Applied Motion Products CANopen Manual

Applied Motion Products

APPLIED MOTION PRODUCTS, INC.

920-0025 Rev L
6/26/2015

Introduction

This manual describes Applied Motion Products CANopen implementation of CiA 301 and CiA 402 specifications. It is expected that the user fully understands both CiA standards. That understanding, along with this specification, will enable the user to develop a distributed motion control system.

The intent of this manual is to highlight manufacturer specific requirements as they pertain to Applied Motion Products drives.

Information and standards may be obtained from the CiA website at http://www.can-cia.com/. Information and software relating directly to the Applied Motion Products CANopen drives, including an open-source example program, may be obtained from our website at http://www.applied-motion.com/.

User Advisory:

Our new manual only pertains to these products:

ST5/10-C

STM17/23/24-C

SV2xxx-C-CE

SV7-C

TSM17/23/24/34C-xxx

TSM23/24C-xxx

TXM24C-xxx

This means that the STAC6-C only supports a limited set of the CANopen objects listed in the manual.

Contents

List of Tables

List of Figures

Reference Documents

Applied Motion Products CANopen Drive Hardware Manual
CiA 301
CiA 303
CiA 402
Bosch CAN Physical Layer Specifications 2.0B
Applied Motion Products Host Command Reference

Commonly Used Acronyms

CAN	Controller Area Network
CiA	CAN in Automation Group (Standards Body)
COB-ID	Communication Object Identification
EDS	Electronic Data Sheet
NMT	Network Management
OD	Object Dictionary
PDS	Power Drive System
PDO	Process Data Object
RPDO	Receive (incoming) PDO
SDO	Service Data Object
TPDO	Transmit (outgoing) PDO

CANopen Network Topology Overview

Applied Motion Products CANopen drives can be integrated into a CANopen system with other device types as shown below.

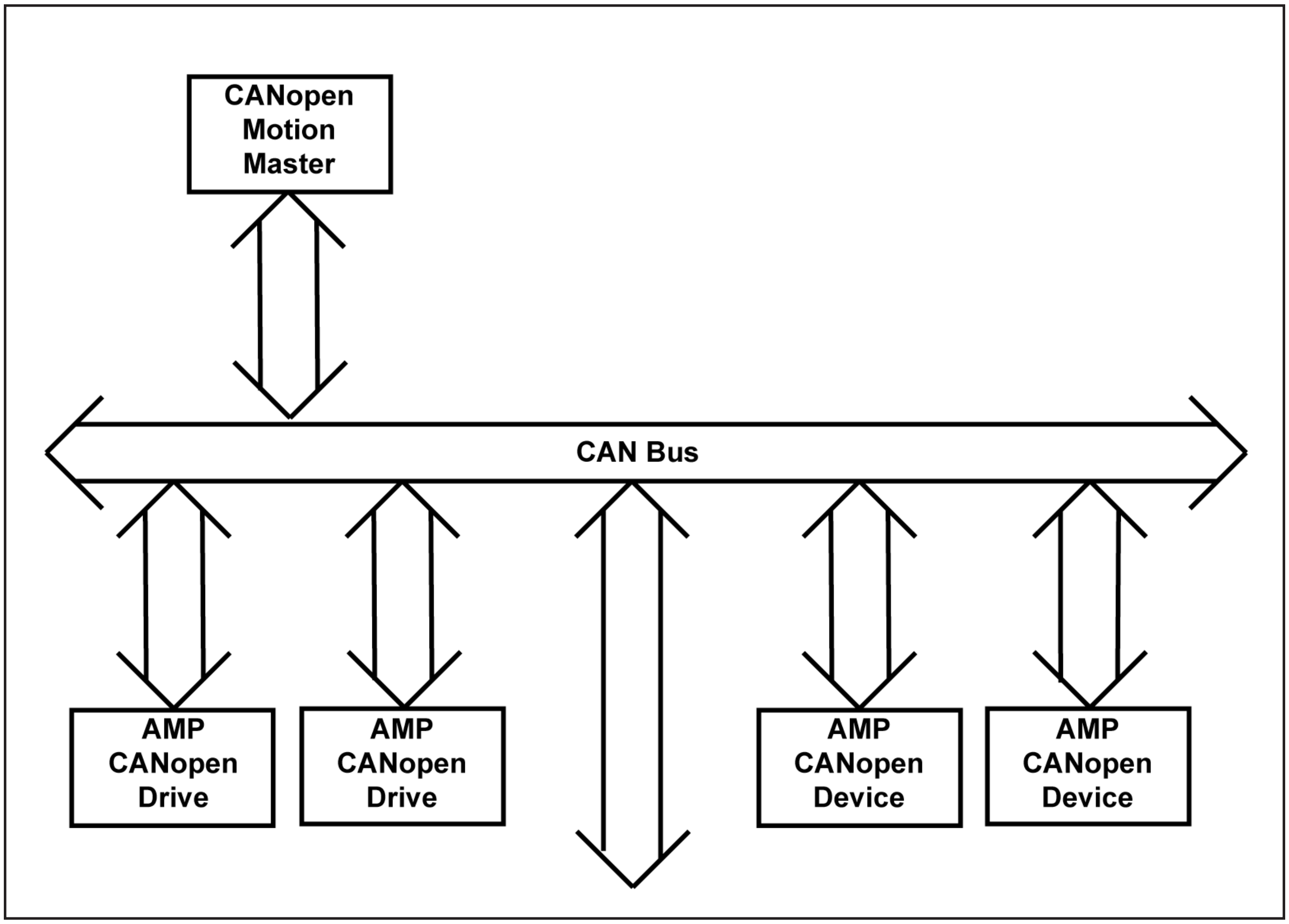


Figure 1: CANopen Network Topology Overview

Drive Setup

There are four phases to setting up a Applied Motion Products CANopen Drive:

- Wiring the power and motor
- Wiring the CANopen connector to the drive
- Setting the Bit Rate and Node ID
- Configuring the Drive

Wiring the Power and Motor

Please refer to the drive's hardware manual for this step. The appropriate manual can be found on our website www.applied-motion.com.

Wiring the CANopen Connector for CANopen Drives

Please refer to Applied Motion Products CANopen Drive Hardware Manual for the Communication connection. As an example, we use the connections for the ST5-C drive.

Applied Motion Products ST5-C drive uses a four-pin spring connector, shown in Figure 2 below, and conforms to CiA303 specification. The connector should be wired in a daisy-chain configuration, as shown in Figure 3 below, with a 120 ohm resistor used to terminate each end. Other wiring topologies, such as star networks, are not recommended due to wave reflection problems. Please reference specific hardware manuals for your drive's wiring configuration.

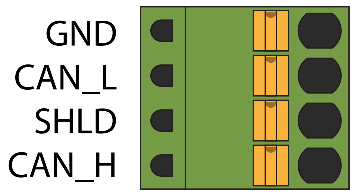
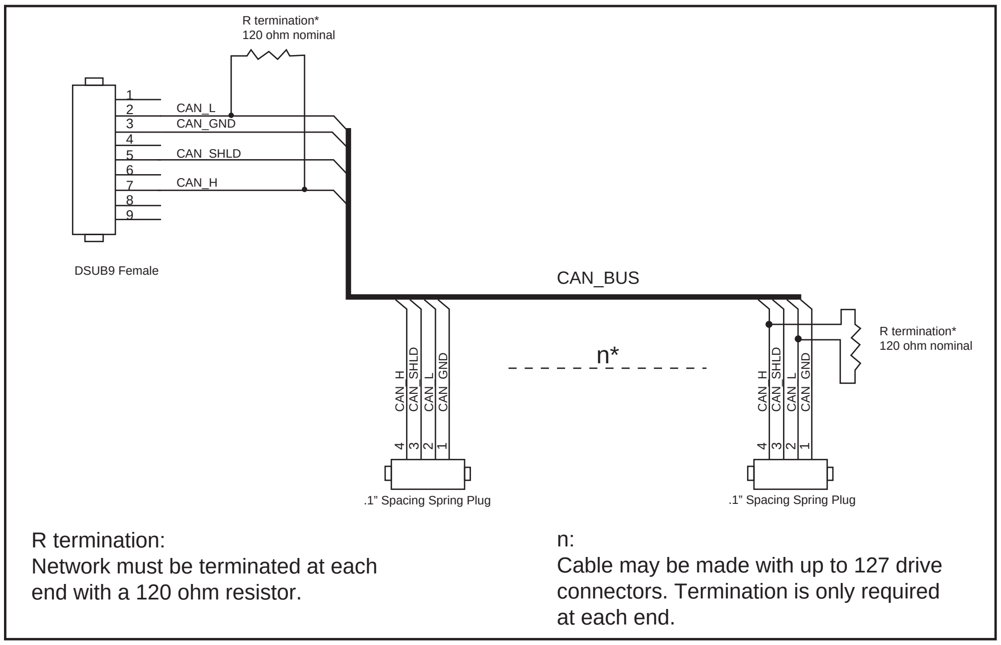


Figure 2: The CANopen Connector

Figure 3 shows a CANopen network with two Applied Motion Products ST5-C drive connectors wired to a Kvaser Leaf USB to CANopen Adapter.



R termination:
Network must be terminated at each end with a 120 ohm resistor.

n:
Cable may be made with up to 127 drive connectors. Termination is only required at each end.

Figure 3: Wiring Schematic

CANopen BitRate & NodeID

Applied Motion Products CANopen drives have three settings, one for Bit Rate and two for Node-ID.

The Bit Rate is configured using an 8-position switch. See Table 1 for the Bit Rate settings. Please reference the drive's hardware manual for the location of the Bit Rate switch.

The Node-ID is configured using a 16-position switch to set the lower 4 bits of the Node-ID and a 8 position switch to set the upper 3 bits of the Node-ID. In some cases, the upper 3 bits of the Node-ID are configured using *ST Configurator*™, *Step-Servo Quick Tuner*™ or our standard *Quick Tuner*™. Please reference the drive's hardware manual for Node-ID switch configuration and setup. Valid ranges for the Node-ID are 01_h through $7F_h$. Node-ID 00_h is reserved in accordance with the CiA 301 specification.

Note: The Node-ID and Bit Rate are captured only after a power cycle, or after a network reset command has been sent. Changing the switches while the drive is powered on will NOT change the Node-ID until one of those conditions has also been met.

Switch Setting	Resultant Bit Rate
0	1 Mbps
1	800 kbps
2	500 kbps
3	250 kbps
4	125 kbps
5	50 kbps
6	20 kbps
7	12.5 kbps

Table 1: Bit Rate Switch Settings

Drive Configuration

Once the CAN connector has been wired to the drive, and the Node-ID and Bit Rate have been set, the drive can be configured. Drive configuration for ST stepper drives and STM Integrated Motors is accomplished using the *ST Configurator*™ software, which can be found on the Applied Motion Products website. Drive configuration and tuning for servo drives are performed using *Quick Tuner*™ or *Step-Servo Quick Tuner*™. In all cases the drive will need to be connected to a Windows PC using the included RS-232 serial cable. Please refer to the appropriate software manual or built-in help screens for details.

Note: When the CANopen drive is first powered on, the drive will automatically send a power-up packet over the RS-232 port. If an Applied Motion Products application is present, it will send a response back to the drive over RS-232 and the drive will hold the CAN node in the Initialization state until the application is closed. If no response is detected, the drive will continue the normal CANopen startup procedure: the drive will power up into the Initialization state, send out a boot-up packet, move into the Pre-Operational state, and start sending out heartbeats with the Pre-Operational state as a status code.

Supported CiA 402 Modes of Operation

Mode	ST	STM/SWM	STAC6	SV/TSM/TXM
Profile Velocity	●	●	●	●
Profile Position	●	●	●	●
Homing	●	●	●	●
Profile Torque				●
Interpolated Position				
Q Program	●	●	●	●

Table 2: Modes of Operation

For detailed information on each mode see the corresponding appendix:

Profile Position Mode - Appendix D

Profile Velocity Mode - Appendix E

Homing Modes - Appendix F

Profile Torque Mode - Appendix G

Q Program Mode - Appendix H

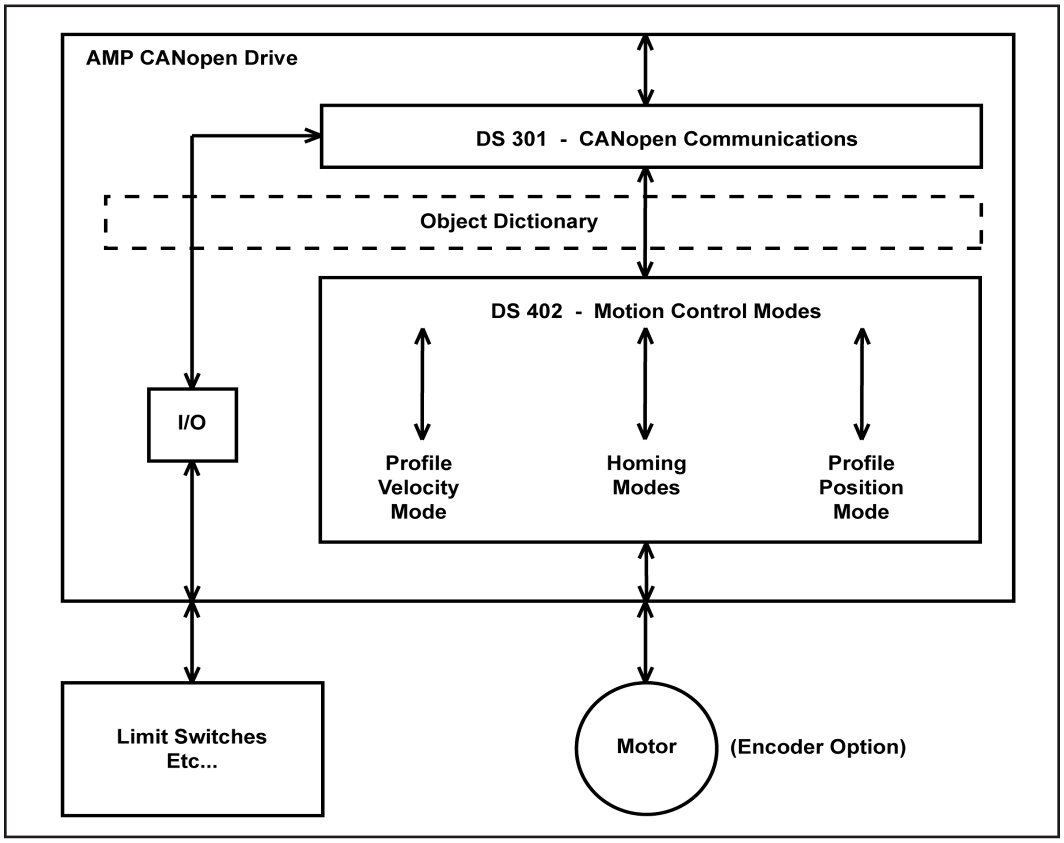


Figure 4: CANopen Drive - Motion Control Modes

Object Dictionary

The Object Dictionary (OD) is the core of any CANopen node. It provides links to all the communication and running parameters of a node. The Object Dictionary is defined in the Electronic Data Sheet (EDS), which lists all supported objects, along with any sub-objects.

For more information: see Appendix A for a detail decription of Object Dictionary.

Any OD Entry may be accessed using the standard Service Data Object (SDO) protocol, while some may be accessed using the low-overhead Process Data Object (PDO) protocol.

For more information: see Appendix J for a description of SDO and PDO Access

Electronic Data Sheet

The EDS, available on Applied Motion Products website, lists all the properties of every supported object in the OD, shown below, a list of Object Dictionary descriptions:

Index	Sub	Comment Fields	Data Type	Access Type	PDO Mapping
1000h	0	Device Type	UNSIGNED32	RO	
1001h	0	Error Register	UNSIGNED8	RO	Yes
1002h	0	Manufacturer status register	UNSIGNED32	RO	Yes
1005h	0	COB-ID Sync message	UNSIGNED32	RW	
1006h	0	Communication cycle period	UNSIGNED32	RW	
1007h	0	Synchronous window length	UNSIGNED32	RW	
1008h	0	Manufacturer Device Name	VISIBLE_STRING	CONST	
1009h	0	Manufacturer hardware version	VISIBLE_STRING	CONST	
100Ah	0	Manufacturer software version	VISIBLE_STRING	CONST	
1010h		Store parameters	ARRAY		
	0	max sub-index	UNSIGNED8	RO	
	1	save all parameters	UNSIGNED32	RW	
1014h	0	COB-ID EMCY	UNSIGNED32	RO	
1017h	0	Producer Heart Beat	UNSIGNED16	RW	
1018h		Identity Object	RECORD		
	0	max sub-index	UNSIGNED8	RO	
	1	Vendor-ID	UNSIGNED32	RO	
	2	Product code	UNSIGNED32	RO	
	3	Revision number	UNSIGNED32	RO	
	4	Serial number	UNSIGNED32	RO	
1200h	0	Server SDO Parameter 0	RECORD		
1400h	0	Receive PDO Communications Parameter 0	RECORD		
1401h	0	Receive PDO Communications Parameter 1	RECORD		
1402h	0	Receive PDO Communications Parameter 2	RECORD		
1403h	0	Receive PDO Communications Parameter 3	RECORD		
1600h	0	Receive PDO Mapping Parameter 0	RECORD		
1601h	0	Receive PDO Mapping Parameter 1	RECORD		
1602h	0	Receive PDO Mapping Parameter 2	RECORD		
1603h	0	Receive PDO Mapping Parameter 3	RECORD		
1800h	0	Transmit PDO Communications Parameter 0	RECORD		
1801h	0	Transmit PDO Communications Parameter 1	RECORD		
1802h	0	Transmit PDO Communications Parameter 2	RECORD		
1803h	0	Transmit PDO Communications Parameter 3	RECORD		
1A00h	0	Transmit PDO Mapping Parameter 0	RECORD		
1A01h	0	Transmit PDO Mapping Parameter 1	RECORD		
1A02h	0	Transmit PDO Mapping Parameter 2	RECORD		
1A03h	0	Transmit PDO Mapping Parameter 3	RECORD		

Table 3: CiA 301 Object Descriptions

For more information: see Appendix A for detail decription of OD.

Index	Sub	Comment Fields	Data Type	Access Type	PDO Mapping
5000h	0	PositionGain_Tuning	UNSIGNED16	RW	
5001h	0	PositionDeriGain_Tuning	UNSIGNED16	RW	
5002h	0	PositionDeriFilter_Tuning	UNSIGNED16	RW	
5003h	0	VelocityGain_Tuning	UNSIGNED16	RW	
5004h	0	VelocityIntegGain_Tuning	UNSIGNED16	RW	
5005h	0	AccFeedForward_Tuning	UNSIGNED16	RW	
5006h	0	PIDFilter_Tuning	UNSIGNED16	RW	
5007h -500Eh	0	NotchFilterX_Tuning	INTEGER16	RW	
500Fh	0	PositionError	UNSIGNED16	RW	
5010h	0	Velocity_MAX	UNSIGNED16	RW	
5011h	0	SmoothFilter	UNSIGNED16	RW	
5012h	0	Driver_Temperature	INTEGER16	RO	

Table 4: Tuning Parameter setting

For more information: see Appendix A for detail decription of OD.

Index	Sub	Comment Fields	Data Type	Access Type	PDO Mapping
603Fh	0	Error Code	UNSIGNED16	RO	Yes
6040h	0	Controlword	UNSIGNED16	WO	Yes
6041h	0	Statusword	UNSIGNED16	RO	Yes

605Ah	0	Quick Stop option code	INTEGER16	RW	
6060h	0	Modes of Operation	INTEGER8	WO	Yes
6061h	0	Modes of Operation Display	INTEGER8	RO	Yes
6064h	0	Position Target value calculated	INTEGER32	RO	Yes
6065h	0	following_error_window	UNSIGNED32	RW	
606Ch	0	Velocity Target value calculated	INTEGER32	RO	Yes
6071h	0	Target Torque (servos only)	INTEGER16	RW	Yes
6073h	0	Max Running Current	UNSIGNED16	RW	Yes
6074h	0	torque_demand_value	INTEGER16	RO	Yes
6078h	0	current_actual_value	INTEGER16	RO	Yes
607Ah	0	Target Position	INTEGER32	RW	Yes
607Ch	0	Home Offset	INTEGER16	RW	Yes
607Eh	0	polarity	UNSIGNED8	RW	Yes
607Fh	0	max_profile_speed	UNSIGNED32	RW	Yes
6081h	0	Profile Velocity	UNSIGNED32	RW	Yes
6083h	0	Profile Acceleration	UNSIGNED32	RW	Yes
6084h	0	Profile Deceleration	UNSIGNED32	RW	Yes
6085h	0	Quick Stop Deceleration	UNSIGNED32	RW	Yes
6087h	0	torque_slope	UNSIGNED32	RW	Yes
6098h	0	Homing Method	INTEGER8	RW	Yes
6099h		Homing Speeds	ARRAY		
	0	max sub-index	UNSIGNED8	RO	
	1	speed_during_search_for_switch	UNSIGNED32	RW	Yes
	2	speed_during_search_for_zero	UNSIGNED32	RW	Yes
609Ah	0	Homing Acceleration	UNSIGNED32	RW	Yes
60FEh		Drive Outputs	RECORD		
	0	max sub-index	UNSIGNED8	RO	
	1	physical_outputs	UNSIGNED32	RW	Yes
60FFh	0	Target Velocity	INTEGER32	RW	Yes
6502h	0	Supported Drive Modes	UNSIGNED32	RO	

Table 5: CiA 402 Objects

For more information: see Appendix A for detail decription of OD.

Index	Sub	Comment Fields	Data Type	Access Type	PDO Mapping
7001h	0	Home Switch	UNSIGNED8	RW	Yes
7002h	0	Idle Current	UNSIGNED16	RW	
7003h	0	Display Drive Inputs	UNSIGNED16	RO	Yes
7005h	0	torque_Constant	UNSIGNED16	RW	
7006h	0	DSP_clearAlarm	UNSIGNED8	WO	
7007h	0	Q Sequence Number	UNSIGNED8	RW	Yes
7009h	0	Velocity Actual Value	UNSIGNED16	RO	Yes
700Ah	0	Position Actual Value	INTEGER32	RO	Yes
700Bh	0	DSP Status Code	UNSIGNED16	RO	Yes
700Ch	0	zero_Position	UNSIGNED8	WO	
700Dh	0	Acceleration_Current	UNSIGNED16	RW	
700Eh	0	Analog Input 1	UNSIGNED16	RO	Yes

Global Control Word and Status Word

Byte #	Bit #	Control Word					Status Word				
		Homing	Position	Velocity	Torque	Q	Homing	Position	Velocity	Torque	Q
1	15	Reserved					Reserved				
	14										
	13						Homing Error	Following Error	Reserved	Reserved	Reserved
	12						Homing Attained	Set Point Ack	Speed	Reserved	Reserved
	11						Internal Limit Active				
	10						Target Reached				
	9	Reserved	Change of Set Point	Reserved	Reserved	Reserved	Remote				
	8	Halt					Reserved				
2	7	Fault Reset					Warning				
	6	Reserved	Abs/Rel	Reserved	Reserved	Reserved	Switch On Disabled				
	5	Reserved	Change Set Point Immediately	Reserved	Reserved	Reserved	Quick Stop				
	4	Homing Operation Start	New Set Point	Reserved	Reserved	Q Program Start	Voltage Enabled				
	3	Enable Operation					Fault				
	2	Quick Stop					Operation Enabled				
	1	Enable Voltage					Switched On				
	0	Switch On					Ready to Switch On				

Figure 5: Global Control Word and Status Word

General Purpose Registers

Applied Motion Products CANopen products provide 23 general purpose registers. These registers are 32 bit read/write registers. This is volatile memory, so data written there will not be saved after a power cycle.

These registers may also be accessed and manipulated via a stored Q Program, if desired. The table below cross-references the CANopen OD entries and the Q Programmer register addresses.

For more information on using these general purpose registers in a Q Program, please see the Host Command Reference, available at www.applied-motion.com.

For more information about the Q Program Mode: see Appendix H

Register Name	OD Address	Q Register Address
User Defined Register 0	4000.01_h *	0
User Defined Register 1	4000.02_h	1
User Defined Register 2	4000.03_h	2
User Defined Register 3	4000.04_h	3
User Defined Register 4	4000.05_h	4
User Defined Register 5	4000.06_h	5
User Defined Register 6	4000.07_h	6
User Defined Register 7	4000.08_h	7
User Defined Register 8	4000.09_h	8
User Defined Register 9	$4000.0A_h$	9
User Defined Register 10	$4000.0B_h$	:
User Defined Register 11	$4000.0C_h$	;
User Defined Register 12	$4000.0D_h$	<
User Defined Register 13	$4000.0E_h$	=
User Defined Register 14	$4000.0F_h$	>
User Defined Register 15	4000.10_h	?
User Defined Register 16	4000.11_h	@
User Defined Register 17	4000.12_h	[
User Defined Register 18	4000.13_h	\
User Defined Register 19	4000.14_h	]
User Defined Register 20	4000.15_h	^
User Defined Register 21	4000.16_h	_
User Defined Register 22	4000.17_h	`

*** Note:** 4000.01h means OD address is index 4000h and sub-index 01h.

Table 7: User Defined Registers in CANopen and Q Programmer

Appendix A - The Object Dictionary

The most important part of a device profile is the Object Dictionary description. The Object Dictionary is essentially a grouping of objects accessible via the network in an ordered pre-defined fashion. Each object within the dictionary is addressed using a 16-bit index.

The overall layout of the standard Object Dictionary is shown below. This layout closely conforms with other industrial serial bus system concepts:

Index (hex)	Object
0000	not used
0001-001F	Static Data Types
0020-003F	Complex Data Types
0040-005F	Manufacturer Specific Complex Data Types
0060-007F	Device Profile Specific Static Data Types
0080-009F	Device Profile Specific Complex Data Types
00A0-0FFF	Reserved for further use
1000-1FFF	Communication Profile Area
2000-5FFF	Manufacturer Specific Profile Area
6000-9FFF	Standardised Device Profile Area
A000-FFFF	Reserved for further use

Table 8: Object Dictionary Structure

Communication Profile

0x1000 - Device type

Contains information about the device type. The object at index 1000h describes the type of device and its functionality. It is composed of a 16-bit field which describes the device profile that is used and a second 16-bit field which gives additional information about optional functionality of the device. The Additional Information parameter is device profile specific. Its specification does not fall within the scope of this document, it is defined in the appropriate device profile. The value 0000h indicates a device that does not follow a standardised device profile. For multiple device modules the Additional Information parameter contains FFFFh and the device profile number referenced by object 1000h is the device profile of the first device in the Object Dictionary. All other devices of a multiple device module identify their profiles at objects 67FFh + x*800h with x = internal number of the device (0-7).

These entries describe the device type of the preceding device.

Object Type	Data Type	Access Type	PDO Mapping	COS[1]	Default value
Var	UNSIGNED32	ro	no	no	0x00000000

bit 0-15: Device profile number

bit 16-31: Additional information

Note 1: COS means: TPDO detects the change of State

0x1001 - Error Register

This object is an error register for the device. The device can map internal errors in this byte. This entry is mandatory for all devices. It is a part of an Emergency object.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	ro	optional	no	0

bit 0: generic error

bit 1: current

bit 2: voltage

bit 3: temperature

bit 4: communication error (overrun, error state)

bit 5: Reserved (always 0)

bit 6: Reserved (always 0)

bit 7: Reserved (always 0)

0x1002 –Manufacturer status register (not yet implemented)

This object is a common status register for manufacturer specific purposes.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	ro	optional	no	0

0x1003 - Pre-defined error field

The object at index 1003h holds the errors that have occurred on the device and have been signaled via the Emergency Object. In doing so it provides an error history.

1. The entry at sub-index 0 contains the number of actual errors that are recorded in the array starting at sub-index 1.

2. Every new error is stored at sub-index 1, the older ones move down the list.

3. Writing a “0” to sub-index 0 deletes the entire error history (empties the array). Values higher than 0 are not allowed. This would lead to an abort message (error code: 0609 0030h).

4. The error numbers are of type UNSIGNED32 and are composed of a 16 bit error code and a 16 bit additional error information field which is manufacturer specific. The error code is contained in the lower 2 bytes (LSB) and the additional information is included in the upper 2 bytes (MSB). If this object is supported it must consist of two entries at least. The length entry on sub-index 0h and at least one error entry at sub-index 1H.

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS	Default value
Array	9	UNSIGNED32	ro	optional	no	0

Subindex	Name	Default value
00	Number of errors	0
01	Standard error field	0
02	Standard error field	0
03	Standard error field	0
04	Standard error field	0
05	Standard error field	0
06	Standard error field	0
07	Standard error field	0
08	Standard error field	0

Number of Errors (subindex = 0) :

bit 0-7: Zero can be written to erase error history

Standard Error Field (subindex = 1~ 8) :

bit 0-15: Error code as transmited in the Emergency object

bit 16-31: Manufacturer specific additional information

0x1005 COB-ID SYNC message

Index 1005h defines the COB-ID of the Synchronisation Object (SYNC). Further, it defines whether the device generates the SYNC. The structure of this object is shown in the following Figure and Table:

UNSIGNED32

MSB LSB

bits	31	30	29	28-11	10-0
11-bit-ID	0/1	0/1	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	11-bit Identifier
29-bit-ID	0/1	0/1	1	29 -bit Identifier	

Figure 6: Structure of TIME COB-ID entry

bit number	value	meaning
31 (MSB)	0 1	Device does not consume TIME message Device consumes TIME message
30	0 1	Device does not produce TIME message Device produces TIME message
29	0 1	11-bit ID (CAN 2.0A) 29-bit ID (CAN 2.0B)

28 -11	0
	X
		if bit 29=0
		if bit 29=1: bits 28-11 of 29-bit-TIME-COB-ID

10-0 (LSB)	X	bits 10-0 of TIME-COB-ID

Table 9: Description of SYNC COB-ID entry

Bits 29, 30 are not changeable. If a device is not able to generate SYNC messages, an attempt to set bit 30 is responded to with an abort message (abort code: 0609 0030h). Devices supporting the standard CAN frame type only either ignore attempts to change bit 29 or respond with an abort message (abort code: 0609 0030h). The first transmission of SYNC object starts within 1 sync cycle after setting Bit 30 to 1. Once the objects exists (Bit 30=1) you are not allowed to change Bit 0-29.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	no	no	0x00000080

0x1006 Communication cycle period

This object defines the communication cycle period in ms. This period defines the SYNC interval. It is 0 if not used. If the communication cycle period on sync producer is changed to a new value unequal 0, the transmission of sync object resumes within 1 sync cycle of the new value.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	no	no	0

bit 0-31: period of SYNC transmission in ms (0 = no transmission, no checking)

0x1007 Synchronous window length

Contains the length of the time window for synchronous PDOs in ms. It is 0 if not used.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	no	no	0

bit 0-31: window length after SYNC when PDOs must be transmited in μs, (0 = not used)

0x1008 Manufacturer device name

Contains the manufacturer device name.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	Visible string	const	no	no	AMP CANopen Motor Driver

Name of the manufacturer as string

0x1009 Manufacturer hardware version

Contains the manufacturer hardware version description.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	Visible string	const	no	no	A001

Name of the hardware version as string.

0x100A Manufacturer software version

Contains the manufacturer software version description.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	Visible string	const	no	no	2.00A

Name of the software version as string.

0x1010 Store parameters

This object supports the saving of parameters in non-volatile memory.

1. Sub-Index 0 contains the largest Sub-Index that is supported.
2. Sub-Index 1 refers to all parameters that can be stored on the device.
3. Sub-Index 2 refers to communication related parameters (Index 1000h - 1FFFh manufacturer specific communication parameters).
4. Sub-Index 3 refers to application related parameters (Index 6000h - 9FFFh manufacturer specific application parameters).
5. At Sub-Index 4 - 127: manufacturers may store their choice of parameters individually.
6. Sub-Index 128 - 254 are reserved for future use.

In order to avoid storage of parameters by mistake, storage is only executed when a specific signature is written to the appropriate Sub-Index. The signature is “save”.

	MSB			LSB
Signature ISO 8859 (“ASCII“)	e	v	a	s
hex	65h	76h	61h	73h

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS	Default value
Array	2	UNSIGNED32	rw	no	no	0

Subindex	Name	Default value
00	max sub-index	1
01	save all parameters	0

Writing value 0x65766173 (‘s’,’a’,’v’,’e’ from LSB to MSB) into this location stores all Manufacturer specific parameters into the EEPROM.

Here list the parameters which can be stored:

0x5000	positionGain_Tuning;
0x5001	positionDeriGain_Tuning;
0x5002	positionDeriFilter_Tuning;
0x5003	velocityGain_Tuning;
0x5004	velocityIntegGain_Tuning;
0x5005	accFeedForward_Tuning;
0x5006	PIDFilter_Tuning;
0x5007	notchFilterA_Tuning;
0x5008	notchFilterB_Tuning;
0x5009	notchFilterC_Tuning;
0x500A	notchFilterD_Tuning;

0x500B	notchFilterE_Tuning;
0x500C	notchFilterF_Tuning;
0x500D	notchFilterG_Tuning;
0x500E	notchFilterH_Tuning;
0x500F	positionError;
0x5010	velocity_MAX;
0x5011	smoothFilter;

0x1011 Restore default parameters (Not yet implemented)

With this object the default values of parameters according to the communication or device profile are restored.

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS	Default value
Var	2	UNSIGNED32	rw	no	no	0

Subindex	Name	Default value
00	max sub-index	1
01	restore all parameters	0

0x1014 COB-ID EMCY

This object defines the COB-ID of the Emergency Object (EMCY).

UNSIGNED32

MSB LSB

bits	31	30	29	28-11	10-0
11-bit-ID	0/1	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	11-bit Identifier
29-bit-ID	0/1	0	1	29 -bit Identifier	

Figure 7: Structure of the EMCY Identifier entry

bit number	value	Meaning
31 (MSB)	0 1	EMCY exists / is valid EMCY does not exist / is not valid
30	0	reserved (always 0)
29	0 1	11-bit ID (CAN 2.0A) 29-bit ID (CAN 2.0B)

28-11	0
	X
		if bit 29=0
		if bit 29=1: bits 28-11 of 29-bit-COB-ID

10-0 (LSB)	X	bits 10-0 of COB-ID

Figure 8: Description of EMCY COB-ID entry

Devices supporting the standard CAN frame type only, an attempt to set bit 29 is responded with an abort message (abort code: 0609 0030h). It is not allowed to change Bits 0-29, while the object exists (Bit 31=0).

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	ro	no	no	0x80+$NODEID

bit 0-10: COB-ID

bit 11-30: set to 0 for 11 bit COB-ID

bit 31: 0(1) - node uses (does NOT use) Emergency object

0x1015 inhibit time EMCY

The inhibit time for the EMCY message can be adjusted via this entry. The time has to be a multiple of 100ms.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	200

bit 0-15: Inhibit time of emergency message in 100µs

0x1017 Producer heartbeat time

The producer hartbeat time defines the cycle time of the heartbeat. The producer heartbeat time is 0 if it not used. The time has to be a multiple of 1ms.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	1000

bit 0-15: Heartbeat producer time in ms (0 = disable transmission)

0x1018 Identity

The object at index 1018h contains general information about the device.

1. The Vendor ID (sub-index 1h) contains a unique value allocated to each manufacturer.

2. The manufacturer-specific Product code (sub-index 2h) identifies a specific device version.

3. The manufacturer-specific Revision number (sub-index 3h) consists of a major revision number and a minor revision number. The major revision number identifies a specific CANopen behaviour. If theCANopen functionality is expanded, the major revision has to be incremented. The minor revision number identifies different versions with the same CANopen behaviour.

4. The manufacturer-specific Serial number (sub-index 4h) identifies a specific device.

Object Type	Sub Number
Record	5

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	max sub-index	UNSIGNED8	ro	no	no	4
01	Vendor-ID	UNSIGNED32	ro	no	no	0x000002D9
02	Product code	UNSIGNED32	ro	no	no	0x00000000
03	Revision number	UNSIGNED32	ro	no	no	0x00000000
04	Serial number	UNSIGNED32	ro	no	no	0x00000000

max sub-index

Vendor-ID

bit 0-31: Assigned by CiA (here is 0x000002D9 for Shanghai AMP and Applied Motion Products Automation)

Product code

bit 0-31: Manufacturer specific

Revision number

bit 0-15: Minor revision num. (CANopen behavior has not changed)

bit 16-31: Major revision number (CANopen behavior has changed)

Serial number

bit 0-31: Manufacturer specific (Lot Number)

0x1019 Synchronous counter overflow value

This object contains the max counters of SYNC message. If it is 0 the producer will send the SYNC message without any data, otherwise with data length 1.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	rw	no	no	0

If value is zero, then SYNC message is transmitted with data length 0.

If value is from 2 to 240, then SYNC message has one data byte, which contains the counter.

Other values are reserved.

0x1029 Error behavior

This object contains the error information of the CANopen driver.

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS
Array	7	UNSIGNED8	rw	no	no

Subindex	Name	Default value	Actual value
00	max sub-index	6	
01	Communication	0x00	
02	Communication other	0x00	
03	Communication passive	0x01	
04	Generic	0x00	
05	Device profile	0x00	
06	Manufacturer specific	0x00	

Value definition for all subindexes:

0x00 - if operational, switch to NMT pre-operational

0x01 - do nothing

0x02 - switch to NMT stopped

01 - Communication error - bus off or Heartbeat consumer error.

02 - Communication other error (critical errors - see ‘Error status bits’) except CAN bus passive but including bus off or Heartbeat consumer.

03 - Communication passive - any communication error including CAN bus passive.

04 - Generic error (critical errors - see ‘Error status bits’).

05 - Device profile error - bit 5 in error register is set.

06 - Manufacturer specific error - bit 7 in error register is set.

0x1200 SDO server parameter

The object holds the COB-ID (communication object ID, also known as CAN message ID) values used to access the driver’s SDO.

1. Sub-index 0 contains the number of sub-elements of this record.
2. Sub-index 1 used by the driver to receive SDO packets. The value is 0x600 + $NODEID.
3. Sub-index 2 gives the COB-ID used by the driver to transmit SDO packets. The value is 0x580 + $NODEID.

Object Type	Sub Number
Record	3

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	max sub-index	UNSIGNED8	ro	no	no	2
01	COB-ID client to server	UNSIGNED32	ro	no	no	0x600+$NODEID
02	COB-ID server to client	UNSIGNED32	ro	no	no	0x580+$NODEID

0x1400~0x1403 RPDO communication

Contains the communication parameters for the PDOs the device is able to receive. The type of the PDO communication parameter (20h) is described in CiA 301 section 9.5.4. The sub-index 0h contains the number of valid entries within the communication record. Its value is at least 2. If inhibit time is supported, the value is 3. At sub-index 1h resides the COB-ID of the PDO. This entry has been defined as UNSIGNED32 in order to allow 11-bit CAN Identifiers (CAN 2.0A) as well as for 29-bit CAN identifiers (CAN 2.0B).

UNSIGNED32

	MSB				LSB
bits	31	30	29	28-11	10-0
11-bit-ID	0/1	0/1	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	11-bit Identifier
29-bit-ID	0/1	0/1	1	29 -bit Identifier	

Figure 9: Structure of PDO COB-ID entry

bit number	value	meaning
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is not valid
30	0	RTR allowed on this PDO
	1	no RTR allowed on this PDO
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28-11	0	if bit 29=0
	X	if bit 29=1: bits 28-11 of 29-bit-COB-ID
10-0 (LSB)	X	bits 10-0 of COB-ID

Table 10: Description of PDO COB-ID entry

The PDO valid/not valid bit allows for the selection of which PDOs are used in the operational state. There may be PDOs fully configured (e.g. by default) but not used, and therefore set to “not valid” (deleted). The feature is necessary for devices supporting more than 4 RPDOs or 4 TPDOs, because each device has only default identifiers for the first four RPDOs/TPDOs. Devices supporting the standard CAN frame type only or not supporting Remote Frames, an attempt to set bit 29 to 1 or bit 30 to 0 will be met with an abort message (abort code: 0609 0030h). Changing bit 0-29 is not allowed while the PDO exists (Bit 31=0).

The transmission type (sub-index 2) defines the transmission/reception character of the PDO. Table 11 describes the usage of this entry. On an attempt to change the value of the transmission type to a value that is not supported by the device an abort message (abort code: 0609 0030h) is generated.

transmission type	PDO transmission				
	cyclic	acyclic	synchronous	asynchronous	RTR only
0		X	X		
1-240	X		X		
241-251	- reserved -				
252			X		X
253				X	X
254				X	
255				X	

Table 11: Description of transmission type

Synchronous (transmission types 0-240 and 252) means that the transmission of the PDO shall be related to the SYNC object. Preferably the devices use the SYNC as a trigger to output or actuate based on the previous synchronous Receive PDO respectively to update the data transmitted at the following synchronous Transmit PDO. Details of this mechanism depend on the device type and are defined in the device profile if applicable.

Asynchronous means that the transmission of the PDO is not related to the SYNC object. A transmission type of zero means that the message shall be transmitted synchronously with the SYNC object but not periodically. A value between 1 and 240 means that the PDO is transferred synchronously and cyclically. The transmission type indicating the number of SYNC which are necessary to trigger PDO transmissions.

Receive PDOs are always triggered by the following SYNC upon reception of data independent of the transmission types 0 - 240. The transmission types 252 and 253 mean that the PDO is only transmitted on remote transmission request. At transmission type 252, the data is updated (but not sent) immediately after reception of the SYNC object.

At transmission type 253 the data is updated at the reception of the remote transmission request (hardware and software restrictions may apply). These value are only possible for TPDOs.

For TPDOs transmission type 254 means, the application event is manufacturer specific (manufacturer specific part of the Object Dictionary), transmission type 255 means, the application event is defined in the device profile. RPDOs with that type trigger the update of the mapped data with the reception.

Sub-index 3h contains the inhibit time. This time is a minimum interval for PDO transmission. The value is defined as multiple of 100µs. It is not allowed to change the value while the PDO exists (Bit 31 of sub-index 1 is 0).Sub-index 4h is reserved. It does not have to be implemented, in this case read or write access leads to Abort SDO Transfer (abort code: 0609 0011h).

In mode 254/255 additionally an event time can be used for TPDO. If an event timer exists for a TPDO (value not equal to 0) the elapsed timer is considered to be an event. The event timer elapses as multiple of 1 ms of the entry in sub-index 5h of the TPDO. This event will cause the transmission of this TPDO in addition to otherwise defined events. The occurrence of the events set the timer. Independent of the transmission type the RPDO event timer is used recognize the expiration of the RPDO.

Object Type	Sub Number
Record	3

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	max sub-index	UNSIGNED8	ro	no	no	2
01	COB-ID used by RPDO	UNSIGNED32	rw	no	no	$NODEID+0x200[2]
02	transmission type	UNSIGNED8	rw	no	no	255

0x1400 - 0x1403 RPDO communication parameter

max sub-index

COB-ID

bit 0-10: COB-ID for PDO.

bit 11-29: set to 0 for 11 bit COB-ID

bit 30: 0(1) - rtr are allowed (are NOT allowed) for PDO

bit 31: 0(1) - node uses (does NOT use) PDO

Transmission type

value = 0-240:	receiving is synchronous, process after next reception of SYNC object
value = 241-253:	not used
value = 254:	manufacturer specific
value = 255:	asynchronous

***Note 2:** Index 1400.01h Default value is 0x200+$NODEID

Index 1401.01h Default value is 0x300+$NODEID

Index 1402.01h Default value is 0x400+$NODEID

Index 1403.01h Default value is 0x500+$NODEID

0x1600~0x1603 RPDO mapping parameter

Contains the mapping for the PDOs the device is able to receive. The type of the PDO mapping

parameter (21h) is described in 9.5.4 of CiA 301. The sub-index 0h contains the number of valid entries within the mapping record. This number of entries is also the number of the application variables which shall be transmitted/received with the corresponding PDO. The sub-indices from 1h to number of entries contain the information about the mapped application variables. These entries describe the PDO contents by their index, sub-index and length . All three values are hexadecimal coded. The length entry contains the length of the object in bit (1..40h). This parameter can be used to verify the overall mapping length. It is mandatory.

The structure of the entries from sub-index 1h - 40h is as follows:

Byte:	MSB		LSB
	index (16 bit)	sub-index (8 bit)	object length (8 bit)

Figure 10: Structure of PDO Mapping Entry

If the change of the PDO mapping cannot be executed (e.g. the PDO length is exceeded or the SDO client attempts to map an object that cannot be mapped) the device responds with an Abort SDO Transfer Service.

Subindex 0 determines the valid number of objects that have been mapped. For changing the PDO mapping first the PDO has to be deleted, the sub-index 0 must be set to 0 (mapping is deactivated). Then the objects can be remapped. When a new object is mapped by writing a subindex between 1 and 64, the device may check whether the object specified by index / sub-index exists. If the object does not exist or the object cannot be mapped, the SDO transfer must be aborted with the Abort SDO Transfer Service with one of the abort codes 0602 0000h or 0604 0041h.

After all objects are mapped subindex 0 is set to the valid number of mapped objects. Finally the PDO will be created by writing to its communication parameter COB-ID. When subindex 0 is set to a value >0 the device may validate the new PDO mapping before transmitting the response of the SDO service. If an error is detected the device has to transmit the Abort SDO Transfer Service with one of the abort codes 0602 0000h, 0604 0041h or 0604 0042h.

When subindex 0 is read the actual number of valid mapped objects is returned. If data types (Index 1h-7h) are mapped they serve as “dummy entries”. The corresponding data in the PDO is not evaluated by the device. This optional feature is useful e.g. to transmit data to several devices using one PDO, each device only utilising a part of the PDO. It is not possible to create a dummy mapping for a TPDO.

A device that supports dynamic mapping of PDOs must support this during the PRE-OPERATIONAL state. If dynamic mapping during the OPERATIONAL state is supported, the SDO client is responsible for data consistency.

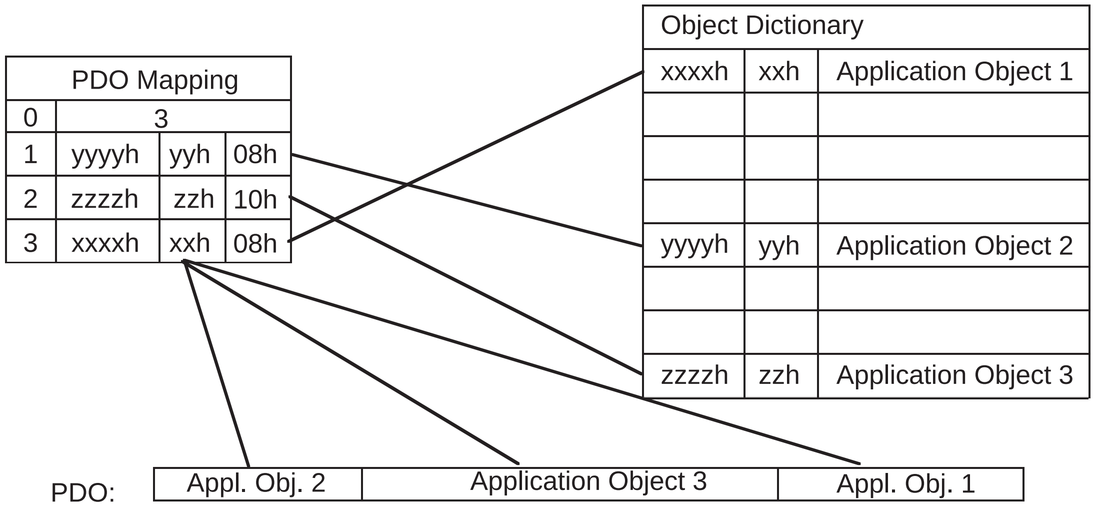


Figure 11: Principle of PDO mapping

Object Type	Sub Number
Record	9

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	Number of mapped objects	UNSIGNED8	rw	no	no	0
01	mapped object 1	UNSIGNED32	rw	no	no	0x00000000
02	mapped object 2	UNSIGNED32	rw	no	no	0x00000000
03	mapped object 3	UNSIGNED32	rw	no	no	0x00000000
04	mapped object 4	UNSIGNED32	rw	no	no	0x00000000
05	mapped object 5	UNSIGNED32	rw	no	no	0x00000000
06	mapped object 6	UNSIGNED32	rw	no	no	0x00000000
07	mapped object 7	UNSIGNED32	rw	no	no	0x00000000
08	mapped object 8	UNSIGNED32	rw	no	no	0x00000000

0x1600 - 0x1603 RPDO mapping parameter (To change mapping, ‘Number of mapped objects’ must be set to 0)

Number of mapped objects

mapped object (subindex 1...8)

bit 0-7:	data length in bits
bit 8-15:	subindex from OD
bit 16-31:	index from OD

0x1800~0x1803 TPDO communication parameter

Contains the communication parameters for the PDOs the device is able to transmit. The type of the PDO communication parameter (20h) is described in 9.5.4 of CiA 301. A detailed description of the entries is done in the section for the Receive PDO Communication Parameter (1400h – 1403h).

Object Type	Sub Number
Record	7

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	max sub-index	UNSIGNED8	ro	no	no	6
01	COB-ID used by TPDO	UNSIGNED32	rw	no	no	0x180+$NODEID3
02	transmission type	UNSIGNED8	rw	no	no	255
03	inhibit time	UNSIGNED16	rw	no	no	100
04	compatibility entry	UNSIGNED8	rw	no	no	0
05	event timer	UNSIGNED16	rw	no	no	0
06	SYNC start value	UNSIGNED8	rw	no	no	0

0x1800 - 0x1803 TPDO communication parameter

max sub-index

COB-ID

bit 0-10: COB-ID for PDONOTE: Bits 0-10 can not be changed.

bit 11-29: set to 0 for 11 bit COB-ID

bit 30: 0(1) - RTR are allowed (are NOT allowed) for PDO

bit 31: 0(1) - node uses (does NOT use) PDO

Transmission type

value = 0: transmiting is synchronous, specification in device profile

value = 1-240: transmiting is synchronous after every N-th SYNC object

value = 241-251: not used

value = 252-253: Transmited only on reception of Remote Transmission Request

value = 254: manufacturer specific

value = 255: asinchronous, specification in device profile

inhibit time

bit 0-15: Minimum time between transmissions of the PDO in 100µs. Zero disables functionality.

compatibility entry

bit 0-7: Not used.

event timer

bit 0-15: Time between periodic transmissions of the PDO in ms. Zero disables functionality.

SYNC start value

value = 0: Counter of the SYNC message shall not be processed.

value = 1-240: The SYNC message with the counter value equal to this value shall be regarded as the first received SYNC message.

Note 3: Index 1800.01h Default Value is 0x180+$NODEID

Index 1801.01h Default Value is 0x280+$NODEID

Index 1802.01h Default Value is 0x380+$NODEID

Index 1803.01h Default Value is 0x480+$NODEID

0x1A00 ~ 0x1A03 TPDO mapping parameter

Contains the mapping for the PDOs the device is able to transmit. The type of the PDO mapping

parameter (21h) is described in 9.5.4 of CiA 301. A detailed description of the entries is done in the section for the Receive PDO Mapping Parameter (1600h – 1603h).

Object Type	Sub Number
Record	9

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	Number of mapped objects	UNSIGNED8	rw	no	no	1
01	mapped object 1	UNSIGNED32	rw	no	no	0x60410010
02	mapped object 2	UNSIGNED32	rw	no	no	0x00000000
03	mapped object 3	UNSIGNED32	rw	no	no	0x00000000
04	mapped object 4	UNSIGNED32	rw	no	no	0x00000000
05	mapped object 5	UNSIGNED32	rw	no	no	0x00000000
06	mapped object 6	UNSIGNED32	rw	no	no	0x00000000
07	mapped object 7	UNSIGNED32	rw	no	no	0x00000000
08	mapped object 8	UNSIGNED32	rw	no	no	0x00000000

0x1A00 - 0x1A03 TPDO mapping parameter. (To change mapping, ‘Number of mapped objects’ must be set to 0).

Number of mapped objects

mapped object (subindex 1...8)

bit 0-7:	data length in bits
bit 8-15:	subindex from OD
bit 16-31:	index from OD

0x1F80 NMT startup (Not implemented)

The object set the NMT status when the device powerup.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	no	no	0x00000000

bit 0: 0(1) - device is not (is) NMT master
bit 1: 0(1) - if bit3=0, start explicitly assigned (all) nodes
bit 2: 0(1) - automaticaly enter (DO NOT automaticaly enter) the operational state on bootup
bit 3: 0(1) - NMT master may (may not) start nodes automatically
bit 4: 0(1) - if monitored node fails heartbeat handle that (all) node(s)
bit 5: 0(1) - flying master process not (yes) supported
bit 6: 0(1) - use bit 4 (ignore bit 4, stop all nodes)
bit 7-31: reserved, set to 0

Manufacturer Specific Objects

The objects described in this section are manufacturer-specific to configure or monitor the AMP CANopen driver.

0x2014~0x2017 PDOx_transmit_mask

The “TPDO detects COS” (means “on change data”) transmit event for PDOs can be mask off by these indexs. By default, all data in a PDO is considered when checking for change.

Each Index has 2 sub-indexes, sub-index 1 is used to mask (or reveal) bits in the lower 32bits (4bytes) of data. Sub-index 2 is the same for the upper 32bits.

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS	Default value
Array	3	UNSIGNED32	rw	no	no	0

Subindex	Name	Default value
00	max sub-index	2
01	PDOx_mask_LowerBytes	0xFFFFFFFF
02	PDOx_mask_upperBytes	0xFFFFFFFF

PDOx_mask_LowerBytes
bit0-31: mask off the lower 4 bytes of a TPDO when the data is “on change”

PDOx_mask_upperBytes
bit0-31: mask off the upper 4 bytes of a TPDO when the data is “on change”

0x2100 Error status bits

Error Status Bits indicates error conditions inside stack or inside application.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	OCTET_STRING	ro	optional	no	00 00 00 00 00 00 00 00 00 00

Default error status bits are:

Communication or protocol errors from driver (informative):
00 - ERROR_NO_ERROR - Error Reset or No Error.
01 - ERROR_CAN_BUS_WARNING - CAN bus warning.
02 - ERROR_RXMSG_WRONG_LENGTH - Wrong data length of received CAN message.
03 - ERROR_RXMSG_OVERFLOW - Previous received CAN message wasn’t processed yet.
04 - ERROR_RPDO_WRONG_LENGTH - Wrong data length of received PDO.
05 - ERROR_RPDO_OVERFLOW - Previous received PDO wasn’t processed yet.
06 - ERROR_CAN_RX_BUS_PASSIVE - CAN receive bus is passive.

07 - ERROR_CAN_TX_BUS_PASSIVE - CAN transmit bus is passive.
Communication or protocol errors from driver (critical):
08 - ERROR_08_reserved - (reserved)
09 - ERROR_09_reserved - (reserved)
0A - ERROR_CAN_TX_BUS_OFF - CAN transmit bus is off.
0B - ERROR_CAN_RXB_OVERFLOW - CAN module receive buffer has overflowed.
0C - ERROR_CAN_TX_OVERFLOW - CAN transmit buffer has overflowed.
0D - ERROR_TPDO_OUTSIDE_WINDOW - TPDO is outside SYNC window.
0E - ERROR_CAN_CONFIGURATION_FAILED - Configuration of CAN module CAN failed (Rx or Tx).
0F - ERROR_0F_reserved - (reserved)

Communication or protocol errors (informative):
10 - ERROR_NMT_WRONG_COMMAND - Wrong NMT command received.
11 - ERROR_SYNC_EARLY - SYNC message was too early.
12 - ERROR_12_reserved - (reserved)
13 - ERROR_13_reserved - (reserved)
14 - ERROR_14_reserved - (reserved)
15 - ERROR_15_reserved - (reserved)
16 - ERROR_16_reserved - (reserved)
17 - ERROR_17_reserved - (reserved)

Communication or protocol errors (critical):
18 - ERROR_SYNC_TIME_OUT - SYNC message timeout.
19 - ERROR_SYNC_LENGTH - Unexpected SYNC data length
1A - ERROR_PDO_WRONG_MAPPING - Error with PDO mapping.
1B - ERROR_HEARTBEAT_CONSUMER - Heartbeat consumer timeout.
1C - ERROR_HEARTBEAT_CONSUMER_REMOTE_RESET - Heartbeat consumer detected remote node reset.
1D - ERROR_1D_reserved - (reserved)
1E - ERROR_1E_reserved - (reserved)
1F - ERROR_1F_reserved - (reserved)

Generic errors (informative):
20 - ERROR_20_reserved - (reserved)
21 - ERROR_21_reserved - (reserved)
22 - ERROR_22_reserved - (reserved)
23 - ERROR_23_reserved - (reserved)
24 - ERROR_24_reserved - (reserved)
25 - ERROR_25_reserved - (reserved)
26 - ERROR_26_reserved - (reserved)
27 - ERROR_27_reserved - (reserved)
Generic errors (critical):
28 - ERROR_WRONG_ERROR_REPORT - Wrong parameters to <CO_errorReport()> function.
29 - ERROR_ISR_TIMER_OVERFLOW - Timer task has overflowed.

2A - ERROR_MEMORY_ALLOCATION_ERROR - Unable to allocate memory for objects.
2B - ERROR_GENERIC_ERROR - Generic error, test usage.
2C - ERROR_MAIN_TIMER_OVERFLOW - Mainline function exceeded maximum execution time.
2D - ERROR_INTERNAL_STATE_APPL - Error in application software internal state.
2E - ERROR_2E_reserved - (reserved)
2F - ERROR_2F_reserved - (reserved)

0x2101 CAN node ID

The object allows the user to set the CAN Node ID of the Node, the change takes effect at next power cycle.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	ro	no	no	3

This object can only be changed through the rotary switch (if available) and AMP configuration software.

0x2102 CAN bit rate

The object allows the user to set the CAN bit rate of the Node, the change takes effect at next power cycle.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	no	no	0

This object can only be changed through the rotary switch (if available) and AMP configuration software.

0x2103 SYNC counter

The object contains the counts of the SYNC message transmits/receives.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

SYNC Counter is incremented each time, SYNC message is received or transmitted.

0x2104 SYNC time

Contains the time information between every adjacent two SYNC message.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	no	no	0

SYNC Time is incremented each timer period and reset to zero, each time SYNC is received or transmitted.

0x3000 switch value (for factory use only)

The object addresses the rotary switch, such as NodeID and BitRate switch settings information. (There are no such switches in IP65 compatible CANopen drivers).

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	no	no	0

0x4000 UserRegisters

This object provide user 23 general purpose registers. They are volatile, so the information sent there will not be saved after a power cycle.

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS	Default value
Array	24	INTEGER32	rw	yes	no	0

Subindex	Name	Default value
0x00	max sub-index	23
0x01	userRegister0	0
0x02	userRegister1	0
0x03	userRegister2	0
0x04	userRegister3	0
0x05	userRegister4	0
0x06	userRegister5	0
0x07	userRegister6	0
0x08	userRegister7	0
0x09	userRegister8	0
0x0A	userRegister9	0
0x0B	userRegister10	0
0x0C	userRegister11	0
0x0D	userRegister12	0
0x0E	userRegister13	0
0x0F	userRegister14	0
0x10	userRegister15	0
0x11	userRegister16	0
0x12	userRegister17	0
0x13	userRegister18	0
0x14	userRegister19	0
0x15	userRegister20	0
0x16	userRegister21	0
0x17	userRegister22	0

0x5000 PositionGain_Tuning

This object shall configure the proportional Gain in Position loop to step-servo drive. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5001 PositionDeriGain_Tuning

This object shall configure the Derivative Gain in Position loop to step-servo drive. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5002 PositionDeriFilter_Tuning

This object provides a very simple single-pole low pass filter that is used to limit this high frequency noise and make the system quieter and more stable.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5003 VelocityGain_Tuning

This object shall configure the proportional Gain in Velocity loop to step-servo drive. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5004 VelocityIntegGain_Tuning

This object shall configure the Integral Gain in Velocity loop to step-servo drive. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5005 AccFeedForward_Tuning

This object adds a feed forward acceleration/deceleration to the torque command to speed the system's response. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5006 PIDFilter_Tuning

This object provide a torque command over-all filter at the end of Velocity loop. The filter is a very simple single-pole low pass filter that is used to limit the high frequency response of the Velocity and therefore the Position control loops. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5007~0x500E NotchFilterx_Tuning[3]

These eight objects shall configure the NotchFilter parameters in torque mode. This object is only available on step-servo driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER16	rw	no	no	0

***Note3:** x should be from A to H

0x500F PositionError

This object shall indicate the configured range of tolerated position values symmetrically to the position demand value. If the position actual value is out of the following error window, a following error (Position Limit) occurs.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

This object is the same functionally as the standard object 0x6065.

0x5010 Velocity_MAX

This object shall configure the max velocity can be allowed in the driver. This value have no effect in Profile Position Mode.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

0x5011 SmoothFilter

This object provides a filter to change in the position command to cause a smoother movement of the motor.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

Please note that a lower filter value will result in much smoother motion but will cause a lag in response.

0x5012 Driver_Temperature

This object contains the information of driver’s temperature.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER16	ro	no	no	0

The unit of this object is 0.1 centigrade.

Device Profile

The objects described in this section are parameters for all motion control modes, including profile position mode, profile velocity mode and also Homing mode.

0x603F DSP Error Code

The DSP error code captures the DSP alarm code of the last error that occurred in the drive.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	yes	no	0

Each bit in DSP Error code indicates one type of alarm or fault status.

(see the detail described in Appendix C)

0x6040 ControlWord

This object is used to control the state and motion control of the drive. It can be used to enable / disable the driver power output, start, and abort moves in all operating modes, and clear fault conditions.

The bits of the controlword are defined as follows:

LSB			MSB					
15 11	10 9	8	7	6 4	3	2	1	0
Manufacturer Specific	Reserved	Halt	Fault Reset	Operation Mode Specific	Enable Operation	Quick Stop	Enable Voltage	Switch On
O	O	O	M	O	M	M	M	M

Table 12: Controlword Data Description

Command	Bit of Controlword(6040h)				
	Fault Reset (Bit7)	Enable Operation (Bit 3)	Quick Stop (Bit 2)	Enable Voltage (Bit 1)	Switch On (Bit 0)
Shutdown	0	×	1	1	0
Switch On	0	0	1	1	1
Switch On	0	1	1	1	1
Disable Voltage	0	×	×	0	×
Quick Stop	0	×	0	1	×
Disable Operation	0	0	1	1	1
Enable Operation	0	1	1	1	1
Fault Reset		×	×	×	×

Table 13: Device Control Command Bit Patterns (Bits 0-3 and 7)

Here is the Command Structure:

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	wo	yes	yes	0

0x6041 StatusWord

The read only object “statusword” indicates the current status of the drive. Each bit is mapped as follows:

MSB								LSB							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High-Byte								L-Byte							

Bit	Name	Mandatory
0	Ready to Switch On	√
1	Switch On	√
2	Operation Enabled	√
3	Fault	√
4	Voltage Disabled	√
5	Quick Stop	√
6	Switch On Disabled	√
7	Temperature	
8	Manufacturer Specific	
9	Remote	√
10	Target Reached	√
11	Internal Limit Active	√
12	Operation Mode Specific	
13	Operation Mode Specific	
14	Manufacturer Specific	
15	Manufacturer Specific	

Table 14: Bits in the statusword

The following bits indicate the status of the device:

State	Bit 6 Switch On Disable	Bit 5 Quick Stop	Bit 3 Fault	Bit 2 Operation Enabled	Bit 1 Switch On	Bit 0 Ready to Switch On
Not Ready to Switch On	0	X	0	0	0	0
Switch On Disabled	1	X	0	0	0	0
Ready to Switch On	0	1	0	0	0	1
Switched On	0	1	0	0	1	1
Operation Enabled	0	1	0	1	1	1
Fault	0	X	1	1	1	1
Fault Reaction Active	0	X	1	1	1	1
Quick Stop Active	0	0	0	1	1	1

Table 15: Device State Bits

Bits marked × are irrelevant for the state. Other bit combinations are not allowed

Bit 9: Remote

If bit 9 is set, then parameters may be modified via the CAN-network, and the drive executes the content of a command message. If the remote bit is reset, then the drive is in local mode and will not execute the command message. The drive may transmit messages containing valid actual values like a position_actual_value, depending on the actual drive configuration. The drive will accept accesses via service data objects (SDOs) in local mode.

Bit 10: Target Reached

If bit 10 is set by the drive, then a setpoint has been reached (torque, speed or position depending on the modes_of_operation). The change of a target value by software alters this bit. If quickstop_option_code is 5, 6, 7 or 8, this bit must be set, when the quick stop operation is finished and the drive is halted. If Halt occured and the drive has halted then this bit is set too.

Bit 11: Internal Limit Active

When this bit is set, the drive indicates that an internal limitation is active (e.g. position_range_limit).

Here is the Command Structure.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	yes	yes	0

Global Control Word and Status Word

Byte #	Bit #	Control Word					Status Word				
		Homing	Position	Velocity	Torque	Q	Homing	Position	Velocity	Torque	Q
1	15	Reserved					Reserved				
	14										
	13						Homing Error	Following Error	Reserved	Reserved	Reserved
	12						Homing Attained	Set Point Ack	Speed	Reserved	Reserved
	11						Internal Limit Active				
	10						Target Reached				
	9	Reserved	Change of Set Point	Reserved	Reserved	Reserved	Remote				
	8	Halt					Reserved				
2	7	Fault Reset					Warming				
	6	Reserved	Abs/Rel	Reserved	Reserved	Reserved	Switch On Disabled				
	5	Reserved	Change Set Point Immediately	Reserved	Reserved	Reserved	Quick Stop				
	4	Homing Operation Start	New Set Point	Reserved	Reserved	Q Program Start	Voltage Enabled				
	3	Enable Operation					Fault				
	2	Quick Stop					Operation Enabled				
	1	Enable Voltage					Switch On				
	0	Switch On					Ready to Switch On				

The above table shows all the bits of “control word and status word” mapped in every operation mode.

0x605A Quick_stop_option_code

The parameter quick_stop_option_code determines what action should be taken if the Quick Stop Function is executed.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER16	rw	No	no	0

quick_stop_option_code	Action
-32768…-1	Manufacturer Specific
0	Disable drive function
1	Slow down on slow down ramp
2	Slow down on quick stop ramp
3	Slow down on the current limit
4	Slow down on the voltage limit
5	Slow down on slow down ramp and stay in Quick-Stop
6	Slow down on quick stop ramp and stay in Quick-Stop
7	Slow down on the current limit and stay in Quick-Stop
8	Slow down on the voltage limit and stay in Quick-Stop
9…32767	reserved

At this time, only option codes 1 and 2 of this feature are active.

0x6060 Modes_of_operation

The parameter modes_of_operation switches the actually chosen operation-mode.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER8	wo	yes	no	0

Mode of Operation	Action
-2	SYNC Q Mode (manufacturer specific)
-1	Normal Q Mode (manufacturer specific)
1	Profile Position Mode
3	Profile Velocity Mode
4	Torque Profile Mode (Servo only)
6	Homing Mode

-1: Normal Q mode (manufacturer specific mode)

-2: SYNC Q mode (manufacturer specific mode)

Velocity Mode and Interpolated Position Mode are not supported in this CANopen driver. Also the Torque profile Mode is only supported for Servo(or Step Servo) CANopen driver.

0x6061 Modes_of_operation_display

The modes_of_operation_display shows the current mode of operation. The meaning of the returned value corresponds to that of the Modes of Operation option code (index 6060h)

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER8	ro	yes	yes	0

This value of operation mode indicates the current mode of operation.

0x6064 Position_value_calculated

This object represents the calculated value of the position at the current moment in user defined units.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER32	ro	yes	yes	0

0x6065 following_error_window

This object shall indicate the configured range of tolerated position values symmetrically to the position demand value. If the position actual value is out of the following error window, a following error (Position Limit) occurs. A following error may occur when a drive is blocked, unreachable profile velocity occurs (Jog mode should be Mode 1), or improperly tuned (Servo and Step Servo only). The value shall be given in user defined position units. If the value of the following error window is 0, the following control shall be switched off.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	no	No	0

This object is the same functionally as Manufacturer specific object 0x500F.

0x606C Velocity_value_calculated

This object represents the calculated value of the velocity at the current moment in user defined units.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER32	ro	yes	yes	0

The value reading from driver should be divided by 240 to change to rps units.
e.g.: a value of 2400 in index 0x606c, means the calculated velocity is 10rps.

0x6071 target_torque

This parameter is the input value for the torque controller in Profile Torque Mode. This object can only be accessed in Servo (or Step Servo) driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER16	rw	yes	no	0

This object parameter is related to the other torque values, such as torque_slope (index 0x6078) and torque_constant (index 0x7005). Please see Appendix G in this document for details.

0x6073 max_current

This object shall configure the max current of the drive.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	yes	no	0

The unit of this object is 0.01Amps.

0x6074 torque_demand_value

This parameter is the output value of the torque limit function (if available within the torque control and power-stage function).

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	yes	no	0

This object is only available on servo/step-servo drives.

The unit of this object is mNm.

0x6078 current_actual_value

The current actual value refers to the instantaneous current in the drive motor.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER16	rw	yes	no	0

This object is only available on servo/step-servo drivers.

The unit of this object is 0.01Amps.

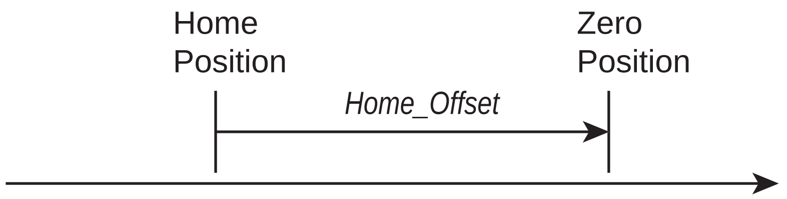
0x607A target_position

The Target Position is the position that the drive should move to in position profile mode using parameters such as velocity, acceleration, and deceleration. The target position is given in terms of Electrical Gear parameters steps per motor shaft revolution. The target position will be interpreted as absolute or relative depending on the absolute relative flag (bit 6) in the controlword.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER32	rw	yes	no	0

0x607C home_offset

The home_offset object is the difference between the zero position for the application and the machine home position (found during homing). During homing the home position is found and, once the homing is completed, the zero position is offset from the home position by adding the home_offset to the home position. All subsequent absolute moves shall be taken relative to this new zero position. This is illustrated in the following diagram.



Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER32	rw	yes	No	0

0x607E polarity

This object contains two individual bits to set the polarity of position and velocity in their own mode. Bit7 (position polarity bit) indicates the position demand value (related to the target_postition object whose index is 0x607A) shall be multiplied by 1 or by –1. The polarity flag shall have no influence on the homing mode. Bit6 (velocity polarity bit) indicates that the velocity demand value (related to the profile velocity whose index is 0x60FF) shall be multiplied by 1 or -1.

Bit	Meaning
7	Position polarity 0=> multiply by 1(default) 1=> multiply by -1
6	Velocity polarity 0=> multiply by 1(default) 1=> multiply by -1

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	rw	yes	No	0

The position polarity bit shall be used only for profile position (pp) mode. The velocity polarity bit shall be used only for profile velocity (pv) mode.

0x607F max_profile_speed

This object configures the maximum speed allowed in either direction in a move profile.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

This object is given the same unit as the objects velocity_value_calculated (index 0x606C) and profile_velocity (0x60FF).

0x6081 P2P_profile_velocity

This object shall configure the velocity normally attained at the end of the acceleration ramp during a profiled move and is valid for both directions of motion. This object sets the velocity value except the velocity parameter in Profile velocity mode (pv).

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

This object is given the same unit as the objects velocity_value_calculated (index 0x606C) and profile_velocity (0x60FF).

0x6083 profile_acceleration

This object shall configure the acceleration ramp in a profiled move.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

The value transmitted to (or read from) the drive should be multiplied by 6 (divided by 6) to change to rps/s unit.

e.g.: To set the acceleration to 100 rps/s, the value transmit to the drive should be 600.

0x6084 profile_deceleration

This object shall configure the deceleration ramp in a profiled move.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

The value transmitted to (or read from) the drive should be multiplied by 6 (divided by 6) to change to rps/s unit.

e.g.: To set the deceleration to 100 rps/s. the value transmitted to the drive should be 600.

0x6085 quick_stop_deceleration

This object configures deceleration used to stop the motor when the quick stop function is activated and the quick stop code object (605Ah) is set to 2.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

The value transmitted to (or read from) the driver should be multiplied by 6 (divided by 6) to change to rps/s unit.

e.g.: To set the quick_stop_deceleration to 1000 rps/s, the value transmitted to the drive should be 6000.

0x6087 torque_slop

This parameter describes the rate of change of torque in units of per thousand of rated torque per second. The units should be Nm/s.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

0x6098 homing_method

This object determines the method that will be used during homing.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER8	rw	yes	No	0

Please see details described in Appendix F of this document.

0x6099 homing_speed

This object determines the speeds that will be used during homing.

There are two parts to define these speeds.

Sub-index 1 to set the speed to search home switch.

Sub-index 2 to set the speed to search zero position.

Object Type	Sub Number	Data Type	Access Type	PDO Mapping	COS	Default value
Array	3	UNSIGNED32	rw	yes	no	0

Subindex	Name	Default value
00	max sub-index	2
01	speed_during_search_for_switch	0
02	speed_during_search_for_zero	0

0x609A homing_acceleration

This object establishes the acceleration to be used for all accelerations and decelerations with the standard homing modes.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	rw	yes	No	0

The units are the same as profile_acceleration/deceleration objects.

0x60FE digital_outputs

This object sets or monitors the digital outputs for drivers. Use sub-index 01 to close one or more outputs by setting the corresponding bit(s) high (1). To open one or more outputs, set the corresponding bit(s) low (0). State of output cannot be changed if it is assigned to a special function, such as Alarm Ouput, Brake Output, etc.

Object Type	Sub Number
Record	2

Subindex	Name	Data Type	Access Type	PDO Mapping	COS	Default value
00	max sub-index	UNSIGNED8	ro	no	no	2
01	physical_outputs	UNSIGNED32	rw	yes	no	0

Bit0-bit3: maximum 4 outputs per drive.

Bit0: output 1

Bit1: output 2

Bit2: output 3

Bit3: output 4

Bit4-31: reserved.

0x60FF profile_velocity

This object configures the velocity parameters in Profile Velocity Mode.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER32	rw	yes	no	0

This object has the same units as the object velocity_value_calculated (index 0x606C).

0x6502 supported_drive_modes

This object provides information on the supported drive modes.

Bit number	Description
0	Profile Position Mode
1	Velocity Mode
2	Profile Velocity Mode
3	Profile Torque Mode
4	reserved
5	Homing Mode
6	Interpolated Position Mode
7	reserved
8	reserved
9	reserved
10…15	reserved
16…31	Manufacturer sepcific

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED32	ro	no	no	0

The supported modes in Applied Motion Products CANopen drives are:

Bit0: Profile Position Mode

Bit2: Profile Velocity Mode

Bit3: Profile Torque Mode (Servo/ Step-Servo only)

Bit5; Homing Mode

0x7001 home_switch

This object shall configure the input number of the Home switch used for Homing.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	rw	yes	no	3

0x7002 idle_current

This object configures/monitors the motor holding current of the device in idle mode.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	yes	no	3

The units of this object is current in Amps multiplied by 100.

e.g.: If the value of this object read from the drive is 210, it means the idle current at this moment is 2.1Amps.

0x7003 drive_inputs

This object contains the information about drive's digital inputs.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	yes	yes	0

Bit0: input 1	Bit1: input 2	Bit2: input 3	Bit3: input 4
Bit4: input 5	Bit5: input 6	Bit6: input 7	Bit7: input 8

Bit8-15: reserved

0x7005 torque_Constant

This object configures the motor's torque constant in manufacturer-specific units. The units are mNm/Amp.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	yes	0

This object is only supported in servo/step-servo drives.

0x7006 DSP_clearAlarm

This object provides a feature to clear an alarm within the drive.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	wo	no	no	0

Set this value to 01h to clear active alarms

0x7007 QSegment.

This object configures the number of Q Segments that will be executed in Q mode.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	rw	yes	no	0

0x7009 velocity_actual_value

This object provides the actual velocity value derived from the position sensor (i.e. encoder)

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	yes	yes	0

This object has the same units as the objects velocity_value_calculated (index 0x606C) and profile_velocity (0x60FF).

0x700A position_actual_value

This object represents the actual position value measured by the position sensor.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	INTEGER32	ro	yes	yes	0

0x700B DSP_statuscode

This object represents the current status code of the driver.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	yes	yes	0

Please see the details in Table 18 in Appendix C of this document.

0x700C zero_Position

This object provides a feature to zero all position parameters, such as position_actual_value (0x700A).

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED8	wo	no	no	0

Set this value to 01h to zero all position parameters.

0x700D Acceleration_Current

This object configures the current of the drive during an acceleration state.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	rw	no	no	0

This object is only available on STM-CANopen drivers.

It has the same units as the objects max_current and idle_current; value should be divided by 100.

0x700E Analog_Input1

This object contains the Analog input value in manufacturer-specific units.

Object Type	Data Type	Access Type	PDO Mapping	COS	Default value
Var	UNSIGNED16	ro	yes	no	0

The value read from 0x700E should be divided by 10 to obtain Volts.

Appendix B - Parameter Unit Scaling

The table below shows conversions from physical units to internal drive units. Use this table to scale parameters before they are sent to the drive. Units must be rounded to the nearest whole number and represented in hexadecimal. Negative numbers should be expressed using two's complement notation.

Parameter Type	Multiplier	Units
Current	0.01	A
Velocity	0.0042	rps
Acceleration	0.1667	rps/s
Distance	1	step

Table 16: Parameter Scaling Chart

Querying the Point to Point Profile Acceleration from the drive:

An SDO read from OD 6083_h returns a value of 226_h, or 550 decimal. Using the acceleration multiplier this yields an acceleration of 91.685 rps/s.

550 * 0.1667 RPS = 91.685 rps/s

Set the Point to Point Acceleration to 10 rps/s:

When setting a known accel value, divide by the acceleration multiplier to obtain the hexadecimal number.

10 rev/s^2 / 0.1667 = 59.988

Using the formula above, and rounding to the nearest whole number, results in a value of 60 decimal, or $3C_h$ to send as an SDO Write to OD 6083_h.

Set the Target Position to -2000 steps:

Because the relationship between physical steps and internal steps is one-to-one, the value -2000 can be sent to OD $607A_h$. Sending a negative number must be done in two's complement notation. To find the two's complement, subtract the value 2000 from 2^{32}, since the Target Position is a 32 bit number.

2^{32} - 2000 = 4,294,965,296 = $FFFFF830_h$

Appendix C - Response Codes

Hex Value	SV/SVAC	SSM/TSM/TXM/SS	MSST/MSSTAC	STM/SWM
0001	***Position Limit***			
0002	CCW Limit			
0004	CW Limit			
0008	***Over Temp***			
0010	***Internal Voltage***	***Internal Voltage***	***Internal Voltage***	***Internal Voltage***
0020	***Over Voltage***			
0040	Under Voltage	***Under Voltage***	Under Voltage	Under Voltage
0080	***Over Current***			
0100	***Bad Hall Sensor***	***Open Motor Winding***		
0200	***Bad Encoder***			Not Used
0400	Comm Error			
0800	Bad Flash			
1000	Wizard Failed	No Move		
2000	Current Foldback		Not Used	Not Used
4000	Blank Q Segment			
8000	No Move	Not Used		

Note: Items in ***bold italic*** represent Drive Faults, which automatically disable the motor. Use the OF command in a Q Program to branch on a Drive Fault.

Table 17: Object 603F$_h$ CiA 402 Error Codes

Hex Value	Status Code Bit Definition
0001	Motor Enabled - motor disabled is this bit = 0
0002	Sampling - for Quick Tuner
0004	Drive Fault - check alarm code
0008	In Position - motor is in position
0010	Moving - motor is moving
0020	Jogging - currently in jog mode
0040	Stopping - in the process of stopping from a stop command
0080	Waiting - for an input
0100	Saving - parameter data is being saved
0200	Alarm present - check alarm code
0400	Homing - executing an SH command
0800	Wait Time - executing a WT command
1000	Wizard running - timing wizard is running
2000	Checking encoder - timing wizard is running
4000	Q Program is running
8000	Initializing

Table 18: Object 700B$_h$ DSP Status Codes

Appendix D - Profile Position Mode

General Mode Description

Profile Position Mode is a point-to-point operating mode using set-points which consist of velocity, acceleration, deceleration, and target position. Once all these parameters have been set, the drive buffers the commands and begins executing the set-point. When using a set of set-points method, a new set-point can be sent to the drive while a previously sent set-point is still executing.

Enable Profile Position Mode

To enable the Profile Position Mode, the value 0001_h must be written to the mode of operation OD entry, located at dictionary address 6060_h. The mode of operation can be verified using OD 6061_h - mode of operation display - which is updated when the current operation mode is accepted.

Set Running Parameters

Set the distance, velocity, acceleration, and deceleration using OD entries $607A_h$, 6081_h, 6083_h, and 6084_h respectively.

Starting/Stopping Motion

After power up or node reset, the drive is in disabled state. The value 0006_h must be written to the control word OD entry, located at dictionary address 6040_h. This will put the drive into “ready to switch on” state and ready to enable drive operation. If the value 0006_h is not written to the control word first, drive operation can not be enabled.

To indicate a new set-point and start motion, toggle bit 4 by sending $001F_h$ to controlword OD entry 6040_h.

To enable drive operation, the value $001F_h$ must be written to the controlword OD entry, located at dictionary address 6040_h. This will also signal that there is a new set-point ready. The drive acknowledges the receipt of a valid set-point using bit 12 of the statusword at OD 6041_h. Because the set-point is edge-triggered, once the drive receives and processes the set-point, the new set-point of the controlword must be cleared by writing $000F_h$ to the controlword register.

While the drive is acting on a set-point, a new set-point may be entered and triggered using the new set-point. The second set-point will be received as soon as it is processed, or at the end of the previous set-point, which ever is later.

Controlword Bits

New Set-point (bit 4) - set this bit high to clock in a new set-point. Once the drive has accepted the set-point, it will respond by setting statusword bit 12 high. Controlword bit 4 should then be taken low.

Change of Set-point (bit 9) - if this bit is low, the previous set-point will be completed and the motor will come to rest before a new set-point is processed. If bit 9 is high, the motor will continue at the speed commanded by the previous set-point until it has reached the position commanded by the previous set-point, then transition to the speed of the new set-point.

Change Set-point Immediately (bit 5) - if this bit is high, the new set-point will take effect immediately. The motor speed will transition to the speed and position commanded by the new set-point.

Abs/rel (bit 6) - if this bit is high, the set-point distance is relative. For example, if the previous motor position was 10,000 steps and a new set-point is issued with a distance of 20,000, the final position will be 30,000. If bit 6 is low, the distance is absolute. If the previous motor position was 10,000 and a new set-point is issued with a distance of 20,000, the new position will be 20,000. (The distance travelled from the previous position to the new position will be 10,000 steps.) For best results, do not change this bit while the motor is moving.

Note: Two set-points can be set up, but if status bit 12 is high, then the buffer is full and another set-point will be ignored.

For more information: See CiA 402 - 2, Profile Position Mode

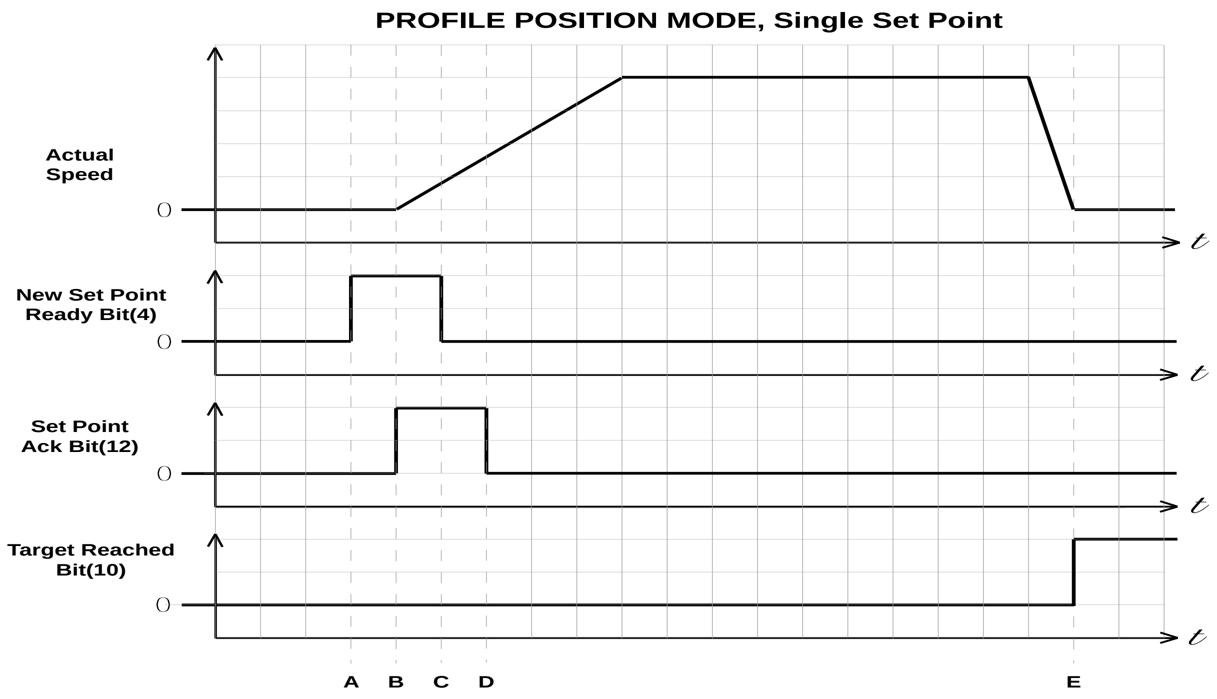


Figure 12: Single Set-Point

Graph Point	New Set-Point Ready Bit	Set-Point Acknowledge Bit	Target Reached Bit	What's Going On
Start	0	0	0	Drive waiting for set-point
A	0 -> 1	0	0	User tells drive a set-point is ready
B	1	0 -> 1	0	Drive acknowledges set-point, starts executing set-point
C	1 -> 0	1	0	User pulls new set-point ready bit low
D	0	1 -> 0	0	Drive pulls set-point ack bit low, indicating ready to receive another set-point
E	0	0	1	The set-point is finished, and the Target Reached bit is set

Table 19: Single Set-Point Profile Position Move

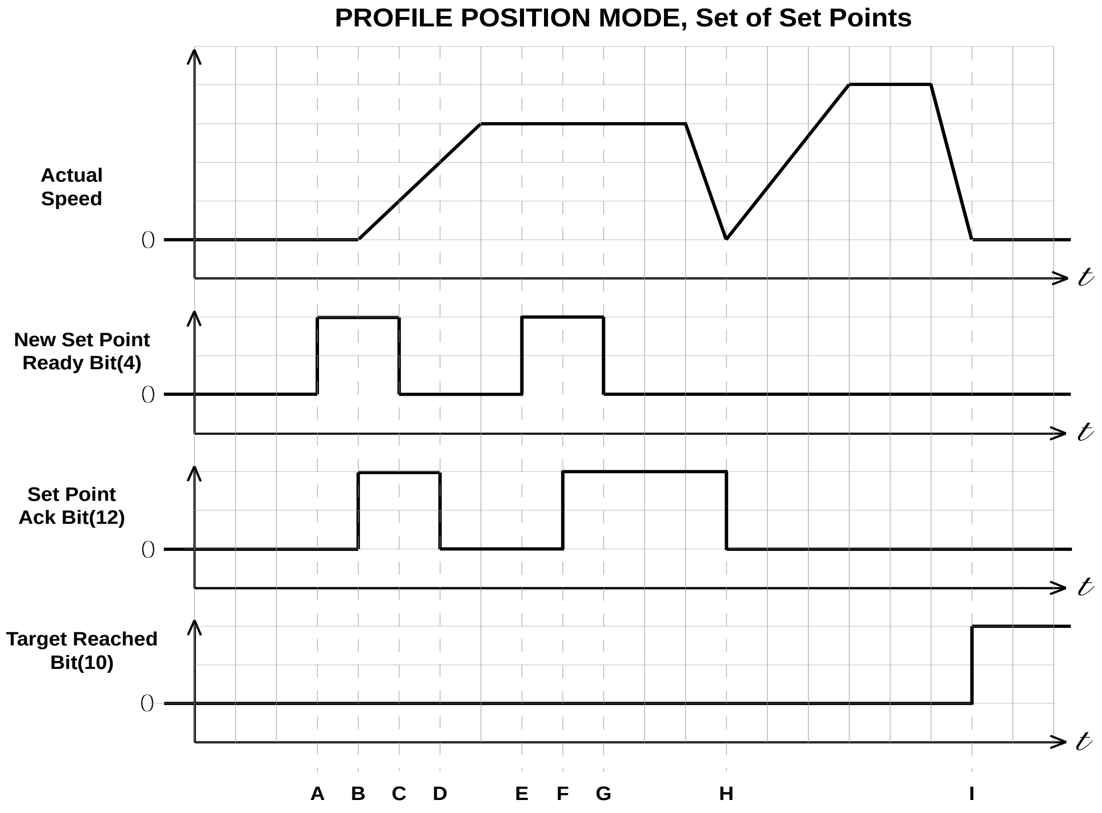


Figure 13: Multiple Set-Points, Stopping Between Moves

In this example, controlword bits 9 (Change of Set-point) and 5 (Change Set Immediately) are 0. The motor comes to rest between moves.

Graph Point	New Set-Point Ready Bit	Set-Point Acknowledge Bit	Target Reached Bit	What's Going On
Start	0	0	0	Drive waiting for set-point
A	0 -> 1	0	0	User tells drive a set-point is ready
B	1	0 -> 1	0	Drive acknowledges set-point, starts executing set-point
C	1 -> 0	1	0	User pulls new set-point ready bit low
D	0	1 -> 0	0	Drive pulls set-point ack bit low, indicat-ing ready to receive another set-point
E	0 -> 1	0	0	User tells drive another set-point is ready
F	1	0 -> 1	0	Drive acknowledges set-point, buffers it as another set-point is still in progress
G	1 -> 0	1	0	User pulls new set-point ready bit low
H	0	1 -> 0	0	Drive pulls set-point ack bit low, starts executing new set-point as soon as old one is finished
I	0	0	1	The set-point is finished, no set-points in buffer, so Target Reached bit is set

Table 20: Multiple Set-Point Profile Position Move with Stopping Between Moves

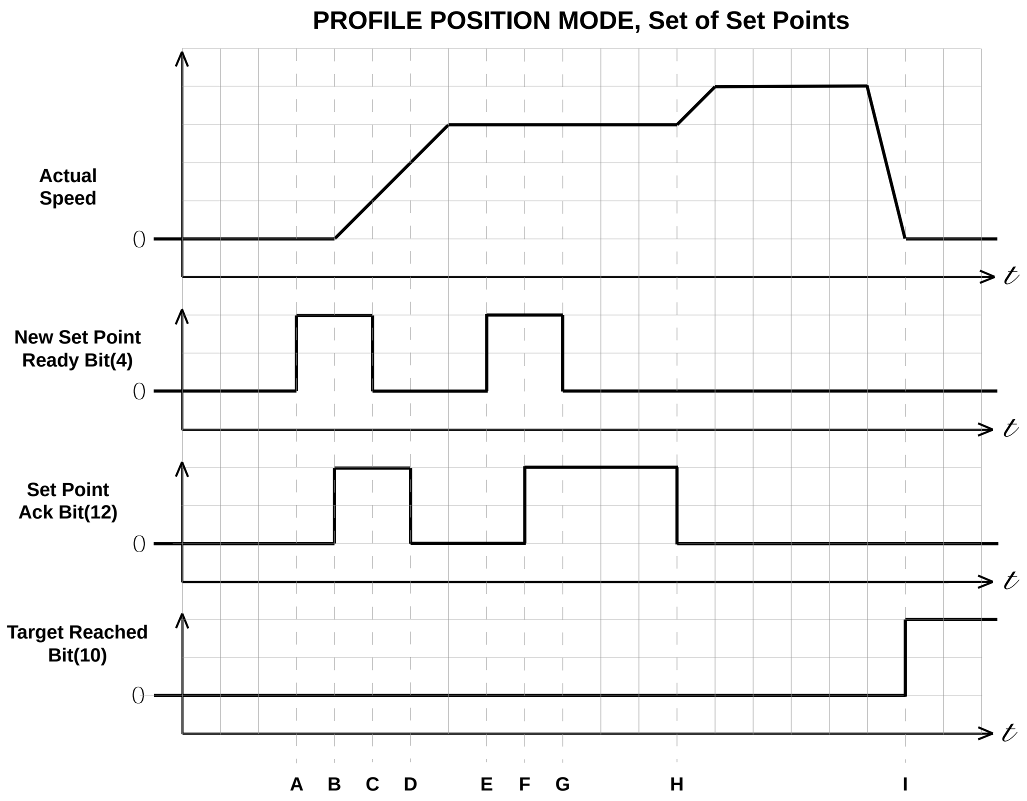


Figure 14: Multiple Set-Points, Continuous Motion

In this example, controlword bit 9 (Change of Set-point) is 1 and controlword bit 5 (Change Set Immediately) is 0. The motor continues at the speed of the first set-point until is reaches the distance of the first set-point, then changes to the new set-point speed. The motion is continuous.

Graph Point	New Set-Point Ready Bit	Set-Point Acknowledge Bit	Target Reached Bit	What's Going On
Start	0	0	0	Drive waiting for set-point
A	0 -> 1	0	0	User tells drive a set-point is ready
B	1	0 -> 1	0	Drive acknowledges set-point, starts executing set-point
C	1 -> 0	1	0	User pulls new set-point ready bit low
D	0	1 -> 0	0	Drive pulls set point ack bit low, indicating ready to receive another set-point
E	0 -> 1	0	0	User tells drive another set-point is ready
F	1	0 -> 1	0	Drive acknowledges set-point, buffers it, as another set-point is still in progress
G	1 -> 0	1	0	User pulls new set-point ready bit low
H	0	1 -> 0	0	Drive pulls set-point ack bit low, starts executing new set-point as soon as the old one is finished
I	0	0	1	The set-point is finished, no set-points in buffer, so Target Reached bit is set

Table 21: Multiple Set-Point Profile Position Move with Continuous Motion

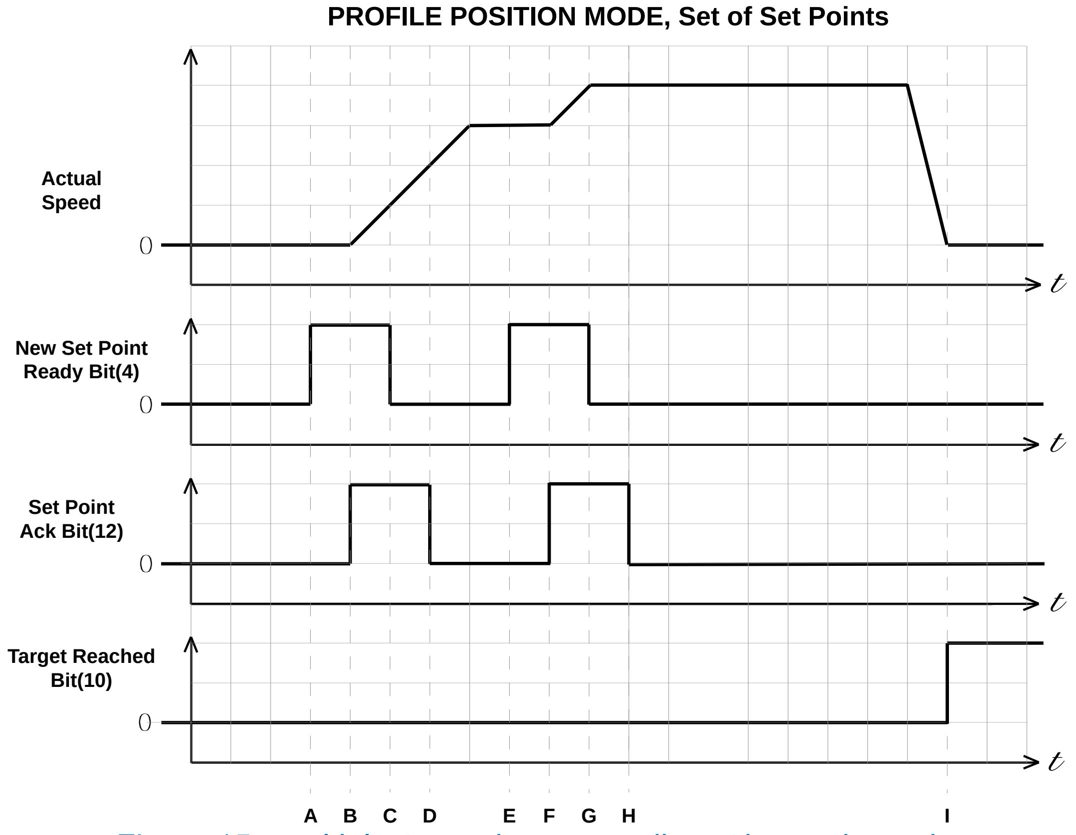


Figure 15: Multiple Set-Points, Immediate Change in Motion

In this example, controlword bit 9 (Change of Set-point) is 1 and controlword bit 5 (Change Set Immediately) is 1. The motor immediately changes to the new set-point speed without completing the first set-point. The motion is continuous.

Graph Point	New Set Point Ready Bit	Set-Point Acknowledge Bit	Target Reached Bit	What's Going On
Start	0	0	0	Drive waiting for set-point
A	0 -> 1	0	0	User tells drive a set-point is ready
B	1	0 -> 1	0	Drive acknowledges set-point, starts executing set-point
C	1 -> 0	1	0	User pulls new set-point ready bit low
D	0	1 -> 0	0	Drive pulls set point ack bit low, indicating ready to receive another set-point
E	0 -> 1	0	0	User tells drive another set-point is ready
F	1	0 -> 1	0	Drive acknowledges set-point, immediately executes it, beginning transition to new set-point speed and position
G	1 -> 0	1	0	User pulls new set-point ready bit low
H	0	1 -> 0	0	Drive pulls set-point ack bit low
I	0	0	1	The set-point is finished, no set-points in buffer, so Target Reached bit is set

Table 22: Multi-Set-Point Profile Position Move with Immediate Change in Motion

Appendix E - Profile Velocity Mode

General Mode Description

Profile Velocity Mode is a relatively simple operating mode. Once the velocity, acceleration, and deceleration are set, the drive will either command the motor to accelerate to the running velocity according to the acceleration parameter, or to halt movement according to the deceleration parameter.

The figure below shows an example of Profile Velocity Mode. The top graph shows the actual speed of the motor, the middle graph the target speed value, and the bottom graph the halt bit in the controlword.

The table below explains how the halt bit and target velocity may be used together to affect motor speed. Between points B and C, the motor does not come to a complete stop, but decelerates according to the profile deceleration value starting at point B. When the halt bit transitions at point C, it accelerates immediately back to the target speed. At Point E, reducing the target speed to zero has the same effect as enabling the halt bit, since the drive is commanding the motor to move at zero speed.

It should be noted that both enabling the halt bit and setting the target velocity to zero keep torque applied to the motor. In order to allow the shaft to move freely, the NMT state must be put in the Drive Disabled state.

Enable Profile Velocity Mode

To enable the profile velocity mode, the value 0003_h must be written to the mode of operation OD entry, located at dictionary address 6060_h.

The mode of operation can be verified using OD 6061_h - mode of operation display - which is updated when the current operation mode is accepted.

Set Running Parameters

Set the velocity, acceleration, and deceleration using OD entries $60FF_h$, 6083_h, and 6084_h respectively.

Enable Drive Operation

After power up or node reset, the drive is in disabled state. The value 0006_h must be written to the control word OD entry, located at dictionary address 6040_h. This will put the drive into “ready to switch on” state and is ready to enable drive operation. If the value 0006_h is not written to the control word first, the drive operation can not be enabled.

To enable drive operation, the value $010F_h$ must be written to the controlword OD entry, located at dictionary address 6040_h. This puts the drive into Operation Enabled state, with the motion halted.

Starting/Stopping Motion

To start and stop motion, toggle the controlword halt bit (bit 8). When the halt bit is set to 0, motion will start or continue; when the halt bit is set to 1, motion will stop. The bit can be toggled by writing $010F_h$ and $000F_h$ to controlword OD entry 6040_h.

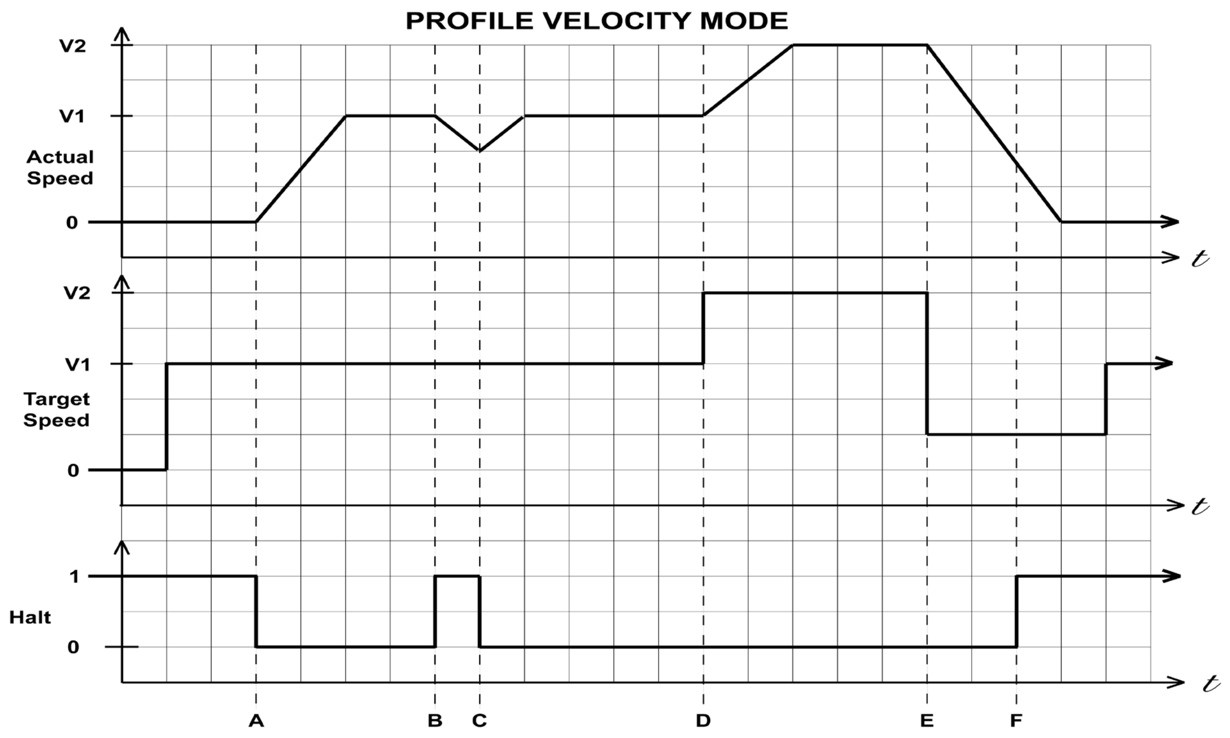


Figure 16: Profile Velocity Mode

Graph Point	Target Speed	Halt Bit	Drive command to Motor
Start	0	1	Motor stopped
A	V1	1 -> 0	Motor accelerates to speed V1
B	V1	0 -> 1	Motor decelerates to stopped
C	V1	1 -> 0	Motor accelerates to V1
D	V1 -> V2	0	Motor accelerates from V1 to V2
E	V2 -> 0	0	Motor decelerates from V2 to 0
F	0	0 -> 1	Motor remains stopped
G	0 -> V1	1	Motor remains stopped

Table 23: Profile Velocity Mode Example

Appendix F - Homing Methods

Set Running Parameters

Set the homing and index velocities, acceleration/deceleration, offset and home sensor (if required) using OD entries 6099_h, $609A_h$, $607C_h$, and 7001_h respectively.

Note: It is important that the limit switch settings have been defined in ST Configurator or Quick Tuner prior to using the CANopen Homing Mode.

Enable Homing Mode

To enable Homing Mode, the value 0006_h must be written to the mode of operation OD entry, located at dictionary address 6060_h. The mode of operation can be verified using OD 6061_h - mode of operation display - which is updated when the current operation mode is accepted.

After power up or node reset, the drive is in disabled state. The value 0006_h must be written to the control word OD entry, located at dictionary address 6040_h. This will put the drive into “ready to switch on” state and is ready to enable drive operation. If the value 0006_h is not written to the control word first, the drive operation can not be enabled.

To put the drive into Operation Enabled Mode, write $000F_h$ to the controlword OD entry, located at dictionary address 6040_h.

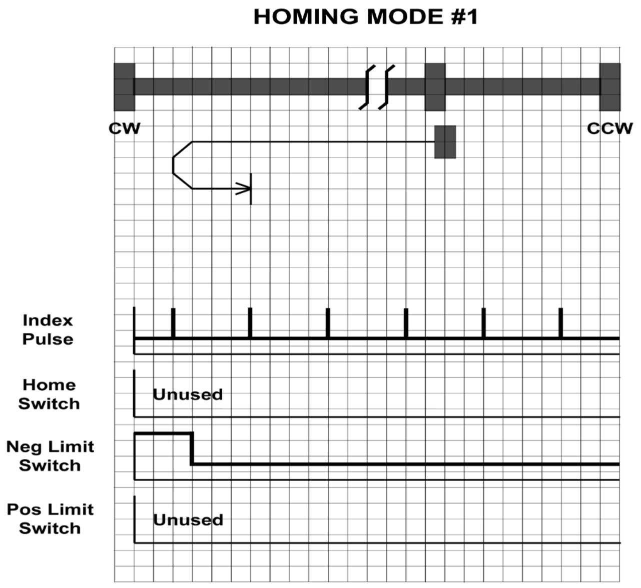
Starting the Homing Procedure

Set the Homing Method required using OD entry 6098_h. To start the homing procedure, bit 4 of the controlword OD entry located at dictionary address 6040_h, must transition from 0 to 1. The status of the homing procedure can be monitored using the statusword OD entry 6041_h.

Homing Method Diagrams

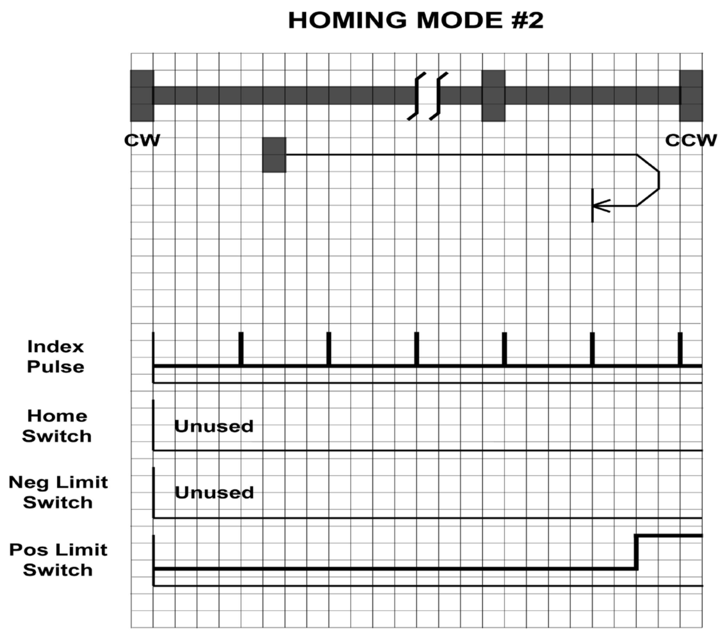
Homing Method 1

Homes to the first index CCW after the CW limit switch is reached.



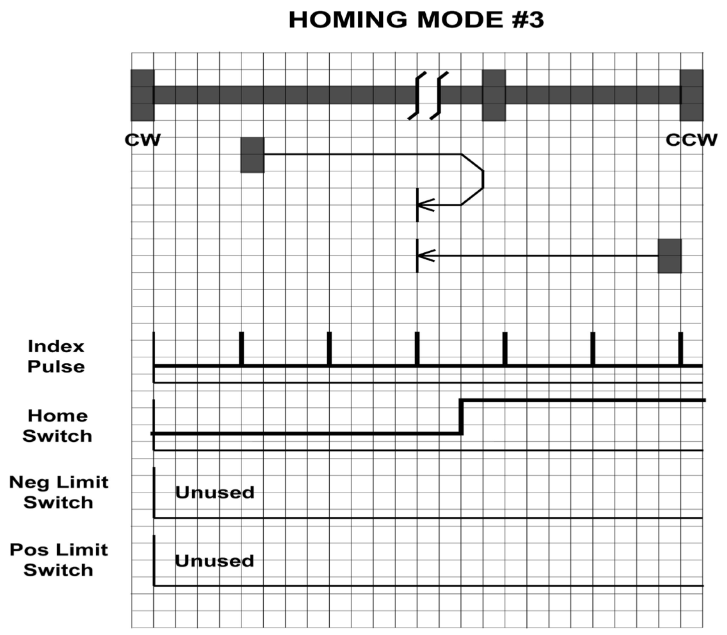
Homing Method 2

Homes to the first index CW after the CCW limit switch is reached.



Homing Method 3

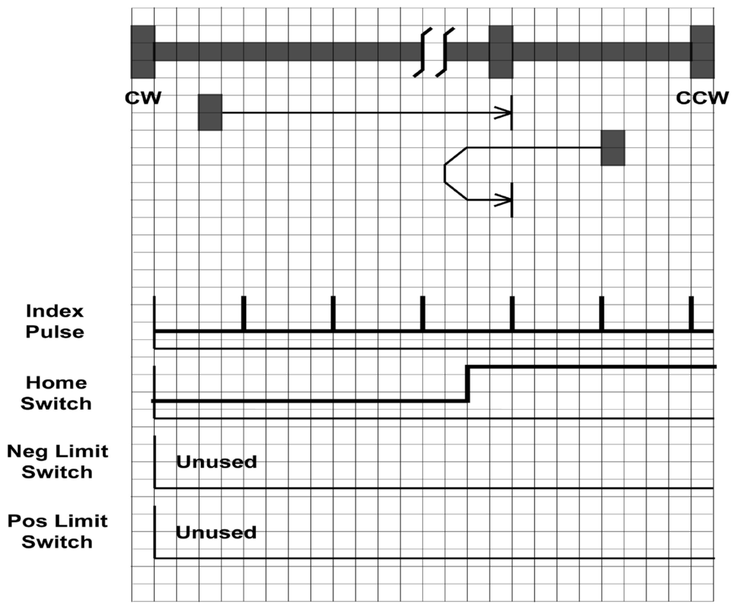
Homes to the first index CW after the positive home switch changes state; the initial direction of motion is dependent on the state of the home switch.



Homing Method 4

Homes to the first index CCW after the positive home switch changes state; the initial direction of motion is dependent on the state of the home switch.

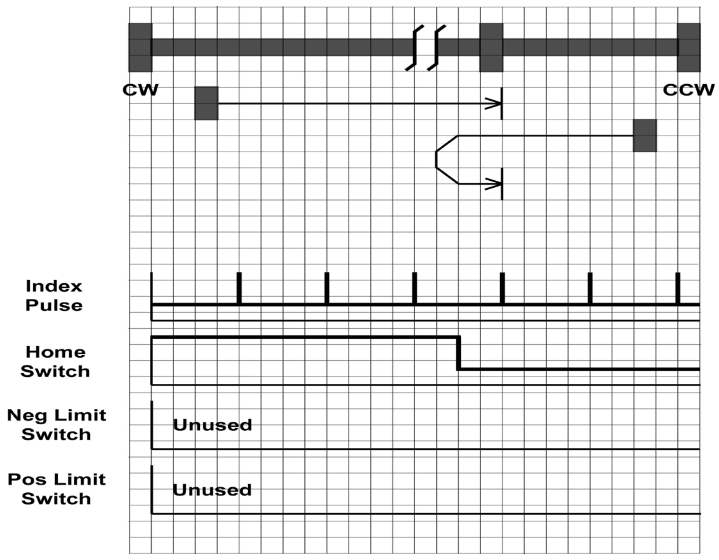
HOMING MODE #4



Homing Method 5

Homes to the first index CCW after the negative home switch changes state; the initial direction of motion is dependent on the state of the home switch.

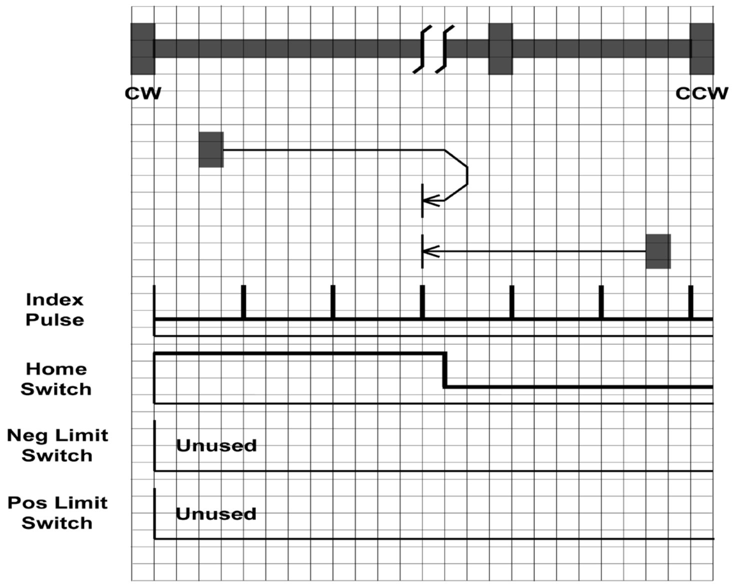
HOMING MODE #5



Homing Method 6

Homes to the first index CW after the negative home switch changes state; the initial direction of motion is dependent on the state of the home switch.

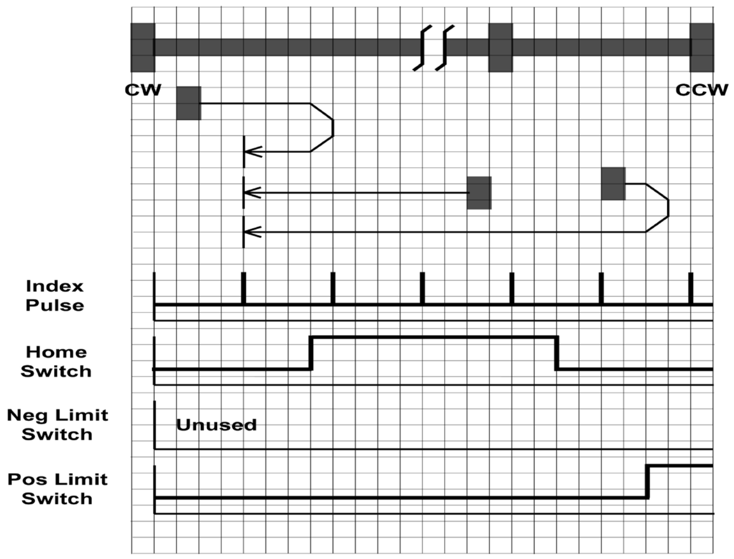
HOMING MODE #6



Homing Method 7

Starts moving CCW (or CW if the home switch is active), and homes to the first index CW of the home switch transition.

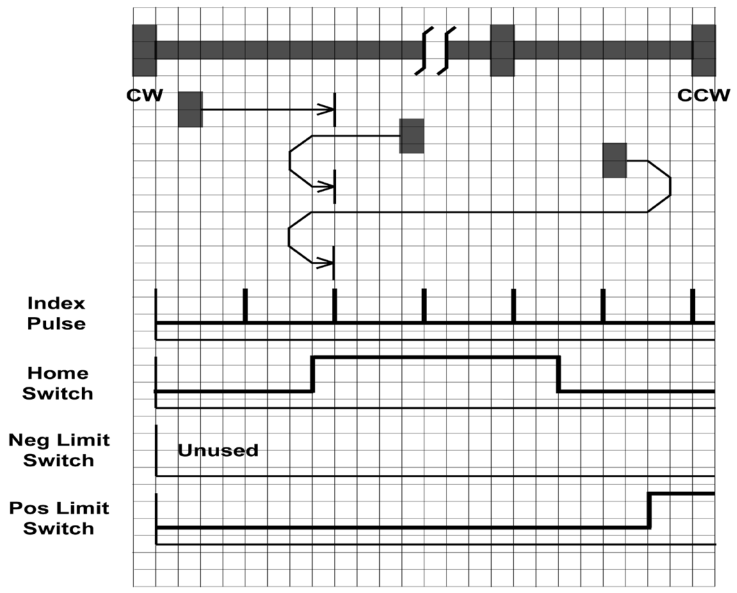
HOMING MODE #7



Homing Method 8

Starts moving CCW (or CW if the home switch is active), and homes to the first index CCW of the home switch transition.

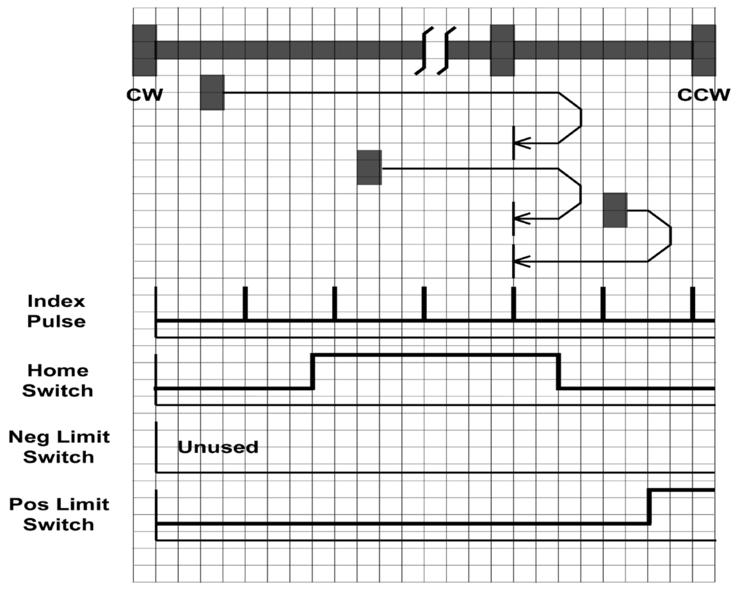
HOMING MODE #8



Homing Method 9

Starts moving CCW and homes to the first index CW of the home switch transition.

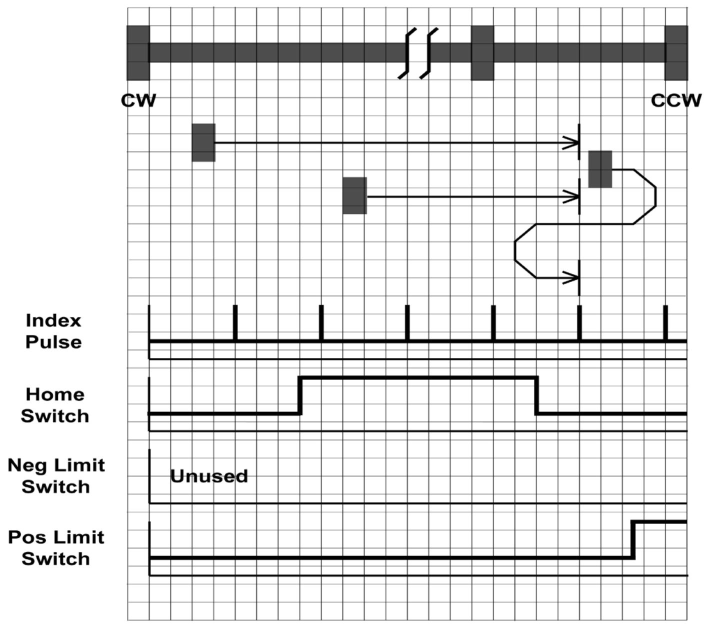
HOMING MODE #9



Homing Method 10

Starts moving CCW and homes to the first index CCW of the home switch transition.

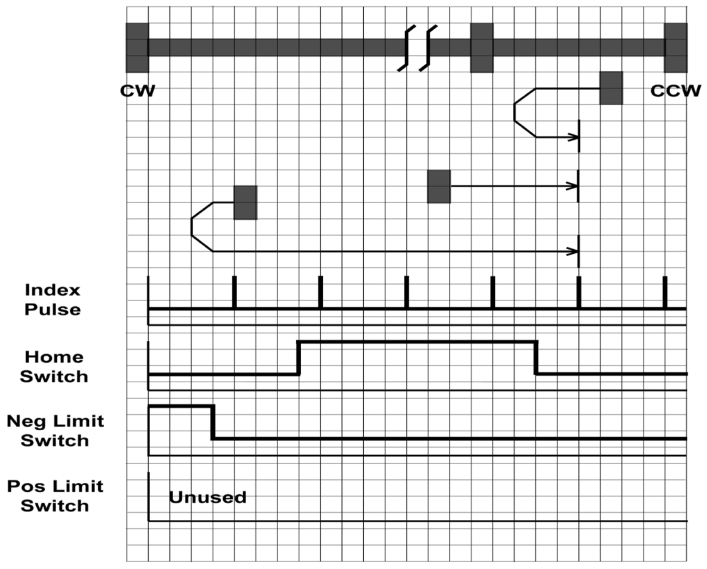
HOMING MODE #10



Homing Method 11

Starts moving CW (or CCW if the home switch is active), and homes to the first index CCW of the home switch transition.

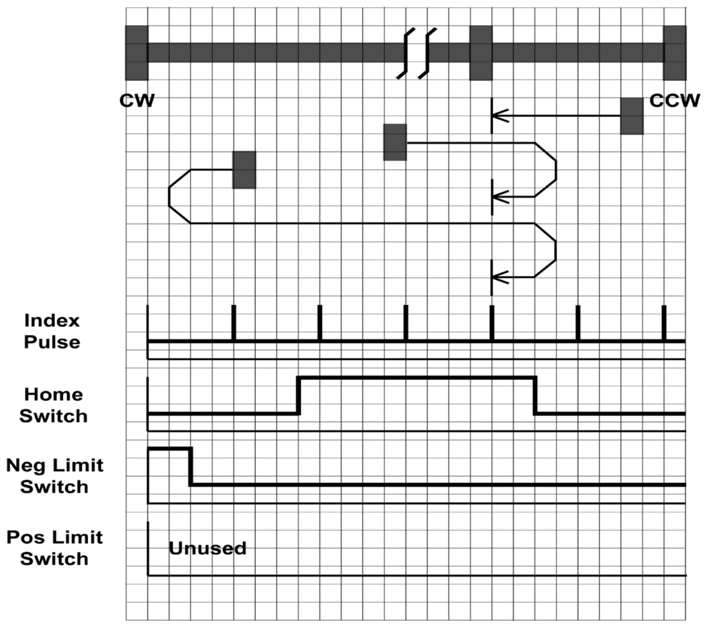
HOMING MODE #11



Homing Method 12

Starts moving CW (or CCW if the home switch is active), and homes to the first index CW of the home switch transition.

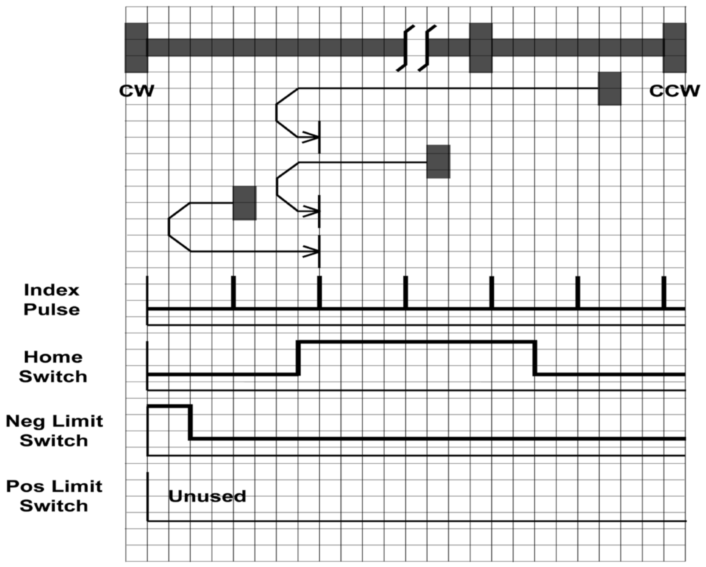
HOMING MODE #12



Homing Method 13

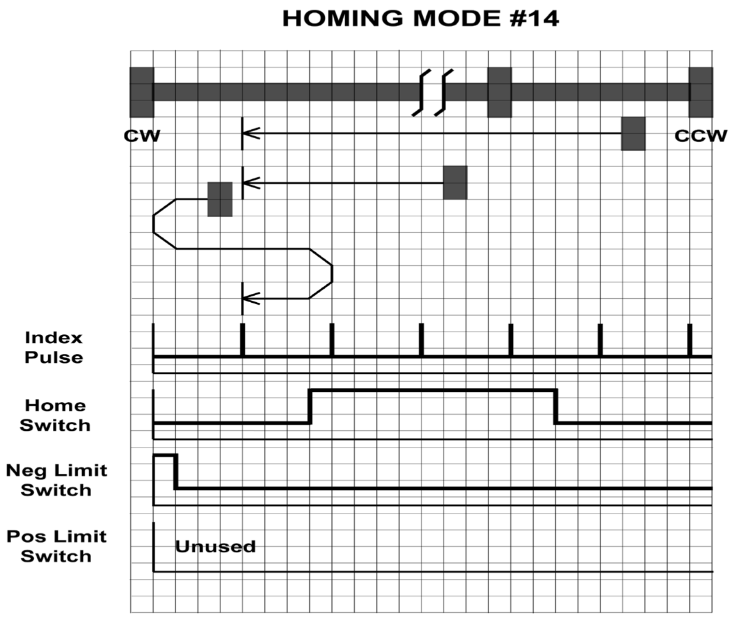
Starts moving CW and homes to the first index CCW of the home switch transition.

HOMING MODE #13



Homing Method 14

Starts moving CW and homes to the first index CW of the home switch transition shown above.

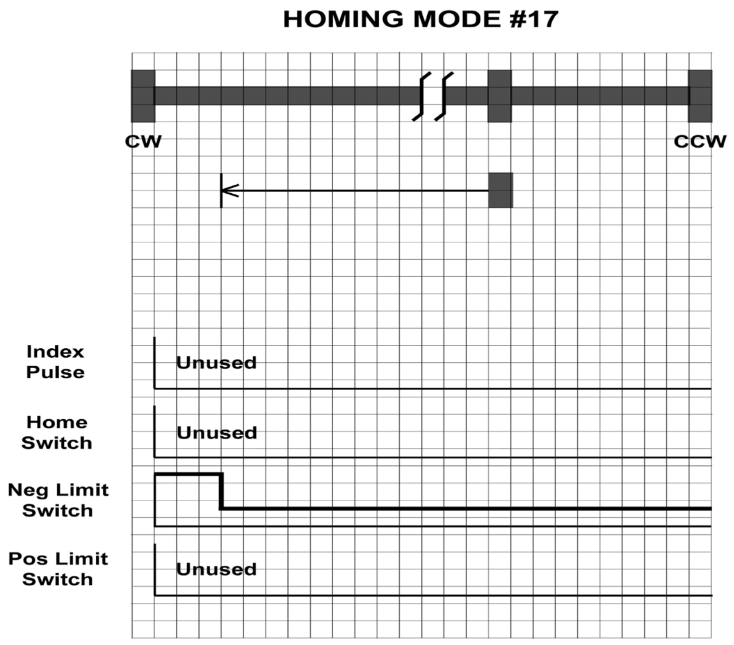


Homing Methods 15 and 16

Homing Methods 15 and 16 are reserved for future expansion.

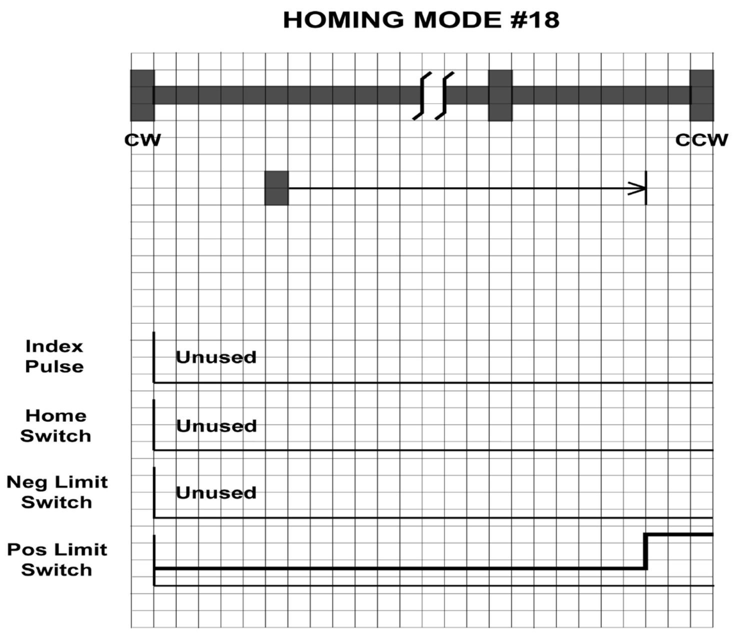
Homing Method 17

Homes to the CW limit switch.



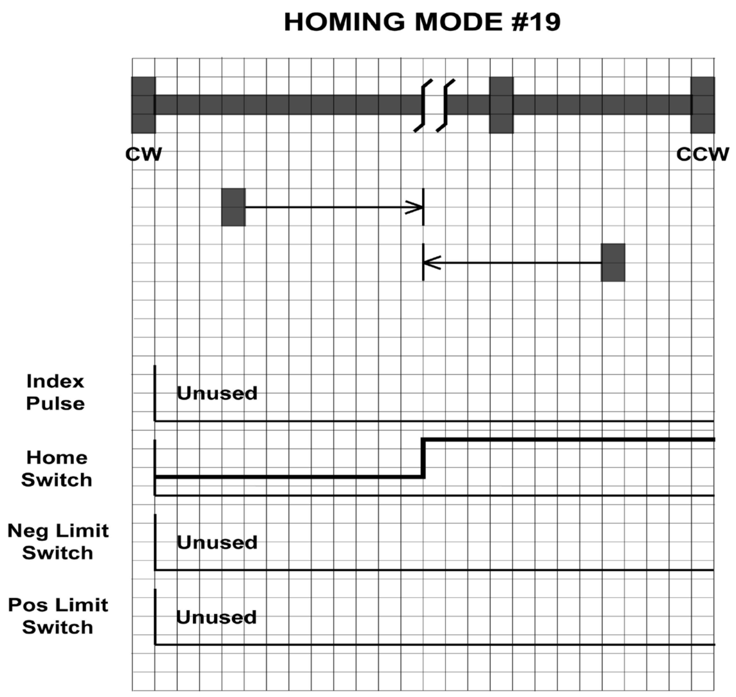
Homing Method 18

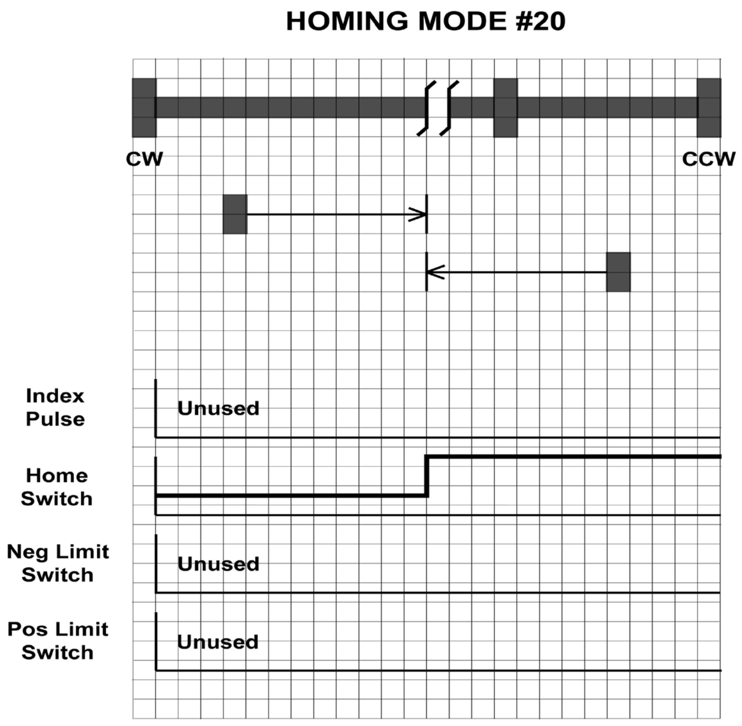
Homes to the CCW limit switch.



Homing Methods 19 and 20

Home to the home switch transition.

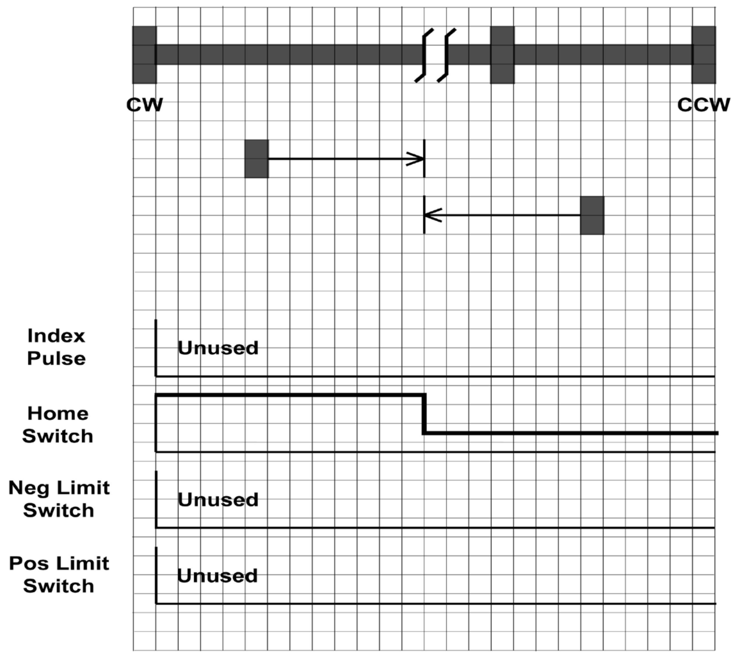




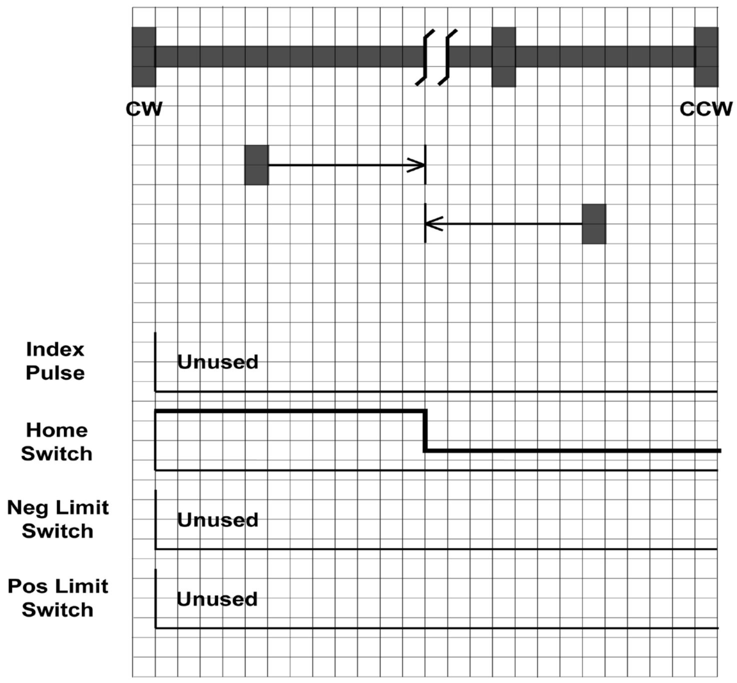
Homing Methods 21 and 22

Home to the home switch transition.

HOMING MODE #21



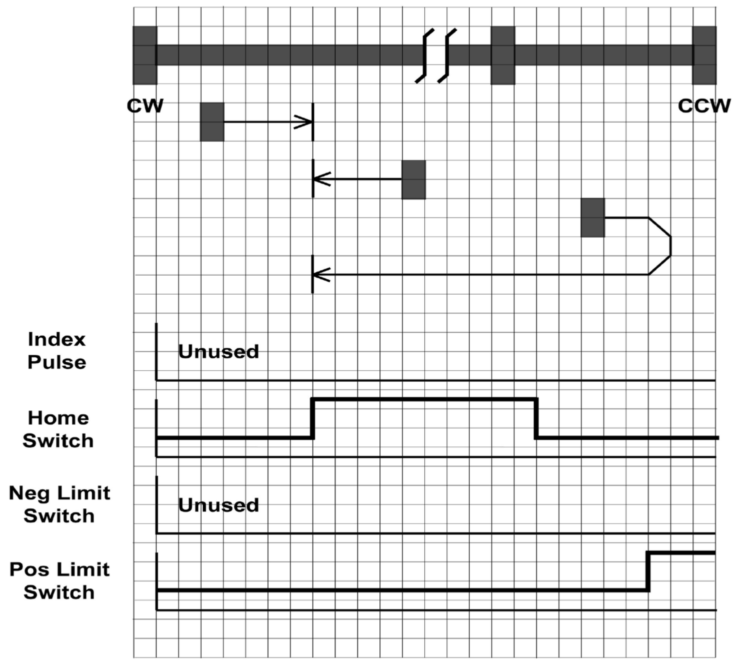
HOMING MODE #22



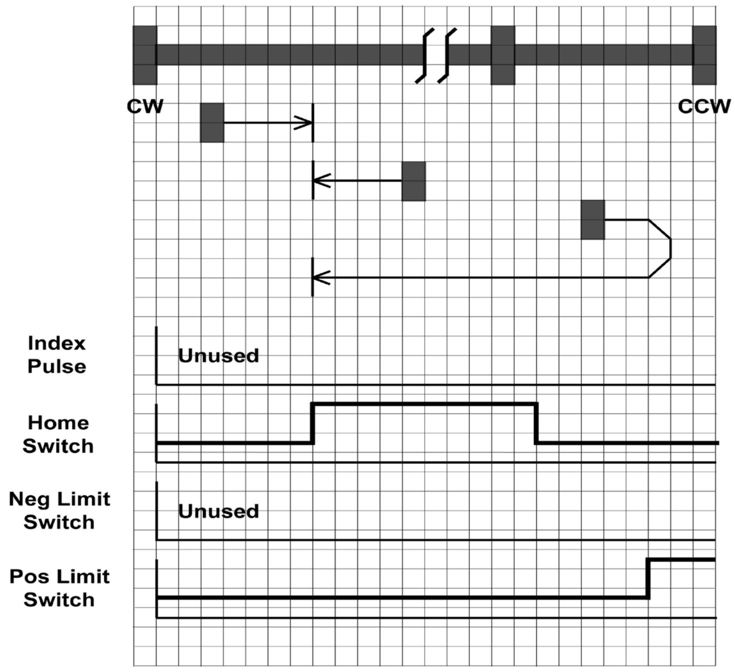
Homing Methods 23 and 24

Home to the home switch transition shown below, and “bounce off” the CCW limit, if required.

HOMING MODE #23



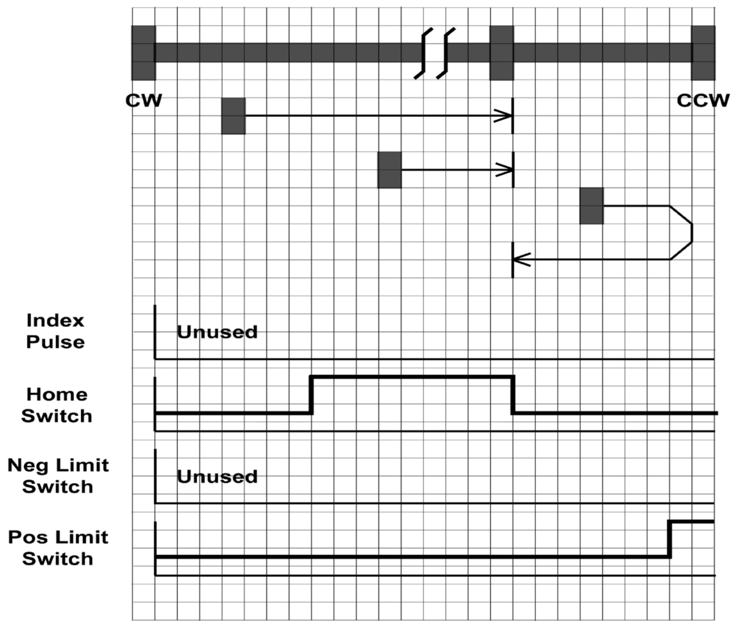
HOMING MODE #24



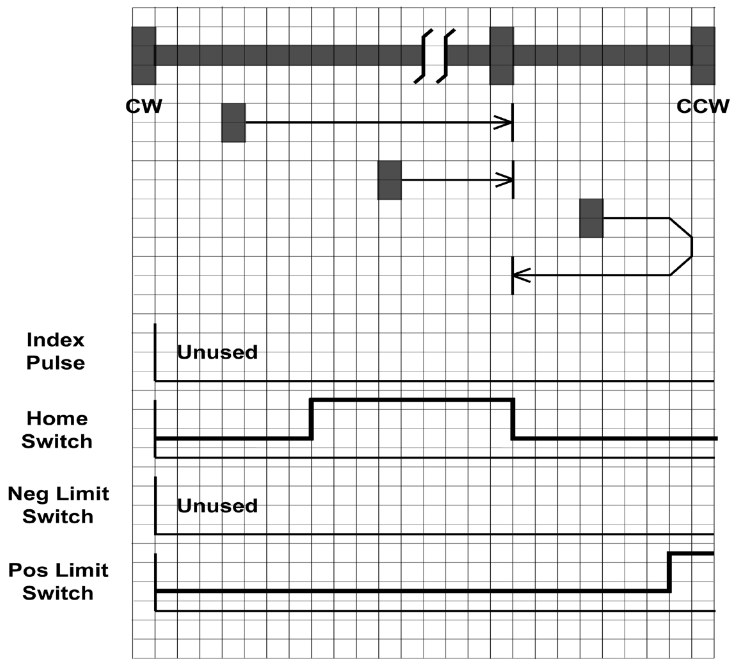
Homing Methods 25 and 26

Home to the home switch transition shown below, and “bounce off” the CCW limit, if required.

HOMING MODE #25



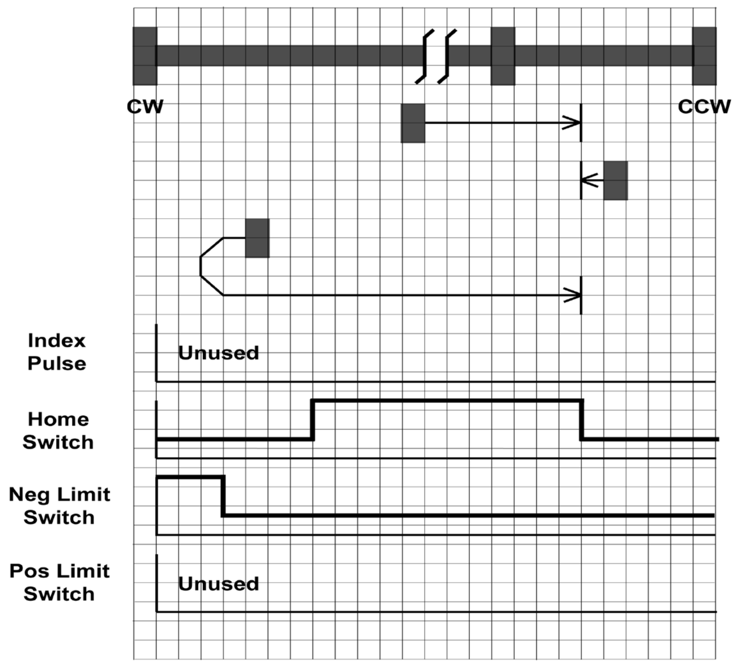
HOMING MODE #26



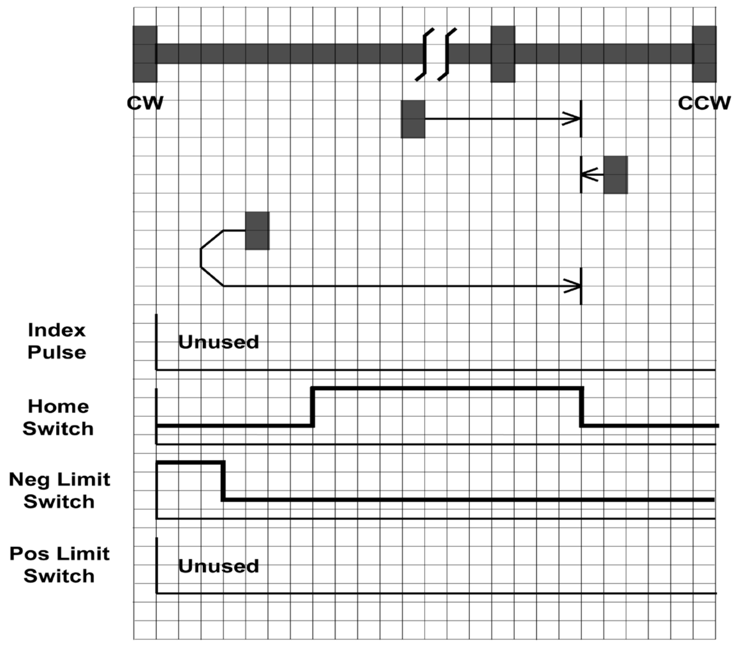
Homing Methods 27 and 28

Home to the home switch transition shown below, and “bounce off” the CW limit, if required.

HOMING MODE #27



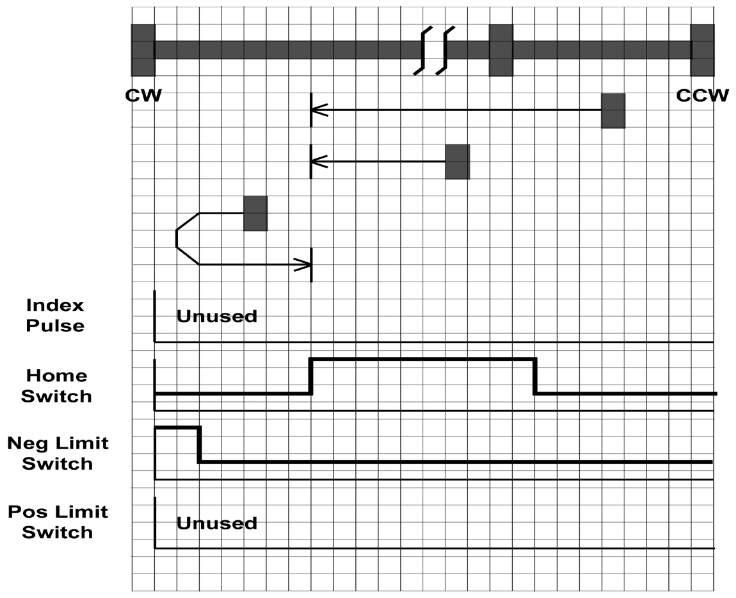
HOMING MODE #28



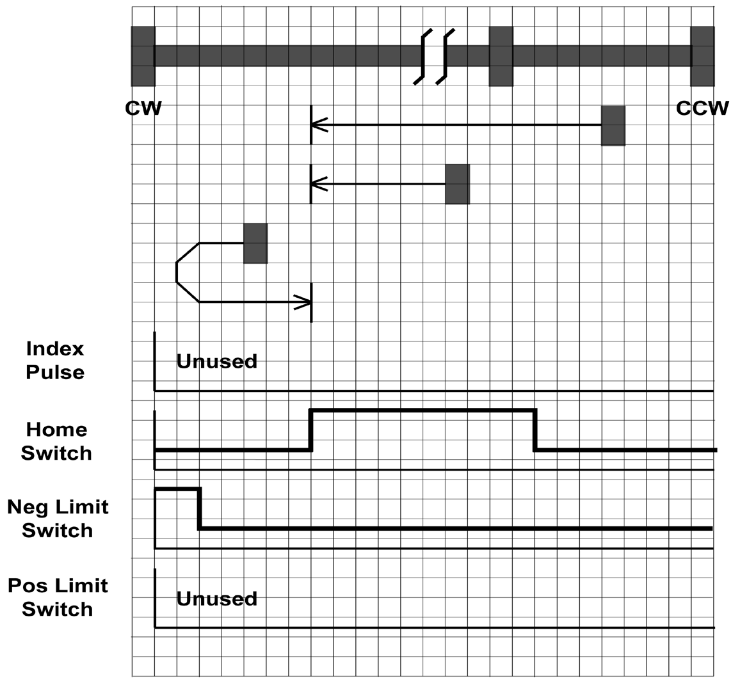
Homing Methods 29 and 30

Home to the home switch transition shown below, and “bounce off” the CW limit, if required.

HOMING MODE #29



HOMING MODE #30



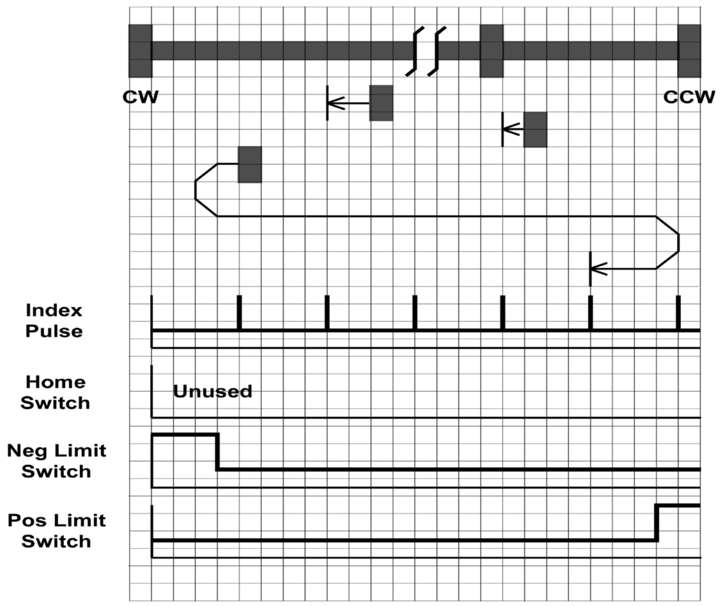
Homing Methods 31 and 32

Homing Methods 31 and 32 are reserved for future expansion.

Homing Method 33

Homes to the next index pulse CW from the current position. If the CW limit is hit, the drive resets to the CCW limit, and continues searching for a limit in the CW direction.

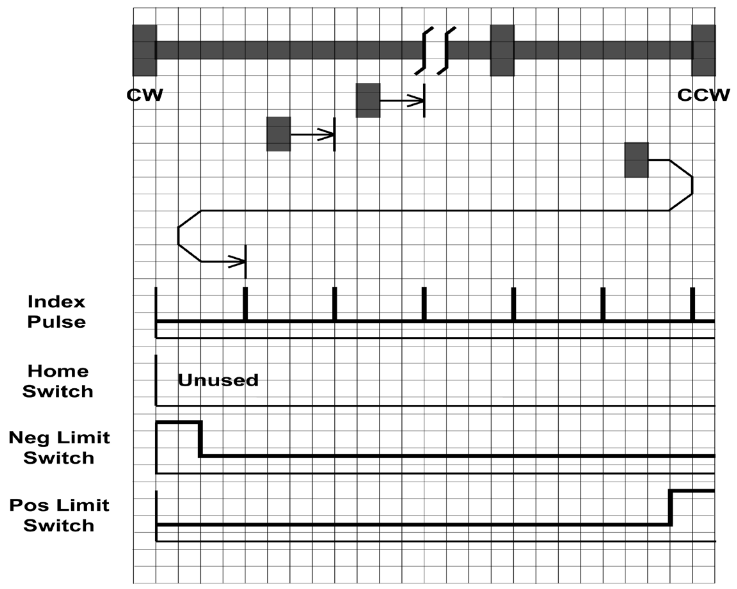
HOMING MODE #33



Homing Method 34

Homes to the next index pulse CCW from the current position. If the CCW limit is hit, the drive resets to the CW limit, and continues searching for a limit in the CCW direction.

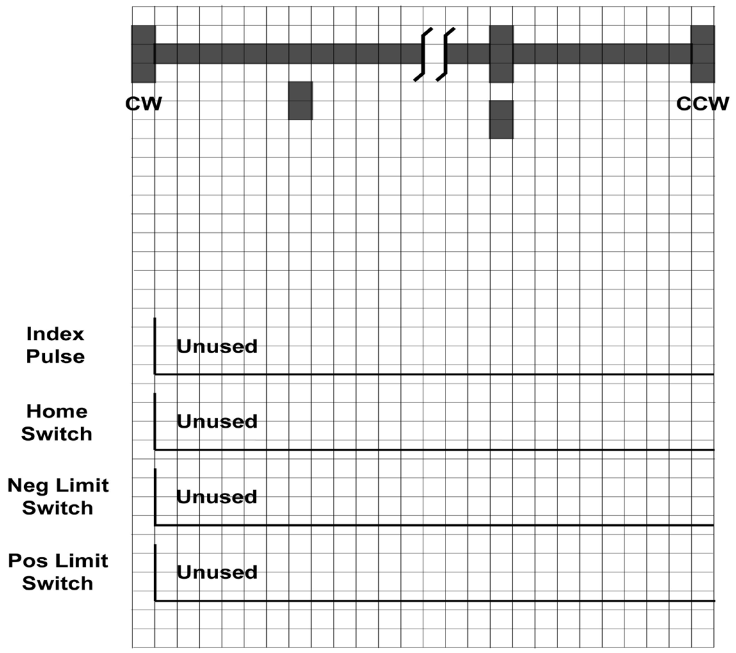
HOMING MODE #34



Homing Method 35

Takes the current position to be the home position; the Home Offset value is ignored, and the motor does not move at all.

HOMING MODE #35



Appendix G - Profile Torque Mode (Servo Only)

General Mode Description

Profile Torque mode is a servo-control torque operating mode. It requires knowledge of the Torque Constant of the motor in Nm/A. This information can be found in the motor print.

Enable Profile Torque Mode

To enable Profile Torque Mode, the value 0004_h must be written to the mode of operation OD entry, located at dictionary address 6060_h.

The mode of operation can be verified using OD 6061_h - mode of operation display - which is updated when the current operation mode is accepted.

Set Running Parameters

To operate in Profile Torque mode, the following parameters must be set:

Parameter Name	Object Dictionary Entry	Length (in bytes)	Units	Description
Torque Constant	7005_h	2	$\frac{m \bullet Nm}{a}$	Motor paramter, found on the motor print
Target Torque	6071_h	2	m • Nm	Torque to be applied to the motor
Torque Slope	6087_h	4	$\frac{m \bullet Nm}{sec}$	Rate at which to ramp torque to new target

Figure 17: Set Running Parameters

Enable Drive Operation

After power up or node reset, the drive is in disabled state. The value 0006_h must be written to the control word OD entry, located at dictionary address 6040_h. This will put the drive into “ready to switch on” state and is ready to enable drive operation. If the value 0006_h is not written to the control word first, the drive operation can not be enabled.

To enable drive operation, the value $000F_h$ must be written to the controlword OD entry, located at dictionary address 6040_h. This puts the drive into the Operation Enabled state with no torque applied.

It should be noted that both enabling the halt bit and setting the target torque to zero will ramp down the torque applied to the motor according to the torque slope. At the end of the slope no torque will be applied to the motor, allowing the shaft to move freely.

Starting/Stopping Torque

To start and stop motion, toggle the controlword halt bit (bit 8). When the halt bit is set to 0, motion will start or continue; when the halt bit is set to 1, motion will stop. The bit can be toggled by writing $010F_h$ and $000F_h$ to controlword OD entry 6040_h.

Parameter Calculations - Example

An application requires a torque of 50 oz-in. and a torque slope of 25 oz-in/sec. The motor print lists the Torque Constant of the motor as 0.07Nm/A. The Nm/A constant given must first be converted into mNm/A, as required by the Torque Constant OD entry. The formula used for this is:

$$0.07\ \frac{\text{Nm}}{\text{A}} \bullet 1000\ \frac{\text{mA}}{\text{A}} = 70\ \frac{\text{m} \bullet \text{Nm}}{\text{A}}$$

As the drive works primarily in Nm, the desired 50 oz-in of torque must also be converted into Nm, using the conversion factor 141.6 oz-in/Nm.

$$141.6\ \frac{\frac{\text{50 oz-in}}{\text{oz-in}}}{\text{Nm}} = 0.3531\ \text{Nm}$$

Now, the resultant torque of 0.3531Nm must be converted into mNm, as required by the Target Torque OD entry.

$$0.3531\ \text{Nm} \bullet 1000\ \frac{\text{m}\bullet\text{Nm}}{\text{Nm}} = 353.1\ \text{m}\bullet\text{Nm}$$

The result is a value of 353 mNm, rounded to the nearest whole number, for the Target Torque OD Entry.

Finally, the desired slope must be converted from the given units of oz-in/sec into the required units of mNm/sec.

$$\left(25\ \frac{\text{oz-in}}{\text{sec}}\right) * \left(\frac{\text{1 Nm}}{\text{141.6 oz-in}}\right) * \left(\frac{\text{1000 m}\bullet\text{Nm}}{\text{1 Nm}}\right) = 176.55\ \frac{\text{m}\bullet\text{Nm}}{\text{sec}}$$

Rounding to the nearest whole number results in a Torque Slope of 177 mNm/sec.

Current Verification - Example

It is important to check that the current required of the drive is within the limits of the servo amplifier. The drive being used, for example, has a continuous rating of 7 amps, and a peak current of 14 amps, which may be held continuously for 2 seconds. This means that a current of 7 amps can be held indefinitely, and currents between 7 and 14 amps may be used in short bursts.

Using the target torque and torque constant from the example above the current draw can be checked, as shown:

$$\frac{0.3531\ \text{Nm}}{0.07\ \frac{\text{Nm}}{\text{A}}} = 5.0443\ \text{A}$$

The resultant current, 5.0443A, is below the 7A continuous current rating of the drive, and well below the peak current rating of 14A. It is possible for the drive to maintain a current of 7A indefinitely, and peak up to 14A for up to two seconds continuously. Values between 7A and 14A may be held proportionally long.

For more information: see CiA 402 - 2, Profile Torque Mode

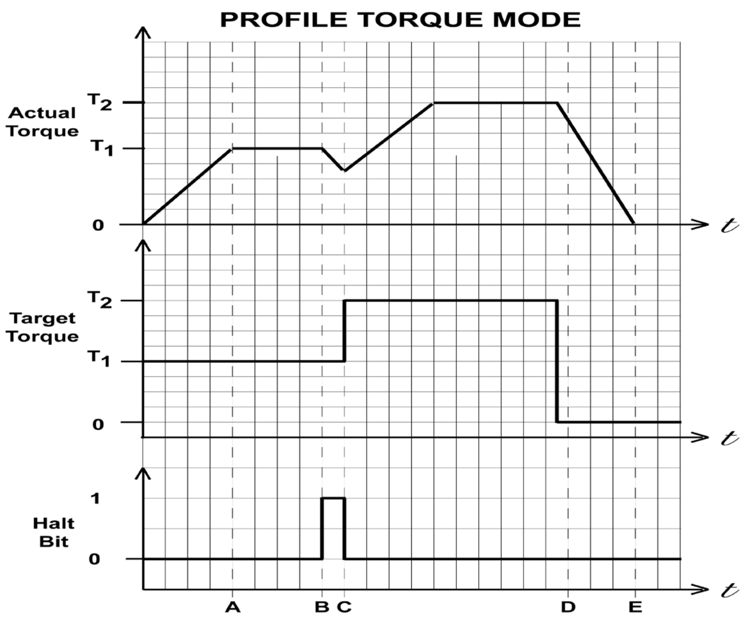


Figure 18: Profile Torque Mode

Graph Point	Target Torque	Halt Bit	Drive command to Motor
Start	T1	0	Ramp torque to T1
A	T1	0	Maintain torque at T1
B	T1	0 -> 1	Ramp torque to zero
C	T1 -> T2	1 -> 0	Ramp torque to T2
D	T2 -> 0	0	Ramp torque to zero
E	0	0	Maintain torque at zero

Table 24: Profile Torque Mode Example

Appendix H - Q Program Mode

General Mode Description

In order to expand the functionality of Applied Motion Products CANopen drives, the Q programming language may be used to execute complex motion profiles that may not be possible within the scope of CiA 402. The Q program must be written and pre-loaded into the CANopen drive using Q Programmer (v1.3.5 or later).

Q Programs may also access and manipulate the CANopen General Purpose registers for use in stored programs. The section above on General Purpose Registers has a chart to convert from the OD entry to the Q address. See object 0x4000.

Loading a Q Program

As with ST Configurator and Quick Tuner, where the drive must be powered up with the RS-232 port connected, Q Programmer must be running for the CANopen drive to delay the normal boot-up procedure. The CAN boot-up will be resumed by closing the Q Programmer application, or by power-cycling the drive with the RS-232 port disconnected. Once Q Programmer is in control of the drive, it may be used in the same way as any other Applied Motion Products Q drive. See the Q Command Reference for more information on Q programming.

Normal Q Program Execution

To execute a stored Q program on a single drive, a value of -1 (FF_h) must be written to the mode of operation OD entry, located at dictionary address 6060_h. The mode of operation can be verified using OD entry 6061_h - mode of operation display - which is updated when the current operation mode is accepted.

Next, the desired Q segment number, 1-12, must be written to the Q Segment Number register, located at address 7007_h.

After power up or node reset, the drive is in disabled state. The value 0006_h must be written to the control word OD entry, located at dictionary address 6040_h. This will put the drive into “ready to switch on” state and is ready to enable drive operation. If the value 0006_h is not written to the control word first, the drive operation can not be enabled.

To enable drive operation, a value of $000F_h$ must be written to the controlword OD entry, located at dictionary address 6040_h. This puts the drive into Operation Enabled state and ready to run the Q program.

To run the selected Q program, a value of $001F_h$ must be written to the controlword. The Q program will then run to completion. The Q program may be re-executed by a 0->1 transition of the Q Program start bit (bit 4) in the controlword.

To halt execution of a Q program, the halt bit (bit 8) of the controlword must be set to 1. The Q program will halt immediately and start from the beginning the next time a 0->1 transition is seen on the Q Program start bit after the halt bit has been cleared.

Synchronous Q Program Execution

To execute a stored Q program on a single drive, a value of -2 (FE$_h$) must be written to the mode of operation OD entry, located at dictionary address 6060$_h$. The mode of operation can be verified using OD entry 6061$_h$ - mode of operation display - which is updated when the current operation mode is accepted.

Next, the desired Q segment number, 1-12, must be written to the Q Segment Number register, located at address 7007$_h$. To enable operation 001F$_h$ must be written to the controlword OD entry 6040$_h$.

After power up or node reset, the drive is in disabled state. The value 0006$_h$ must be written to the control word OD entry, located at dictionary address 6040$_h$. This will put the drive into “ready to switch on” state and is ready to enable drive operation. If the value 0006$_h$ is not written to the control word first, the drive operation can not be enabled.

To enable drive operation, a value of 000F$_h$ must be written to the controlword OD entry, located at dictionary address 6040$_h$. This puts the drive into Operation Enabled state and ready to run the Q program.

To run the drive based on the SYNC pulse, the pulse must be set in the COB-ID SYNC register located at OD entry 1005$_h$. A standard value for the SYNC pulse is 80$_h$, but any unused COB-ID may be used. Refer to CiA301 for a list of reserved COB-IDs.

Once the SYNC pulse has been set, and the desired Q segment has been set, the drive will execute the Q segment every time it receives a SYNC pulse. In this way, multiple drives may be instructed to start a Q program with a single, network-wide instruction.

To halt execution of a Q program, set the halt bit (bit 8) of the controlword to 1. The Q program will halt immediately, and start from the beginning the next time a SYNC pulse is sent after the halt bit has been cleared.

More Information

For more information: see CiA 301, COB-IDs

For more information on Q programming: see the Host Command Reference.

Appendix I - Understanding NMT States

Under normal operating conditions, a Network Management (NMT) state machine will power up into an Initialization state, send out a boot-up packet, move into a Pre-Operational state, and start sending out heartbeats with the Pre-Operational state status code.

NMT Mode	NMT Control Command	NMT Status Code (Heartbeat)
Initialization/Node Reset	129	0
Pre-Operational	128	127
Operational	1	5
Stopped	2	4

Table 25: Understanding NMT States

Building a CANopen NMT Data Frame - Example

This example will show how to send a broadcast message to all CANopen nodes to put them into an operational NMT state. The COB-ID for NMT commands is always 0. This ensures that an NMT command has the highest priority on the bus and will never be preempted, except by another node sending out an NMT command.

The first data byte of an NMT command contains the NMT Control Command, which is 1 (Operational) in this case.

The second data byte contains either the Node-ID of a target Node, or, in the event that the NMT master is requesting that all nodes change their NMT Mode, a zero. To send a broadcast message, a zero is used.The completed data frame is below.

COB ID	Data Length	Data Byte 0	Data Byte 1
0	2	1	0

Table 26: Example - NMT Data Frame

Appendix J - SDO and PDO Access

Enable SDO Use

To enable Service Data Object (SDO) use, the NMT state must be either Pre-Operational or Operational. Send an NMT message to put the node into either state. When completed, the heartbeat should return either 127 (Pre-Operational) or 5 (Operational). The drive is now ready to read and/or write all OD entries.

Building an SDO Read Data Frame - Example

This example will show how to read the heartbeat time of node $2E_h$. An SDO read request must be sent to the drive. The default COB-ID for SDO requests is 600_h (see CiA 301), plus the Node-ID of $2E_h$. This results in a specific COB-ID for this message of $62E_h$. The first data byte is reserved for the control byte, which is always 40_h for an SDO Read. The next 2 bytes are reserved for the OD entry address in Little Endian format. To obtain OD entry 1017, we stuff data byte 1 with 17_h and data byte 2 with 10_h. Data byte 3 is reserved for the sub-index of the OD entry, which in this case is zero. The last 4 bytes are unused for SDO reads. The whole message appears as:

COB-ID	Data Length	Data Byte 0	1	2	3	4	5	6	7
$62E_h$	8	40_h	17_h	10_h	00_h	00_h	00_h	00_h	00_h
						DATA BYTES			

Table 27: Enable SDO Use

The drive will respond with a message with COB-ID 580_h + Node ID, or $5AE_h$.

For more information: see CiA 301, SDO protocol

PDO Access

Enable PDO Use

To enable Process Data Object (PDO) use, the NMT state must be set to Operational and an NMT message sent to enable the Operational state. When completed, the heartbeat should return a 5. The drive is now ready to receive RPDOs, and will transmit TPDOs depending on the Transmission Type.

TPDO Transmission Types

There are several triggering options for Transmit PDOs, which are controlled by OD entries 1800_h to 1803_h, and their associated sub-entries.

Possible TPDO Triggers

SYNC pulse - Node will send TPDO after receiving one or multiple SYNC pulses

Event/Timer - Node will issue TPDO based on an internal event or timer

Remote Request - Node will send TPDO after a remote request

0	PDO transmitted on the next SYNC pulse after the statusword has changed
1	PDO transmitted on every SYNC pulse
2-240	PDO transmitted on every n SYNC pulses
254-255	PDO transmitted every time statusword changes, or the Event Timer has expired

Table 28: TPDO Transmission Types

PDO Mapping

The Applied Motion Products CANopen allows you to map objects to PDOs to allow for the transfer of application data more efficiently. By using the PDO mapping, the user can map multiple mappable Objects to a PDO, (max length for a PDO is 8 bytes). The following section describes the process of a TPDO mapped to several objects.

TPDO Index	Sub-Index	Mapped To Index	# Bytes
$1A01_h$	00_h		
$1A01_h$	01_h	6041_h	2
$1A01_h$	02_h	$700A_h$	4

Table 29: PDO Mapping

PDO Mapping Procedure

PDO Mapping Example 1: Mapping StatusWord and Position_Actual_Value to TPDO2					
Step	Action	Index	Sub	Bytes	Value
1	Return back “PreOperation“ State				
2	Turn off TPDO2	1801_h	01_h		80000280_h
3	Set the number of Mapped objects to 0	$1A01_h$	00_h		00_h
4	Map StatusWord to TPDO2 mapped object1	$1A01_h$	01_h	2	60410010_h
5	Map Position_actual_value to TPDO2 mapped object2	$1A01_h$	02_h	4	$700A0020_h$
6	Set the number of Mapped objects to 2	$1A01_h$	00_h		02_h
7	Turn On TPDO2	1801_h	01_h		00000280_h
8					
9					

Table 30: PDO Mapping Example

Default Mapping Example - Consumer PDO 2

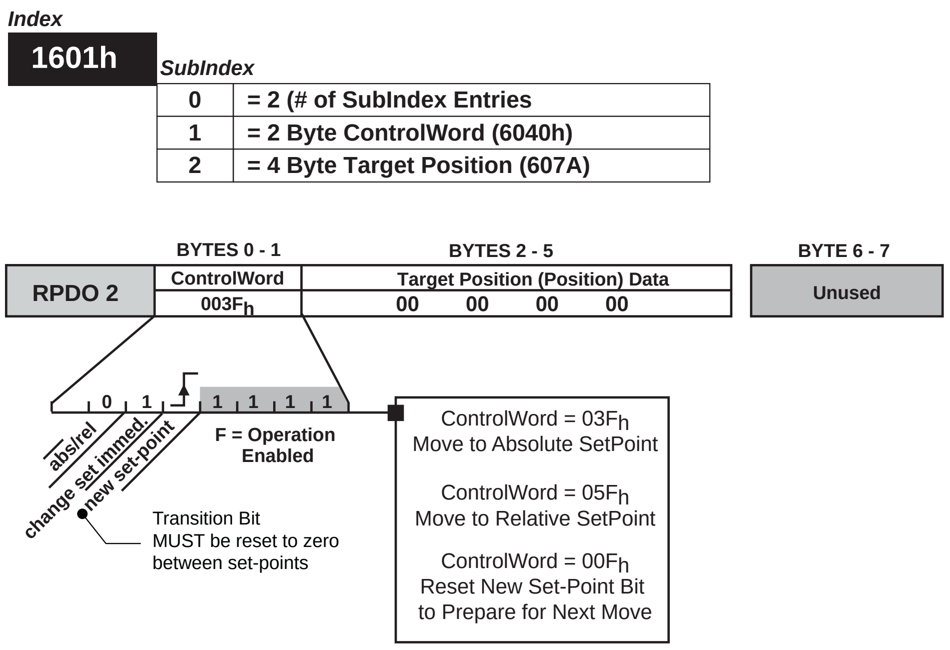

Index
1601h
SubIndex
0 = 2 (# of SubIndex Entries
1 = 2 Byte ControlWord (6040h)
2 = 4 Byte Target Position (607A)
BYTES 0 - 1
BYTES 2 - 5
BYTE 6 - 7
RPDO 2
ControlWord
003Fh
Target Position (Position) Data
00 00 00 00
Unused
0 1 1 1 1 1
abs/rel
change set immed.
new set-point
F = Operation Enabled
Transition Bit MUST be reset to zero between set-points
ControlWord = 03Fh Move to Absolute SetPoint
ControlWord = 05Fh Move to Relative SetPoint
ControlWord = 00Fh Reset New Set-Point Bit to Prepare for Next Move

Figure 19: PDO Mapping Showing the Default Mapping for RPDO2

Default PDO Mapping - Stepper

PDO Name	First Mapped Parameter	OD Entry	# Bytes	Second Mapped Parameter	OD Entry	# Bytes	# Bytes Total
TPDO1	Statusword	6041_h	2				2
TPDO2	Statusword	6041_h	2	Target Position	6064_h	4	6
TPDO3	Statusword	6041_h	2	Target Velocity	$606C_h$	2	4
TPDO4	Input Status	7003_h	1				1
RPDO1	Controlword	6040_h	2				2
RPDO2	Controlword	6040_h	2	Target Distance	$607A_h$	4	6
RPDO3	Controlword	6040_h	2	Target Velocity	$60FF_h$	2	4
RPDO4	Output State	$60FE_h$	1				1

Table 31: PDO Mapping - Stepper

Default PDO Mapping - Servo

PDO Name	First Mapped Parameter	OD Entry	# Bytes	Second Mapped Parameter	OD Entry	# Bytes	# Bytes Total
TPDO1	Statusword	6041_h	2				2
TPDO2	Statusword	6041_h	2	Actual Position	$700A_h$	4	6
TPDO3	Statusword	6041_h	2	Actual Velocity	7009_h	2	4
TPDO4	Input Status	7003_h	1				1
RPDO1	Controlword	6040_h	2				2
RPDO2	Controlword	6040_h	2	Target Distance	$607A_h$	4	6
RPDO3	Controlword	6040_h	2	Target Velocity	$60FF_h$	2	4
RPDO4	Output State	$60FE_h$	1				1

Table 32: PDO Mapping - Servo

PDO COB-ID

Because PDOs are directly mapped to OD entries, no overhead is required when working with them. RPDOs may be sent directly with the COB-ID being the default RPDO COB-ID plus the Node-ID. For example, the default RPDO1 COB-ID is 200_h. Therefore, the COB-ID for RPDO1 to Node $2E_h$ would be $200_h + 02E_h = 22E_h$. The default COB-IDs for each PDO may be found in CiA 301 on page 78.

Building an RPDO Data Frame - Example

This example will show how to set the controlword of node $2E_h$ to $7E4F_h$ using RPDO1. The paragraph above shows how the COB-ID of $22E_h$ was determined. The mapping table above that shows that the first two message bytes will contain the controlword. According to Endianness, the first data byte will be $4F_h$ and the second will be $7E_h$. The entire message will then read:

COB ID	Data Length	Data Byte 0	Data Byte 1
$22E_h$	2	$4F_h$	$7E_h$

Table 33: PDO COB-IDs

For more information on PDO mapping: see CiA 402-3

For more information on the PDO protocol: see CiA 301

Appendix K - Operation Mode Specific Control and Status Word

Control Word of Profile Position Mode

Bit #	Name	Value	Description
4	New Set Point	0	Toggle this bit from 0->1 to clock in a new set point
		1	
5	Change Set Point Immediately	0	Positioning shall be completed (target reached) before the next one gets started
		1	Next positioning shall be started immediately
6	Abs/Rel	0	Target position shall be an absolute value
		1	Target position shall be a relative value
8	Halt	0	Positioning shall be executed or continued
		1	Axis shall be stopped
9	Change of Set Point	0	The previous set-point will be completed and the motor will come to rest before a new set point is processed
		1	The motor will continue at the speed commanded by the previous set point until it has reached the position commanded by the previous set point, then transition to the speed of the new set point

Status Word of Profile Position Mode

Bit #	Name	Value	Description
10	Target Reached	0	Halt (Bit 8 in controlword) = 0: Target position not reached Halt (Bit 8 in controlword) = 1: Axis decelerates
		1	Halt (Bit 8 in controlword) = 0: Target position reached Halt (Bit 8 in controlword) = 1: Velocity of axis is 0
12	Set Point Ack	0	Previous set point already processed, waiting for new set point
		1	Previous set point still in process, set point overwriting shall be accepted
13	Following Error	0	No following error
		1	Following error

Control Word of Profile Velocity Mode

Bit #	Name	Value	Description
8	Halt	0	The motion shall be executed or continued
		1	Axis shall be stopped

Status Word of Profile Velocity Mode

Bit #	Name	Value	Description
10	Target Reached	0	Halt (Bit 8 in controlword) = 0: Target velocity not reached Halt (Bit 8 in controlword) = 1: Axis decelerates
		1	Halt (Bit 8 in controlword) = 0: Target velocity reached Halt (Bit 8 in controlword) = 1: Velocity of axis is 0
12	Speed	0	Speed is not equal to 0
		1	Speed is equal 0

Control Word of Homing Mode

Bit #	Name	Value	Description
4	Homing Operation Start	0	Do not start homing procedure
		1	Start or continue homing procedure
8	Halt	0	Enable bit 4
		1	Stop axis

Status Word of Homing Mode

Bit #	Name	Value	Description
10	Target Reached	0	Halt (Bit 8 in controlword) = 0: Home position not reached Halt (Bit 8 in controlword) = 1: Axis decelerates
		1	Halt (Bit 8 in controlword) = 0: Home position reached Halt (Bit 8 in controlword) = 1: Velocity of axis is 0
12	Homing Attained	0	Homing Mode not yet complete
		1	Homing Mode carried out successfully
13	Homing Error	0	No homing error
		1	Homing error

Control Word of Q Mode

Bit #	Name	Value	Description
4	Q Program Start	0	Toggle this bit from 0->1 to run Q program
		1	
8	Halt	0	Enable bit 4
		1	Stop axis

Status Word of Q Mode

Bit #	Name	Value	Description
10	Target Reached	0	Halt (Bit 8 in controlword) = 0: Q program running Halt (Bit 8 in controlword) = 1: Axis decelerates
		1	Halt (Bit 8 in controlword) = 0: Q program finishes or not started Halt (Bit 8 in controlword) = 1: Velocity of axis is 0

Appendix L – Example Programs

Profile Position Mode

**** Enable Motor Power - CiA 402 State Machine ****

ID	DLC	Data
$0603	$8	$2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on
$0603	$8	$2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603	$8	$2B $40 $60 $00 $0F $00 $00 $00 ‘Operation Enabled

**** Set to Profile Position Mode ****

$0603	$8	$2F $60 $60 $00 $01 $00 $00 $00 ‘Set to Profile Position Mode

**** Set Motion Parameters ****

$0603	$8	$23 $81 $60 $00 $F0 $00 $00 $00 ‘Set Profile Velocity to 1 rps
$0603	$8	$23 $83 $60 $00 $58 $02 $00 $00 ‘Set Acceleration to 100 rps/s
$0603	$8	$23 $84 $60 $00 $58 $02 $00 $00 ‘Set Deceleration to 100 rps/s

Single Move Absolute

$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603	$8	$2B $40 $60 $00 $1F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $0F $00 $00 $00 ‘Clear New Set Point Bit

Single Move Relative

$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603	$8	$2B $40 $60 $00 $5F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $4F $00 $00 $00 ‘Clear New Set Point Bit

Multiple Move, Stopping between Moves

$0603	$8	$23 $81 $60 $00 $B0 $04 $00 $00 ‘Set Profile Velocity to 5 rps
$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603	$8	$2B $40 $60 $00 $5F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $4F $00 $00 $00 ‘Clear New Set Point Bit
$0603	$8	$23 $81 $60 $00 $60 $09 $00 $00 ‘Set Profile Velocity to 10 rps
$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 600000 steps
$0603	$8	$2B $40 $60 $00 $5F $00 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $4F $00 $00 $00 ‘Clear New Set Point Bit

Multiple Move, Continuous Motion

$0603	$8	$23 $81 $60 $00 $B0 $04 $00 $00 ‘Set Profile Velocity to 5 rps
$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603	$8	$2B $40 $60 $00 $5F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $4F $02 $00 $00 ‘Clear New Set Point Bit
$0603	$8	$23 $81 $60 $00 $60 $09 $00 $00 ‘Set Profile Velocity to 10 rps
$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 600000 steps
$0603	$8	$2B $40 $60 $00 $5F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $4F $02 $00 $00 ‘Clear New Set Point Bit

Multiple Move, Immediate Change in Motion

$0603	$8	$23 $81 $60 $00 $B0 $04 $00 $00 ‘Set Profile Velocity to 5 rps
$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 200000 steps
$0603	$8	$2B $40 $60 $00 $7F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $6F $02 $00 $00 ‘Clear New Set Point Bit
$0603	$8	$23 $81 $60 $00 $60 $09 $00 $00 ‘Set Profile Velocity to 10 rps
$0603	$8	$23 $7A $60 $00 $40 $0D $03 $00 ‘Set Target Position to 600000 steps
$0603	$8	$2B $40 $60 $00 $7F $02 $00 $00 ‘Set New Set Point Bit to 1
$0603	$8	$2B $40 $60 $00 $6F $02 $00 $00 ‘Clear New Set Point Bit

Profile Velocity Mode

**** Enable Motor Power - CiA 402 State Machine ****

ID	DLC	Data
$0603	$8	$2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on
$0603	$8	$2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603	$8	$2B $40 $60 $00 $0F $01 $00 $00 ‘Operation Enabled; Motion Halted

**** Set to Profile Velocity Mode ****

$0603	$8	$2F $60 $60 $00 $03 $00 $00 $00 ‘Set to Profile Velocity Mode

**** Set Motion Parameters ****

$0603	$8	$23 $FF $60 $00 $F0 $00 $00 $00 ‘Set Target Velocity to 1 rps
$0603	$8	$23 $83 $60 $00 $58 $02 $00 $00 ‘Set Acceleration to 100 rps/s
$0603	$8	$23 $84 $60 $00 $58 $02 $00 $00 ‘Set Deceleration to 100 rps/s

**** Start/Stop Motion ****

$0603	$8	$2B $40 $60 $00 $0F $00 $00 $00 ‘Motion Starts
$0603	$8	$23 $FF $60 $00 $60 $09 $00 $00 ‘Change Target Velocity to 10 rps
$0603	$8	$2B $40 $60 $00 $0F $01 $00 $00 ‘Motion Halts

Homing Mode

**** Enable Motor Power - CiA 402 State Machine ****

ID	DLC	Data
$0603	$8	$2B $40 $60 $00 $06 $00 $00 $00 ‘Ready to Switch on
$0603	$8	$2B $40 $60 $00 $07 $00 $00 $00 ‘Switched on
$0603	$8	$2B $40 $60 $00 $0F $00 $00 $00 ‘Operation Enabled

**** Set to Homing Mode ****

$0603	$8	$2F $60 $60 $00 $06 $00 $00 $00 ‘Set to Homing Mode
$0603	$8	$2F $98 $60 $00 $13 $00 $00 $00 ‘Set Homing Method to 19

**** Set Motion Parameters ****

$0603	$8	$23 $9A $60 $00 $58 $02 $00 $00 ‘Set Homing Acceleration to 100rps/s
$0603	$8	$23 $99 $60 $01 $F0 $00 $00 $00 ‘Set Homing Velocity (Search for Switch) to 1rps
$0603	$8	$23 $99 $60 $02 $78 $00 $00 $00 ‘Set Index Velocity (Search for Index or Zero) to 0.5rps
$0603	$8	$23 $7C $60 $00 $40 $9C $00 $00 ‘Set Homing Offset to 40000 Steps
$0603	$8	$2F $01 $70 $00 $03 $00 $00 $00 ‘Set Homing Switch to Input 3

**** Start/Stop Homing ****

$0603	$8	$2B $40 $60 $00 $1F $00 $00 $00 ‘Homing Starts
$0603	$8	$2B $40 $60 $00 $1F $01 $00 $00 ‘Homing Stops

Normal Q Mode

**** Enable Motor Power - CiA 402 State Machine ****

ID	DLC	Data	
$0603	$8	$2B $40 $60 $00 $06 $00 $00 $00	‘Ready to Switch on
$0603	$8	$2B $40 $60 $00 $07 $00 $00 $00	‘Switched on
$0603	$8	$2B $40 $60 $00 $0F $00 $00 $00	‘Operation Enabled

**** Set to Normal Q Mode ****

$0603	$8	$2F $60 $60 $00 $FF $00 $00 $00	‘Set to Normal Q Mode
$0603	$8	$2F $07 $70 $00 $01 $00 $00 $00	‘Set Q Segment Number to 1

**** Start/Stop Q Program ****

$0603	$8	$2B $40 $60 $00 $1F $00 $00 $00	‘Q Program Starts
$0603	$8	$2B $40 $60 $00 $1F $01 $00 $00	‘Q Program Halts

Sync Q Mode

**** Enable Motor Power - CiA 402 State Machine ****

ID	DLC	Data	
$0603	$8	$2B $40 $60 $00 $06 $00 $00 $00	‘Ready to Switch on
$0603	$8	$2B $40 $60 $00 $07 $00 $00 $00	‘Switched on
$0603	$8	$2B $40 $60 $00 $0F $00 $00 $00	‘Operation Enabled

**** Set to Sync Q Mode ****

$0603	$8	$2F $60 $60 $00 $FE $00 $00 $00	‘Set to Sync Q Mode
$0603	$8	$2F $07 $70 $00 $01 $00 $00 $00	‘Set Q Segment Number to 1
$0603	$8	$23 $05 $10 $00 $80 $00 $00 $00	‘Set Sync Pulse to 0x80

**** Start/Stop Q Program ****

$80	$0		‘Q Program Starts
$0603	$8	$2B $40 $60 $00 $0F $01 $00 $00	‘Q Program Halts

PDO Mapping

****Mapping TPDO2 ****

$0000	$2	$80 $03	‘Return back to “PreOperation” Mode
$0603	$8	$23 $01 $18 $01 $80 $02 $00 $80	‘Turn off the TPDO2
$0603	$8	$2F $01 $1A $00 $00 $00 $00 $00	‘Set Number of Mapped objects to zero
$0603	$8	$23 $01 $1A $01 $10 $00 $41 $61	‘Map object1(0x6041) to TPDO2 subindex1.
$0603	$8	$23 $01 $1A $02 $20 $00 $0A $70	‘Map object2(0x700A) to TPDO2 subindex2.
$0603	$8	$2F $01 $1A $00 $02 $00 $00 $00	‘Set Number of total Mapped objects to Two
$0603	$8	$23 $01 $18 $01 $80 $02 $00 $00	‘Turn on the TPDO2

Applied Motion Products, Inc.

404 Westridge Drive Watsonville, CA 95076

Tel (831) 761-6555 (800) 525-1609 Fax (831) 761-6544