued)

3.2.3 Configure the SD4840EK

- 1) Click on the “CoE - Online” tab.
- 2) Click on the [+] button next to the 6000:0 Index field (Inputs) to expand it. Note the value of the STATUS_word1 register (6000:01). This value is typically 0x4408.
- 3) Click on the [+] button next to the 8010:0 Index field to expand it.
- 4) The configuration data is available under the nine sub-index fields of the 8010:0 Index. Configuration data can be temporarily changed here. Double click on any field to open the Set Value Dialog screen to set the parameter to the desired value. Table R5.3, Configuration Word 0 Bits found on page 55, allows you to calculate the proper hexadecimal value of CFG_word0 for your application.
- 5) Once you set a parameter, check the value of the STATUS_word1 register (6000:01). If the value has bit 13 set to “1”, there is an error in your configuration setting. (An example is a STATUS_word1 register change from 0x4408 to 0x6408.)
- 6) To make these changes permanent, you must define values under the “Startup” tab. Click on the “Startup” tab.
- 7) Right click on the blank surface and select “Add New Item...” in the resulting pop up menu.
- 8) Click on the [+] button next to the 8010:0 Index field to expand it.
- 9) Double click on the field that must be changed. This opens a dialog that will allow you to set the field with a decimal or hexadecimal value.
- 10) Click the [OK] button in the dialog box to set the new value.
- 11) Click the [OK] button in the “Edit CANopen Startup Entry” screen to accept the new startup parameter setting.

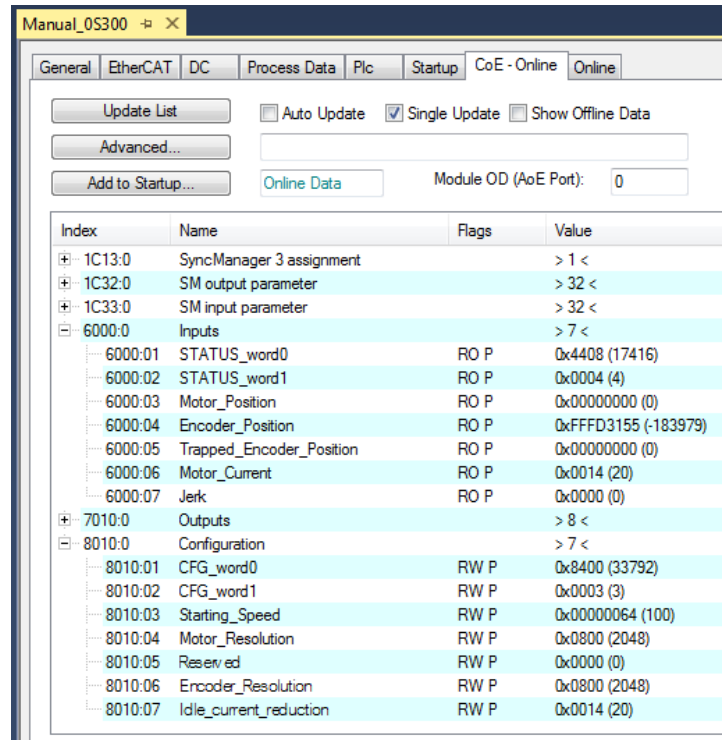


Figure T3.3 SD4840EK Configuration Registers

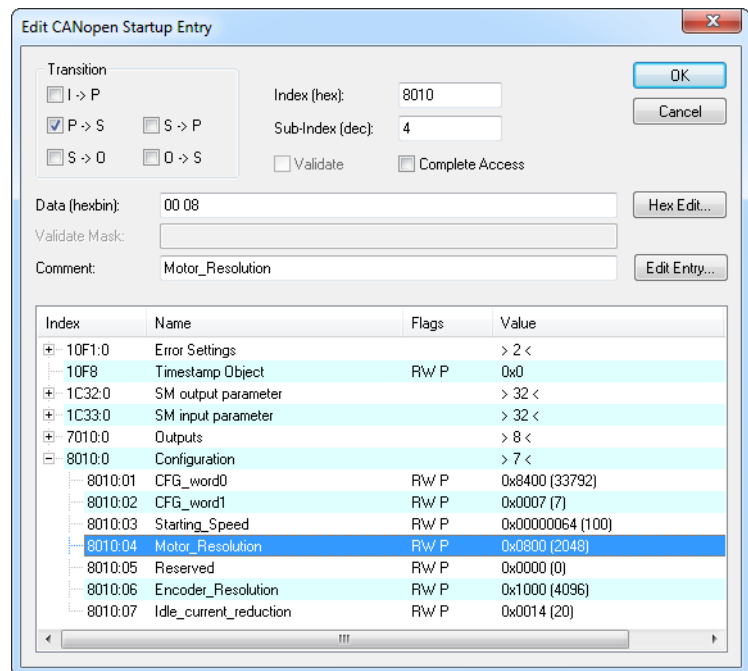


Figure T3.4 SD4840EK Configuration Registers

3.3 Create a DUT

AMCI sample programs use a DUT (Defined User Type) to group the I/O variables associated with each SD4840EK. Once the DUT is defined, a separate variable is created for each device on the machine.

- 1) If needed, create a PLC for the project.
- 2) Expand out the PLC tree until the DUT field is available.
- 3) Right click on DUT and select Add → DUT.
- 4) In the window that opens, name the DUT and give it a Structure data type.
- 5) The following is the variable layout used in the sample programs. This layout can be used as is, or can be modified to fit your specific application.

```

TYPE DUT_AMCI_SD4840EK :
STRUCT
  //Outputs:
  nCMD0 AT %Q*      : UINT;
  nCMD1 AT %Q*      : UINT := 16#8000;
  nTargetPos AT %Q* : DINT;
  nTargetVel AT %Q* : UDINT;
  nAccel AT %Q*     : UINT;
  nDecel AT %Q*     : UINT;
  nMtrCurr AT %Q*   : UINT;
  nJerk AT %Q*      : UINT := 0;
  //Inputs:
  nStatus0 AT %I*   : UINT;
  nStatus1 AT %I*   : UINT;
  nMotorPos AT %I*  : DINT;
  nEncPos AT %I*    : DINT;
  nTrappedEnc AT %I* : DINT;
  nLastCurrent AT %I* : UINT;
  nLastJerk AT %I*  : UINT;
END_STRUCT
END_TYPE

```

3.4 Create Variables Based on the DUT

The location of the actual variable(s) used in the system's program depends on programming style and program complexity. Typically, variables are declared in the Main program or in a Global Variable List.

```

st_Axis1_SD4840EK : DUT_AMCI_SD4840EK;
st_Axis2_SD4840EK : DUT_AMCI_SD4840EK;

```

3.5 Link Variable Names to I/O Words

- 1) From the menu, select *Build* → *Build Solution*. (Ctrl+Shift+B). The build will fail with a message that at least one variable must be linked to a task variable. Click [OK] to close the message.
- 2) In the I/O tree, expand the SD4840EK until the input and output words are visible.
- 3) Right click on the STATUS_word0 input word. In the pop up menu that opens, select "Change Link..."

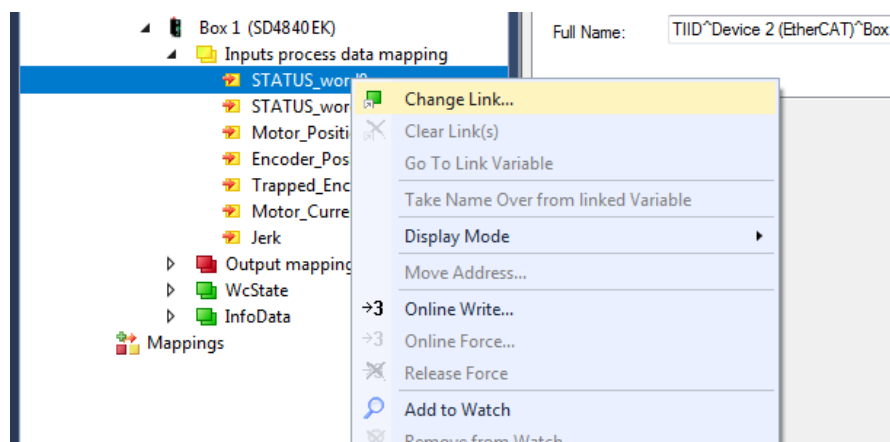


Figure T3.5 Change Link Pop up Menu

3.5 Link Variable Names to I/O Words (continued)

- 4) In the *Attach Variable* window that opens, select the variable to link to the STATUS_word0 input word and click [OK].

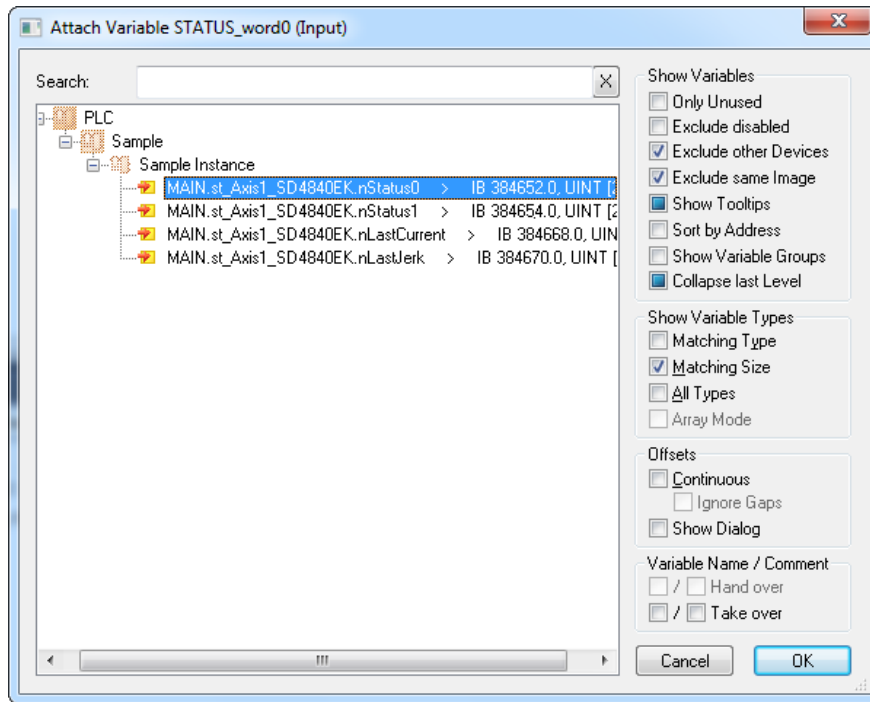


Figure T3.6 Change Link Pop up Menu

- 5) Repeat steps three and four for the remaining input and output words of the SD4840EK.

TASK 4 (OPTIONAL)

DISTRIBUTED CLOCK - SYNC0 SETUP

This chapter outlines the steps need to configure the SD4840EK to synchronize to the Sync0 signal from the Distributed Clock (DC) feature instead of the SyncManager 2 event. This configuration is optional.

By default, the SD4840EK acts on the data from the EtherCAT master as soon as it is transmitted to the device. This mode of operation is called “SM-Synchron”. The data transfer is synchronized with the SyncManager 2 event. This synchronization is more than adequate for most machine types.

On very high speed machines, or large machines that require more than one EtherCAT transfer to update all of the I/O, it may be necessary to use the EtherCAT Distributed Clock mechanism to synchronize the start of moves with other motion axes or other devices on the EtherCAT network.

Using the Distributed Clock feature adds complexity to the system that may not be needed. On large machines that require more than one EtherCAT transfer to update all of the I/O, configure the network to update all of the axes with one transfer. This may eliminate the need to use the DC functionality on large machines.

4.1 Verify Main PLC Task Timing

- 1) In the *Solutions Explorer*, expand out *System* → *Tasks* so that the tasks are visible. Double click on the task assigned to the main PLC. This task is typically named *PlcTask*. In the windows that open, note the value for Cycle Ticks and its time in milliseconds. These values are typically 10 ticks and 10.000 milliseconds.
 - ▶ You can change the timing of the main task here, but this will affect the timing of every program that runs under this task.

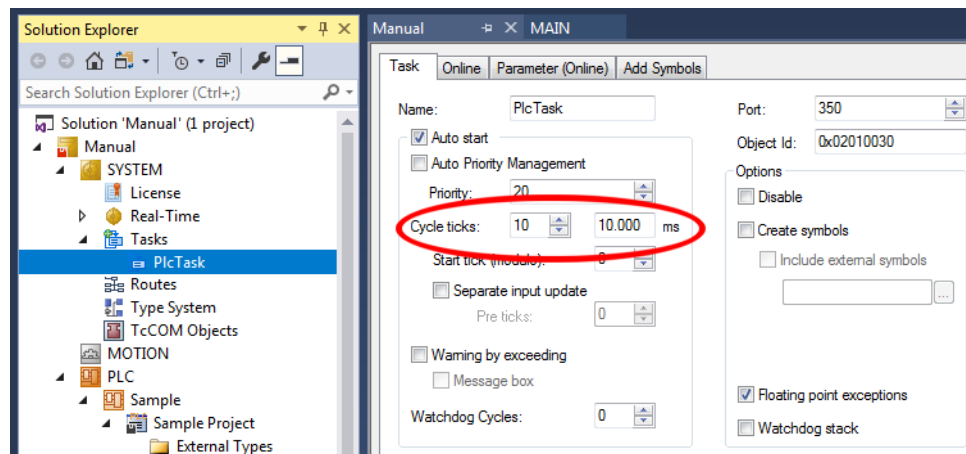


Table T4.1 Main PLC Task Timing

- 2) If this value is between 2 and 50 milliseconds and the correct update time for the SD4840EK, skip to step 4.3. When setting the *SYNC 0* → *Sync Unit Cycle* multiplier in step 6 of 4.3, use a value of 1.
- 3) If the timing in step 1 above is not correct for the SD4840EK, but the correct time is an integer multiple of this timing, then skip to step 4.3. When setting the *SYNC 0* → *Sync Unit Cycle* multiplier in step 6 of 4.3, use the correct integer multiplier.
- 4) If the timing in step 1 above is not correct for the SD4840EK, and the correct time is *not* an integer multiple of this timing, then proceed with step 4.2.

4.2 Create New PLC Task

You only have to follow this step if the Main PLC task timing verified in step 4.1 above is not the correct update time for the SD4840EK and the correct time is not an integer multiple of the Main PLC task timing.

- 1) In the *Solutions Explorer*, expand out *System* → *Tasks* so that the tasks are visible. Right click on *Tasks* and select “Add New Item...”.
- 2) In the window that opens, name the new task and select “TwinCAT Task With Image” as the *Type*. Click the [OK] button to accept the values.

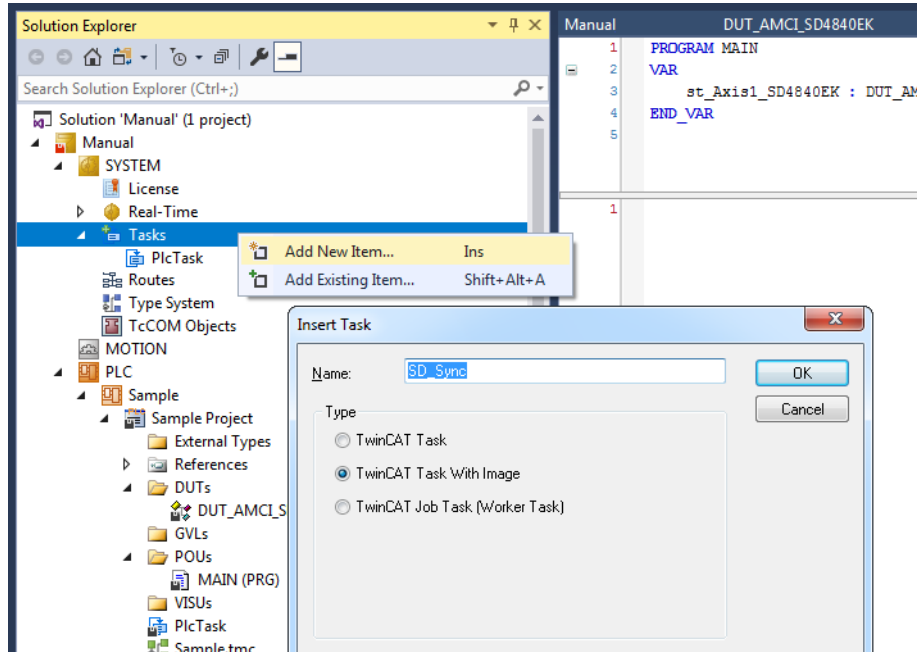


Figure T4.1 Create New Task

- 3) In the window that open, enter the desired update time in Cycle Ticks. By default, each Cycle Tick is 1 millisecond.
- 4) If needed, expand out the new task so that the Outputs icon is visible.
- 5) Right click on the Outputs icon and select “Add New Item...”. The *Insert Variable* window will open.
- 6) Name the new variable. Under >*Data Type*, select “UINT”. Click the [OK] button to close the *Insert Variable* window.

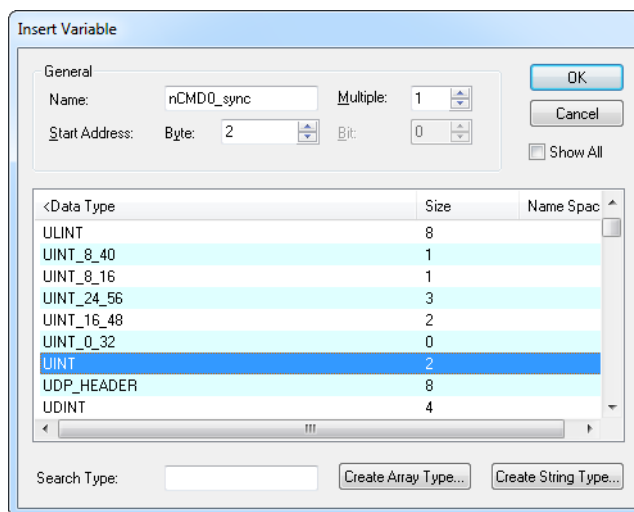


Figure T4.2 Setting Task Variable Name and Size

4.2 Create New PLC Task (continued)

- 7) In the *Solutions Explorer*, click on the name of the new variable to select it. The information on the variable will appear in a pane. Click on the [Linked to...] button in this pane to open the *Attach Variable* window.
- 8) Double click on the `CMD_word0` variable of the correct SD4840EK. This will link the `CMD_word0` of the unit to the variable created for the task. The SD4840EK will update at the time specified by the new task.

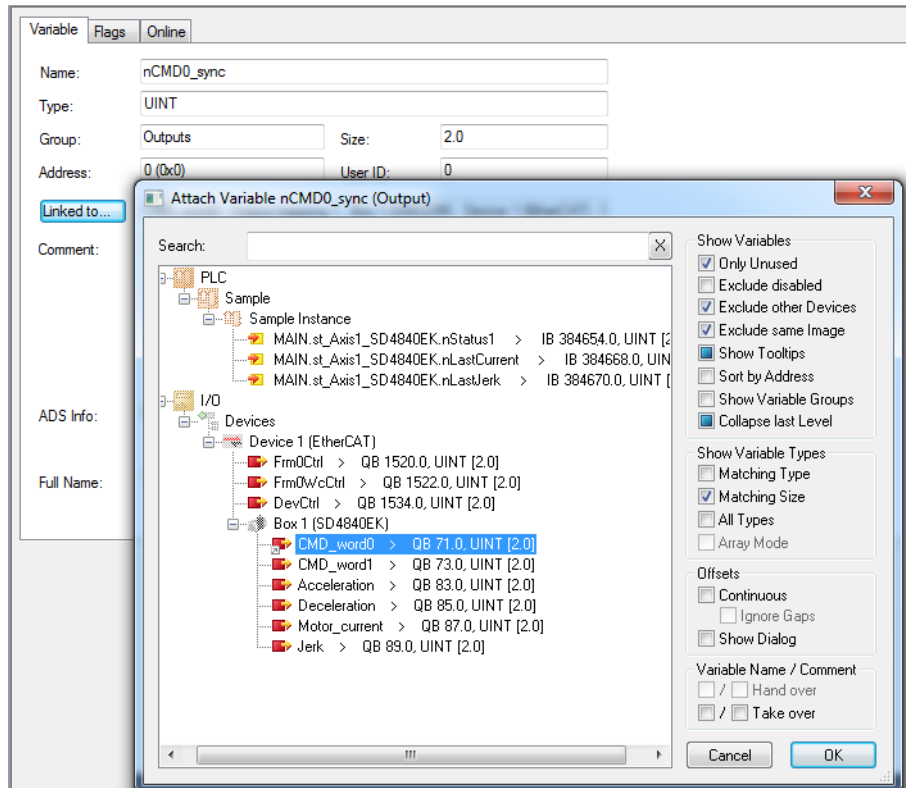


Figure T4.3 Linking Task Variable to the SD4840EK

4.3 Set Operational Mode

- 1) If necessary, expand the I/O Device tree until the SD4840EK is visible.
- 2) Double click on the SD4840EK to open the setting window for the device.
- 3) Click on the “DC” tab.
- 4) Under the Operation Mode setting, click on the drop down menu and select “DC_Synchron”.
- 5) Click on the [Advanced Settings...] button.
- 6) Verify or set the following:
 - *Cyclic Mode* -> *Enable* checkbox is checked.
 - *Sync Unit Cycle* (μs) is equal to the time of the correct task. If not, the variables are not linked correctly.
 - *SYNC 0* -> *Sync Unit Cycle* radio button is enabled. In the drop down menu, set the multiplier as needed so that the *Sync Unit Cycle* (μs) time multiplied by the multiplier value is between 2 and 50 milliseconds.
 - *Enable SYNC 0* checkbox is checked.
 - *Enable SYNC 1* checkbox is not checked.
- 7) Click [OK] to close the window.

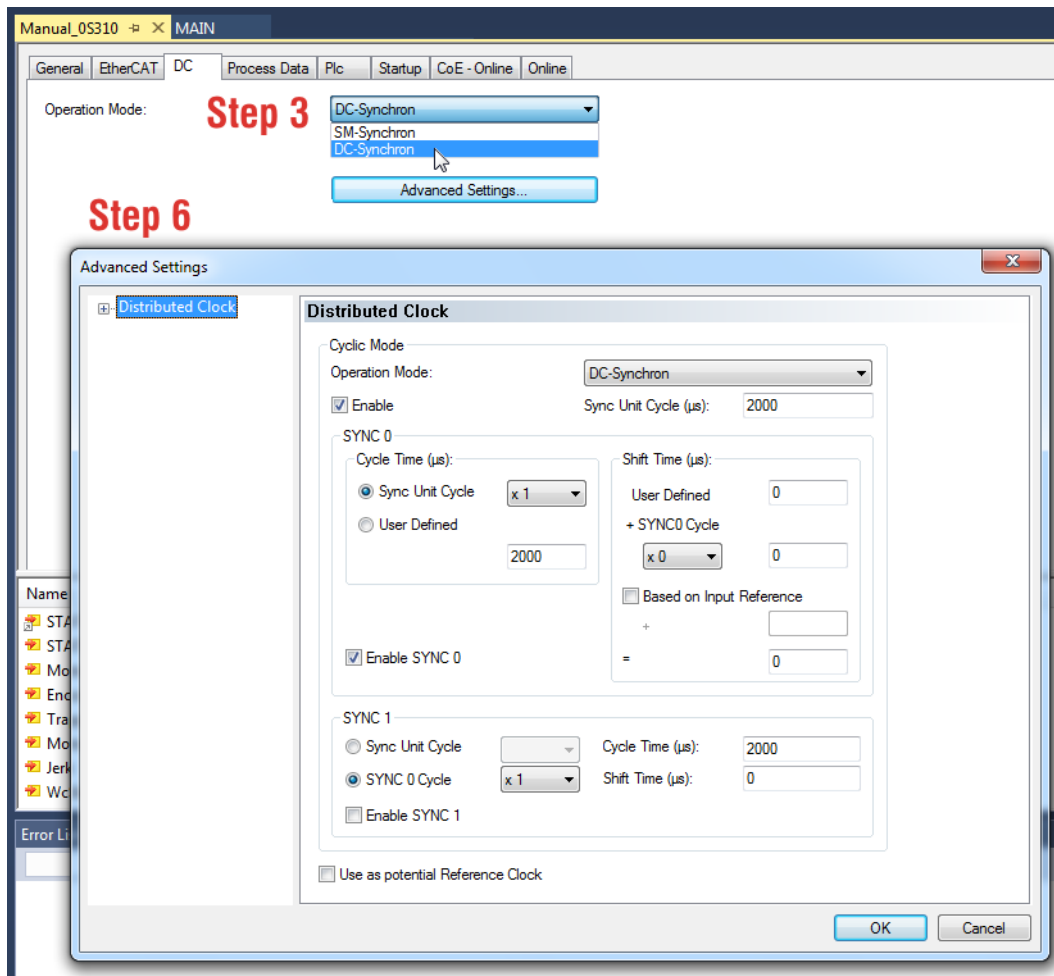


Figure T4.4 Distributed Clock Settings

4.4 Set CFG_word1 Value to Enable SYNC0

Continuing from step 4.3...

- 1) Click on the “CoE - Online” tab in the settings window.
- 2) Expand the 8010:0 tree to see the value of the 8010:02 register. (CFG_word1 value)
- 3) This value must have bit 6 set to enable the synchronous mode in the SD4840EK. If this value is greater than or equal to 64, then the bit is set and you are finished with this step of the procedure. If the value is less than 64, then the bit is not set. This new value must be specified in the Startup tab in order to make the change permanent.
- 4) Click on the “Startup” tab.
- 5) Right click on the blank surface and select “Add New Item...” in the resulting pop up menu.
- 6) Click on the [+] button next to the 8010:0 Index field to expand it.
- 7) Double click on the 8010:02 field. This opens a dialog that will allow you to set the field with a decimal or hexadecimal value. If setting in decimal, the new value is the present value from step 3 + 64.
- 8) Click the [OK] button in the dialog box to set the new value.
- 9) Click the [OK] button in the Edit CANopen Startup Entry screen to accept the new startup parameter setting.

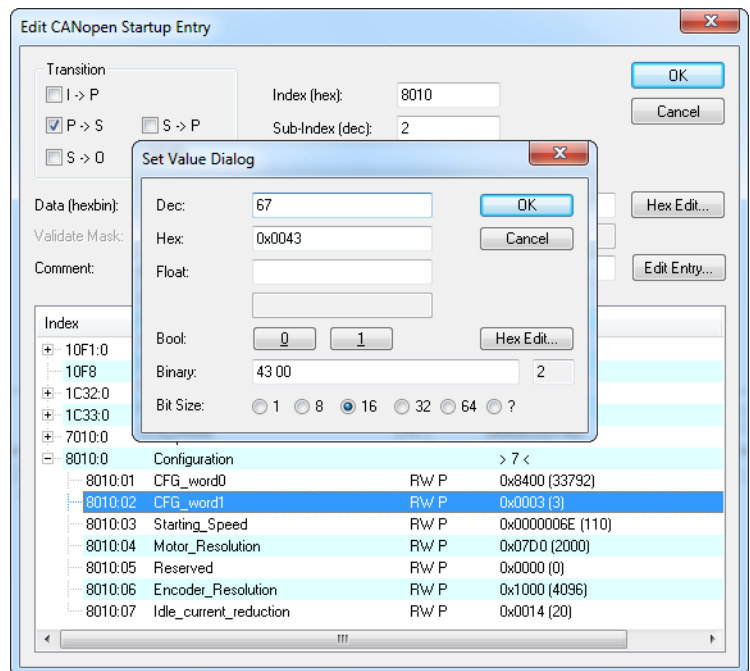


Figure T4.5 Set DC_Sync Bit in Configuration Word 1



ADVANCED MICRO CONTROLS INC.

20 GEAR DRIVE, TERRYVILLE, CT 06786 T: (860) 585-1254 F: (860) 584-1973

www.amci.com

LEADERS IN ADVANCED CONTROL PRODUCTS