Lenze
AC Tech

PositionServo

CANopen Communication Module

Communications Interface Reference Guide

P94CAN01B

About These Instructions

This documentation applies to the optional CANopen Communication Module for the PositionServo drive and should be used in conjunction with the PositionServo User Manual (S94P01) that shipped with the drive. These documents should be read carefully as they contain important technical data and describe the installation and operation of the drive and this option module. This manual describes the CANopen implementation developed by AC Tech Corporation for the PositionServo drives. It contains information for anyone who participates in the evaluation of or design of a distributed motion control system. The user should have prior knowledge of motion control, networks, and CANopen before implementing a CANopen program.

This document is printed in the United States of America

Table of Contents

1. Safety Information

1.1 Warnings, Cautions & Notes

General

Some parts of Lenze controllers (frequency inverters, servo inverters, DC controllers) can be live, with the potential to cause attached motors to move or rotate. Some surfaces can be hot.

Non-authorized removal of the required cover, inappropriate use, and incorrect installation or operation creates the risk of severe injury to personnel or damage to equipment.

All operations concerning transport, installation, and commissioning as well as maintenance must be carried out by qualified, skilled personnel (IEC 364 and CENELEC HD 384 or DIN VDE 0100 and IEC report 664 or DIN VDE0110 and national regulations for the prevention of accidents must be observed).

According to this basic safety information, qualified skilled personnel are persons who are familiar with the installation, assembly, commissioning, and operation of the product and who have the qualifications necessary for their occupation.

Application as directed

Drive controllers are components which are designed for installation in electrical systems or machinery. They are not to be used as appliances. They are intended exclusively for professional and commercial purposes according to EN 61000-3-2. The drive user manual includes information on compliance with EN 61000-3-2.

When installing the drive controllers in machines, commissioning (i.e. the starting of operation as directed) is prohibited until it is proven that the machine complies with the regulations of the EC Directive 98/37/EEC (Machinery Directive); EN 60204 must be observed.

Commissioning (i.e. starting of operation as directed) is only allowed when there is compliance with the EMC Directive (89/336/EEC).

The drive controllers meet the requirements of the Low Voltage Directive 73/23/EEC. The harmonised standards of the series EN 50178/DIN VDE 0160 apply to the controllers.

The availability of controllers is restricted according to EN 61800-3. These products can cause radio interference in residential areas.

Installation

Ensure proper handling and avoid excessive mechanical stress. Do not bend any components and do not change any insulation distances during transport or handling. Do not touch any electronic components and contacts.

Controllers contain electrostatically sensitive components, which can easily be damaged by inappropriate handling. Damaging or destroying electrical components may pose a danger to your health.

Electrical connection

When working on live drive controllers, applicable national regulations for the prevention of accidents (e.g. VBG 4) must be observed.

The electrical installation must be carried out according to the appropriate regulations (e.g. cable cross-sections, fuses, PE connection). Additional information can be obtained from the national regulation documentation. In the United States, electrical installation is regulated by the National Electric Code (nec) and NFPA 70 along with state and local regulations.

The standards documentation contains information about installation in compliance with EMC (shielding, grounding, filters and cables). These notes must also be observed for CE-marked controllers.

The manufacturer of the system or machine is responsible for compliance with the required limit values demanded by EMC legislation.

Operation

Systems including controllers must be equipped with additional monitoring and protection devices according to the corresponding standards (e.g. technical equipment, regulations for prevention of accidents, etc.). You are allowed to adapt the controller to your application as described in the documentation.

	**DANGER**! • After the controller has been disconnected from the supply voltage, do not touch live components or the power connection until capacitors have had enough time to discharge. Wait at least 3 minutes before servicing the drive. Observe the corresponding notes on the controller. • Do not continuously cycle input power to the controller more than once every three minutes. • Please close all protective covers and doors during operation.
	**WARNING!** Network control permits automatic operation of the inverter drive. The system design must incorporate adequate protection to prevent personnel from accessing moving equipment while power is applied to the drive system.

Pictographs used in these instructions:

Pictograph	Signal Word	Meaning	Consequence if Ignored
	**DANGER!**	Warning of Hazardous Electrical Voltage.	Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	**WARNING!**	Impending or possible danger to personnel	Death or injury
	**STOP!**	Possible damage to equipment	Damage to drive system or its surroundings
	**NOTE**	Useful tip: If note is observed, it will make using the drive easier	

1.2 Reference Documents

- CAN and CANopen Specifications: “CAN in Automation (CiA)”; visit: http://www.can-cia.de/
- PositionServo Programming Manual: PM94P01; Refer to: http://www.actech.com
- PositionServo User Manual: S94P01; Refer to: http://www.actech.com
- MotionView Software Manual: IM94MV01; Refer to: http://www.actech.com

1.3 Conventions for Object Descriptions

In this manual, object descriptions are as illustrated in the sample below. Each object description includes summary information including object type, access, units, range, map PDO and memory capability in tabular format. The table header includes the object title and index/sub-index number.

Object				Index	Sub-Index
SERVER SDO Parameters				**0x1200**	
Type	Access	Units	Range	Map PDO	Memory
Record	RO	--	--	NO	--
Description The Server SDO object holds the COB-ID values needed for access the drive’s SDO. Sub-index 0 holds the number of sub-elements of the record. The COB-ID is the communication object ID or the CAN message ID.					

Object				Index	Sub-Index
SDO Receive COB-ID				**0x1200**	**1**
Type	Access	Units	Range	Map PDO	Memory
Unassigned 32	RO	--	0x600 – 0x671	NO	--
Description The SDO Receive COB-ID is the CANopen object ID used by the drive to receive an SDO packet. The SDO Receive COB-ID value is 0x600 plus the drive’s CAN node ID.					

Relationships of Sub-Index Objects

This manual describes both objects and sub-index objects. Object descriptions and Sub-Index object descriptions are included in the bottom portion of the table. The Sub-index object 0 always contains the number of elements contained by the record.

Object Summary Description Fields

Field Name	Description
Type	The object type: Record, Array, Float, Visible String, Integer (8/16/32), Unassigned (8/16/32)
Access	The object’s access type: RO, WO, RW
	RO: read only
	WO: write only
	RW: read and write
Units	The units used to express the object’s value.
Range	The acceptable range of values if less then that specified by Type.
Map PDO	Object PDO map capability: YES, NO, EVENT
	YES: the object can be mapped to a PDO
	NO: the object cannot be mapped to a PDO
	EVENT: the object can be mapped and can be set to event triggering.
Memory	F: object can be held in the amplifier’s flash memory
	R: object can be held in the amplifier’s RAM
	RF: object can be held in the amplifier’s flash memory and RAM
	-- (dash):object can not be stored or object contains sub-index objects

1.4 Commonly Used Terms, Acronyms & Definitions

CAN	Control Area Network
CANopen	Communication protocol to open and communicate with the Control Area Network
CMS	CAN-based Message Specification
COB-ID	Communication Object Identifier
CP	Communication Profile
CRC	Cyclic Redundancy Check
DCF	Device Configuration File: an ASCII file containing a description of the object configuration of an individual device
DP	Device Profile: defines the OD objects for a particular type of device
EDS	Electronic Data Sheet: an ASCII file containing a device's communication functionality and objects; plus its device-specific objects and their default values
EMCY	Emergency Object
Index	4-digit hexadecimal code used to identify an object: 16-bits Sub-Index: decimal code to further identify object's parameters: 8-bits
NMT	Network Management
OD	Object Dictionary
Object	Communication message - 4 types: Administrative: NMT Service Data Object: SDO Process Data Object: PDO Pre-Defined or Special Function Object: SYNC; TIME STAMP; EO
PDO	Process Data Object:
RPDO	Receive PDO
SDO	Service Data Object:
SYNC	Synchronization Special Function Object
TIME STAMP	Provides a common time reference, implemented as a CMS object
TPDO	Transmit PDO (also known as TxPDO)
Transferring Data:	SDO: for large low priority data transfer between devices (i.e. configuring devices on CANopen network) PDO: for fast data transfer of 8 bytes or less without protocol overhead (i.e. the data content has been previously defined)
Transmission Modes:	Sychronous: synchronization by receipt of a SYNC message Cyclic: transmission triggered periodically after every 'n' SYNC message Acylic: transmission 'pre-triggered' by remote transmission request from another device or by the occurrence of an object-specific event specified in the device profile Asychronous: transmission is triggered by a remote transmission request from another device (by a CAN Frame)

2 Installation

2.1 Mechanical Installation

Install the CAN Communications Module as illustrated in Figure 1.

Disconnect power from drive and wait 3 minutes.

Remove the two COMM module screws that secure Option Bay 1.

With a flat head screwdriver, pry up the Option Bay 1 cover plate.

Install the CAN COMM Module (E94ZACAN1) in Option Bay 1.

Secure with two COMM module screws (max torque: 0.3Nm/3lb-in).

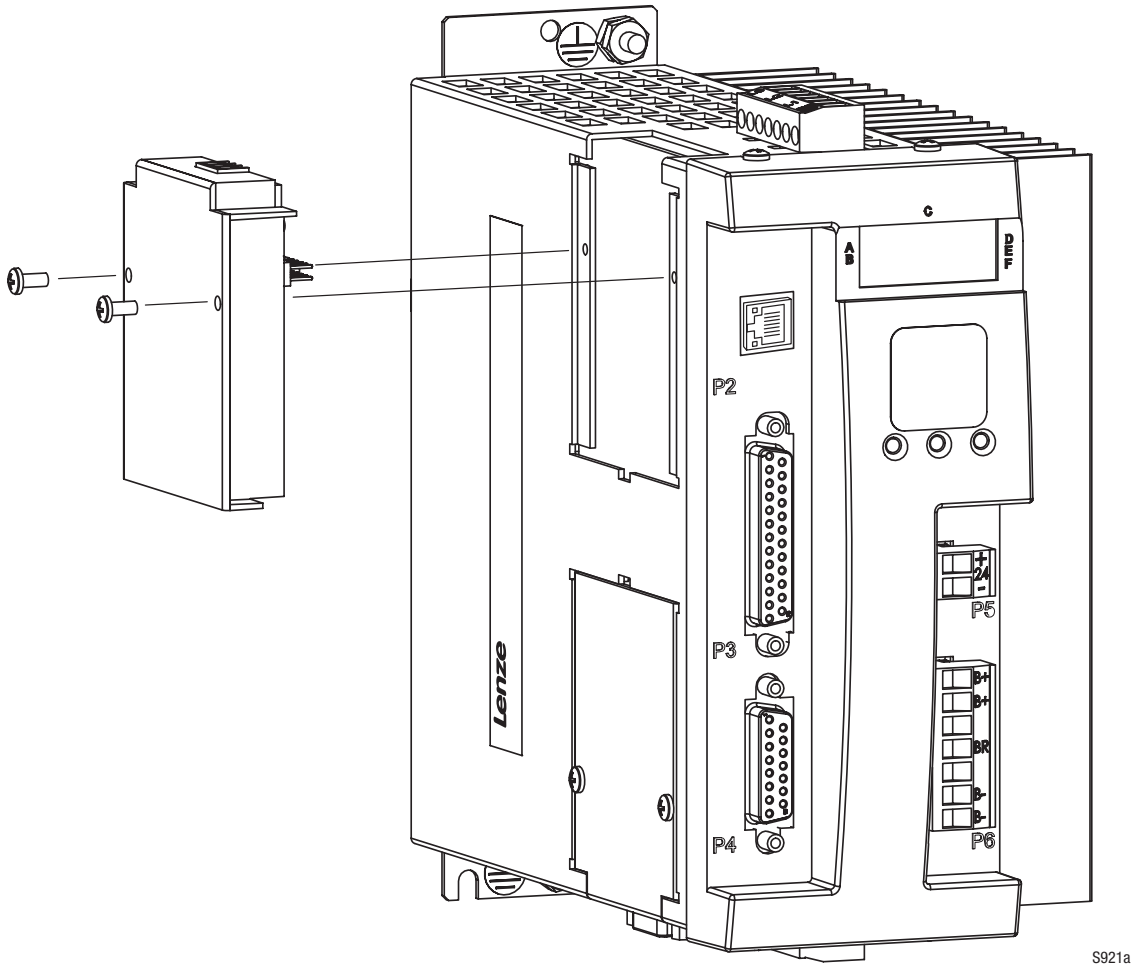

Figure 1: Installation of CAN Communications Module

2.2 Electrical Installation

Table 1 and Figure 2 illustrate the pinout of the PositionServo CAN Module connector. This connector provides 2-wire plus isolated ground connection to the network.

Table 1: CAN Bus Interface Pin Assignments

Terminal	Name	Description
1	ICOM	Isolated Common
2	CAN L	CAN Bus Low
3	CAN H	CAN Bus High

Figure 2: CAN Bus Interface Pinout

Connections and Shielding

Figure 3 illustrates the connection of the cables for a PositionServo drive in a CAN master/slave network. A 120ohm (1%) termination is necessary between the high and low terminals of the first and last slaves in the network.

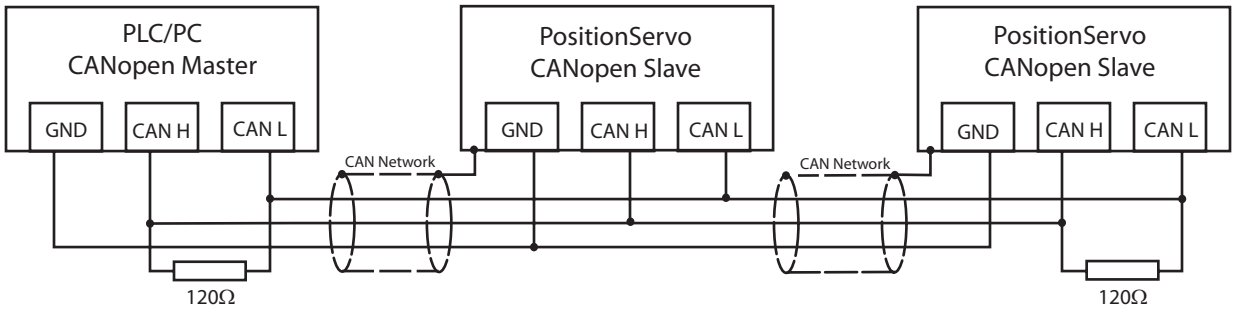

Figure 3: 120Ω (1%) Termination in CAN Network

3 Introduction

This reference guide assumes that the reader has a working knowledge of CANopen protocol and familiarity with the programming and operation of motion control equipment. This guide is intended as a reference only. The optional CANopen communication module (P/N E94ZACAN1) is required for the PositionServo drive to communicate on a CAN network.

3.1 CAN Overview

The backbone of CANopen is CAN, a serial bus network originally designed by Robert Bosch GmbH to coordinate multiple control systems in automobiles. The CAN model lends itself to distributed control. Any device can broadcast messages on the network. Each device receives all messages and uses filters to accept only the appropriate messages. Thus, a single message can reach multiple nodes, reducing the number of messages that need to be sent. This also greatly reduces bandwidth required for addressing, allowing distributed control at real-time speeds across the entire system.

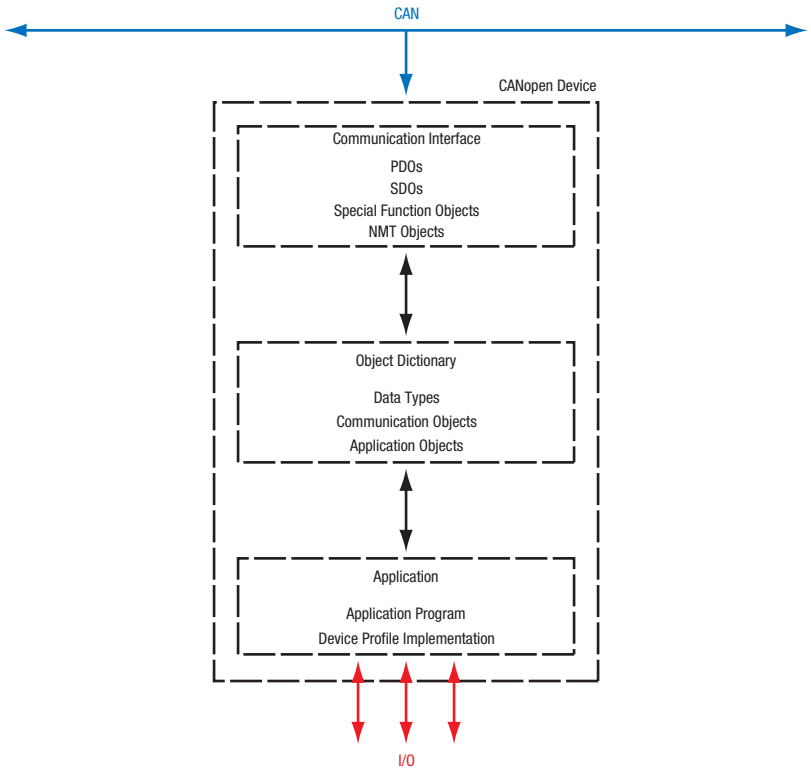

Figure 4: CANbus Network

CAN and CANopen Architecture

CAN specifies the data link and physical connection layers of a fast, reliable network. The CANopen profiles specify how various types of devices, including motion control devices, can use the CAN network in a more efficient manner.

In a CANopen motion control system, control loops are closed on the individual amplifiers, not across the network. A master application coordinates multiple devices, using the network to transmit commands and receive status information. Each device can transmit to the master or any other device on the network. CANopen provides the protocol for mapping device and master internal commands to messages that can be shared across the network. A CANopen network can support up to 127 nodes. Each node has a seven-bit node ID in the range of 1-127. (Node ID 0 is reserved and should not be used.)

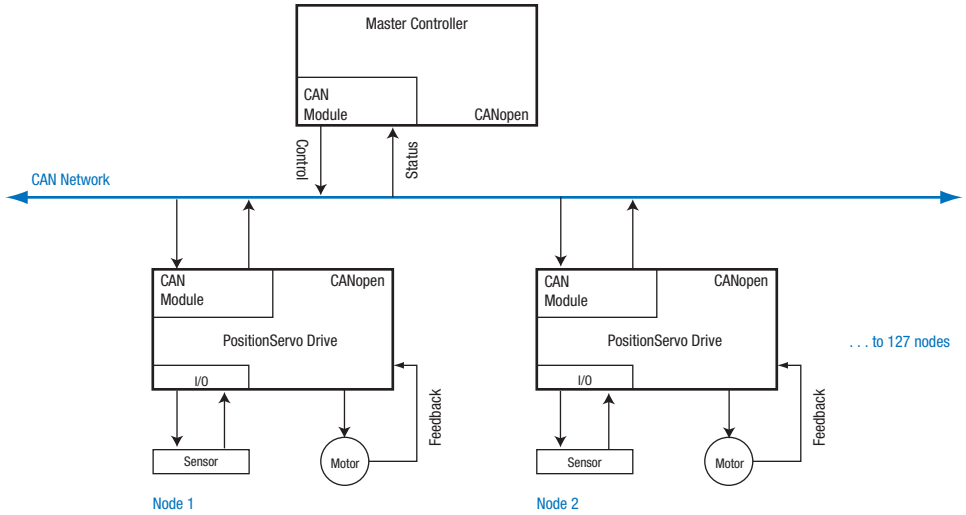

Figure 5: CANopen Network

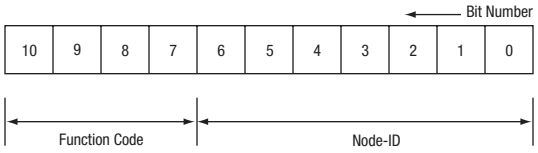

Figure 6: 11-bit CAN Identifier

Example of a CANopen Move Sequence:

- CANopen master transmits a control word to initialize all devices.
- Devices transmit messages indicating their status (in this example, all are operational).
- CANopen master transmits a message instructing devices to perform homing operations.
- Devices indicate that homing is complete.
- CANopen master transmits messages instructing devices to enter position profile mode (point-to-point motion mode) and issues first set of point-to-point move coordinates.
- Devices execute their moves, using local position, velocity, and current loops, and then transmit actual position information back to the network.
- CANopen master issues next set of position coordinates.

3.2 PositionServo Drive Configuration

Before AC Tech drives can be used in a CANopen network they need to be properly connected and configured. Refer to the PositionServo User's Manual for details on hardware connection.

There are a few parameters that need to be configured before the PositionServo can operate in a CANopen network. These parameters are listed under the <Communication> <CAN> folder in the MotionView software program. Alternatively these parameters can be reached from the drive's front display and keypad. CAN related parameters are explained herein:

- CAN Control — Enabled/Disabled: Use this parameter to enable or disable CAN followed by reboot. This parameter takes effect after the drive has been re-booted (power cycled).
- CAN baud rate — 10k- 1000k: Parameter takes effect after drive has been re-booted (power cycled).
- CAN address — 1-127: sets drive's CAN ID. This parameter takes effect after the drive has been re-booted (power cycled).

- CAN Boot Up Mode — Pre-Operational, Operational or Pseudo Master modes are available after power up.
  - Pre-Operational — default mode for CAN Open slave. Drive will await message from master to enter Operational mode
  - Operational — drive will enter Operational mode immediately after power up without receiving activation message from master. This feature is useful in a master-less network.
  - Pseudo Master — in this mode drive will send activation message (with specified delay, see below) for all CAN slaves waiting in Pre-Operational mode. This mode is useful when emulating master functionality and activating passive slaves. Only one drive can be configured as the pseudo master and only when there is no other master device.
- CAN Boot up delay — If drive is configured for Pseudo Master mode it will send activation message with delay specified in this parameter. Delay is used to allow specified slaves to boot up and configure their hardware to listen to the Master messages.

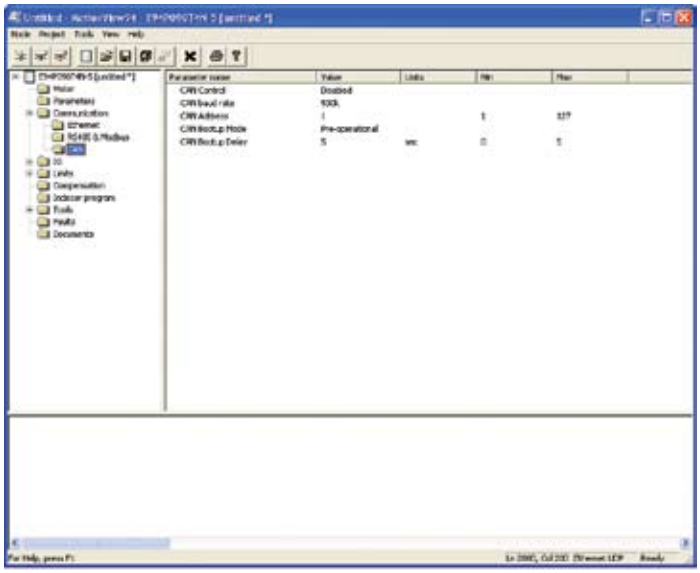

Figure 7: MotionView CANopen Communication Parameters

CANopen configuration parameters in MotionView

To configure the drive for CANopen operations follow steps:

1. Start MotionView and perform connection to the drive.
2. Open <Communication><CAN> folder to see CAN related parameters.
3. Set proper communication speed
4. Set drive's CAN ID
5. Set the [CAN Bootup Mode]. Select pre-operational mode if the drive is going to be part of CANopen network with a CANopen Master controller (or the other drive is set to Pseudo-Master mode)
6. If you selected "Pseudo Master Mode" in step 5, set the [CAN Bootup Delay].
7. Set the [CAN Control] parameter to "Enable"
8. Re-boot the drive.
9. After drive boots it enters selected boot up mode and is ready for operation under CAN control
10. To turn off CAN mode, set parameter [CAN Control] to disable and re-boot the drive. Drive will return to normal operation mode thereafter.

NOTE:

1. The user program is disabled while in CAN control mode and attempting to access it will result in a warning message. The program however is not erased from the memory and will be made available for execution upon return from CANopen mode ([CAN Control] set to Disable).

2. MotionView can be connected to drive for monitoring of parameters, reviewing faults, oscilloscope operations etc. Drive's variables can be viewed from <Tools><Diagnostic> folder.

All CAN parameters can be accessed from the front panel. The parameter names as seen from the built in display are:

Name	Description	Input	Value
CAnb	CAN baud rate	1	10k
		2	25k
		3	50k
		4	125k
		5	250k
		6	500k
		7	800k
		8	1000k
CAnA	CAN address	1 - 127	
Cano	Bootup Op mode	0	pre-operational
		1	operational
		2	pseudo master mode
CAnE	CAN control parameter	0	disabled
		1	enabled

NOTE:
CAnE , CAnA and CAnb require a reboot to change.

3.3 CAN Protocol

The physical layer of CAN is a differentially driven, two-wire bus, terminated by 120-ohm resistors at each end. The maximum bit rate supported by CAN is 1,000,000 bits/second for up to 25 meters. Lower bit rates may be used for longer network lengths.

The CAN Message

CANopen messages are transmitted within CAN messages.

CAN Message Format

CAN messages are communicated over the bus in the form of network packets. Each packet consists of an identifier (CAN message ID), control bits, and zero to eight bytes of data. (The CAN message is sometimes referred to as a communication object or COB and the CAN message ID as a COB-ID.)

CRC Error Checking

Each packet is sent with CRC (cyclic redundancy check) information to allow controllers to identify and re-send incorrectly formatted packets.

CAN Message ID

Every CAN message has a CAN message ID. The message ID plays two important roles:

- It provides the criteria by which the message is accepted or rejected by a node.
- It determines the message's priority.

CAN Message Priority

The priority of a CAN message is encoded in the message ID. The lower the value of the message ID, the higher the priority of the message. When two or more devices attempt to transmit packets at the same time, the packet with the highest priority succeeds. The other devices back off and re-attempt later.

Objects and Dictionaries

The primary means of controlling a device on a CANopen network is by writing to device parameters, and reading device status information. For this purpose, each device defines a group of parameters that can be written, and status values that can be read. These parameters and status values are collectively referred to as the device's objects. These objects define and control every aspect of a device's identity and operation. Some objects define basic information such as device type, model, and serial number. Others are used to check device status and deliver motion commands. The entire set of objects defined by a device is called the device's Object Dictionary (OD). Every device on a CANopen network must define an object dictionary, and nearly every CANopen network message involves reading values from or writing values to the object dictionaries of devices on the network.

Object Dictionary as Interface

The object dictionary is an interface between a device and other entities on the network.

CANopen Profiles and the Object Dictionary

The CANopen profiles specify the mandatory and optional objects that comprise most of an object dictionary. The Communication Profile specifies how all devices must communicate with the CAN network. The Communication Profile specifies dictionary objects that set up a device's ability to send and receive messages. The device profiles specify how to access particular functions of a device. The Profile for Drives and Motion Control specifies objects used to control device homing and position control. In addition to the objects specified in the Application Layer and Communication Profile and device profiles, CANopen allows manufacturers to add device-specific objects to a dictionary.

Object Dictionary Structure

An object dictionary is a lookup table. Each object is identified by a 16-bit index with an eight-bit sub-index. Most objects represent simple data types, such as 16-bit integers, 32-bit integers, and strings. These can be accessed directly by the 16-bit index.

Other objects use the sub-index to represent groups of related parameters. For instance, the Motor Data object (index 0x6410, paragraph 6.5, page 47) has 24 sub-index objects defining basic motor characteristics such as motor type, motor wiring configuration, and hall sensor type. (The subindex provides up to 255 sub-entries for each index.)

The organization of the dictionary is specified in the profiles, as shown herein.

Index Range	Objects
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 (including those specific to motion control)
0080-009F	Device Profile Specific Complex Data Types (including those specific to motion control)
00A0-0FFF	Reserved for further use
1000-1FFF	Communication Profile Area (DS 301)
2000-5FFF	Manufacturer Specific Profile Area
6000-9FFF	Standardized Device Profile Area (including Profile for Motion Control)
A000-FFFF	Reserved for future use

3.4 Accessing the Object Dictionary

CANopen provides two methods to access a device's object dictionary:

- The Service Data Object (SDO)
- The Process Data Object (PDO)

Each can be described as a channel for access to an object dictionary.

3.4.1 SDOs and PDOs

The basic characteristics of PDOs and SDOs are listed in Table 2. For help deciding whether to use an SDO or a PDO refer to paragraph 3.4.4 "SDO or PDO? Design Considerations".

Table 2: Basic PDO & SDO Characteristics

SDO	PDO
The SDO protocol allows any object in the object dictionary to be accessed, regardless of the object's size. This comes at the cost of significant protocol overhead.	One PDO message can transfer up to eight bytes of data in a CAN message. There is no additional protocol overhead for PDO messages.
Transfer is always confirmed.	PDO transfers are unconfirmed.
Has direct, unlimited access to the object dictionary.	Requires prior setup, wherein the CANopen master application uses SDOs to map each byte of the PDO message to one or more objects. Thus, the message itself does not need to identify the objects, leaving more bytes available for data.
Employs a client/server communication model, where the CANopen master is the sole client of the device object dictionary being accessed.	Employs a peer-to-peer communication model. Any network can initiate a PDO communication, and multiple nodes can receive it.
An SDO has two CAN message identifiers: a transmit identifier for messages from the device to the CANopen master, and a receive identifier for messages from the CANopen master.	Transmit PDOs are used to send data from the device, and receive PDOs are used to receive data.
SDOs can be used to access the object dictionary directly.	A PDO can be used only after it has been configured using SDO transfers.
Best suited for device configuration, PDO mapping, and other infrequent, low priority communication between the CANopen master and individual devices. Such transfers tend to involve the setting up of basic node services; thus, the term service data object.	Best suited for high-priority transfer of small amounts of data, such as delivery of set points from the CANopen master or broadcast of a device's status. Such transfers tend to relate directly to the application process; thus, the term process data object.
For more information about SDOs, refer to "SDOs: Description and Examples", paragraph 3.4.2, page 17	For more information about PDOs, refer to "PDOs: Description and Examples", paragraph 3.4.3, page 18

The Communication Profile requires the support of at least one SDO per device. (Without an SDO, there would be no way to access the object dictionary.) It also specifies default parameters for four PDOs. AC Tech CANopen amplifiers each support 1 SDO and 16 PDOs (eight transmit PDOs and eight receive PDOs).

3.4.2 SDOs: Description and Examples

Each PositionServo amplifier provides one SDO. The CANopen master can use this SDO to configure, monitor, and control the device by reading from and writing to its object dictionary.

SDO CAN Message IDs

The SDO protocol uses two CAN message identifiers. One ID is for messages sent from the CANopen master (SDO client) to the amplifier (SDO server). The other ID is for messages sent from the SDO server to the SDO client.

The CAN message ID numbers for these two messages are fixed by the CANopen protocol. They are based on the device's node ID (which ranges from 1 to 127). The ID used for messages from the SDO client to the SDO server (i.e. from the CANopen master to the amplifier) is the hex value 0x600 + the node ID. The message from the SDO server to the SDO client is 0x580 + the node ID. For example, an amplifier with node ID 7 uses CAN message IDs 0x587 and 0x607 for its SDO protocol.

Client/ Server Communication

The SDO employs a client/server communication model. The CANopen master is the sole client. The device is the server. The CANopen master application should provide a client SDO for each device under its control. The CAN message ID of an SDO message sent from the CANopen master to a device should match the device's receive SDO message identifier. In response, the CANopen master should expect an SDO message whose CAN message ID matches the device's transmit SDO message identifier.

SDO Message Format

The SDO uses a series of CAN messages to send the segments that make up a block of data. The full details of the SDO protocol are described in the CANopen Application Layer and Communication Profile.

Confirmation

Because an SDO transfer is always confirmed, each SDO transfer requires at least two CAN messages (one from the master and one from the slave).

Confirmation Example

To update an object that holds an eight-byte long value requires six CAN messages:

1. The master sends a message to the device indicating its intentions to update the object.
2. The message includes the object's index and sub-index values as well as the size (in bytes) of the data to be transferred.
3. The device responds to the CANopen master indicating that it is ready to receive the data.
4. The CANopen master sends one byte of message header information and the first 7 bytes of data. (Because SDO transfers use one byte of the CAN message data for header information, the largest amount of data that can be passed in any single message is 7 bytes.)
5. The device responds indicating that it received the data and is ready for more.
6. The CANopen master sends the remaining byte of data.
7. The device responds indicating success.

Segmented, Expedited and Block Transfers

As in the example above, most SDO transfers consist of an initial transfer request from the client, followed by series of confirmed eight-byte messages. Each message contains one byte of header information and a segment (up to seven bytes long) of the data being transferred. For the transfer of short blocks of data (four bytes or less), the Communication Profile specifies an expedited SDO method. The entire data block is included in the initial SDO message (for uploads) or in the response (for downloads). Thus, the entire transfer is completed in two messages.

The Communication Profile also describes a method called block SDO transfers, where many segments can be transferred with a single acknowledgment at the end of the transfer. AC Tech CANopen amplifiers do not require the use of the block transfer protocol.

Using an SDO

To use an SDO, the CANopen master needs an SDO client to communicate with the SDO on each device.

3.4.3 PDOs: Description and Examples

PositionServo amplifiers provide eight transmit PDOs and eight receive PDOs. A transmit PDO is used to transmit information from the device to the network. A receive PDO is used to update the device.

Default PDO Message Identifiers

The Communication Profile reserves four CAN message identifiers for transmit PDOs and four identifiers for receive PDOs. Refer to the sections on Receive PDO Communication Parameters and Transmit PDO Communication Parameters.

The first four transmit PDOs and receive PDOs provided in AC Tech CANopen amplifiers use these default addresses. The addresses of the remaining four transmit PDOs and receive PDOs are null by default. Any PDO message identifier can be reconfigured by the programmer.

PDO Peer-to-Peer Communication

Peer-to-peer relationships match the transmit PDO identifier of the sending node to a receive PDO identifier of one or more other nodes on the network. Any device can broadcast a PDO message using one of its eight transmit PDOs. The CAN identifier of the outgoing message matches the ID of the sending PDO. Any node with a matching receive PDO identifier will accept the message.

PDO Peer-to-Peer Example:

Node 1 = transmit PDO 1, has a CAN message ID of 0x0189. Node 2 = receive PDO 1 has a matching ID, as does Node 3. They both accept the message. Other nodes do not have a matching receive PDO, so they do not accept the message.

PDO Mapping

For optimal of the CAN message's eight-byte data area use PDO mapping. For each byte in the PDO message, mapping uses the SDO to configure dictionary objects in both the sending and the receiving node to know:

- The index and sub-index which objects are to be accessed
- The type of data
- The length of the data

The PDO message itself carries no transfer control information, leaving all eight bytes available for data. (Contrast this with the SDO, which uses one byte of the CAN message data area to describe the objects being written or read, and the length of the data.)

Default PDO Mappings

The Profile for Drives and Motion Control specifies default mappings for the first eight transmit PDOs and the first eight receive PDOs. AC Tech CANopen amplifiers are shipped with these default PDO mappings. These default PDO mappings can be re-mapped by the programmer.

Mappable Objects

Not all objects in a device's object dictionary can be mapped to a PDO. This manual notes this ability (or lack thereof) in the description of each object.

Dynamic PDO Mapping

AC Tech supports dynamic PDO mapping, which allows the CANopen master to change the mapping of a PDO during operation. For instance, a PDO might use one mapping in Homing Mode, and another mapping in Profile Position Mode.

PDO Transmission Modes

PDOs can be sent in one of two transmission modes:

- Synchronous: Messages are sent only after receipt of a specified number of synchronization (SYNC) objects, sent at regular intervals by a designated synchronization device. (For more information on the SYNC object, see SYNC and high-resolution Time Stamp messages)
- Asynchronous: The receipt of SYNC messages does not govern message transmission.

Synchronous transmission can be cyclic, where the message is sent after a predefined number of SYNC messages, or acyclic, where the message is triggered by some internal event but does not get sent until the receipt of a SYNC message.

PDO Triggering Modes

The transmission of a transmit PDO message from a node can be triggered in one of three ways:

Trigger	Description
Event	Message transmission is triggered by the occurrence of an object specific event. For synchronous PDOs this is the expiration of the specified transmission period, synchronized by the reception of the SYNC object. For acyclicly transmitted synchronous PDOs and asynchronous PDOs the triggering of a message transmission is a device specific event specified in the device profile.
SYNC message	For synchronous PDOs, the message is transmitted after a specified number of SYNC cycles have occurred.
Remote Request	The transmission of an asynchronous PDO is initiated upon receipt of a remote request initiated by any other device.

PDO Examples

The programmer has broad discretion in the use of PDOs. Consider some of the default receive PDO mappings specified in the Profile for Drives and Motion Control:

PDO	Default Objects Mapped	Purpose
Receive PDO 1	Control Word (0x6040)	Controls the state of the device
Receive PDO 2	Control Word (0x6040) Mode of Operation (0x6060)	Controls the state and operating mode of the device
Receive PDO 3	Control Word (0x6040) Target Position (0x607a)	Controls the state and target position of the amplifier in profile position mode

Here are some other examples:

- On the device designated as the SYNC message and time stamp producer, map a transmit PDO to transmit the high-resolution time stamp message on a periodic basis. Map receive PDOs on other devices to receive this object.
- Another transmit PDO could transmit general amplifier status updates.

3.4.4 SDO or PDO? Design Considerations

Differences Between SDO and PDO

As stated earlier, SDOs and PDOs can both be described as channels through which CAN messages are sent, and both provide access to a device's object dictionary. However, each has characteristics that make it more appropriate for certain types of data transfers. Table 3 provides a review of the differences between SDOs and PDOs, and some design considerations indicated by those differences.

Table 3: SDO & PDO Design Considerations

SDO	PDO	Design Considerations
The accessed device always confirms SDO messages. This makes SDOs slower.	PDO messages are unconfirmed. This makes PDOs faster.	To transfer 8 bytes or less at real-time speed, use a PDO. For instance, to receive control instructions and transmit status updates. To transfer large amounts of low priority data, use the SDO. Also, if confirmation is absolutely required, use an SDO.
One SDO transfer can send long blocks of data, using as many CAN messages as required.	A PDO transfer can only send small amounts of data (up to eight bytes) in a single CAN message. Mapping allows very efficient use of those eight bytes.	
Asynchronous.	Synchronous or asynchronous. Cyclic or acyclic.	Use PDO when synchronous or broadcast communications are required. For instance, to communicate set points from the master to multiple devices for a multi-axis move, or to have a device broadcast its status.
The SDO employs a client-server communication model. The CANopen master is the client. It reads from and writes to the object dictionaries of devices. The device being accessed is the server.	The PDO employs a peer-to-peer communication model. Any device can send a PDO message, and a PDO message can be received and processed by multiple devices..	
All communications can be performed through the SDO without using any PDOs.	The CANopen master application uses SDO messages to map the content of the PDO, at a cost of increased CPUcycles on the CANopen master and increased bus traffic.	If the application does not benefit from the use of a PDO for a certain transfer,consider using SDO to avoid the extra overhead. For instance, if an object's value is updated only once (as with many configuration objects), the SDO is more efficient. If the object's value is updated repeatedly, a PDO is more efficient.

3.4.5 Mapping a PDO

Reasons for mapping or remapping a PDO include:

- changes to the amplifier's Manufacturer Status Register object (index 0x1002, paragraph 6.2, page 36)
- changes in the CANopen status word
- amplifier I/O change
- amplifier PVT status changes

Two objects in the device's object dictionary define a PDO:

- A PDO's communication object defines the PDO's CAN message ID and its communication type (synchronous or asynchronous) and triggering type (event-drive or cyclic).
- A PDOs mapping object maps every data byte in the PDO message to an object in the device's object dictionary.

Mapping a PDO is the process of configuring the PDO's communication and mapping objects.

To Map a Receive PDO

The general procedure for mapping a receive PDO is listed in Table 4. The procedure for mapping a transmit PDO is similar.

Table 4: Mapping a Receive PDO

Stage	Step	Sub-Steps / Comments
1	Disable the PDO	In the PDO's mapping object (Receive PDO Mapping Parameters, index 0x1601), set the sub-index 0 (NUMBER OF MAPPED OBJECTS) to zero. This disables the PDO
2	Set the communication parameters	If necessary, set the PDO's CAN message ID (PDO COB-ID) using sub-index 1 of the PDO's RECEIVE PDO Communication Parameters (index 0x1401). Choose the PDO's transmission type (PDO TYPE) in sub-index 2 of object 0x1401. A value in the range [0-240] = synchronous; [254-255] = asynchronous.
3	Map the data	Using the PDO's mapping parameters (sub-indexes 1-4 of Receive PDO Mapping Parameters, index 0x1601), you can map up to 4 objects (whose contents must total to no more than 8 bytes), as follows: In bits 0-7 of the mapping value, enter the size (in bits) of the object to be mapped, as specified in the object dictionary. In bits 8-15, enter the sub-index of the object to be mapped. Clear bits 8-15 if the object is a simple variable. In bits 16-31, enter the index of the object to be mapped.
4	Set the number of mapped objects and enable the PDO	In the PDO's Receive PDO Mapping Parameters (index 0x1601), set sub-index 0 (NUMBER OF MAPPED OBJECTS) to the actual number of objects mapped. This properly configures the PDO. Also, the presence of a non-zero value in the NUMBER OF MAPPED OBJECTS object enables the PDO

Example: Mapping a Receive PDO

This example illustrates the general procedure for mapping a receive PDO. In the example, the second receive PDO is mapped to the device's Control Word object (index 0x6040) to receive device state change commands and to the Mode of Operation object (index 0x6060) to receive mode change commands.

3.5 Objects that Define SDO's and PDO's

To define an SDO or PDO, use the objects included in Table 5.

Table 5: Objects the Define an SDO or PDO

Object	Index	Sub-Index
Receive Object		
Receive PDO Communication Parameters	0x1400 - 0x1407	--
PDO COB_ID Index	0x1400-7	1
PDO Type	0x1400-7	2
Receive PDO Mapping Parameters	0x1600 - 0x1607	--
Number of Mapped Objects	0x1600-7	0
PDO Mapping	0x1600-7	1-8
Transmit Object		
Transmit PDO Communication Parameters	0x1800 - 0x1807	--
PDO COB-ID	0x1800-7	1
PDO Type	0x1800-7	2
Transmit PDO Mapping Parameters	0x1A00 - 0x1A07	--
Number of Mapped Objects	0x1A00-7	0
PDO Mapping	0x1A00-7	1-8

Object				Index	Sub-Index
RECEIVE PDO COMMUNICATION PARAMETERS				**0x1200**	**1**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	NO	--
Description These objects allow configuration of the communication parameters of each receive PDO. Subindex 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
PDO COB-ID				**0x1200**	**1**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	Refer to Description	NO	R

Description
CAN message ID used by the PDO. The ID is formatted as follows:

Bit	Description
0-10	Give the 11-bit identifier for standard (CAN 2.0A) identifiers, or the lower 11 bits for extended (CAN 2.0B) identifiers.
11-28	Give the upper 18 bits of extended identifiers. For standard identifiers these bits should be written as zeros.
29	Defines the identifier format. This bit is clear for standard (11-bit) identifiers, and set for extended (29-bit) identifiers.
30	Reserved for future use.
31	Identifies the PDO as valid if clear. If set, the PDO is disabled and its mapping may be changed.

Default Values
The default values for this object are specified in the DS-301 CANopen specification. These values are:

Index	Default ID
0x1400	0x00000200 + amplifier CAN node ID.
0x1401	0x00000300 + amplifier CAN node ID.
0x1402	0x00000400 + amplifier CAN node ID.
0x1403	0x00000500 + amplifier CAN node ID.
0x1404	0x80000000
0x1405	0x80000000
0x1406	0x80000000
0x1407	0x80000000

Object				Index	Sub-Index
PDO Type				**0x1400 – 7**	**2**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RW	--	Refer to Description	NO	R

Description
This object controls the behavior of the PDO when new data is received. The following codes are defined for receive PDOs:

Code	Behavior
0-240	The received data is held until the next SYNC message. When the SYNC message is received the data is applied.
241-253	Reserved.
254-255	The received data is applied to its mapped objects immediately upon reception.

Object				Index	Sub-Index
Receive PDO Mapping Parameters				**0x1600 – 0x1607**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	NO	--
Description These objects allow the mapping of each of the receive PDO objects to be configured.					

Object				Index	Sub-Index
Number of Mapped Objects				**0x1600 – 7**	**0**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RW	--	0 - 8	NO	R

Description
This value gives the total number of objects mapped to this PDO. It can be set to 0 to disable the PDO operation and must be set to 0 before changing the PDO mapping. Once the PDO mapping has been established by configuring the objects in sub-indexes 1 – 8, this value should be updated to indicate the actual number of objects mapped to the PDO.

Object				Index	Sub-Index
PDO Mapping				**0x1600 – 7**	**1 - 8**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	Refer to Description	NO	R

Description
When a PDO message is received, the data passed with the PDO message (up to 8 bytes) is used to update the objects mapped to the PDO. The values in the PDO mapping objects identify which object(s) the PDO data maps to. The first object is specified by the value in sub-index 1; the second object is identified by sub-index 2, etc. Each of the PDO mapping values consist of a 32-bit value structured as follows:

Bit	Description
0-7	Size (in bits) of the object being mapped. Must match the actual object size as defined in the object dictionary.
8-15	Sub-index of the object to be mapped.
16-31	Index of the object to be mapped.

Object				Index	Sub-Index
Transmit PDO Communication Parameters				**0x1800 – 0x1807**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	NO	--

Description
These objects allow configuration of communication parameters of each transmit PDO object. Sub-index 0 contains the number of sub-elements of this record.

Object				Index	Sub-Index
PDO COB-ID				**0x1800 – 7**	**1**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	Refer to Description	NO	R

Description
This object holds the CAN object ID used by the PDO. The ID is formatted as follows:

Bit	Description
0-10	11-bit identifier for standard (CAN 2.0A) identifiers, or the lower 11 bits for extended (CAN 2.0B) identifiers.
11-28	Upper 18 bits of extended identifiers. For standard identifiers these bits should be written as zeros.
29	Identifier format. This bit is clear for standard (11-bit) identifiers, and set for extended (29-bit) identifiers.
30	If set, remote transmit requests (RTR) are not allowed on this PDO. If clear, the PDO is transmitted in response to a remote request.
31	Identifies the PDO as valid if clear. If set, the PDO is disabled and its mapping may be changed.

Default Values
The default values for this object are specified in the DS-301 CANopen specification. These values are:

Index	Default ID
0x1800	0x00000180 + amplifier CAN node ID.
0x1801	0x00000280 + amplifier CAN node ID.
0x1802	0x00000380 + amplifier CAN node ID.
0x1803	0x00000480 + amplifier CAN node ID.
0x1804	0x80000000
0x1805	0x80000000
0x1806	0x80000000
0x1807	0x80000000

Object				Index	Sub-Index
PDO Type				**0x1800 – 7**	**2**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RW	--	Refer to Description	EVENT	R

Description
This object identifies which events trigger a PDO transmission:

Code	Behavior
0	The PDO is transmitted on the next SYNC message following a PDO event. See PDO Events, below, for a description of a PDO event.
1-240	The PDO is transmitted every N SYNC messages, where N is the PDO type code. For example, a PDO with type code 7 would be transmitted on every 7th SYNC message.
241-251	Reserved.
252	The PDO is transmitted on the SYNC message following a remote request.
253	The PDO is transmitted immediately in response to a remote request.
254-255	The PDO is transmitted immediately in response to an internal PDO event.

PDO Events

Some objects in the object dictionary have special PDO events associated with them. If such an object is mapped to a transmit PDO, then the PDO may be configured with a code that relies on this event to trigger its transmission. The codes that use PDO events are 0, 254, and 255.

An example of an object that has a PDO event associated with it is the Device Status object (index 0x6041). This object triggers an event to any mapped transmit PDO each time its value changes.

A transmit PDO which included this object in its mapping would have its event signaled each time the status register changed. Most objects in the object dictionary do not have PDO events associated with them. Those that do are identified by the word EVENT in the PDO Mapping fields of their descriptions.

Object				Index	Sub-Index
Transmit PDO Mapping Parameters				**0x1A00 – 0x1A07**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	NO	--

Description
These objects allow the mapping of each of the transmit PDO objects to be configured.

Object				Index	Sub-Index
Number of Mapped Objects				**0x1A00 – 7**	**0**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RW	--	0 - 8	NO	R

Description
Total number of objects mapped to this PDO. It can be set to 0 to disable the PDO operation, and must be set to 0 before changing the PDO mapping. Once the PDO mapping has been established by configuring the objects in sub-indexes 1 – 4, this value should be updated to indicate the actual number of objects mapped to the PDO.

Object				Index	Sub-Index
PDO Mapping				**0x1A00 – 7**	**1 - 8**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	Refer to Description	NO	R

Description
When a PDO message is transmitted, the data passed with the PDO message (up to 8 bytes) is gathered from the objects mapped to the PDO. The values in the PDO Mapping objects identify which object(s) the PDO data maps to. The first object is specified by the value in sub-index 1; the second object is identified by sub-index 2, etc. Each of the PDO mapping values consist of a 32-bit value structured as follows:

Bit	Description
0-7	Size (in bits) of the object being mapped. This value must match the actual object size as defined in the object dictionary.
8-15	Sub-index of the object to be mapped.
16-31	Index of the object to be mapped.

4 Network Management

This chapter describes the messages, methods, and objects used to manage devices on a CANopen network.

4.1 Network Management Overview

This section describes the objects, messages, and methods used to control the CANopen network.

4.1.1 Network Management Services and Objects

Network management services on the CANopen network include device state control, device monitoring, synchronization, and emergency handling. Special communication objects, as summarized herein, provide these services.

Object	Description
Network Management (NMT)	This object provides services to control the state of the device, including the initialization, starting, monitoring, resetting, and stopping of nodes. It also provides device-monitoring services (node-guarding and heartbeat).
Synchronization (SYNC)	Broadcast periodically by a specified device or the CANopen master to allow synchronized activity among multiple devices. The CAN message ID of the SYNC message is 80.
Time Stamp	Broadcast periodically by a specified device or the CANopen master to allow devices to synchronize their clocks.
Emergency	Transmitted by a device when an internal error occurs.

Network Manager Node

In general applications, a single node (such as a PC) is designated as the network manager. The network manager runs the software that issues all NMT messages. The network manager node can be the same node that runs the CANopen master application.

4.1.2 General Device State Control

State Machine

Every CANopen device implements a simple state machine. The machine defines three states (pre-operational, operational and stopped). The network manager application uses NMT messages to interact with the state machine and control state changes.

Device States

The following states are defined for AC Technology CANopen amplifiers:

State	Description
Pre-operational	Every node enters this state after power-up or reset. In this state, the device is not functional, but will communicate over the CANopen network. PDO transfers are not allowed in pre-operational state, but SDO transfers may be used.
Operational	This is the normal operating state for all devices. SDO and PDO transfers are both allowed.
Stopped	No communication is allowed in this state except for network management messages. Neither SDO nor PDO transfers may be used.

State Control Messages

NMT messages can be used to control state changes on network devices. The following NMT messages are sent by the network manager to control these state changes. Each of these messages can be either sent to a single node (by node ID), or broadcast to all nodes:

NMT Message	Cause/Effect
Reset	Causes each receiving node to perform a soft reset and come up in pre-operational state.
Reset communications	Causes each receiving node to reset its CANopen network interface to power-on state, and enter pre-operational state. This is not a full device reset, just a reset of the CANopen interface.
Pre-operational	Causes the receiving node(s) to enter pre-operational state. No reset is performed.
Start	Causes the node(s) to enter operational state.
Stop	Causes the node(s) to enter stopped state.

4.1.3 Device Monitoring

Monitoring Protocols

In addition to controlling state machines, NMT messages provide services for monitoring devices on the network. Monitoring services use one of two protocols: heartbeat and node guarding.

Heartbeat Protocol

The heartbeat protocol allows the network manager application to detect problems with a device or its network connection. The CANopen master configures the device to periodically transmit a heartbeat message indicating the device's current state (pre-operational, operational, or stopped). The network manager monitors the heartbeat messages. Failure to receive a node's heartbeat messages indicates a problem with the device or its connection to the network.

Node-guarding Protocol

The node-guarding protocol is similar to the heartbeat, but it allows both the device and the network manager to monitor the connection between them. The network manager configures the device (node) to expect node-guarding messages at some interval. The network manager then sends a message to the configured device at that frequency, and the device responds with a node-guarding message. This allows both the network manager and the device to identify a network failure if the guarding messages stop.

SYNC and High-resolution Time Stamp Messages

The SYNC message is a standard CANopen message used to synchronize multiple devices and to trigger the synchronous transmission of PDOs. In addition, to allow more accurate synchronization of device clocks, CANopen amplifiers use the optional high-resolution time stamp message specified in the Communication Profile. Normally, a single device produces both the SYNC message and the high-resolution time stamp message.

4.1.4 Time Stamp PDOs

The device designated as the time stamp producer should have a transmit PDO mapped for the high-resolution time stamp message. This PDO should be configured for synchronous transmission, based on the SYNC message. It is recommended to send this message approximately every 100 milliseconds.

Every other device (all time stamp consumers) should have a receive PDO mapped for the high resolution time stamp message. The message ID of each receive PDO used to receive a time stamp should match the ID of the transmit PDO used to send the time stamp.

Configuring the devices in this fashion causes the time stamp producer to generate a transmit PDO for every N sync messages. This PDO is received by each of the time stamp consumers on the network and causes them to update their internal system times based on the message content. The result is that all devices on the network act as though they share the same clock input, and remain tightly synchronized.

4.1.5 Emergency Messages

A device sends an 8-byte emergency message (EMCY) when an error occurs in the device. It contains information about the error type, and AC Tech-specific information. A device need only send one EMCY message per event. Any device can be configured to accept EMCY messages.

EMCY Message Structure

The EMCY message is structured as follows:

Bytes	Description
0, 1	Emergency error code. See EMCY Message CANopen Error Codes
2	Error register object value See Error Register object index 0x1001
3	Reserved for future use (0 for now).
4,5	Bit mask of active error conditions on the amplifier EMCY: AC Tech-Specific Error Conditions.
6,7	Reserved for future use (0 for now).

EMCY Message CANopen Error Codes

Bytes 0 and 1 of the EMCY message describe the standard CANopen error codes. AC Tech amplifiers support Error codes 00xx and 10xx.

Error Code (hex	Description	Error Code (hex	Description
00xx	Error Reset or No Error	62xx	User Software
10xx	Generic Error	63xx	Data Set
20xx	Current	70xx	Additional Modules
21xx	Current, device input side	80xx	Monitoring
22xx	Current inside the device	81xx	Communication
23xx	Current, device output side	8110	CAN Overrun (Objects lost)
30xx	Voltage	8120	CAN in Error Passive Mode
31xx	Mains Voltage	8130	Life Guard Error or Heartbeat Error
32xx	Voltage inside the device	8140	recovered from bus off
33xx	Output Voltage	8150	Transmit COB-ID
40xx	Temperature	82xx	Protocol Error
41xx	Ambient Temperature	8210	PDO not processed due to length error
42xx	Device Temperature	8220	PDO length exceeded
50xx	Device Hardware	90xx	External Error
60xx	Device Software	F0xx	Additional Functions
61xx	Internal Software	FFxx	Device specific. XX represents fault number specified in Programmer's manual

4.2 Network Management Objects

This section describes objects closely related to network management. They include:

COB-ID SYNC MESSAGE INDEX 0X1005

PRODUCER HEARTBEAT TIME INDEX 0X1017

EMERGENCY OBJECT ID INDEX 0X1014

EMERGENCY OBJECT ID INHIBIT TIME INDEX 0X1015

Object				Index	Sub-Index
COB-ID SYNC MESSAGE				**0X1005**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	See SYNC ID Format	NO	R

Description
This object establishes an amplifier as the SYNC producer and defines the CAN object ID (COBID) associated with the SYNC message. The SYNC message is a standard CANopen message type used to synchronize multiple devices on a CANopen network.
SYNC ID Format: The SYNC message ID is formatted as follows:

Bits	Description
0-10	Give the 11-bit identifier for standard (CAN 2.0A) identifiers, or the lower 11 bits for extended (CAN 2.0B) identifiers.
11-28	Give the upper 18 bits of extended identifiers. For standard identifiers these bits should be written as zeros.
29	Identifier format. This bit is clear for standard (11-bit) identifiers, and set for extended (29-bit) identifiers.
30	If set, the amplifier is configured as the SYNC message producer. This bit should be set in at most one amplifier on a network.
31	Reserved

Object				Index	Sub-Index
PRODUCER HEARTBEAT TIME				**0X1017**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	milliseconds	--	NO	R

Description
This object gives the frequency at which the amplifier will produce heartbeat messages. This object may be set to zero to disable heartbeat production. Note that only one of the two nodeguarding methods may be used at once. If this object is non-zero, then the heartbeat protocol is used regardless of the settings of the node-guarding time and lifetime factor.

Object				Index	Sub-Index
EMERGENCY OBJECT ID				**0X1014**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	--	NO	R

Description
CAN message ID used with the emergency object. Refer to the CANopen Application Layer and Communication Profile (DS 301).

Object				Index	Sub-Index
EMERGENCY OBJECT ID INHIBIT TIME				**0X1015**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	milliseconds	--	NO	R

Description
Inhibit time for the emergency object. Refer to the CANopen Application Layer and Communication Profile (DS 301).

5 Device Configuration and Control through Native Variables List

5.1 Native Control

Every aspect of the PositionServo can be manipulated by writing or reading the drive's internal variable(s). All variables are addressed by their respective index number. Variables are listed in the "complete list of variables" in the PositionServo Programmer's Manual.

Every variable can be interpreted as 32 bit Integer or as DOUBLE. Each variable has its native format inside and without regards to how the value was sent, it will be casted to its natural format by the drive. For example: Drive variable #30 is the drive's current limit. Its natural format is float. However this variable can be set as Integer type as well. Of course the fractional portion of the value can't be changed by Integer type but if integer precision is enough it can be used.

All variables are located in RAM but some of them have non-volatile copy in EPM memory. Implementation of the CANopen interface provides 4 types of objects to access the drive's variables. Objects with indexes 0x2000 – 0x23FF read or write RAM time copies of the variables as 32 bit integers. Objects with indexes 0x2400 – 0x 27FF read or write RAM time copies of the variables as Float numbers. Objects with indexes 0x3000-0x33FF write RAM and EPM copies and read EPM copies of the variables as Integer 32 numbers. Objects with indexes 0x3400-0x37FF write RAM and EPM copies and read EPM copies of the variables as Float numbers.

This organization gives the user unified access to variables and simplifies implementation of communication software.

5.2 Objects to Access the Drive's RAM Variables

Object				Index	Sub-Index
RAM VARIABLES				**0X2000-0X23FF**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	--	NO	R
Description Objects in this range Read or Write corresponding internal drive's RAM copies of the variables as Integer 32 values. Object Index to access particular variable calculated as: Object Index = 0x2000 + VarID , where: VarID is the variable ordinal index from the variable table. (Variable table is provided in Programmer's Manual). For Units and Range of every particular variable please refer to Complete list of Variables in Programmer's Manual.					

Object				Index	Sub-Index
RAM VARIABLES				**0X2400-0X27FF**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	R/W	--	--	NO	R
Description Objects in this range Read or Write corresponding internal drive's RAM copies of the variables as Float values. Object Index to access particular variable calculated as: Object Index = 0x2400 + VarID, where: VarID is the variable ordinal index from the variable table. (Variable table is provided in Programmer's Manual). For Units and Range of every particular variable please refer to Complete list of Variables in Programmer's Manual.					

Object				Index	Sub-Index
RAM VARIABLES				**0X3000-0X33FF**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	R/W	--	--	NO	RF
Description Objects in this range: • Write corresponding internal drive's RAM and EPM copies of the variables as Integer 32 values. • Read corresponding internal drive's EPM copies of the variables as Integer 32 values Object Index to access particular variable calculated as: Object Index = 0x2000 + VarID, where: VarID is the variable ordinal index from the variable table. (Variable table is provided in Programmer's Manual). For Units and Range of every particular variable please refer to Complete list of Variables in Programmer's Manual.					

Object				Index	Sub-Index
RAM VARIABLES				**0X3400-0X37FF**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	R/W	--	--	NO	RF
Description Objects in this range: • Write corresponding internal drive's RAM and EPM copies of the variables as Float values. • Read corresponding internal drive's EPM copies of the variables as Float values Object Index to access particular variable calculated as: Object Index = 0x2000 + VarID , where: VarID is the variable ordinal index from the variable table. (Variable table is provided in Programmer's Manual). For Units and Range of every particular variable please refer to Complete list of Variables in Programmer's Manual.					

6 Device Control, Configuration and Status

This chapter describes a wide range of device control, configuration, and status methods and objects.

6.1 Device Control and Status Overview

This section describes the objects and functions used to control the status of an amplifier including:

6.1.1 Control Word, Status Word, and Device Control Function

Device Control Function Block

The Profile for Drives and Motion Control describes control of the amplifier in terms of a control function block with two major sub-elements: the operation modes and the state machine.

Control and Status Words

As illustrated in Figure 8, the Control Word object (index 0x6040, paragraph 6.2, page 34) manages device mode and state changes. The Status Word object (index 0x6041, paragraph 6.2, page 34) identifies the current state of the amplifier. Other factors affecting control functions include: digital input signals, fault conditions, and settings in various dictionary objects.

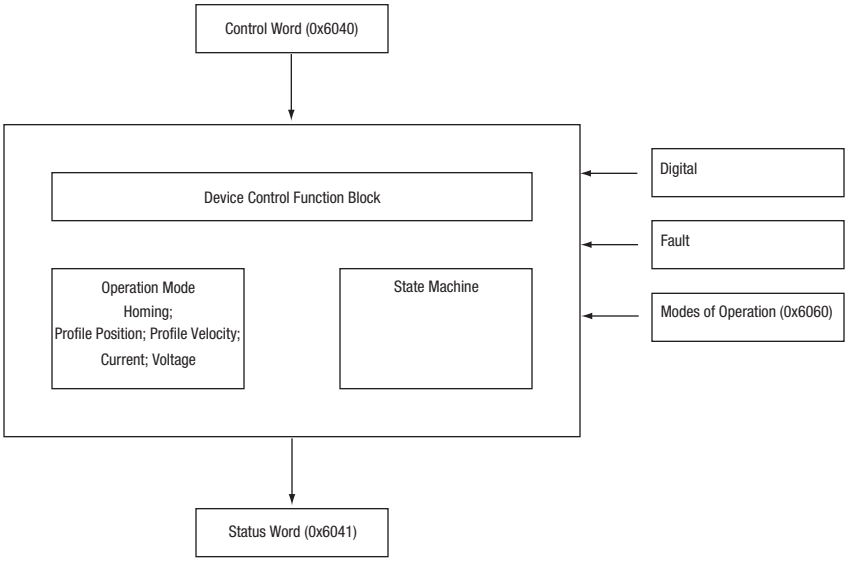

Figure 8: Control and Status Words

Operation Modes

As mentioned elsewhere in this manual, AC Tech CANopen amplifiers support these operation modes:

- Homing
- Profile position
- Profile velocity
- Current follower
- Velocity follower

State Machine Nesting and States

Note that the Communication Profile also specifies a state machine, with three states: preoperational, operational, and stopped. The entire device control function block described in this chapter, including the device state machine, operates in the operational state of the Communication Profile state machine.

The state machine describes the status and possible control sequences of the drive. The state also determines which commands are accepted. States are described herein:

State	Description
Not Ready to Switch On	Low-level power (e.g. _ 15V, 5V) has been applied to the drive. The drive is being initialized or is running self-test. A brake, if present, is applied in this state. The drive function is disabled.
Switch On Disabled	Drive initialization is complete. The drive parameters have been set up. Drive parameters may be changed. The drive function is disabled.
Ready to Switch On	The drive parameters may be changed. The drive function is disabled.
Switched On	High voltage has been applied to the drive. The power amplifier is ready. The drive parameters may be changed. The drive function is disabled.
Operation Enable	No faults have been detected. The drive function is enabled and power is applied to the motor. The drive parameters may be changed. (This corresponds to normal operation of the drive.)

Quick Stop Active	The drive parameters may be changed. The quick stop function is being executed. The drive function is enabled and power is applied to the motor. If the 'Quick-Stop-Option-Code' is switched to 5 (Stay in Quick-Stop), the amplifier cannot exit the Quick-Stop-State, but can be transmitted to 'Operation Enable' with the command 'Enable Operation."
Fault Reaction Active	The drive parameters may be changed. A non-fatal fault has occurred in the drive. The quick stop function is being executed. The drive function is enabled and power is applied to the motor.
Fault	The drive parameters may be changed. A fault has occurred in the drive. The drive function is disabled.

6.1.2 State Changes Diagram

The diagram illustrated in Figure 9 is from the Profile for Drives and Motion Control and shows the possible state change sequences of an amplifier. Each transition is numbered and described in the legend herein.

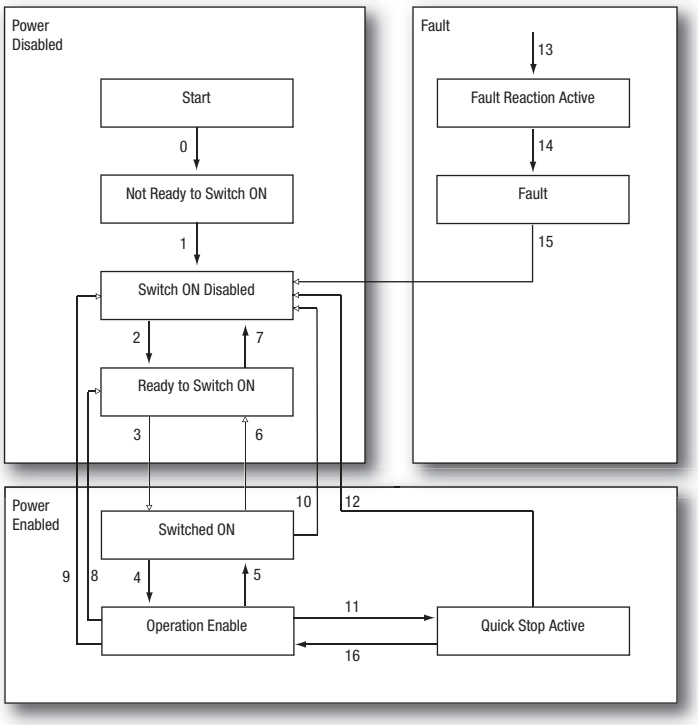

Figure 9: State Changes

State Changes Diagram Legend

#	From State	To State	Event/Action
0	Startup	Not Ready to Switch On	Event: Reset. Action: The drive self-tests and/or self-initializes
1	Not Ready to Switch On	Switch On Disabled	Event: The drive has self-tested and/or initialized successfully. Action: Activate communication and process data monitoring
2	Switch On Disabled	Ready to Switch On	Event: 'Shutdown' command received from host. Action: None
3	Ready to Switch On	Switched On	Event: 'Switch On' command received from host. Action: The power section is switched on if it is not already switched on
4	Switched On	Operation Enable	Event: 'Enable Operation' command received from host. Action: The drive function is enabled.
5	Operation Enable	Switched On	Event: 'Disable Operation' command received from host. Action: The drive operation is disabled.
6	Switched On	Ready to Switch On	Event: 'Shutdown' command received from host. Action: The power section is switched off.
7	Ready to Switch On	Switch On Disabled	Event: 'Quick stop' command received from host. Action: None
8	Operation Enable	Ready to Switch On	Event: 'Shutdown' command received from host. Action: The power section is switched off immediately, and the motor is free to rotate if unbraked
9	Operation Enable	Switch On Disabled	Event: 'Disable Voltage' command received from host. Action: The power section is switched off immediately, and the motor is free to rotate if unbraked
10	Switched On	Switch On Disabled	Event: 'Disable Voltage' or 'Quick Stop' command received from host.Action: The power section is switched off immediately, and the motor is free to rotate if unbraked
11	Operation Enable	Quick Stop Active	Event: 'Quick Stop' command received from host. \ Action: The Quick Stop function is executed.
12	Quick Stop Active	Switch On Disabled	Event: 'Quick Stop' is completed or 'Disable Voltage' command received from host. This transition is possible if the Quick-Stop-Option-Code is not 5 (Stay in Quick-Stop) Action: The power section is switched off.
13	FAULT	Fault Reaction Active	Event: A fatal fault has occurred in the drive. Action: Execute appropriate fault reaction.
14	Fault Reaction Active	Fault	Event: The fault reaction is completed. Action: The drive function is disabled. The power section may be switched off.
15	Fault	Switch On Disabled	Event: 'Fault Reset' command received from host. Action: A reset of the fault condition is carried out if no fault exists currently on the drive. After leaving the 'Fault' state the Bit 'Fault Reset' of the Control Word has to be cleared by the host.
16	Quick Stop Active	Operation Enable	Event: 'Enable Operation' command received from host. This transition is possible if the Quick-Stop-Option-Code is 5, 6, 7, or 8 (see the Quick Stop Option Code object, index 0x6085, paragraph 6.2, page 35). Action: The drive function is enabled.

6.2 Device Control and Status Objects

This section describes the objects used to control the status of an amplifier including:

CONTROL WORD	INDEX 0X6040
STATUS WORD	INDEX 0X6041
QUICK STOP OPTION CODE	INDEX 0X605A
SHUTDOWN OPTION CODE	INDEX 0X605B
DISABLE OPERATION OPTION CODE	INDEX 0X605C
MODE OF OPERATION	INDEX 0X6060
MODE OF OPERATION DISPLAY	INDEX 0X6061
REFERENCE SOURCE	INDEX 0X2025
MANUFACTURER STATUS REGISTER	INDEX 0X1002

Object				Index	Sub-Index
CONTROL WORD				**0X6040**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	--	See Description	YES	R

Description
This object is used to controls the state of the amplifier. It can be used to enable / disable the amplifier output, start, and abort moves in all operating modes, and clear fault conditions.
Control Word Bit Mapping: The value programmed into this object is bit-mapped as follows:

Bits	Description
0	Switch On. This bit must be set to enable the amplifier.
1	Enable Voltage. This bit must be set to enable the amplifier.
2	Quick Stop. If this bit is clear, then the amplifier is commanded to perform a quick stop.
3	Enable Operation. This bit must be set to enable the amplifier.
4-6	Operation mode specific. Descriptions appear in the sections that describe the various operating modes
7	Reset Fault. A low-to-high transition of this bit makes the amplifier attempt to clear any latched fault condition.
8-15	Reserved for future use.

Object				Index	Sub-Index
STATUS WORD				**0X6041**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RO	--	Refer to Description	YES	--

Description
This object identifies the current state of the amplifier and is bit-mapped as follows:

Bits	Description
0	Ready to switch on.
1	Switched on.
2	Operation Enabled. Set when the amplifier is enabled.
3	Fault. If set, a latched fault condition is present in the amplifier.
4	Voltage enabled. Set if the amplifier bus voltage is above the minimum necessary for normal operation.
5	Quick Stop. When clear, the amplifier is performing a quick stop.
6	Switch on disabled.
8	Set if the last trajectory was aborted rather then finishing normally. Remote. Set when the amplifier is being controlled by the CANopen interface. When clear, the amplifier may be monitored through this interface, but some other input source is controlling it.
10	Target Reached. This bit is set when the motor has come to rest at the target position. This bit is cleared when a trajectory is running, or when the position error is greater then the tracking window value.
11	Internal Limit Active. This bit is set when amplifier limits current.
12-13	The meanings of these bits are operation mode specific.
14-15	Set when the amplifier is performing a move and cleared when the trajectory finishes. This bit is cleared immediately at the end of the move, not after the motor has settled into position.

Object				Index	Sub-Index
QUICK STOP OPTION CODE				**0X605A**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	--	Refer to Description	NO	R

Description
This object defines the behavior of the amplifier when a quick stop command is issued. The following values are defined:

Value	Description
0	Disable the amplifier's outputs
1	Slow down using the slow down ramp (i.e. the normal move deceleration value). When the move has been successfully aborted the amplifier's state will transition to the 'switch on disabled' state.
2	Slow down using the quick stop ramp, then transition to 'switch on disabled'.
5	Slow down using the slow down ramp. The amplifier state will remain in the 'quick stop' state after the move has been finished.
6	Slow down using the quick stop ramp and stay in 'quick stop' state.

All other values will produce unspecified results and should not be used.

Object				Index	Sub-Index
SHUTDOWN OPTION CODE				**0X605B**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	--	Refer to Description	NO	R

Description
This object defines the behavior of the amplifier when the amplifier's state is changed from 'operation enabled' to 'ready to switch on'. The following values are defined:

Value	Description
0	Disable the amplifier's outputs
1	Slow down using the slow down ramp (i.e. the normal move deceleration value).

All other values will produce unspecified results and should not be used.

Object				Index	Sub-Index
DISABLE OPERATION OPTION CODE				**0X605C**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	--	Refer to Description	NO	R

Description
This object defines the behavior of the amplifier when the amplifier's state is changed from 'operation enabled' to 'switched on'. The following values are defined:

Value	Description
0	Disable the amplifier's outputs
1	Slow down using the slow down ramp (i.e. the normal move deceleration value).

All other values will produce unspecified results and should not be used.

Object				Index	Sub-Index
MODE OF OPERATION				**0X6060**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RW	--	Refer to Description	YES	R

Description
This object selects the amplifier's mode of operation. The modes of operation presently supported by this device are:

Mode	Description
-1	Single loop Current follower
-2	Single loop Velocity follower
1	Profile Position mode
3	Profile Velocity mode
6	Homing mode

The amplifier will not accept other values. Note that there may be some delay between setting the mode of operation and the amplifier assuming that mode. To read the active mode of operation, use object 0x6061.

Object				Index	Sub-Index
MODE OF OPERATION DISPLAY				**0X6061**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RO	--	Refer to Description	YES	--

Description
This object displays the current mode of operation. Refer to Mode of Operation (index 0x6060, paragraph 6.2, page 35).

Object				Index	Sub-Index
REFERENCE SOURCE				**0X2025**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	--	Refer to Description	YES	--

Description
This object configures type of the reference for Velocity follower or Current follower modes. Refer to Mode of Operation (index 0x6060, paragraph 6.2, page 35) ,NON PROFILED OPERATING MODES (paragraph 8, page 62) and Control Loop Configuration (paragraph 7, page 51). Possible values are:

Value	Description
0	Reference configured to be Analog input #1
1	Reference configured to a digital value. Object number depends on the operating mode

Object				Index	Sub-Index
MANUFACTURER STATUS REGISTER				**0X1002**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	Refer to Description	YES	--

Description
This 32-bit object is a bit-mapped status register with the following fields:

Bit in register	Description
0	Set when drive enabled
1	Set if DSP subsystem at any fault
2	Set if drive has a valid program
3	Set if byte-code or system or DSP at any fault
4	Set if drive has a valid source code
5	Set if motion completed and target position is within specified limits
6	Set when scope is triggered and data collected
7	Set if motion stack is full
8	Set if motion stack is empty
9	Set if byte-code halted
10	Set if byte-code is running
11	Set if byte-code is set to run in step mode
12	Set if byte-code is reached the end of program
13	Set if current limit is reached
14	Set if byte-code at fault
15	Set if no valid motor selected
16	Set if byte-code at arithmetic fault
17	Set if byte-code at user fault
18	Set if DSP initialization completed
19	Set if registration has been triggered
20	Set if registration variable was updated from DSP after last trigger
21	Set if motion module at fault
22	Set if motion suspended
23	Set if program requested to suspend motion
24	Set if system waits completion of motion
25	Set if motion command completed and motion Queue is empty
26	Set if byte-code task requested reset
27	If set interface control is disabled. This flag is set/clear by ICONTROL ON/OFF statement.
28	Set if positive limit switch reached
29	Set if negative limit switch reached
30	Events disabled. All events disabled when this flag is set. After executing EVENTS ON all events previously enabled by EVENT EventName ON statements become enabled again
31	Set if 'under voltage' condition detected

Object				Index	Sub-Index
EXTENDED STATUS REGISTER				**0X2053**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	Refer to Description	YES	--

Description
This 32-bit object is a bit-mapped extended status register with the following fields:

Bit in register	Description
0	DSP subsystem in run mode
1	Velocity in specified velocity window
2	Registration input detected
3	DSP system in fault state
4	DSP system ready to run
5	Velocity within Zero limits
6	Reserved
7	Reserved
8	Reserved
9	Encoder lost
10	Encoder data invalid
11	Regen output ON (active)
12	Motor powered
13	Over-current flag
14	Reserved
15	Reserved
16	Event processor is running
17	Events are set to run in step mode
18	Reserved
19	Set if parameter's flash file is not valid (checksum doesn't match)
20	Set if indexer program protected by password
21	If set then PositionServo is in Test Mode

6.3 Error Management Objects

This section describes the objects used to view error status and define error limits and error handling. They include:

PRE-DEFINED ERROR OBJECT	INDEX 0X1003	
NUMBER OF ERRORS	INDEX 0X1003	SUB-INDEX 0
STANDARD ERROR FIELD	INDEX 0X1003	SUB-INDEX 1-8
ERROR REGISTER	INDEX 0X1001	
FAULT STATUS	INDEX 0X2009	

This is the FAULT History access:

Object				Index	Sub-Index
PRE-DEFINED ERROR OBJECT				**0X1003**	**--**
Type	Access	Units	Range	Map PDO	Memory
Array	RW	--	--	NO	R

Description
This object provides an error history. Each sub-index object holds an error that has occurred on the device and has been signaled via the Emergency Object. Refer to Emergency Messages. The entry at sub-index 0 contains the number of errors that are recorded in the array starting at sub-index 1. Each new error is stored at sub-index 1. Older errors move down the list.

Object				Index	Sub-Index
NUMBER OF ERRORS				**0X1003**	**0**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RW	--	0 - 8	NO	R
Description This object provides the number of errors in the error history (number of sub-index objects 1-8). Writing a 0 deletes the error history (empties the array). Writing a value higher than 0 results in an error.					

Object				Index	Sub-Index
STANDARD ERROR FIELD				**0X1003**	**1-8**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RW	--	--	NO	R
Description One sub-index object for each error found, up to 8 errors. Each is composed of a 16-bit error code and a 16-bit additional error information field. The error code is contained in the lower 2 bytes (LSB) and the additional information is included in the upper 2 bytes (MSB).					

Object				Index	Sub-Index
ERROR REGISTER				**0X1001**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 8	RO	--	Refer to Description	YES	--

Description
This object is a bit-mapped list of error conditions present in the amplifier. The bits used in this register are mapped as follows:

Bits	Description
0	Generic error. This bit is set any time there is an error condition in the amplifier.
1	Current error. Indicates either a short circuit on the motor outputs, or excessive current beyond amplifier capability was detected.
2	Voltage error. The DC bus voltage supplied to the amplifier is either over or under the amplifier's limits.
3	Temperature error. Either the amplifier or motor is over temperature. Note that the amplifier will only detect a motor over temperature condition if an amplifier input has been configured to detect this condition.
4	Communication error. The amplifier does not presently use this bit.
5	Reserved. Undefined
6	Reserved. Always 0.
7	The following errors cause this bit to be set; tracking error, limit switch active.

Object				Index	Sub-Index
FAULT STATUS				**0X2009**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	Refer to Description	YES	R

Description

This object allows latching fault conditions to be viewed. When the object is read, any set bit will indicate a latching fault condition:

Fault ID	Associated Flags*	Description
1	1, 3	Overvoltage
2	1, 3	Invalid hall sensors code
3	1, 3	Overcurrent
4	1, 3	Overtemperature
5	1, 3	Reserved
6	1, 3	Overspeed. (Overspeed limit set by motor capability in motor file)
7	1, 3	Position error excess.
8	1, 3	Attempt to enable while motor data array invalid or motor was not selected.
9	1,3	Motor overtemperature switch activated
10	1,3	Subprocessor error
11-13	-	Reserved
14	1,3	Undervoltage
15	1,3	Hardware current trip protection
16	-	Reserved
17	3	Unrecoverable error.
18	16	Division by zero
19	16	Arithmetic overflow
20	3	Subroutine stack overflow. Exceeded 16 levels subroutines stack depth.
21	3	Subroutine stack underflow. Attempt to execute RETURN statement without preceding call to subroutine.
22	3	Variable evaluation stack overflow. Expression too complicated for compiler to process.
23	21	Motion Queue overflow. 32 levels depth exceeded
24	21	Motion Queue underflow. Last queued MDV statement has non 0 target velocity
25	3	Unknown opcode. Byte code interpreter error
26	3	Unknown byte code. Byte code interpreter error
27	21	Drive disabled. Attempt to execute motion while drive is disabled.
28	16, 21	Accel too big. Motion statement parameters result in calculating too big Accel value for system to handle
29	16, 21	Accel too small. Motion statement parameters result in calculating too small Accel value for system to handle
30	16, 21	Velocity too big. Motion statement parameters result in calculating too big velocity value for system to handle
31	16, 21	Velocity too small. Motion statement parameters result in calculating too small velocity value for system to handle

* Associated Flags in Status Register (Object Index 0x1002, paragraph 6.2, page 36)

6.4 Basic Amplifier Configuration Objects

Objects described in this section provide access to basic amplifier parameters. They include:

DEVICE TYPE	INDEX 0X1000	
DEVICE NAME	INDEX 0X1008	
HARDWARE VERSION STRING	INDEX 0X1009	
SOFTWARE VERSION NUMBER	INDEX 0X100A	
IDENTITY OBJECT	INDEX 0X1018	
VENDOR ID	INDEX 0X1018	SUB-INDEX 1
PRODUCT CODE	INDEX 0X1018	SUB-INDEX 2
REVISION NUMBER	INDEX 0X1018	SUB-INDEX 3
SERIAL NUMBER	INDEX 0X1018	SUB-INDEX 4
AMPLIFIER NAME	INDEX 0X2060	
CANOPEN NETWORK CONFIGURATION	INDEX 0X20EA	
SUPPORTED DRIVE MODES	INDEX 0X6502	
AMPLIFIER MODEL NUMBER	INDEX 0X6503	
AMPLIFIER MANUFACTURER	INDEX 0X6504	
MANUFACTURER'S WEB ADDRESS	INDEX 0X6505	
AMPLIFIER DATA	INDEX 0X6510	
AMPLIFIER SERIAL NUMBER	INDEX 0X6510	SUB-INDEX 1
AMPLIFIER DATE CODE	INDEX 0X6510	SUB-INDEX 2
AMPLIFIER PEAK CURRENT	INDEX 0X6510	SUB-INDEX 3
AMPLIFIER CONTINUOUS CURRENT	INDEX 0X6510	SUB-INDEX 4
AMPLIFIER PEAK CURRENT TIME	INDEX 0X6510	SUB-INDEX 5
AMPLIFIER MAXIMUM VOLTAGE	INDEX 0X6510	SUB-INDEX 6
AMPLIFIER MINIMUM VOLTAGE	INDEX 0X6510	SUB-INDEX 7
AMPLIFIER MAXIMUM TEMPERATURE	INDEX 0X6510	SUB-INDEX 8
AMPLIFIER CURRENT LOOP PERIOD	INDEX 0X6510	SUB-INDEX 9
AMPLIFIER SERVO LOOP PERIOD	INDEX 0X6510	SUB-INDEX 10
AMPLIFIER TYPE CODE	INDEX 0X6510	SUB-INDEX 11
DEVICE TYPE	INDEX 0X67FF	

Object				Index	Sub-Index
DEVICE TYPE				**0X1000**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	Refer to Description	NO	--
Description Describes the type of device and its functionality. This 32-bit value is composed of two 16-bit components. The lower two bytes identify the device profile supported by the device. This amplifier supports the DSP402 device profile, indicated by the value 0x0192. The upper two bytes give detailed information about the type of motors the drive can control. The bit mapping of this value is defined by the Profile for Drives and Motion Control. CANopen amplifiers, this value is 0x0002, indicating that supports servo devices.					

Object				Index	Sub-Index
DEVICE NAME				**0X1008**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	--
Description An ASCII string which gives the amplifier's model number.					

Object				Index	Sub-Index
HARDWARE VERSION STRING				**0X1009**	**--**
Type	Access	Units	Range	Map PDO	Memory
String	Const	--	--	NO	--
Description Describes amplifier hardware version.					

Object				Index	Sub-Index
SOFTWARE VERSION NUMBER				**0X100A**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	--
Description Contains an ASCII string listing the software version number of the amplifier.					

Object				Index	Sub-Index
IDENTITY OBJECT				**0X1018**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RO	--	--	NO	--
Description This object can uniquely identify an amplifier by unique manufacturer ID, serial number, and product revision information. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
VENDOR ID				**0X1018**	**1**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	0x0000019C	NO	--
Description A unique identifier assigned to AC Technology Corp. The value of this identifier is fixed at: 0x0000019C					

Object				Index	Sub-Index
PRODUCT CODE				**0X1018**	**2**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	Refer to Description	NO	--
Description A value that uniquely identifies the amplifier type. The currently defined values for this object are: Value Product 940 940 Position Servo					

Object				Index	Sub-Index
REVISION NUMBER				**0X1018**	**3**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	--	NO	--
Description Identifies the revision of the CANopen interface.					

Object				Index	Sub-Index
SERIAL NUMBER				**0X1018**	**4**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	--	NO	--
Description Identifies the amplifier's serial number.					

Object				Index	Sub-Index
AMPLIFIER NAME				**0X2002**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible string	RW	--	--	NO	F
Description This object may be used to assign a name an amplifier. The data written here is stored to flash memory and is not used by the amplifier. Although this object is documented as holding a string (i.e. ASCII data), any values may be written here.					

Object				Index	Sub-Index
CANOPEN NETWORK CONFIGURATION				**0X20EA**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 16	RW	--	Refer to Description	NO	RF

Description
This object is used to configure the CANopen network bit rate and node ID for the amplifier. Values written here are stored to flash memory. The new network configuration will not take effect until the amplifier is reset.

Bit	Description
0-6	Node ID value.
7	Reserved for future use.
8-11	Reserved
12-15	Network bit rate setting.

On power-up (or after a reset), the amplifier will determine its node ID programmed in bits 0-6. Note that the node ID value zero is not a legal CANopen ID, and will result in unspecified action.The network bit rate is encoded as one of the following values:

Code	Bit Rate (bits / second)
0	1,000,000
1	800,000
2	500,000
3	250,000
4	125,000
5	50,000
6	20,000
7-15	Reserved for future use

Object				Index	Sub-Index
SUPPORTED DRIVE MODES				**0X6502**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	Refer to Description	NO	--

Description
This bit-mapped value gives the modes of operation supported by the amplifier. The standard device profile (DSP402) defines several modes of operation. Each mode is assigned one bit in this variable. A drive indicates its support for the mode of operation by setting the corresponding bit. The modes of operation supported by this device, and their corresponding bits in this object, are as follows:

Bit	Description
0	*Position profile mode*
1	Velocity mode
2	*Profile velocity mode*
3	Profile torque mode
5	*Homing mode*
6	Interpolated Position Mode
7-15	Reserved
16	*Single loop current follower (AC Technology specific)*
17	*Single loop velocity follower (AC Technology specific)*

The current version of amplifier firmware supports only modes that are *initialized*. Their bits are st in the word and therefore the expected value of this object is 0x00030025.

Object				Index	Sub-Index
AMPLIFIER MODEL NUMBER				**0X6503**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	--

Description
This ASCII string gives the amplifier model number (ID string).

Object				Index	Sub-Index
AMPLIFIER MANUFACTURER				**0X6504**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	--

Description
This ASCII string identifies the amplifier's manufacturer as "AC Tech Corp."

Object				Index	Sub-Index
MANUFACTURER'S WEB ADDRESS				**0X6505**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	--

Description
This ASCII string gives the web address of AC Technology Corp. (www.actech.com)

Object				Index	Sub-Index
AMPLIFIER DATA				**0X6510**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RO	--	--	NO	--

Description
This record lists various amplifier parameters. Sub-index 0 contains the number of sub-elements of this record.

Object				Index	Sub-Index
AMPLIFIER SERIAL NUMBER				**0X6510**	**1**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RO	--	--	NO	--
Description Gives the amplifier serial number.					

Object				Index	Sub-Index
AMPLIFIER BUILD AND DATE CODE NUMBER				**0X6510**	**2**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	--
Description This ASCII string gives the manufacturing build code.					

Object				Index	Sub-Index
AMPLIFIER PEAK CURRENT				**0X6510**	**3**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	0.01 A	--	NO	--
Description The amplifier's peak current rating in 0.01-amp units. Peak current rating for PositionServo amplifiers are 3X continues current (sub-index 4).					

Object				Index	Sub-Index
AMPLIFIER CONTINUOUS CURRENT				**0X6510**	**4**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	0.01 A	--	NO	--
Description The amplifier's continuous current rating in 0.01-amp units.					

Object				Index	Sub-Index
AMPLIFIER PEAK CURRENT TIME				**0X6510**	**5**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	milliseconds	--	NO	--
Description The time the amplifier is rated to output peak current in milliseconds. Expected value for PositionServo amplifier 2000 (mS)					

Object				Index	Sub-Index
AMPLIFIER MAXIMUM VOLTAGE				**0X6510**	**6**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	0.1 V	--	NO	--
Description Maximum bus voltage rating for amplifier in 0.1-volt units.					

Object				Index	Sub-Index
AMPLIFIER MINIMUM VOLTAGE				**0X6510**	**7**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	0.1 V	--	NO	--
Description Minimum bus voltage rating for amplifier in 0.1-volt units. Over-voltage hysteresis for amplifier in 0.1-volt units.					

Object				Index	Sub-Index
AMPLIFIER MAXIMUM TEMPERATURE				**0X6510**	**8**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	degrees C	--	NO	--
Description Temperature limit for amplifier in degrees centigrade. PositionServo has set to 100 degree centigrade.					

Object				Index	Sub-Index
AMPLIFIER CURRENT LOOP PERIOD				**0X6510**	**9**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	10ns	--	NO	--
Description Current loop update period in 10-nanosecond units.					

Object				Index	Sub-Index
AMPLIFIER SERVO LOOP PERIOD				**0X6510**	**10**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	--	--	NO	--
Description Servo loop update period as a multiple of the current loop period. (4 for PositionServo models.)					

Object				Index	Sub-Index
AMPLIFIER TYPE CODE				**0X6510**	**11**
Type	Access	Units	Range	Map PDO	Memory
String	RO	--	Refer to Description	NO	--

Description
3 digit string "X XX" that identifies the type of amplifier in terms of mains voltage (V) and output current capability (A rms).

Digit 1	Description	Digits 2 & 3	Description		
X	B = 240V AC	XX	02 = 2A rms	06 = 6A rms	10 = 10A rms
	C = 480V AC		04 = 4A rms	08 = 8A rms	12 = 12A rms
			05 = 5A rms	09 = 9A rms	18 = 18A rms

Object				Index	Sub-Index
DEVICE TYPE				**0X67FF**	**--**
Type	Access	Units	Range	Map PDO	Memory
Unsigned 32	RO	--	--	NO	--
Description Holds the same data as object 0x1000. Repeated as required by the CANopen specification.					

6.5 Basic Motor Configuration Objects

Objects described in this section provide access to basic motor parameters. They include:

MOTOR MODEL NUMBER	INDEX 0X6403	
MOTOR MANUFACTURER	INDEX 0X6404	
MOTOR DATA	INDEX 0X6410	
MOTOR TYPE	INDEX 0X6410	SUB-INDEX 1
MOTOR TYPE	INDEX 0X6410	SUB-INDEX 2
SYMBOLIC MOTOR MODEL	INDEX 0X6410	SUB-INDEX 3
MOTOR VENDOR NAME	INDEX 0X6410	SUB-INDEX 4
FEEDBACK CONFIGURATION	INDEX 0X6410	SUB-INDEX 5
HALLCODE INDEX	INDEX 0X6410	SUB-INDEX 6
HALL OFFSET	INDEX 0X6410	SUB-INDEX 7
Zero OFFSET	INDEX 0X6410	SUB-INDEX 8
ICTRL (RESERVED)	INDEX 0X6410	SUB-INDEX 9
MOTOR INERTIA	INDEX 0X6410	SUB-INDEX 10
MOTOR BACK EMF	INDEX 0X6410	SUB-INDEX 11
MOTOR TORQUE CONSTANT	INDEX 0X6410	SUB-INDEX 12
MOTOR INDUCTANCE	INDEX 0X6410	SUB-INDEX 13
MOTOR RESISTANCE	INDEX 0X6410	SUB-INDEX 14
MOTOR MAX CONT. CURRENT	INDEX 0X6410	SUB-INDEX 15
MOTOR MAX VELOCITY	INDEX 0X6410	SUB-INDEX 16
MOTOR POLES	INDEX 0X6410	SUB-INDEX 17
ENCODER COUNT	INDEX 0X6410	SUB-INDEX 18
MOTOR NOMINAL TERMINAL VOLTAGE	INDEX 0X6410	SUB-INDEX 19
MOTOR FEEDBACK DEVICE TYPE	INDEX 0X6410	SUB-INDEX 19

Object				Index	Sub-Index
MOTOR MODEL NUMBER				**0X6403**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RW	--	--	NO	F
Description This object gives a location to store the motor's model number for future reference. M_MODEL. Note that this parameter is always stored to non-volatile memory on the amplifier. The programmed value is preserved across power cycles.					

Object				Index	Sub-Index
RESERVED				**0X6404**	**--**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	--	NO	F
Description Reserved for future use.					

Object				Index	Sub-Index
MOTOR DATA				**0X6410**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	NO	--
Description This record holds a variety of motor parameters. Note that all motor parameters are stored to non-volatile memory on the amplifier. The programmed values are preserved across power cycles. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
MOTOR ID				**0X6410**	**1**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	Refer to Description	NO	F
Description Returns motor catalog ID. This field needs to be set to "-1" for any motor which doesn't have AC Technology Corp. catalog ID assigned.					

Object				Index	Sub-Index
MOTOR TYPE				**0X6410**	**2**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	Refer to Description	NO	F

Description
M_SERIES
Defines the type of motor connected to the amplifier. For proper operation, this object must be set to one of the following values:

Type	Description
1-999	Rotary AC Synchronous servo motor
1000-1999	Linear AC Synchronous servo motor
2000-2999	Rotary AC induction motor

Object				Index	Sub-Index
RESERVED				**0X6410**	**3**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RO	--	Refer to Description	NO	F
Description This index is reserved for future use.					

Object				Index	Sub-Index
MOTOR VENDOR NAME				**0X6410**	**4**
Type	Access	Units	Range	Map PDO	Memory
Visible String	RW	--	--	NO	F
Description Defines symbolic motor's vendor name. Example: "Lenze"					

Object				Index	Sub-Index
FEEDBACK CONFIGURATION				**0X6410**	**5**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	Refer to Description	NO	F

Description
Describes motor's feedback device configuration data as follows:

Value	Description
0	reserved
1	encoder feedback
2	resolver feedback
3	Absolute encoder (BiSS, SPI)
4	Absolute encoder (EnDat)
5	Absolute encoder (HyperFace)

Object				Index	Sub-Index
HALL CODE				**0X6410**	**6**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 - 5	NO	F

Description
Defines motor hallcode index. Index shows hall sensors mapping to the corresponding motor phases U,V or W:

Code	Hall ordering
0	U V W
1	U W V
2	V U W
3	V W U
4	W V U
5	W U V

Object				Index	Sub-Index
HALL OFFSET				**0X6410**	**7**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RO	--	0	NO	F

Description
Reserved for future use by AC Technology Corp. Must be set to 0.

Object				Index	Sub-Index
Zero OFFSET				**0X6410**	**8**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	0.1 mech. degree	36000 - 36000	NO	F

Description
Description (Value /100) * (65536/360) Value/100 must be modulo 360 - positive. Sets the resolver offset value in 0.01 mechanical degree. Must be 0 for motors with incremental encoders.

Object				Index	Sub-Index
ICTRL (RESERVED)				**0X6410**	**9**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	--	NO	F

Description
Reserved. Must be set to 0.

Object				Index	Sub-Index
MOTOR INERTIA				**0X6410**	**10**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.000001 kg / cm2	0 -2,147,483,647	NO	F
Description M_JM - kg / m^2 Value in * 10e2 - gives kg/m^2 - write to PID Motor inertia in units of 0.000001 kg / cm2.					

Object				Index	Sub-Index
MOTOR BACK EMF				**0X6410**	**11**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.01 V/kRPM	0 - 2,147,483,647	NO	F
Description M_KE – Value * 100 ->PID Motor back-EMF constant. Units are 0.01 V/kRPM.					

Object				Index	Sub-Index
MOTOR TORQUE CONSTANT				**0X6410**	**12**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.001 Nm / Amp	0 - 2,147,483,647	NO	F
Description M_KT Value * 1000 - > PID Motor torque constant, units: 0.001 Nm / Amp.					

Object				Index	Sub-Index
MOTOR INDUCTANCE				**0X6410**	**13**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	0.01 milli Henry	0 - 32,767	NO	F
Description M_LS Value *100 -> PID Motor winding inductance, in 0.01-milli Henry units.					

Object				Index	Sub-Index
MOTOR RESISTANCE				**0X6410**	**14**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	0.01 Ohm	0 - 32,767	NO	F
Description M_RS Value * 100 -> PID Motor winding resistance, in 0.01-Ohm units.					

Object				Index	Sub-Index
MOTOR MAX CONT. CURRENT				**0X6410**	**15**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	0.01 A(Rms)	0 - 32,767	NO	F
Description M_MAXCURRENT Value * 100 -> PID Motor continous RMS current per phase in 0.01 A Rms units.					

Object				Index	Sub-Index
MOTOR MAX VELOCITY				**0X6410**	**16**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	1 RPM	0 - 32,767	NO	F
Description M_MAXVELOCITY -> direct Motor maximum velocity in RPM.					

Object				Index	Sub-Index
MOTOR POLES				**0X6410**	**17**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 - 32,767	NO	F
Description M_NPOLES -> direct Number of motor poles per rotation. For proper operation this value must be set correctly for the motor being controlled. This parameter is only used for rotary motors. For linear motors its value needs to be set to 2.					

Object				Index	Sub-Index
ENCODER COUNTS				**0X6410**	**18**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts / rev	0 - 2,147,483,647	NO	F
Description M_ENCODER -> Value *4 For rotary motors gives the number of encoder counts / motor revolution. For linear motors this parameter represents ration between pole pair pitch and linear encoder resolution. Value to write for linear motors can be calculated as follows: Encoder counts = Lpp / LEres where Lpp - pole pair pitch (S-S or N-N distance in linear motor) LEres – linear encoder resolution.					

Object				Index	Sub-Index
MOTOR NOMINAL TERMINAL VOLTAGE				**0X6410**	**19**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	1 Volt	0 - 32,767	NO	F
Description M_TERMVOLTAGE - direct Sets motor normal terminal voltage in 1 Volt units.					

Object				Index	Sub-Index
MOTOR FEEDBACK DEVICE TYPE				**0X6410**	**20**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 - 32,767	NO	F

Description
Describes motor's feedback device configuration data as follows:

Value	Description
0	reserved
1	encoder feedback
2	resolver feedback
3	Absolute encoder (BiSS, SPI)
4	Absolute encoder (EnDat)
5	Absolute encoder (HyperFace)

7 Control Loops

This chapter describes the nested control loop model used by AC Tech amplifiers to control the position of the motor.

7.1 Control Loop Configuration

This section provides an overview of the control loops. Topics include:

7.1.1 Nested Control Loops

AC Tech amplifiers use up to three nested control loops - current, velocity, and position – to control a motor in three associated operating modes. In position mode, The amplifier uses all three loops, as shown in Figure 10. The loops are nested: the current loop within the velocity loop, within the position loop. Stated another way, the position loop drives the velocity loop, which drives the current loop

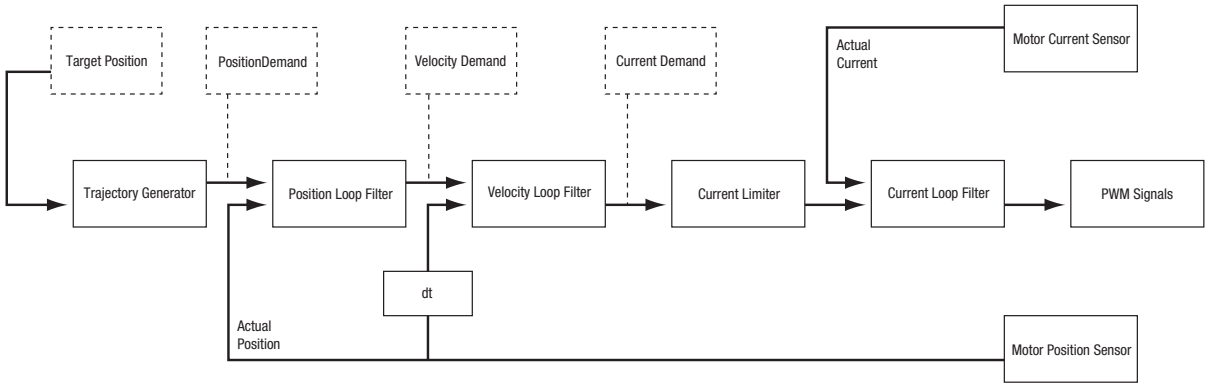

Figure 10: Nested Loops

Basic Attributes of All Control Loops

These loops (and servo control loops in general) share several common attributes including command input, limits, feedback, gain and outputs.

Loop Attribute	Description
Command Input	Every loop is given a value which it will attempt to control. For example, the velocity loop receives a velocity demand that is the desired motor speed.
Limits	Limits are set on current loop to protect the motor and/or mechanical system.
Feedback	The nature of servo control loops is that they receive feedback from the device they are controlling. For example, the position loop uses the actual motor position as feedback.
Gains	These are constant values that are used in the mathematical equation of the servo loop. The values of these gains can be adjusted during amplifier setup to improve the loop performance. Adjusting these values is often referred to as tuning the loop.
Output	The loop generates a control signal. This signal can be used as the command signal to another control loop or the input to a power amplifier.

7.1.2 The Position Loop

The CANopen master provides a target position to the amplifier's internal trajectory generator. In turn the generator provides the position loop a position demand and velocity and acceleration limit values. The position loop applies corrective gains in response to feedback to forward a velocity demand to the velocity loop. The inputs to the position loop vary with different operating modes.

Trajectory Generator Inputs

The inputs to the trajectory generator include profile position, velocity, and acceleration values. They are accessed through different sets of mode-specific objects as summarized in Table 7.

Table 7: Trajectory Generator Inputs

Mode	Input Object Name	Object ID	Description
Homing	Homing Method	0x6098	Defines the method to find the motor home position
	Homing Speeds	0x6099	The sub-index objects of 0x6099 hold the two velocities (fast and slow) used when homing
	Homing Acceleration	0x609A	Defines the acceleration used for all homing moves
	Home Offset	0x607C	Used in homing mode as an offset between the home sensor position and the zero position
Profile Position	Motion Profile Type	0x6086	Selects the type of trajectory profile to use. Choices are trapezoidal and s-curve.
	Target Position	0x607A	Destination position of the move
	Profile Velocity	0x6081	The velocity that the trajectory generator attempts to achieve when running in position profile mode
	Profile Acceleration	0x6083	Acceleration that the trajectory generator attempts to achieve when running in position profile mode
	Profile Deceleration	0x6084	Deceleration that the trajectory generator attempts to achieve at the end of a trapezoidal profile when running in position profile mode

Position Loop Feedback

The feedback to the loop is the actual motor position, obtained from a position sensor attached to the motor (most often a quadrature encoder). This is provided by the Position Actual Value object (index 0x6063, paragraph 7.2, page 55).

Position Loop Gains

The following gains are used by the position loop to calculate the output value: PP, PI, PL, and PD.

Gain	Description
PP - Position loop proportional	The loop calculates its position error as the difference between the Position Actual Value and the Position Demand Value. This error in turn is multiplied by the proportional gain value. The primary effect of this gain is to reduce the following error.
PI – Integral	The position error gets accumulated. The primary effect of this gain is to decrease steady error. Accumulated error multiplies on PI gain value and added to the velocity demand
PL - Limit	Limit influence produced by PI gain to allow faster settling time
PD – Differential	The position error in previous step gets substructed from the current position error forming error change rate. Error rate of change multiplies on PD gain and added to velocity demand. This gain primarily contributes to stability of the loop acting as a “shock absorber”.

These gains are accessed through the sub-index objects of the Position Loop Gains object (index 0x60FB, paragraph 7.2, page 57).

7.1.3 The Velocity Loop

As shown the summing junction takes the velocity demand from position loop, subtracts the actual velocity, represented by the feedback signal, and produces an error signal. This error signal is then processed using the integral and proportional gains to produce a current demand.

During normal operation in position mode velocity loop driven by the position loop. When device placed in velocity mode velocity loop can be driven by analog reference, or value taken from internal variable manipulated by any of the device’s interfaces such as RS-232, RS-485, Ethernet, CAN etc. Set of limiting parameters for this occasion is set by following objects:

Object Name	Object ID	Object Note
*Analog Input #1 Velocity scaling factor	0x2424	(used only without position loop)
*Velocity Loop Acceleration	0x244C	(used only without position loop)
*Velocity Loop Deceleration	0x244D	(used only without position loop)

*Not used when the velocity loop is controlled by the position loop.

Velocity Loop Input

If drive is in position mode then output of the position loop is the input of the velocity loop . If drive is in velocity follower mode then input of the velocity loop could be an internal object (index 0x248B) or analog input #1. Selection is done using the Reference Source object (index 0x2025, paragraph 6.2, page 36).

Velocity Loop Gains

The velocity loop uses the following gains. Refer to the Velocity Loop Gains object (index 0x60F9, paragraph 7.3, page 60).

Gain	Name	Description
Vp	Velocity loop proportional	The velocity error (the difference between the actual and the limited commanded velocity) is multiplied by this gain. The primary effect of this gain is to increase bandwidth (or decrease the step-response time) as the gain is increased.
Vw	Velocity wide	This is additional proportional gain applied to output of velocity loop to allow wider range of gain values. This gain can range from -16 to +4
Vi	Velocity loop integral	The integral of the velocity error is multiplied by this value. Integral gain reduces the velocity error to zero over time. It controls the DC accuracy of the loop, or the flatness of the top of a square wave signal. The error integral is the accumulated sum of the velocity error value over time.

7.1.4 The Current Loop

Current loop starts from current limiter.The current limiter accepts a current demand from the velocity loop, applies limits, and passes a limited current value to the summing junction. The summing junction takes the commanded current, subtracts the actual current (represented by the feedback signal), and produces an error signal. This error signal is then processed using the integral and proportional gains to produce a voltage command. This command is then applied to the amplifier's power stage.

Current Loop Input

If drive is in position or velocity mode then output of the velocity loop is the input of the current loop . If drive is in current mode then input of the current loop could be an internal object (index 0x248B) or analog input #1. Selection is done by the Reference Source object (index 0x2025). In case if analog loop is driven by analog input #1 scale value applied by object Analog Input #1 Current scaling factor (index 0x2423)

Current Loop Limits

The commanded current value is first reduced based on a set of current limit parameters designed to protect the motor. These current limits are accessed through the following objects:

Object Name	Object ID	Description
Peak Current Limit 16	0x241F	Maximum current that can be generated by the amplifier if carrier frequency selected is 16kHz for a short duration of time. This value cannot exceed the peak current rating of the amplifier.
Peak Current Limit 8	0x2420	Maximum current that can be generated by the amplifier if carrier frequency selected is 8kHz for a short duration of time. This value cannot exceed the peak current rating of the amplifier.
Continuous Current Limit	0x241E	Maximum current that can be constantly generated by the amplifier.
Analog Input # 1 CSF	0x2423	(CSF: Current Scaling Factor) Current scaling value when driven by analog input #1

7.2 Position Loop Configuration Objects

This section describes the objects used to configure the position control loop.

POSITION DEMAND VALUE	INDEX 0X6062	
POSITION ACTUAL VALUE	INDEX 0X6063	
POSITION ACTUAL VALUE	INDEX 0X6064	
FOLLOWING ERROR WINDOW	INDEX 0X6065	
FOLLOWING ERROR TIME	INDEX 0X6066	
POSITION WINDOW	INDEX 0X6067	
ACTUAL VELOCITY	INDEX 0X6069	
VELOCITY DEMAND VALUE	INDEX 0X606B	
ACTUAL VELOCITY	INDEX 0X606C	
FOLLOWING ERROR	INDEX 0X60F4	
POSITION LOOP GAINS	INDEX 0X60FB	
POSITION LOOP PROPORTIONAL GAIN	INDEX 0X60FB	SUB-INDEX 1
POSITION LOOP INTEGRAL GAIN	INDEX 0X60FB	SUB-INDEX 2
POSITION LOOP DERIVIATIVE GAIN	INDEX 0X60FB	SUB-INDEX 3
POSITION LOOP INTEGRAL GAIN LIMIT	INDEX 0X60FB	SUB-INDEX 4
SOFTWARE POSITION LIMITS	INDEX 0X607D	
NEGATIVE SOFTWARE LIMIT POSITION	INDEX 0X607D	SUB-INDEX 1
POSITIVE SOFTWARE LIMIT POSITION	INDEX 0X607D	SUB-INDEX 2
SOFTWARE LIMITS MODE	INDEX 0X607D	SUB-INDEX 3

Object				Index	Sub-Index
POSITION DEMAND VALUE				**0X6062**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RO	encoder counts	--	YES	--
Description This is the motor position (in units of encoder counts) to which the amplifier is currently trying to move the axis. This value is updated every servo cycle based on the amplifier's internal trajectory generator. Units: encoder counts.					

Object				Index	Sub-Index
POSITION ACTUAL VALUE				**0X6063**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	--	YES	R
Description This is the actual motor position in units of encoder counts as calculated by the amplifier every servo cycle based on the state of the encoder input lines.					

Object				Index	Sub-Index
POSITION ACTUAL VALUE				**0X6064**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	--	YES	R
Description This object holds the same value as Position Actual Value object (index 0x6063).					

Object				Index	Sub-Index
FOLLOWING ERROR WINDOW				**0X6065**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	0 – 32767	YES	RF
Description This object holds allowable following error. If at any time following error exceeds this value for time more then specified by Following Error Time object (index 0x6066) amplifier will enter fault state.					

Object				Index	Sub-Index
FOLLOWING ERROR TIME				**0X6066**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	ms	0 – 8000	YES	RF
Description This object holds allowable following error time i.e maximum time following error can set by object index 0X6065 before fault will be generated.					

Object				Index	Sub-Index
POSITION WINDOW				**0X6067**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	0 - 2,147,483,647	YES	RF
Description Size of the amplifier's tracking window. The "target reached" bit of the amplifier's status word is set when the amplifier is not running a trajectory, and the position error has been within the tracking window for the programmed tracking window time. Bit #5 in manufacturer status window is affected by this object as well. Bit #5 is set when current profile command is completed and motor actual position is within specified position window.					

Object				Index	Sub-Index
ACTUAL VELOCITY				**0X6069**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RO	0.1 enc counts / sec	--	YES	--
Description Actual motor velocity in units of 0.1 encoder counts / second.					

Object				Index	Sub-Index
VELOCITY DEMAND VALUE				**0X606B**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RO	0.1 enc counts / sec	--	YES	--
Description Velocity that the velocity loop is currently trying to attain. When the amplifier is running in homing or profile position mode, the velocity demand value is the output of the position loop, and the input to the velocity loop. AC Tech CANopen amplifiers support some modes in which the velocity demand is produced from a source other then the position loop. In these modes the demand velocity comes from the analog reference input, or the internal velocity preset memory location.					

Object				Index	Sub-Index
ACTUAL VELOCITY				**0X606C**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RO	0.1 enc counts / sec	--	YES	--
Description This object contains exactly the same information as object 0x6069.					

Object				Index	Sub-Index
FOLLOWING ERROR				**0X60F4**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RO	encoder counts	--	YES	--
Description This object gives the difference, in units of encoder counts, between the Position Actual Value object (index 0x6063) and the Position Demand Value object (index 0x60fc). This value is calculated as part of the position control loop. It is also the value that the various tracking windows are compared to. Refer to the Position Window object (index 0x6067, paragraph 7.2, page 56), and the Error Window object (index 0x6065, page 56).					

Object				Index	Sub-Index
POSITION LOOP GAINS				**0X60FB**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	YES	--
Description This object contains the various gain values used to optimize the position control loop. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
POSITION LOOP PROPORTIONAL GAIN				**0X60FB**	**1**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 –32,767	YES	RF
Description This gain value is multiplied by the position loop error. The position loop error is the difference between the instantaneous commanded position and the actual motor position.					

Object				Index	Sub-Index
POSITION LOOP INTEGRAL GAIN				**0X60FB**	**2**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 –16,383	YES	RF
Description This value is multiplied by position loop error over the time. The position loop error is the difference between the instantaneous commanded position and the actual motor position.					

Object				Index	Sub-Index
POSITION LOOP DERIVIATIVE GAIN				**0X60FB**	**3**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 –32,767	YES	RF
Description This value is multiplied by position loop error's difference. Error difference is taking by subtracting loop error value in last servo cycle and loop error value in present servo cycle.					

Object				Index	Sub-Index
POSITION LOOP INTEGRAL GAIN LIMIT				**0X60FB**	**4**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 –2000	YES	RF
Description This value limits influence of the integral gain (object 0x60FB sub index 2.) term. This value internaly scaled to 1/2000 meaning that 2000 would represent 100% of the term influence.					

Object				Index	Sub-Index
SOFTWARE POSITION LIMITS				**0X607D**	**--**
Type	Access	Units	Range	Map PDO	Memory
Array	RW	--	--	YES	--
Description This array holds the two software position limit values Negative Software Limit Position and Positive Software Limit Position. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
NEGATIVE SOFTWARE LIMIT POSITION				**0X607D**	**1**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	--	YES	RF
Description The Software Position Limits array holds the two software position limit values: Negative Software Limit Position and Positive Software Limit Position. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
POSITIVE SOFTWARE LIMIT POSITION				**0X607D**	**2**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	--	YES	RF
Description The Software Position Limits array holds the two software position limit values: Negative Software Limit Position and Positive Software Limit Position. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
SOFTWARE LIMITS MODE				**0X607D**	**3**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	--	0-2	YES	RF
Description Enables or Disables software position limits. Possible settings 0-disabled or 1-enabled(will disable and fault when hit), 2- enabled (will decelerate to stop and generate fault).					

7.3 Velocity Loop Configuration Objects

This section describes the objects used to configure the velocity control loop including:

VELOCITY LOOP MAXIMUM ACCELERATION	INDEX 0X244C	
VELOCITY LOOP MAXIMUM DECELERATION	INDEX 0X244D	
VELOCITY LIMITS ENABLED	INDEX 0X204B	
ANALOG INPUT VELOCITY SCALE	INDEX 0X2424	
PROGRAMMED VELOCITY	INDEX 0X248B	
VELOCITY LOOP GAINS	INDEX 0X60F9	
VELOCITY LOOP PROPORTIONAL GAIN	INDEX 0X60F9	SUB-INDEX 1
VELOCITY LOOP INTEGRAL GAIN	INDEX 0X60F9	SUB-INDEX 2
VELOCITY LOOP GAIN SCALER	INDEX 0X60F9	SUB-INDEX 3

Object				Index	Sub-Index
VELOCITY LOOP MAXIMUM ACCELERATION				**0X244C**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	RPM*Sec	0.1 – 5,000,000	YES	RF
Description This acceleration value limits the maximum rate of change of the commanded velocity input to the velocity loop. This limit only applies when the absolute value of the velocity change is positive (i.e. the speed is increasing in either direction). Units are RPM*Sec. This value is only used if velocity demand is produced by a source other than the Position Loop. The value is applied if Accel/ Decel limiting is enabled by object 0x204B.					

Object				Index	Sub-Index
VELOCITY LOOP MAXIMUM DECELERATION				**0X244D**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	RPM*Sec	0.1 – 5,000,000	YES	RF
Description This acceleration value limits the maximum rate of change of the commanded velocity input to the velocity loop. This limit only applies when the absolute value of the velocity change is negative (i.e. the speed is decreasing in either direction). Units are RPM*Sec. This value is only used if velocity demand is produced by a source other than the Position Loop. The value is applied if Accel/Decel limiting is enabled by object Velocity Limits Enabled (index 0x204B).					

Object				Index	Sub-Index
VELOCITY LIMITS ENABLED				**0X204B**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 – 1	YES	RF
Description This object enables or disables acceleration and deceleration limiting for sources other than the position loop output. Possible values: 0 – limits disabled and 1 – limits enabled.					

Object				Index	Sub-Index
ANALOG INPUT VELOCITY SCALE				**0X2424**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	RPM/V	-2000 to +2000	YES	--
Description When analog input #1 is used as velocity loop reference (set velocity) ,this object gives the scaling factor in RPM/V units.					

Object				Index	Sub-Index
PROGRAMMED VELOCITY				**0X248B**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	RPS	-350 to +350	YES	RF
Description Gives the commanded velocity value when running in velocity follower mode (Operating mode= -2)					

Object				Index	Sub-Index
VELOCITY LOOP GAINS				**0X60F9**	**--**
Type	Access	Units	Range	Map PDO	Memory
Record	RW	--	--	YES	--
Description This object contains the various gain values used to optimize the velocity control loop. Sub-index 0 contains the number of sub-elements of this record.					

Object				Index	Sub-Index
VELOCITY LOOP PROPORTIONAL GAIN				**0X60F9**	**1**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 – 32,767	YES	RF
Description This gain value is multiplied by the velocity error value. The velocity error is the difference between the desired and actual motor velocity.					

Object				Index	Sub-Index
VELOCITY LOOP INTEGRAL GAIN				**0X60F9**	**2**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	0 – 32,767	YES	RF
Description This gain value is multiplied by the integral of the velocity loop error.					

Object				Index	Sub-Index
VELOCITY LOOP GAIN SCALER				**0X60F9**	**3**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	-16 to +4	YES	RF
Description A shift value that allows more range on the velocity loop gains when using a different encoders. The output of the velocity loop is multiplied by velocity loop gain scaler after the other gains have been used.					

7.4 Current Loop Configuration Objects

This section describes the objects used to configure the current control loop including:

USER PEAK CURRENT LIMIT16 INDEX 0X241F

USER PEAK CURRENT LIMIT8 INDEX 0X2420

USER CONTINUOUS CURRENT LIMIT INDEX 0X241E

ACTUAL MOTOR CURRENT INDEX 0X24BC

PROGRAMMED CURRENT INDEX 0X248B

ANALOG INPUT CURRENT SCALE INDEX 0X2423

Object				Index	Sub-Index
USER PEAK CURRENT LIMIT16				**0X241F**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	Amps (RMS)	0 – 50	YES	RF
Description Specifies a peak current limit in phase Amps (RMS) when drive PWM carrier frequency is set to 16kHz					

Object				Index	Sub-Index
USER PEAK CURRENT LIMIT8				**0X2420**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	Amps (RMS)	0 – 50	YES	RF
Description Specifies a peak current limit in phase Amps (RMS) when drive PWM carrier frequency is set to 8kHz.					

Object				Index	Sub-Index
USER CONTINUOUS CURRENT LIMIT				**0X241E**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	Amps (RMS)	0 – 50	YES	RF
Description Specifies a continues current limit in phase Amps (RMS).					

Object				Index	Sub-Index
ACTUAL MOTOR CURRENT				**0X24BC**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RO	Amps (RMS)	--	YES	--
Description Actual motor RMS per phase current.					

Object				Index	Sub-Index
PROGRAMMED CURRENT				**0X248B**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	Amps (RMS)	-50 to +50	YES	RF
Description This object gives the programmed current value (0.01 Amps) used when running in current (torque) mode.					

Object				Index	Sub-Index
ANALOG INPUT CURRENT SCALE				**0X2423**	**--**
Type	Access	Units	Range	Map PDO	Memory
Float	RW	Amps (RMS)	-5 to +5	YES	--
Description When analog input #1 is used as a current loop reference (set current), this object gives the scaling factor in A/V units.					

8 Non Profiled Operating Modes

This chapter describes the operation of an amplifier in non-profiles modes such as velocity follower and current follower. Contents include:

8.1 Current Follower Mode

Current follower mode is set by setting the Mode of Operation object (index 0x6060, paragraph 6.2, page 35) to -1. In this mode the current loop input is disconnected from the velocity loop output. Reference for the current loop could be the value of the Programmed Current object (index 0x248B) or analog input #1. If the analog signal #1 is used for the current loop reference then the analog signal at the analog input #1 is multiplied by the Analog Input Current Scale object (index 0x2423) and then fed to the current loop summing junction (reference). The Reference Source object (index 0x2025) controls the reference source configuration.

8.2 Velocity Follower Mode

Velocity follower mode is set by setting object Mode of Operation (index 0x6060, paragraph 6.2, page 35) to -2. In this mode the velocity loop input is disconnected from the position loop output. Reference for the velocity loop could be the value of the Programmed Velocity object (index 0x248B) or analog input #1. If analog signal #1 is used for the velocity loop reference then the analog signal at the analog input #1 is multiplied by the Analog Input Velocity Scale object (index 0x2424) and then fed to the velocity reference limiter and then to the velocity loop summing junction. The Reference Source object (index 0x2025) controls the reference source configuration. The velocity reference limiter has 3 objects: Velocity Loop Maximum Acceleration (index 0x244C); Velocity Loop Maximum Deceleration (index 0x244D) and Velocity Limit Enabled (index 0x204B).

9 Homing Mode

9.1 Homing Mode Operation

This section describes control of the amplifier in homing mode. Contents of this section include:

9.1.1 Homing Overview

What is Homing? Homing is the method by which a drive seeks the home position (also called the datum, reference point, or zero point). There are various methods of achieving this using:

- limit switches at the ends of travel, or
- a dedicated home switch.

Most of the methods also use the index pulse input from an incremental encoder.

The Homing Function

The homing function provides a set of trajectory parameters to the position loop, as shown in Figure 11. The parameters are generated by the homing function and are not directly accessible through CANopen dictionary objects. They include the profile mode and velocity, acceleration, and deceleration data.

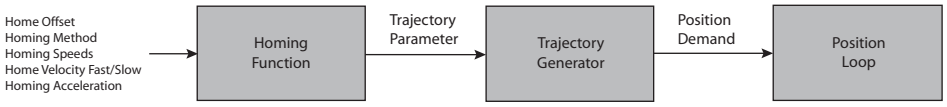

Figure 11: Homing Function

Initiating and Verifying a Homing Sequence

For the amplifier to operate in homing mode, the Mode of Operation object (index 0x6060, paragraph 6.2, page 35) should be set to 6. A homing move is started by setting bit 4 of the Control Word object (index 0x6040, paragraph 6.2, page 34). The results of a homing operation can be accessed in the Status Word (index 0x6041, paragraph 6.2, page 34).

Home Offset

The home offset 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 Figure 12.

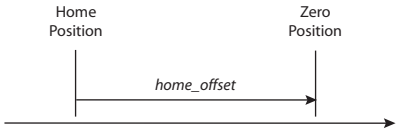

Figure 12: Home Offset

Homing Speeds

There are two homing speeds: fast and slow. The fast speed is used to find the home switch and he slow speed is used to find the index pulse. Refer to the Homing Speeds object (index 0x6099, paragraph 9.2, page 70)

Homing Acceleration

Homing Acceleration (index 0x609A) establishes the acceleration to be used for all accelerations and decelerations with the standard homing modes. Note that in homing, it is not possible to program a separate deceleration rate.

9.1.2 Homing Methods

There are several homing methods, each supported by objects described later in this chapter. Each method establishes the:

- Homing signal (positive limit switch, negative limit switch, home switch)
- Direction of actuation and, where appropriate, the position of the index pulse.

Legend to Homing Method Descriptions

Homing method descriptions and diagrams in this manual are based on those in theCANopen Profile for Drives and Motion Control (DSP 402). As highlighted in the example below, each homing method diagram shows the motor in the starting position on a mechanical stage. The arrow line indicates direction of motion, and the circled number indicates the homing method (the mode selected in the Homing Method object). The location of the circled method number indicates the home position reached with that method. Solid line stems on the index pulse line indicate index pulse locations. Longer dashed lines overlay these stems as a visual aid. Finally, the relevant limit switch is represented, showing the active and inactive zones and transition.

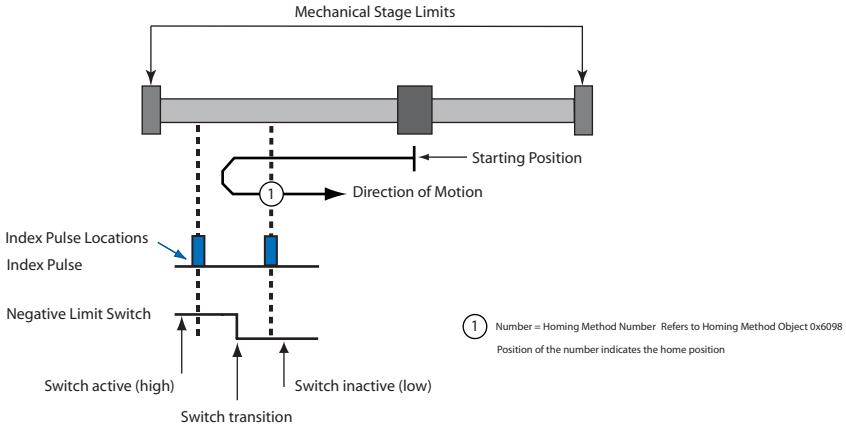

Figure 13: Homing Method Descriptions

Note that in the homing method descriptions, negative motion is leftward and positive motion is rightward.

9.1.3 Homing Method 1: Homing on the Negative Limit Switch

Using this method, the initial direction of movement is leftward if the negative limit switch is inactive (here shown as low). The home position is at the first index pulse to the right of the position where the negative limit switch becomes inactive.

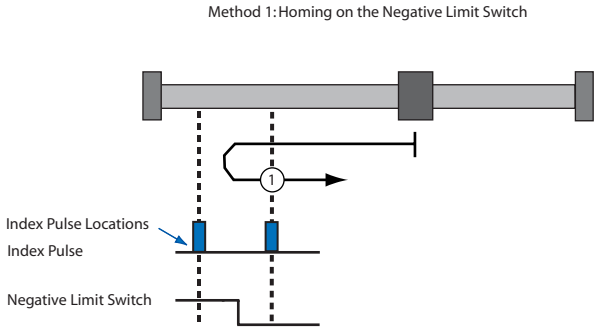

Figure 14: Homing on the Negative Limit Switch

9.1.4 Homing Method 2: Homing on the Positive Limit Switch

Using this method the initial direction of movement is rightward if the positive limit switch is inactive (here shown as low). The position of home is at the first index pulse to the left of the position where the positive limit switch becomes inactive.

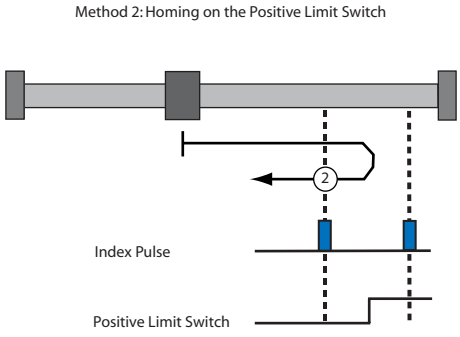

Figure 15: Homing on the Positive Limit Switch

9.1.5 Homing Method 3 and 4: Homing on the Positive Home Switch and Index Pulse

Using methods 3 or 4, the initial direction of movement depends on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is located so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

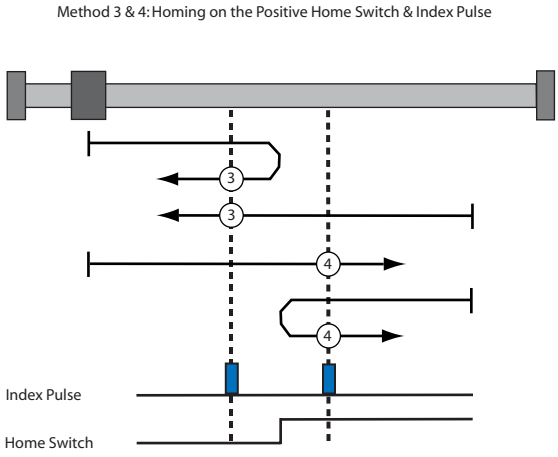

Figure 16: Homing on the Positive Home Switch & Index Pulse

9.1.6 Homing Methods 5 and 6: Homing on the Negative Home Switch and Index Pulse

Using methods 5 or 6, the initial direction of movement depends on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is located so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

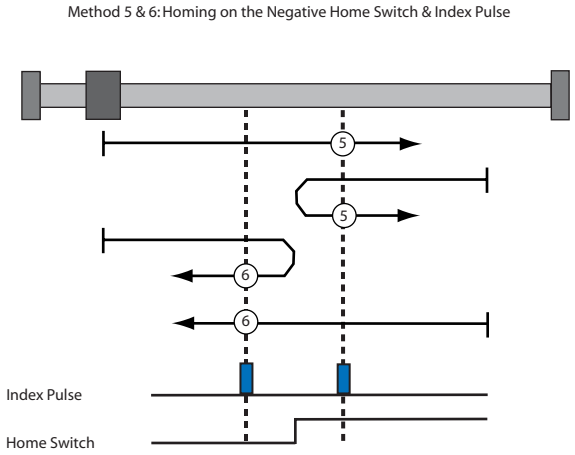

Figure 17: Homing on the Negative Home Switch & Index Pulse

9.1.7 Homing Methods 7-14: Homing on the Home Switch and Index Pulse

These methods use a home switch, which is active over only a portion of the travel. In effect, the switch has a momentary action as the axis sweeps past the switch.

Using methods 7 to 10, the initial direction of movement is to the right. Using methods 11 to 14 the initial direction of movement is to the left, unless the home switch is active at the start of the motion. In this case the initial direction of motion depends on the edge being sought. The home position is at the index pulse on either side of the rising or falling edges of the home switch, as shown in the following two diagrams. If the initial direction of movement leads away from the home switch, the drive must reverse on encountering the relevant limit switch.

Figure 18 illustrates a homing sequence on the home switch and index pulse with a positive initial move.

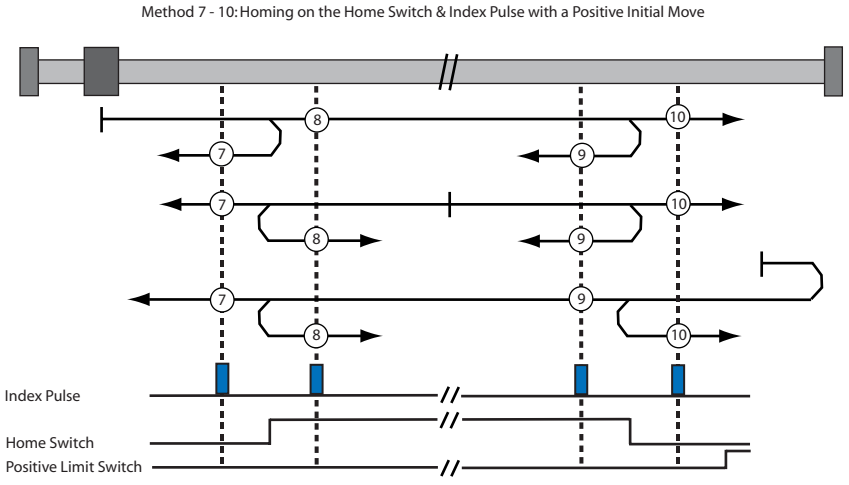

Figure 18: Homing on the Home Switch & Index Pulse w/ Positive Initial Move

Figure 19 illustrates a homing sequence on the home switch and index pulse with a negative initial move.

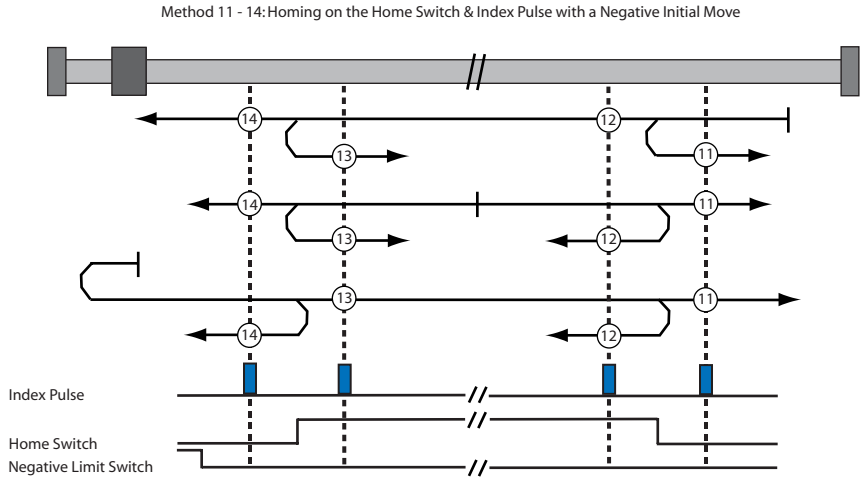

Figure 19: Homing on the Home Switch & Index Pulse w/ Negative Initial Move

9.1.8 Homing Methods 15, 16, 20, 22, 24, 26, 28, and 30: Reserved

Homing methods 15, 16, 20. 22, 24, 26, 28 and 30 are reserved for future use.

9.1.9 Homing Methods 17 and 18: Homing without an Index Pulse

These methods are similar to methods 1-2, except that the home position is not dependent on the index pulse but only on the relevant limit switch translation. Method 17 uses the negative limit switch, and method 18 uses the positive limit switch.

9.1.10 Homing Methods 19, 21, 23, 25, 27, and 29: Homing without an Index Pulse

These methods are similar to methods 1 to 14, except that the home position does not depend on the index pulse. Instead, it depends on the relevant home or limit switch transitions. For example, method 19 is similar to method 3 as shown in the following diagram.

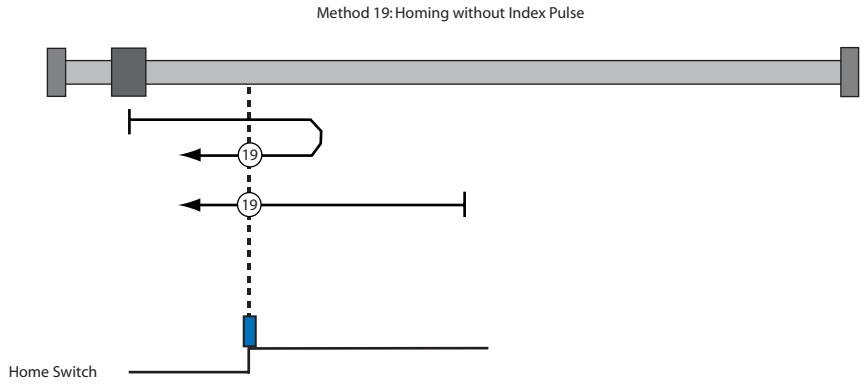

Figure 20: Homing without an Index Pulse

This means method 19 and 20 (as described in the Profile for Drives and Motion Control) both imply the same home algorithm and location, because methods 3 and 4 are only different in which index pulse the locate. Likewise, 22, 24, 26, 28, and 30 (as described in the Profile for Drives and Motion Control) are redundant. For this reason, in AC Tech amplifiers, the following redundant home methods are reserved: 20, 22, 24, 26, 28, and 30. The equivalent home method (one less then each of these values) should be used instead.

9.1.11 Homing Methods 31 and 32: Reserved

Homing methods 31 and 32 are reserved for future use.

9.1.12 Homing Methods 33 and 34: Homing on the Index Pulse

Using methods 33 or 34 the direction of homing is negative or positive respectively. The home position is at the index pulse found in the selected direction.

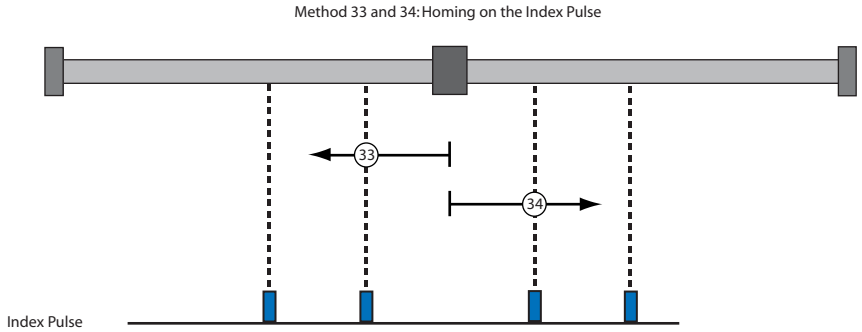

Figure 21: Homing on the Index Pulse

9.1.13 Homing Method 35: Homing on the Current Position

In homing method 35, the current position is the homing position.

9.2 Homing Mode Operation Objects

This section describes the objects that control the operation of the amplifier in homing mode. They include:

HOMING METHOD	INDEX 0X6098	
HOMING SPEEDS	INDEX 0X6099	
HOME VELOCITY – FAST	INDEX 0X6099	SUB-INDEX 1
HOME VELOCITY – SLOW	INDEX 0X6099	SUB-INDEX 2
HOMING ACCELERATION	INDEX 0X609A	
HOME OFFSET	INDEX 0X607C	

Object				Index	Sub-Index
HOMING METHOD				**0X6098**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 8	RW	--	Refer to Description	YES	RF

Description
Defines the method to find the motor home position in homing mode. Supported methods:

Mode	Home position
0	The current position.
1	The location of the first encoder index pulse on the positive side of the negative limit switch.
2	The location of the first encoder index pulse on the negative side of the positive limit switch.
3	The location of the first index pulse on the negative side of a positive home switch. A positive home switch is one that goes active at some position, and remains active for all positions greater then that one.
4	The location of the first index pulse on the positive side of a positive home switch.
5	The location of the first index pulse on the positive side of a negative home switch. A negative home switch is one that goes active at some position, and remains active for all positions less then that one.
6	The location of the first index pulse on the negative side of a negative home switch.
7	The location of the first index pulse on the negative side of the negative edge of an intermittent home switch. An intermittent home switch is one that is only active for a limited range of travel.
8	The location of the first index pulse on the positive side of the negative edge of an intermittent home switch.
9	The location of the first index pulse on the negative side of the positive edge of an intermittent home switch.
10	The location of the first index pulse on the positive side of the positive edge of an intermittent home switch.
11	The location of the first index pulse on the positive side of the positive edge of an intermittent home switch.
12	The location of the first index pulse on the negative side of the positive edge of an intermittent home switch.
13	The location of the first index pulse on the positive side of the negative edge of an intermittent home switch.
14	The location of the first index pulse on the negative side of the negative edge of an intermittent home switch.
15-16	Reserved for future use.
17	The edge of a negative limit switch.
18	The edge of a positive limit switch.
19	The edge of a positive home switch.
20	Reserved for future use.
21	The edge of a negative home switch.
22	Reserved for future use.
23	The negative edge of an intermittent home switch.
24	Reserved for future use.
25	Positive edge of an intermittent home switch.
26	Reserved for future use.
27	The positive edge of an intermittent home switch.
28	Reserved for future use.
29	Negative edge of an intermittent home switch.
30-32	Reserved for future use.
33	The first index pulse on the negative side of the current position.
34	The first index pulse on the positive side of the current position.
35	Set current position to home and move to new zero position (including home offset). This is the same as mode 0 except that mode 0 does not do the final move to the home position.

Note that these homing methods only define the location of the home position. The zero position is always the home position adjusted by the homing offset. Refer to Homing Methods.

Object				Index	Sub-Index
HOMING SPEEDS				**0X6099**	**--**
Type	Access	Units	Range	Map PDO	Memory
Array	RW	--	--	YES	--
Description This array holds the two velocities used when homing. Sub-index 0 contains the number of subelements of this record.					

Object				Index	Sub-Index
HOME VELOCITY – FAST				**0X6099**	**1**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.1 encoder cnts/sec	0 – 500,000,000	YES	RF
Description This velocity value is used during segments of the homing procedure that may be handled at high speed. Generally, this means move in which the home sensor is being located, but the edge of the sensor is not being found. Units are 0.1 encoder counts / second.					

Object				Index	Sub-Index
HOME VELOCITY – SLOW				**0X6099**	**2**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.1 encoder cnts/sec	0 – 500,000,000	YES	RF
Description This velocity value is used for homing segment that require low speed such as cases where the edge of a homing sensor is being sought. Units are 0.1 encoder counts / second.					

Object				Index	Sub-Index
HOMING ACCELERATION				**0X609A**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	10 encoder cnts/sec^2	0 – 200,000,000	YES	RF
Description This value defines the acceleration used for all homing moves. It is specified in units of 10 encoder counts / second^2. The same acceleration value is used at the beginning and ending of moves (i.e. there is no separate deceleration value).					

Object				Index	Sub-Index
HOME OFFSET				**0X607C**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	--	YES	RF
Description This offset value (in units of encoder counts) is used in homing mode as an offset between the home sensor position and the zero position. Refer to Home Offset for more information.					

10 Profile Position and Profile Velocity Mode Operation

This chapter describes the operation of an amplifier in profile position and profile velocity modes. Contents include:

10.1 Profile Position Mode Operation Overview

This section provides an overview of profile position mode operation. Contents of this section include:

10.1.1 Point-to-Point Motion Profiles

In profile position mode, an amplifier receives set points from the trajectory generator to define a target position and moves the axis to that position at a specified velocity and acceleration. This is known as a point-to-point move. For the amplifier to operate in profile position mode, the Mode of Operation object (index 0x6060, paragraph 6.2, page 35) should be set to 0x0001. AC tech amplifiers also support special multi-segment type of moves where target position and velocities can be defined as set allowing complicated segment motion profiles.

Trapezoidal and S-curve Motion Profiles

In a point-to-point move, the rate of change in acceleration is known as jerk. Some applications can tolerate jerk, whereas in others, high rates of jerk can cause excessive mechanical wear or material damage. To support systems with varying levels of jerk tolerance, the profile position mode supports two motion profiles: the trapezoidal profile, which has unlimited jerk, and the jerklimited S-curve profile.

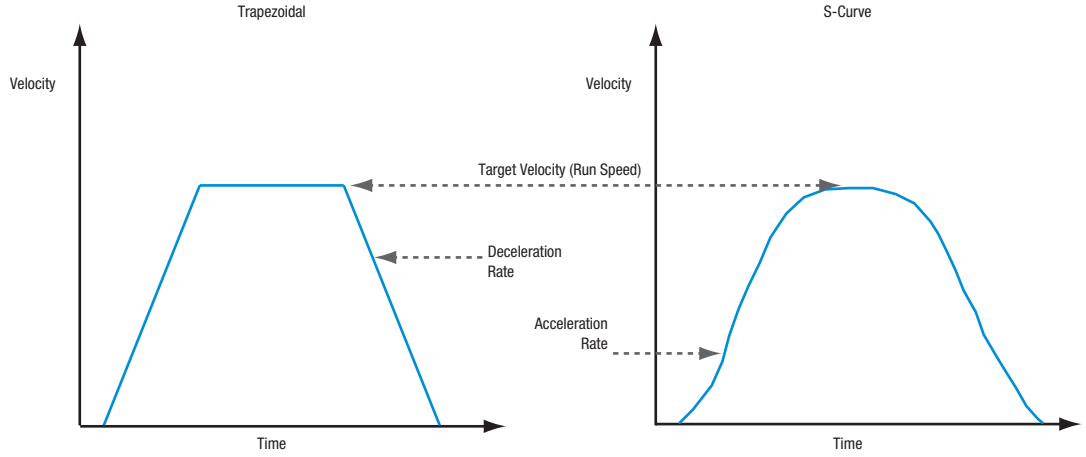

Figure 22:Trapezoidal & S-Curve Motion Profiles

In a trapezoidal profile, jerk is unlimited at the corners of the profile (start of the move, target velocity, start of deceleration, and end of the move). S-curve profiling limits jerk or “smoothes” the motion. Note that an S-curve profile move does support an independent deceleration rate. The Motion Profile Type object (index 0x6086) controls which type of profile is used.

Relative vs. Absolute Moves

In a relative move, the target position is added to the instantaneous commanded position, and the result is the destination of the move. In an absolute move, the target position is offset from the home position.The instantaneous commanded position (called the demand position in the CANopen specification) is the output of the trajectory generator. During the course of the profile this position changes constantly. It is possible to update the profile's target position while the move is in progress. If this update is performed as a relative move, then the target position value will be added to the instantaneous commanded position at the time the update is received. This type of update is most useful when the motor needs to be moved a set distance beyond the point where some asynchronous event occurs.

10.1.2 Handling a Series of Point-to-Point Moves

There are two methods for handling a series of point-to-point moves:

- As a series of discrete profiles (supported in both trapezoidal and S-curve profile moves)
- As one continuous profile (supported in both trapezoidal and S-curve profile moves)

General descriptions of the two methods follow. Detailed procedures appear later in the chapter.

A Series of Discrete Profiles

The simplest way to handle a series of point-to-point moves is to start a move to a particular position, wait for the move to finish, and then start the next move. As shown in Figure 23, each move is discrete. The motor accelerates, runs at target velocity, and then decelerates to zero before the next move begins.

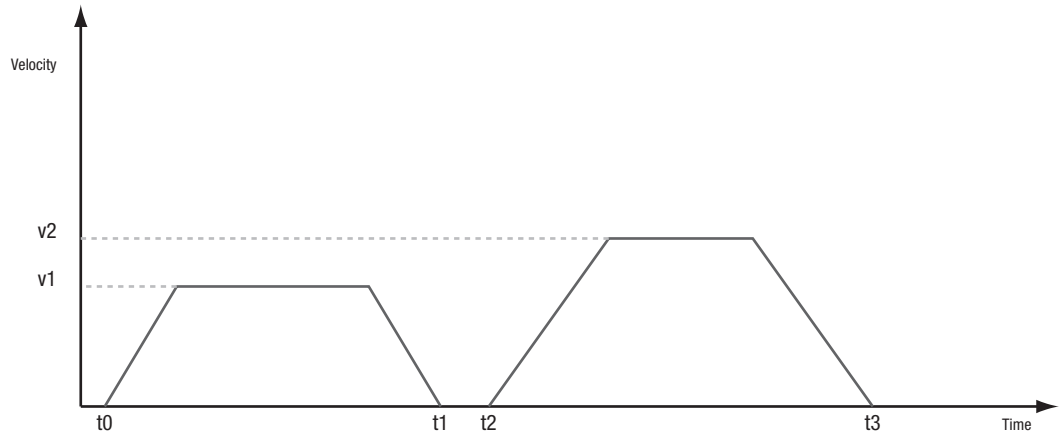

Figure 23: Discrete Profiles

The Profile for Drives and Motion Control refers to this method as the "single setpoint" method. AC Tech CANopen amplifiers allow use of this method with both trapezoidal and S-curve profile moves.

One Continuous Profile (segmented moves)

Alternately, a series of trapezoidal or S-curved profile moves can be treated as a continuous move. AC Tech amplifiers use special technique called Segmented Moves. Each segment of the complex move consists of Target position and Target velocity. As shown in Figure 24, the motor does not stop between moves. Instead, motion profile defined as set of Target Position – Profile Velocity pairs. If this method is used then profile acceleration and deceleration are not used and calculated automatically by amplifier. Their values are calculated based on segment start – segment stop velocities and segment length in position units. This method gives user grater flexibility rather then change point on the fly method since whole profile can be described before motion is started. At the same time loading the next value of the target position and target velocity could be done on the fly. This is particularly useful in systems where calculation by the motion master must also be done on the fly. The motion buffer for the segmented move is 32 entries deep.

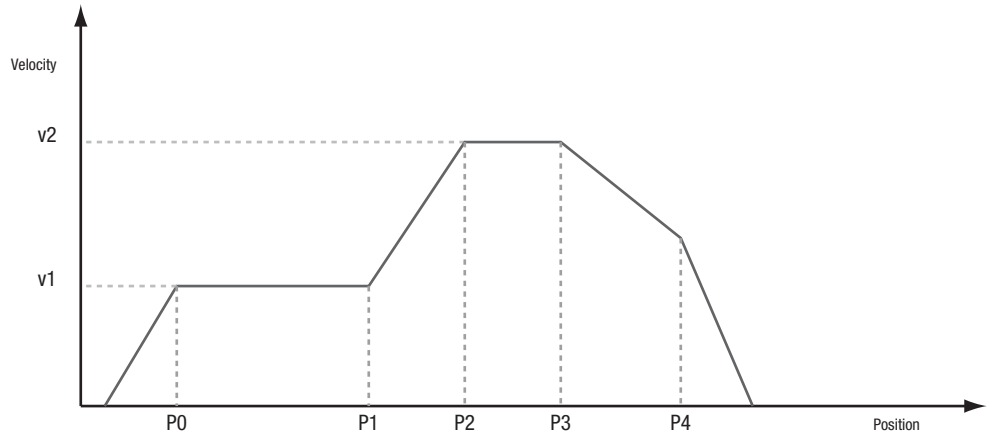

Figure 24: One Continuous Profile

The Profile for Drives and Motion Control refers to this method as the “set of setpoints” method. AC Tech CANopen amplifiers allow use of this method with trapezoidal or S-curve types of moves.

10.1.3 Point-to-Point Move Parameters and Related Data

Move Parameters

Each point-to-point move is controlled by a set of parameters, accessed through the following objects.

Object Name	Object ID	Description
Target Position	0x607A	When running in position profile mode, this object holds the destination position of the trajectory generator.
Profile Velocity	0x6081	Velocity that the trajectory generator will attempt to achieve when running in position profile mode.
Profile Acceleration	0x6083	Acceleration that the trajectory generator attempts to achieve when running in position profile mode.
Profile Deceleration	0x6084	Note that an S-curve profile move does not use a deceleration rate. Instead, the acceleration rate is applied to both the acceleration and deceleration of the move.
Quick Stop Deceleration	0x6085	Deceleration value used when a trajectory needs to be stopped as the result of a quick stop command. Note that unlike most trajectory configuration values, this value is NOT buffered. Therefore, if the value of this object is updated during an abort, the new value is used immediately.
Motion Profile Type	0x6086	Trapezoidal or S-Curve

The Point-to-Point Move Buffer

In profile position mode, the amplifier uses a buffer to store the parameters (listed in the Move Parameters table) for the next point-to-point move, or for next move segment consisting of the Target position-Profile velocity pair. The move buffer can be modified at any point before a control sequence copies the “next-move” parameters to the active move registers.

Move-Related Control Word and Status Word Bit Settings

An amplifier's Control Word (index 0x6040) and Status Word (index 0x6041) play an important role in the initiation and control of point-to-point move sequences, as described herein.

Object Name	Object ID	Bit#	Bit Name	Description
Control Word	0x6040	4	New Setpoint	The transition of bit 4 from 0 to 1 is what causes the amplifier to copy a set of move parameters from the buffer to the active register, thus starting the next move.
		5	Motion Suspended	Allows or not start motion on change bit #4. If this bit is set then transition of bit #4 copies motion parameters to motion buffer but doesn't start the motion until this bit is cleared. If there is an entries in motion buffer clearing this bit will start the motion.
		6	Absolute/ Relative	Value = 0: Move is absolute (based on home position). Value = 1: Move is relative (based on current commanded position).
		8	Halt	Value = 1: Interrupts the motion of the drive. Wait for release to continue.
Status Word	0x6041	10	Target Reached	Amplifier sets bit 10 to 1 when target position has been reached. Amplifier clears bit 10 to zero when new target is received.
		12	Setpoint Acknowledged	Set by the amplifier when Control Word bit 4 goes from 0 to 1. Cleared when Control Word bit 4 is cleared if there is more room in motion buffer (buffer depth is 32 entry). If there is no room in the buffer (buffer full) bit will stay set until buffer has at least one vacant entry.

Refer to paragraph 6.2, page 34 for more Control Word (index 0x6040) and Status Word (index 0x6041) information.

10.1.4 Point-To-Point Move Sequence Examples

Figures 25 and 26 illustrate how to perform:

- A series of moves treated as a Series of Discrete Profiles
- A series of trapezoidal or S-curve position multi-segment moves treated as One Continuous Profile

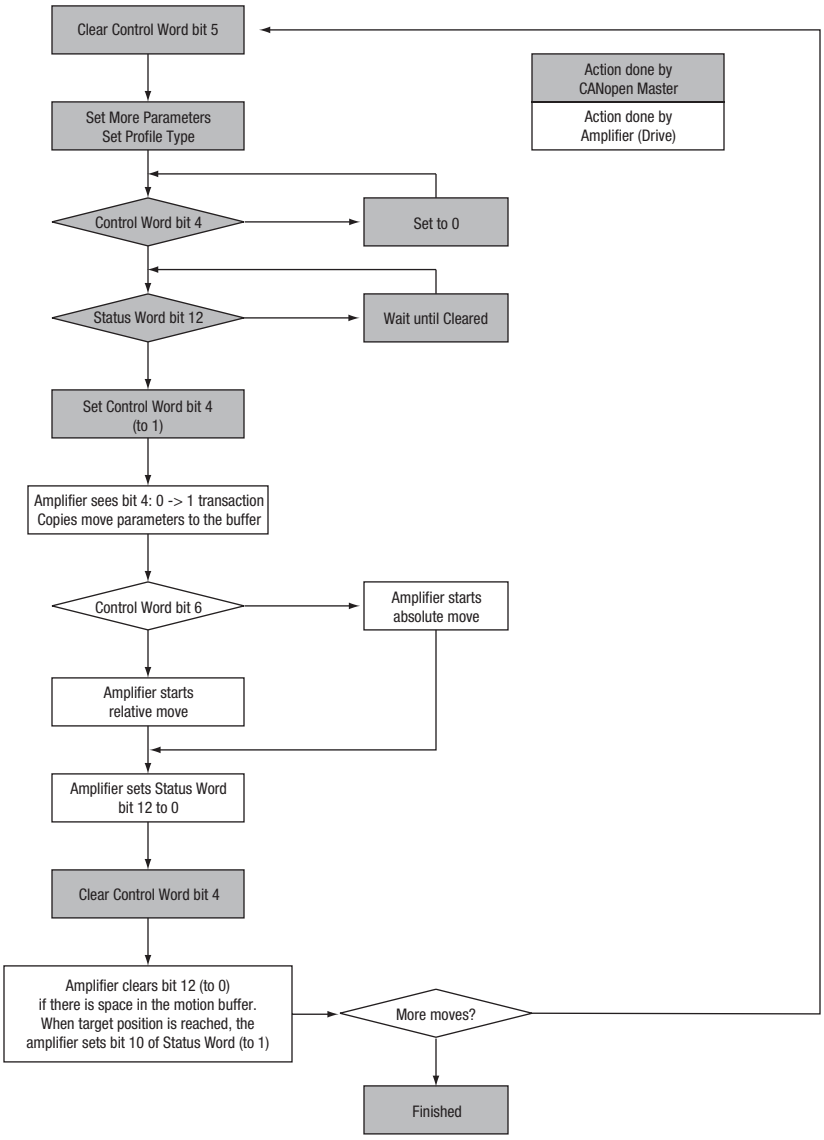

Figure 25: A series of moves treated as a Series of Discrete Profiles

Notes:

1. This type of move is supported as a trapezoidal profile or S-curve profile.
2. Control Word bit 4 is “new setpoint.” It needs to be 0 because the move will be triggered by a 0->1 transition.
3. Bit 4, value of 1 indicates that valid data has been sent to amplifier and new move should begin. Bit 5 is “change set immediately.” A value of 1 tells the amplifier to update the current profile immediately by copying the contents of the move buffer to the active registers (without waiting for move to finish).
4. Amplifier must detect bit 4 0-1 transition to begin move. Bit 5 value 1 allows immediate update.
5. Control word bit 6: value 0 causes absolute move; value 1 causes relative move.
6. Status Word bit 12 is “setpoint acknowledge.” A value of 1 indicates the amplifier has received a setpoint and has started the move.
7. Control Word bit 4 is “new setpoint.” It needs to be 0 to allow the next move will be triggered by a 0->1 transition. Also, the 1->0 transition causes the amplifier to clear bit 12.
8. Amplifier detects 0->1 transition of Control Word bit 4 and clears bit 12 in response. When the motor reaches the target position, the amplifier sets Status Word bit 10 (“target reached”) to 1.

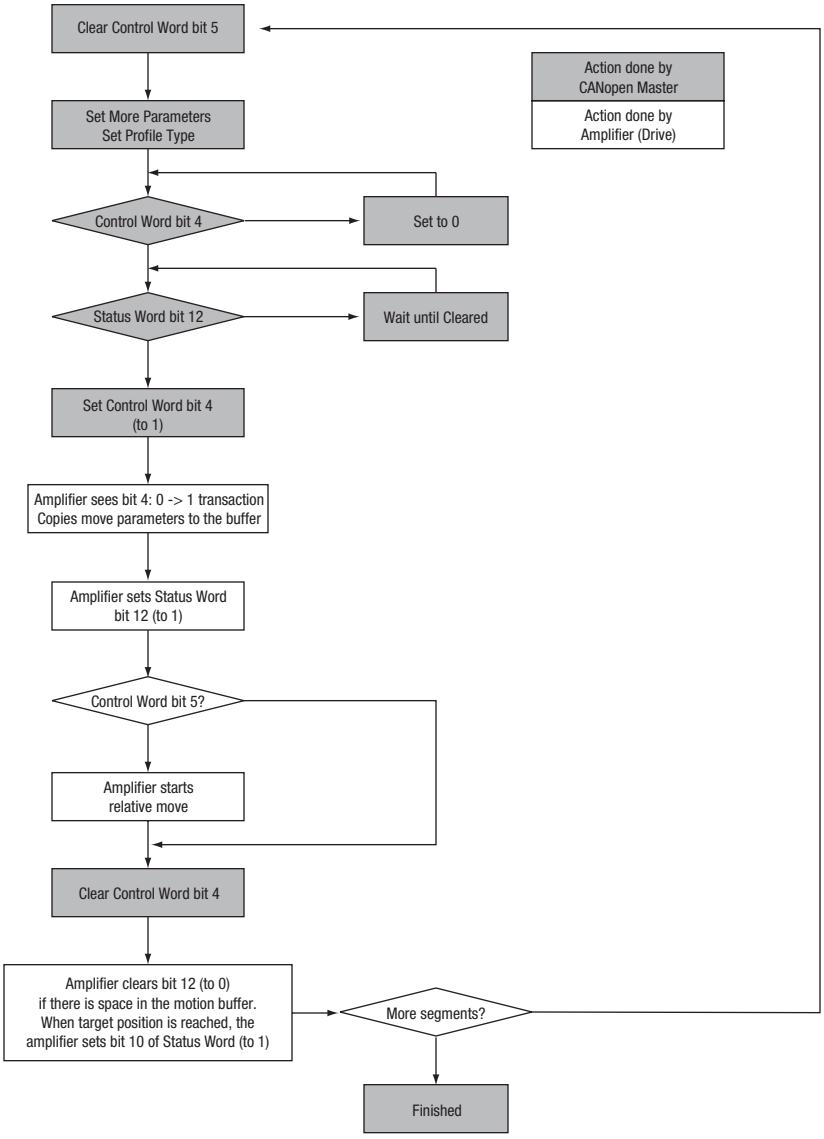

Figure 26: A series of trapezoidal or S-curve position multi-segment moves treated as One Continuous Profile.

10.2 Profile Velocity Mode Operation

10.2.1 Position and Velocity Loops

In profile velocity mode, both the velocity and position loops are used. Profile velocity moves are controlled by some of the same gains and limits objects used in profile position mode. In addition, a Target Velocity object (index 0x60FF) provides the target velocity. For the amplifier to operate in profile velocity mode, the Mode of Operation object (index 0x6060, paragraph 6.2, page 35) should be set to 0x0003.

Controlling motion in Profile Velocity Mode

In profile velocity mode, motion is started by giving a non-zero value to the Target Velocity (index 0x60FF). Motion is stopped by setting this object to zero. In profile velocity mode, the target velocity is updated as soon as the Target Velocity object (index 0x60FF) is set. In this mode, Control Word bits 4, 5, and 6 are not used. To start a move in profile velocity mode, set the profile parameters (profile accel, profile decel, and target velocity).

10.3 Profile Position, Profile Velocity Mode Objects.

This section describes the objects that control the operation of the amplifier in profile position mode. They include:

TARGET POSITION	INDEX 0X607A
PROFILE VELOCITY	INDEX 0X6081
TARGET VELOCITY	INDEX 0X60FF
PROFILE ACCELERATION	INDEX 0X6083
PROFILE DECELERATION	INDEX 0X6084
QUICK STOP DECELERATION	INDEX 0X6085
MOTION PROFILE TYPE	INDEX 0X6086

Object				Index	Sub-Index
TARGET POSITION				**0X607A**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	encoder counts	--	YES	RF
Description When running in position profile mode, this object holds the destination position of the trajectory generator. The units of this object are fixed at encoder counts. Note that the target position programmed here is not passed to the internal trajectory generator until the move has been started or updated using the Control Word. Refer to paragraph 10.1, page 71 “Profile Position Mode Operation Overview” for more information.					

Object				Index	Sub-Index
PROFILE VELOCITY				**0X6081**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.1 encoder cnts/sec	0 – 500,000,000	YES	RF
Description Velocity that the trajectory generator attempts to achieve when running in position profile mode. Note that the value programmed here is not passed to the internal trajectory generator until the move has been started or updated using the Control Word. Refer to paragraph 10.1, page 71 “Profile Position Mode Operation Overview” for more information.					

Object				Index	Sub-Index
TARGET VELOCITY				**0X60FF**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	0.1 encoder cnts/sec	-500,000,000 to +500,000,000	YES	R
Description In profile velocity mode, this object is an input to the amplifier’s internal trajectory generator. Any change to the target velocity triggers an immediate update to the trajectory generator. Note that this is different from the way the profile position works. In that mode, changing the trajectory input parameters doesn’t affect the trajectory generator until bit 4 of the Control Word object (index 0x6040) has been changed from 0 to 1.					

Object				Index	Sub-Index
PROFILE ACCELERATION				**0X6083**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	10 encoder cnts/sec^2	0 -200,000,000	YES	RF
Description In profile position mode, this value (specified in units of 10 encoder counts / second^2) is the acceleration that the trajectory generator attempts to achieve. Note that the value programmed here is not passed to the internal trajectory generator until the move has been started or updated using the Control Word. Refer to paragraph 10.1, page 71 “Profile Position Mode Operation Overview” for more information.					

Object				Index	Sub-Index
PROFILE ACCELERATION				**0X6083**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	10 encoder cnts/sec^2	0 -200,000,000	YES	RF
Description In profile position mode, this value (specified in units of 10 encoder counts / second^2) is the acceleration that the trajectory generator attempts to achieve. Note that the value programmed here is not passed to the internal trajectory generator until the move has been started or updated using the Control Word. Refer to paragraph 10.1, page 71 “Profile Position Mode Operation Overview” for more information.					

Object				Index	Sub-Index
PROFILE DECELERATION				**0X6084**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	10 encoder cnts/sec^2	0 -200,000,000	YES	RF
Description This value (specified in units of 10 encoder counts / second2) is the deceleration that the trajectory generator attempts to achieve at the end of a trapezoidal profile when running in position profile mode. Note that this value is only used when running trapezoidal or profile position special velocity mode profiles. The S-curve profile generator uses the Profile Acceleration object (index 0x6083) as the acceleration target for both the start and end of moves. Note that the value programmed here is not passed to the internal trajectory generator until the move has been started or updated using the Control Word. Refer to paragraph 10.1, page 71 “Profile Position Mode Operation Overview” for more information.					

Object				Index	Sub-Index
QUICK STOP DECELERATION				**0X6085**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 32	RW	10 encoder cnts/sec^2	0 -200,000,000	YES	RF
Description Deceleration value used when a trajectory needs to be stopped as the result of a quick stop command. When a quick stop command is issued, the demand velocity is decreased by this value until it reaches zero. This occurs in all position modes (homing, profile position, and interpolated position modes), and for all trajectory generators (trapezoidal and s-curve). Note that unlike most trajectory configuration values, this value is NOT buffered. Therefore, if the value of this object is updated during an abort, the new value is used immediately.					

Object				Index	Sub-Index
MOTION PROFILE TYPE				**0X6086**	**--**
Type	Access	Units	Range	Map PDO	Memory
Integer 16	RW	--	Refer to Description	YES	RF

Description
Type of trajectory profile to use when running in profile position mode. Supported values are:

Mode	Description
0	Trapezoidal profile mode.
3	S-curve profile mode.
-1	Trapezoidal multi-segmented move
-2	S-curve multi-segmented move

The amplifier will not accept other values. See Profile Position Mode Operation Overview, for more information.

Note that the value programmed here is not passed to the internal trajectory generator until the move has been started or updated using the Control Word bit #4. Refer to paragraph 10.1, page 71 “Profile Position Mode Operation Overview” for more information.

AC Technology Corporation
www.actech.com
630 Douglas Street
Uxbridge, MA 01569
Telephone: (508) 278-9100
Facsimile: (508) 278-7873

P94CAN01B