Barrett Puck® P4™ CANopen Manual

Table of Contents

Index of Tables

Table of Figures

Introduction

Overview

The 4th-generation Barrett Puck® (P4™) is an ultraminiature, brushless motor controller with a built-in encoder. Designed to be mounted directly to the motor, P4 simplifies system architecture by replacing home-run wiring between motors, controllers, and sensors with a convenient bus for power and CANopen communications.

Block Diagram

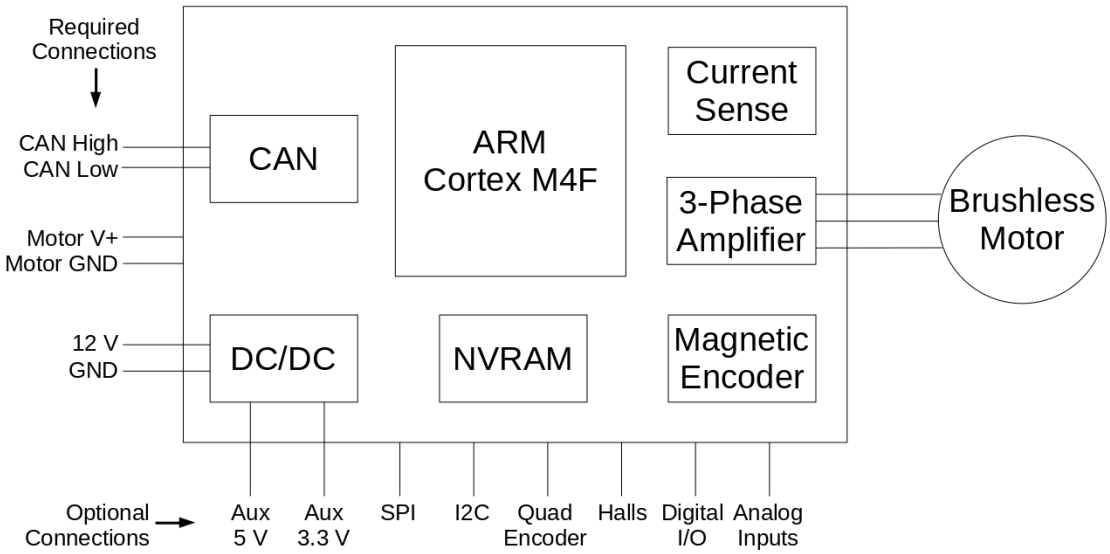


Figure 1: Puck (P4) Motor Controller

Hardware Features

- 120 MHz 32-bit ARM Cortex M4F DSP
- Non-Volatile RAM for storing configuration parameters
- Integrated current sensing
- Space-vector commutation
- Built-in encoder:
  - 4096 cts/rev
  - Absolute within a single revolution
  - Impervious to dust & debris
- External encoder support
  - SPI
  - Quadrature
- 12-48 V motor bus
- ISO 11898-2 CAN physical layer
- 3.3 and 5 V auxiliary outputs
- Internal temperature sensing
- In-system upgradeable firmware
- PWM frequency adjustable up to 80 kHz
- Digital Hall-effect feedback
- Dual analog inputs
- Up to 4 digital I/O
- SPI/BiSS/I2C masters for external sensors

Supported CANopen Features

- Network Management (NMT) messages
- Heartbeat producer
- Expedited Service Data Objects (SDOs)
- Byte-level Receive/Transmit Process Data Objects (RPDOs/TPDOs)
- Sync messages for RPDOs/TPDOs
- Up to 4 RPDOs and 4 TPDOs
- Up to 4 mappable objects per PDO
- Dynamic RPDO/TPDO configuration
- SDO abort message generation
- Optional “Boot to Operational State”
- Single pair of static SDO Connection Object Identifiers (COB IDs), 0x600/0x580 + NodeID
- 11-bit CAN 2.0A identifiers
- 1 Mbps CAN frame bitrate
- Emergency (EMCY) messages
- Error registers (0x1001, 0x1002, 0x1003, 0x603F)
- Object Dictionary “dummy” entries
- Drive modes (CANopen DS402):
  - Idle (0)
  - Profile Position (1)
  - Profile Velocity (3)
  - Profile Torque (4)
  - Homing Immediate (6)
  - Cyclic Synchronous Position (8)
  - Cyclic Synchronous Velocity (9)
  - Cyclic Synchronous Torque (10)
  - Phase Voltage with Angle (12)
- Control word (0x6040), except halt
- Status word (0x6041), except remote

CANopen Features Not Yet Supported

These features may be added in future firmware releases, based on customer feedback.

- Heartbeat consumption
- Timestamps
- Segmented SDOs
- Block transfer SDOs
- SDO size indication (all Tx SDOs are 8 bytes regardless of payload type)
- Bit-level RPDOs/TPDOs
- Drive modes:
  - Velocity (2)
  - Hard-stop, Limit-switch Homing (6)
  - Interpolated Position (7)
  - Cyclic Synchronous Torque with Angle (11)
- Position scaling, gear ratios, feed constants (user-defined units)
- Brake actuation
- Quick-Stop, Halt/Resume
- Deceleration ramps when switching from Operation Enabled
- Remote Transmit Requests (RTR)
- Multiple/dynamic SDO COB IDs
- NMT master discovery
- TPDO event timers (tx on data change)
- TPDO inhibit timers (but don’t tx too often)
- 29-bit CAN 2.0B identifiers
- CAN bitrates other than 1 Mbps
- Digital input triggers
- Analog torque/speed control
- Safe Torque Off (STO) input
- Layer-Setting Services (LSS) for configuring CAN ID and bitrate

Software Installation

Please refer to our Quick Start Guide for software installation and initial testing:

https://web.barrett.com/support/Puck_Documentation/QuickStart_P4_Devkit.pdf

Configuration

CANopen Object Dictionary CSV

The Puck Utility application can configure the CANopen object dictionary using a Comma-Separated Values (CSV) file containing default values for many important object dictionary entries. Please use the provided "config/P4-16-DevKit24V.csv" file as a starting point. You can make a copy of this file and modify it to match your application's communication requirements and your motor's parameters.

Configure PDOs

Any object in P4's Object Dictionary can be mapped to any PDO.

Receive Process Data Objects (RPDOs)

P4 can listen for up to 4 RPDOs, and each RPDO can contain up to 4 object dictionary entries (up to the CAN frame limit of 8 bytes). Each RPDO can be configured to apply its data immediately or on the nth SYNC message.

Transmit Process Data Objects (TPDOs)

P4 can transmit up to 4 TPDOs, and each TPDO can contain up to 4 object dictionary entries (up to the CAN frame limit of 8 bytes). Each TPDO can be configured to transmit on the nth SYNC message. The ability to transmit on data change is not yet supported.

Object Mapping Value Format

The format for mapped RPDO and TPDO Entry Values in the CSV file is: [0x][Index][SubIdx][Length], where [Index] is the 16-bit object dictionary index (in hex), [SubIdx] is the 8-bit Sub-Index (in hex), and [Length] is the bit-length of the mapped object dictionary entry (in hex). [Length] will either be [08], [10], or [20] corresponding to 8 bits, 16 bits, or 32 bits, respectively.

Also note that the Type of a mapping entry (e.g. 0x1600,1) is always UNSIGNED32, while the [Length] of the mapped value must match the entry being mapped (e.g. the length of 0x6040,0 = [10]).

Example 1: Torque Control with Position Feedback

This is a simple configuration which allows the Puck to receive Target Torques and reply with its Actual Position. In this case, setting the Control Word and Mode of Operation must be performed using Service Data Objects (SDOs).

Description	Index	SubIdx	Type	Value
Disable RPDO1 Entries	0x1600	0	UNSIGNED8	0
RPDO1 COB ID	0x1400	1	UNSIGNED32	0x200 | $ID
RPDO1 Receive type (APPLY ON EVERY SYNC)	0x1400	2	UNSIGNED8	0
RPDO1 Entry 1 (TORQUE TARGET)	0x1600	1	UNSIGNED32	0x60710010
Enable RPDO1 Entries	0x1600	0	UNSIGNED8	1
Disable TPDO1 Entries	0x1A00	0	UNSIGNED8	0
TPDO1 COB ID	0x1800	1	UNSIGNED32	0x180 | $ID
TPDO1 Transmit type (TX ON EVERY SYNC)	0x1800	2	UNSIGNED8	0
TPDO1 Entry 1 (ACTUAL POSITION)	0x1A00	1	UNSIGNED32	0x60640020
Enable TPDO1 Entries	0x1A00	0	UNSIGNED8	1

Here are the CAN frames you can use to test this configuration:

MsgID	DLC	Data	Description
0x601	6	22 40 60 00 86 00	Clear Faults, and go to Ready To Switch On
0x601	6	22 40 60 00 0F 00	Go to Operation Enabled
0x601	5	22 60 60 00 04	Set Mode: Profile Torque
0x201	2	64 00	Set Torque: 0x0064 = 100 or 1/10 of Peak Torque
0x080	0		SYNC
0x181	*4*	*01 02 03 04*	*Response: Actual Position = 0x04030201*

Example 2: Pos/Vel/Trq Control with Pos/Vel/Current Feedback

This is how the default CSV configures the P4. This very flexible configuration allows both the CANopen Control Word along with the Mode of Operation to be controlled via PDO along with a Target Position, Target Velocity, and a Target Torque. The P4 will select from one of these three control targets based on the present Mode of Operation. For feedback, the P4 will report its Status Word, and Mode Display, along with the Actual Position, Actual Velocity, and Actual Current values. This configuration works for both “Profile” and “Cyclic Synchronous” control mode styles.

Description	Index	SubIdx	Type	Value
Disable RPDO1 Entries	0x1600	0	UNSIGNED8	0
RPDO1 COB ID	0x1400	1	UNSIGNED32	0x200 | $ID
RPDO1 Receive type (APPLY ON EVERY SYNC)	0x1400	2	UNSIGNED8	0
RPDO1 Entry 1 (CONTROL WORD)	0x1600	1	UNSIGNED32	0x60400010
RPDO1 Entry 2 (MODE OF OPERATION)	0x1600	2	UNSIGNED32	0x60600008
RPDO1 Entry 3 (TORQUE TARGET)	0x1600	3	UNSIGNED32	0x60710010
Enable RPDO1 Entries	0x1600	0	UNSIGNED8	3
Disable RPDO2 Entries	0x1601	0	UNSIGNED8	0
RPDO2 COB ID	0x1401	1	UNSIGNED32	0x300 | $ID
RPDO2 Receive type (APPLY ON EVERY SYNC)	0x1401	2	UNSIGNED8	0
RPDO2 Entry 1 (TARGET VELOCITY)	0x1601	1	UNSIGNED32	0x60FF0020
RPDO2 Entry 2 (TARGET POSITION)	0x1601	2	UNSIGNED32	0x607A0020
Enable RPDO2 Entries	0x1601	0	UNSIGNED8	2
Disable TPDO1 Entries	0x1A00	0	UNSIGNED8	0
TPDO1 COB ID	0x1800	1	UNSIGNED32	0x180 | $ID
TPDO1 Transmit type (TX ON EVERY SYNC)	0x1800	2	UNSIGNED8	0
TPDO1 Entry 1 (STATUS WORD)	0x1A00	1	UNSIGNED32	0x60410010
TPDO1 Entry 2 (MODE DISPLAY)	0x1A00	2	UNSIGNED32	0x60610008
TPDO1 Entry 3 (ACTUAL POSITION)	0x1A00	3	UNSIGNED32	0x60640020
Enable TPDO1 Entries	0x1A00	0	UNSIGNED8	3
Disable TPDO2 Entries	0x1A01	0	UNSIGNED8	0
TPDO2 COB ID	0x1801	1	UNSIGNED32	0x280 | $ID
TPDO2 Transmit type (TX ON EVERY SYNC)	0x1801	2	UNSIGNED8	0
TPDO2 Entry 1 (ACTUAL VELOCITY)	0x1A01	1	UNSIGNED32	0x606C0020
TPDO2 Entry 2 (ACTUAL CURRENT)	0x1A01	2	UNSIGNED32	0x60780010
Enable TPDO2 Entries	0x1A01	0	UNSIGNED8	2

Example 3: Use TPDO3 to Get Temperature Feedback at a Lower Rate

For some types of sensors, you may not need to receive sensor feedback at the same rate you are sending SYNC messages. This can also be helpful to reduce CAN traffic. By adjusting your TPDO Transmit Type, you can have the device report a value after every nth SYNC.

Transmit Type	Description
0	Transmit on every SYNC
1-240	Transmit on nth SYNC
252	Update on Sync & Tx on RTR (not supported)
253	Update & Tx on RTR (not supported)
254	Manufacturer-specific (not supported)
255	Async (not supported)

Description	Index	SubIdx	Type	Value
Disable TPDO3 Entries	0x1A02	0	UNSIGNED8	0
TPDO3 COB ID	0x1802	1	UNSIGNED32	0x380 | $ID
TPDO3 Transmit type (TX ON EVERY 10TH SYNC)	0x1802	2	UNSIGNED8	10
TPDO3 Entry 1 (AMPLIFIER TEMPERATURE)	0x1A02	1	UNSIGNED32	0x30000210
Enable TPDO3 Entries	0x1A02	0	UNSIGNED8	1

Example 4: Using Dummy Entries

Dummy entries are an advanced technique to get individual nodes to pick off different entries of a single PDO based on their PDO configuration. For example, if you wanted to pack a set of two Target Torques destined for nodes 1 & 2 into a single RPDO that they both listen to, you could save a lot of CAN bandwidth by configuring the nodes this way. Notice that RPDO entries 1&2 are reversed for Node 2, when compared with Node 1. In this way, Node 1 will essentially discard/ignore Node 2's torque target, and Node 2 will ignore Node 1's torque target, even though both torque targets are found in the same CANopen frame.

For Node 1:

Description	Index	SubIdx	Type	Value
Disable RPDO1 Entries	0x1600	0	UNSIGNED8	0
RPDO1 COB ID	0x1400	1	UNSIGNED32	*0x201*
RPDO1 Receive type (APPLY ON EVERY SYNC)	0x1400	2	UNSIGNED8	0
RPDO1 Entry 1 (TORQUE TARGET)	0x1600	1	UNSIGNED32	0x60710010
RPDO1 Entry 2 (DUMMY 16-BIT)	0x1600	2	UNSIGNED32	0x00030010
Enable RPDO1 Entries	0x1600	0	UNSIGNED8	2

For Node 2:

Description	Index	SubIdx	Type	Value
Disable RPDO1 Entries	0x1600	0	UNSIGNED8	0
RPDO1 COB ID	0x1400	1	UNSIGNED32	*0x201*
RPDO1 Receive type (APPLY ON EVERY SYNC)	0x1400	2	UNSIGNED8	0
RPDO1 Entry 1 (DUMMY 16-BIT)	0x1600	1	UNSIGNED32	0x00030010
RPDO1 Entry 2 (TORQUE TARGET)	0x1600	2	UNSIGNED32	0x60710010
Enable RPDO1 Entries	0x1600	0	UNSIGNED8	2

Configure Warnings and Faults

P4 implements this subset of the CiA DS402 warnings and faults.

Description	Index	SubIdx	Type	Value
Encoder magnet fault enable (bool)	0x2403	0	UNSIGNED8	0
Motor temp limit (deg C)	0x220A	0	UNSIGNED8	70
Bus overvoltage limit (0.1 V)	0x2384	6	UNSIGNED16	520
Bus undervoltage limit (0.1 V)	0x2384	7	UNSIGNED16	180
Amplifier overtemperature (deg C)	0x2384	9	UNSIGNED8	70
CAN overrun fault enable (bool)	0x2404	0	UNSIGNED8	1
CAN bus-off recovery fault enable (bool)	0x2405	0	UNSIGNED8	1
SYNC loss fault timeout (ms)	0x2406	0	UNSIGNED16	500
Velocity tracking warning window (cts/sec) - 1%	0x606D	0	UNSIGNED16	0
Velocity tracking warning timeout (ms)	0x606E	0	UNSIGNED16	100
Velocity tracking fault window (cts/sec)	0x2104	0	UNSIGNED16	0
Velocity tracking fault timeout (ms)	0x2105	0	UNSIGNED16	100
Position tracking warning window (cts)	0x6065	0	UNSIGNED32	0
Position tracking warning timeout (ms)	0x6066	0	UNSIGNED16	500
Position tracking fault window (cts)	0x2120	0	UNSIGNED32	0
Position tracking fault timeout (ms)	0x2121	0	UNSIGNED16	500

Configure Drive Parameters

For effective motion control, you will need to ensure the drive parameters in your motor's configuration file (CSV) match your motor specifications and application requirements.

Description	Index	SubIdx	Type	Value
# Motor specifications (from datasheet)				
Motor rated torque (mNm)	0x6076	0	UNSIGNED32	10000
Max motor vel (cts/sec)	0x6080	0	UNSIGNED32	52500
Motor poles	0x3011	3	UNSIGNED8	16
Motor Kt (mNm/A)	0x3011	4	UNSIGNED16	530
Max continuous current (mA)	0x3011	8	UNSIGNED16	3430
Peak current (mA)	0x3011	9	UNSIGNED16	17152
Peak current duration (ms)	0x3011	10	UNSIGNED16	500
Motor calibration current (mA)	0x3011	11	UNSIGNED16	500
# Profile Mode				
Acceleration (cts/s^2)	0x6083	0	UNSIGNED32	2000000
Deceleration (cts/s^2)	0x6084	0	UNSIGNED32	2000000
Quick-stop deceleration (cts/s^2)	0x6085	0	UNSIGNED32	2000000
Torque slope (rated trq/1000/s)	0x6087	0	UNSIGNED32	5000

# Position control				
Position proportional gain (Kp)	0x2382	1	UNSIGNED16	10
Velocity feed-forward gain (Kvff)	0x2382	2	UNSIGNED16	0
Acceleration feed-forward gain (Kaff)	0x2382	3	UNSIGNED16	0
Output multiplier (100=1.0)	0x2382	4	UNSIGNED16	100
Position minimum (cts)	0x607D	1	INTEGER32	-0x7FFFFFFF
Position maximum (cts)	0x607D	2	INTEGER32	0x7FFFFFFF
# Velocity control				
Velocity proportional gain (Kp)	0x2381	1	UNSIGNED32	0x3A9D4952
Velocity integral gain (Ki)	0x2381	2	UNSIGNED32	0x38FDC161
Encoder velocity, low-pass filter (Fc, Hz)	0x2100	1	UNSIGNED16	200
Velocity controller input, low-pass filter (Fc, Hz)	0x2100	2	UNSIGNED16	0
Velocity controller ouput, low-pass filter (Fc, Hz)	0x2100	3	UNSIGNED16	0
# Current control				
DQ-Axis current proportional gain (Kp)	0x2380	1	UNSIGNED32	0x3CB50B0F
DQ-Axis current integral gain (Ki)	0x2380	2	UNSIGNED32	0x48042F87
# Amplifier settings				
PWM frequency (Hz)	0x3001	1	UNSIGNED32	16000
Dead time (ns)	0x3001	2	UNSIGNED16	200
Max propagation delay (ns)	0x3001	3	UNSIGNED16	160
Max settling time (ns)	0x3001	5	UNSIGNED16	250
Sampling time (ns)	0x3001	6	UNSIGNED16	400
Conversion time (ns)	0x3001	7	UNSIGNED16	1475
Max sensor current (mA)	0x3001	8	UNSIGNED16	10000
Nominal bus voltage (V*10)	0x3001	9	UNSIGNED16	240
Gate driver input inversion? (bool)	0x3001	10	UNSIGNED8	0
# iSense circuit values				
Alpha Gain (*1000)	0x3008	4	UNSIGNED16	7490
Alpha Shunt (mOhms)	0x3008	5	UNSIGNED16	20
Beta Gain (*1000)	0x3009	4	UNSIGNED16	7490
Beta Shunt (mOhms)	0x3009	5	UNSIGNED16	20
# Encoder settings				
Encoder resolution (cts)	0x3013	1	UNSIGNED32	4096
Encoder user zero (cts)	0x3013	2	UNSIGNED32	0
Encoder user polarity	0x3013	3	UNSIGNED8	0x01
Encoder type (0 = Internal)	0x3013	4	UNSIGNED8	0
Encoder lag factor	0x3013	5	UNSIGNED16	0

Calibration

The Puck Utility application's "Calibrate" menu provides access to several calibration steps. Whenever a new motor is paired with a puck, or when any motor constants are updated in the CANopen CSV, these calibration steps should be executed in the order shown here.

Current Sense Bias

The P4 needs to know its current sensors' internal bias voltages when no motor current is flowing, so this routine samples the sensors when the motor phases are not powered.

Current Sense Gain Factor

The current sense gain factor routine ensures that the current sensor amplitudes are matched to each other. This results in more balanced commutation currents and smoother operation. For this step, it is important to ensure that the motor shaft is free to spin during the calibration. Any load will invalidate the calibration.

Encoder Zero

Because the encoder magnet is attached at an arbitrary angle with respect to the rotor magnets, the P4 needs to record the raw encoder value when the rotor is stalled under Phase A. The P4 uses this calibration value to set the electromagnetic field orientation for optimal torque generation. This calibration is performed using Mode 12: Phase Voltage with Angle. The phase voltage is automatically derived from the Motor Calibration Current (CANopen dictionary entry 0x3011,11). To calculate an appropriate Motor Calibration Current for your motor, try using 20% of the Continuous Current rating for your motor, in mA.

Example for the Maxon EC-Max 22, model 283860:

Parameter	Value
Continuous current	716 mA
Motor calibration current	716 * 20% = 143 mA

For this step, it is important to ensure that the motor shaft is free to spin during the calibration. Any load will invalidate the calibration.

This calibration routine also determines and records the motor's phase wiring order so that the commutation direction is aligned with the raw encoder direction. This is required for proper commutation, but it does not affect the observed motor direction.

Encoder Lag

The commutation code requires an accurate estimation of the motor's electrical angle in order to orient its electromagnetic field for optimum torque. However, there are delays between reading the encoder, calculating the electrical angle, and generating the electromagnetic field. Accounting for these delays requires predicting where the rotor will be at the moment the electromagnetic field is updated. This calibration step analyzes the motor's performance in both directions while varying the "encoder lag" constant to find the optimum calibration value.

CANopen Information

High-level overview

CAN is a multi-drop differential serial bus that allows nodes to communicate reliably at rates up to 1 Mbps across distances up to 40 meters, even in electrically-noisy environments. CANopen is a protocol designed for use on top of the CAN physical layer.

For more technical information about CAN, see the “CAN Physical Layer” section of the Appendix.

CiA DS301 — Communication

The CAN in Automation (CiA) industry group governs the DS301 standard for CANopen communication. This standard organizes device parameters into an Object Dictionary (OD) of data records. Each record has a 16-bit index, and each value in a record is accessible by an 8-bit sub-index. Values can be 8/16/32 bits long, signed or unsigned.

Table 1: Object Dictionary Key

Term	Definition
U08	Unsigned 8-bit integer
U16	Unsigned 16-bit integer
U32	Unsigned 32-bit integer
I08	Signed 8-bit integer
I16	Signed 16-bit integer
I32	Signed 32-bit integer
RO	Read-Only
RW	Read-Write

Table 2: Communication Profile Area (0x1000 to 0x1FFF)

Index	SubIdx	Type	Access	Description
0x0002	0	I08	RW	Dummy entry, for PDO padding
0x0003	0	I16	RW	Dummy entry, for PDO padding
0x0004	0	I32	RW	Dummy entry, for PDO padding
0x0005	0	U08	RW	Dummy entry, for PDO padding
0x0006	0	U16	RW	Dummy entry, for PDO padding
0x0007	0	U32	RW	Dummy entry, for PDO padding
0x1000	0	U32	RO	Device Type: [16-bit extra info | 16-bit device profile]
0x1001	0	U08	RO	Error bitfield (see below)
0x1002	0	U32	RO	Manufacturer Status Register
0x1003	0	U08	RO	Error history, record count
0x1003	1-8	U32	RO	Error history, most recent error is at 0x1003,1
0x1005	0	U32	RW	Sync COB ID
0x1006	0	U32	RO	Sync period in microseconds (for producer)
0x1008	0	STR	RO	Mfg device name (4 chars)

0x1009	0	STR	RO	Mfg hardware version (4 chars)
0x1010	1	U32	RW	Save All, write 0x65766173 “SAVE”
0x1010	2	U32	RW	Save Communication parameters, write 0x65766173
0x1010	3	U32	RW	Save Application parameters, write 0x65766173
0x1010	4	U32	RW	Save single entry, write [Idx << 8 | SubIdx]
0x100A	0	STR	RO	Mfg software version (4 chars)
0x1014	0	U32	RW	Emergency COB ID (default = 0x80 + NodeID)
0x1015	0	U16	RW	Emergency inhibit time
0x1017	0	U16	RW	Heartbeat period in milliseconds
0x1018	1	U32	RO	Vendor ID
0x1018	2	U32	RO	Product code
0x1018	3	U32	RO	Revision number
0x1018	4	U32	RO	Serial number
0x1200	1	U32	RO	SDO receive COB ID, default = 0x600 + NodeID
0x1200	2	U32	RO	SDO transmit COB ID, default = 0x580 + NodeID
0x1F80	0	U32	RW	NMT startup (0 = Boot to Op, 4 = Boot to Pre-op)

Table 3: PDO Configuration

Index	SubIdx	Type	Access	Description
0x1400-0x1403	1	U32	RW	RPDO COB ID
0x1400-0x1403	2	U08	RW	RPDO Rx type: 0-240 = Apply on nth Sync, 254 = Mfg specific, 255 = Async (Apply upon receipt)
0x1600-0x1603	0	U08	RW	RPDO mapping, number of mapped objects (0-4)
0x1600-0x1603	1-4	U32	RW	RPDO mapping, bits 16-31: Index, bits 8-15: SubIdx, bits 0-7 bit length (8, 16, or 32 bits)
0x1800-0x1803	1	U32	RW	TPDO COB ID
0x1800-0x1803	2	U08	RW	TPDO Tx type: 0-240 = Tx on nth Sync, 254 = Mfg specific, 255 = Async
0x1800-0x1803	3	U16	RW	TPDO inhibit time in multiples of 100 uS (not implemented)
0x1800-0x1803	4	U08	RO	Unused
0x1800-0x1803	5	U16	RW	TPDO event timer in ms (not implemented)
0x1A00-0x1A03	0	U08	RW	TPDO mapping, number of mapped objects (0-4)
0x1A00-0x1A03	1-4	U32	RW	TPDO mapping, bits 16-31: Index, bits 8-15: SubIdx, bits 0-7 bit length (8, 16, or 32 bits)

Network Management (NMT) State Machine

A CANopen device can be in one of four states:

Boot-up (0x00)

- When a CANopen device is powered on, it begins in a “Boot-up” state where it initializes the Application and Communication areas of its object dictionary.
- The device does not consume any messages
- It emits a single boot-up (heartbeat) message
- The device automatically enters Pre-op or Operational state after initialization
  - Next state depends on the value of the device’s Startup entry (0x1F80,0).
  - 0 = Boot to Operational, 4 = Boot to Pre-Op.
  - If periodic heartbeats are configured (0x1017,0 > 0), then it will begin sending heartbeats with the new state.

Stopped (0x04)

- Communication limited to NMT and heartbeats only

Pre-operational (0x7F)

- All messages except RPDOs are allowed
- Send TPDO responses to SYNC messages

Operational (0x05)

- All messages are allowed

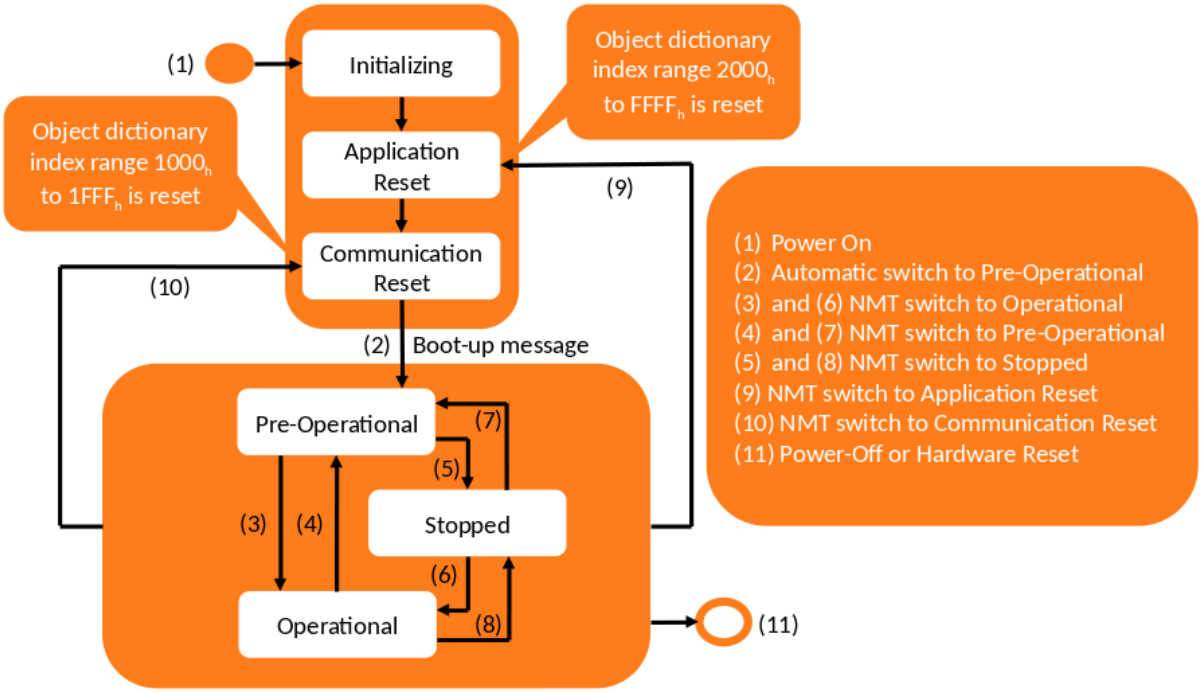


Figure 2: NMT State Machine

The NMT state can be controlled by sending NMT messages to the device. The device state is reported in the device’s heartbeat. For more information about the format and payload of NMT messages, see the Appendix.

CiA DS402 — Device Profile for Drives and Motion Control

CiA defines standard device profiles (collections of well-defined object dictionary records) for many different types of devices in an effort to make devices more inter-operable with each other on a CANopen network. P4 uses device profile DS402 for Drives and Motion Control. Standard objects are located from 0x6000 to 0x6FFF. Manufacturer-specific objects are located from 0x2000 to 0x3FFF.

States, Modes, and Codes

Table 4: States and Modes

Index	SubIdx	Type	Access	Description
0x6040	0	U16	RW	Control Word
0x6041	0	U16	RO	Status Word
0x605A	0	U16	RW	Quick stop option code
0x605B	0	U16	RW	Shutdown option code
0x605C	0	U16	RW	Disable operation option code
0x605D	0	U16	RW	Halt option code
0x605E	0	U16	RW	Fault reaction option code
0x6060	0	U08	RW	Set mode of operation (see table below)
0x6061	0	U08	RO	Read mode of operation (see table below)
0x6502	0	U32	RO	Bitfield of supported drive modes

Control Word

9	8	7	6	5	4	3	2	1	0
OMS	H	FR	OMS	OMS	OMS	EO	QS	EV	SO

Key	Meaning
MS	Manufacturer-Specific
R	Reserved (0)
OMS	Operation Mode Specific
H	Halt
FR	Fault Reset
EO	Enable Operation
QS	Quick Stop
EV	Enable Voltage
SO	Switch On

The Control Mode Display (0x6061,0) is cleared to zero upon entering the state Operation Enabled.

Status Word

13	12	11	10	9	8	7	6	5	4	3	2	1	0
OMS	OMS	ILA	TR	RM	MS	W	SOD	QS	VE	F	OE	SO	RTSO

Key	Meaning
MS	Manufacturer-Specific
OMS	Operation Mode Specific
ILA	Internal Limit Active
TR	Target Reached
RM	Remote
W	Warning
SOD	Switch On disabled
QS	Quick Stop
VE	Voltage Enabled
F	Fault
OE	Operation enabled
SO	Switched On
RTSO	Ready To Switch On

When the Status Word (0x6041,0) indicates that the drive is in the “Operation Enabled” state (when its value = 0x37, typically), the drive will accept mode transition requests to its Control Mode (0x6060,0) entry.

Changing the Control Mode clears the mode-specific status bits of the Status Word.

Stopping Methods

* There are 5 different stopping methods, each configurable, * = default:
* - 605A Quick Stop (ControlWord[2])
* 1-4:OpEnabled -> QuickStop -> SOD
* 1 - Use Slow Down Ramp (0x6084)
* 2*- Use Quick Stop Ramp (0x6085)
* 5-8:OpEnabled -> QuickStop
* 5 - Use Slow Down Ramp (0x6084)
* 6 - Use Quick Stop Ramp (0x6085)
* - 605B Shutdown (ControlWord[0], OpEnabled -> RTSO)
* 0*- Immediate, No Ramp
* 1 - Use Slow Down Ramp (0x6084)
* - 605C Disable (OpEnabled -> SwitchedOn)
* 0 - Immediate, No Ramp
* 1*- Use Slow Down Ramp (0x6084)
* - 605D Halt (ControlWord[8], Remain OpEnabled)
* 1*- Use Slow Down Ramp (0x6084)
* 2 - Use Quick Stop Ramp (0x6085)
* - 605E Fault Reaction (AnyState -> FaultReactionActive -> Fault)
* 1 - Use Slow Down Ramp (0x6084)
* 2*- Use Quick Stop Ramp (0x6085)

Modes of Operation (0x6060, 0x6061)

Write to 0x6060 to set the mode of operation. But depending on the state of the actuator, you may not be able to set the mode of operation. Read from 0x6061 to determine the actuator's present mode of operation.

Table 5: List of Motion Control Modes

Mode	Description
0	Idle - Current controller turned off, motor phase leads tied to ground.
1	Profile Position
2	Velocity *(not yet implemented)*
3	Profile Velocity
4	Profile Torque
5	Reserved
6	Homing
7	Interpolated Position *(not yet implemented)*
8	Cyclic Synchronous Position - Latch a position target upon SYNC, follow a path that will reach the target position by the next SYNC
9	Cyclic Synchronous Velocity - Latch a velocity target upon SYNC, accelerate such that the target velocity will be reached by the next SYNC
10	Cyclic Synchronous Torque - Latch a torque target upon SYNC, alter the applied torque incrementally until the target torque is reached by the next SYNC
11	Cyclic Synchronous Torque with Commutation Angle *(not yet implemented)*
12	Phase Voltage with Commutation Angle *(used during motor calibration)*

Bitfield of supported drive modes (0x6502)

Bit	11	10	9	8	7	6	5	4	3	2	1	0
Mode	PVCA	CSTCA	CST	CSV	CSP	IP	HM	R	PT	PV	VL	PP
Default	1	0	1	1	1	0	1	0	1	1	0	1

PP: Profile Position, VL: Velocity, PV: Profile Velocity, PT: Profile Torque, R: Reserved, HM: Homing, IP: Interpolated Position, CSP: Cyclic Synchronous Position, CSV: Cyclic Synchronous Velocity, CST: Cyclic Synchronous Torque, CSTCA: Cyclic Synchronous Torque with Commutation Angle, PVA: Phase Voltage with Commutation Angle.

Motor Parameters

Table 6: Motor Parameters

Index	SubIdx	Type	Access	Description
0x6076	0	I32	RW	Rated torque (mNm)
0x6080	0	I32	RW	Rated speed (cts/sec)
0x60EA	0	I16	RW	Electrical angle (-2^15 to 2^15-1), corresponding to -180 to +180 degrees. Writeable only in modes 11 & 12.
0x3011	1	I16	RW	Electrical zero (cts), the raw encoder reading when the rotor is stalled under Phase A (calibrated)
0x3011	2	I08	RW	Electrical polarity (-1 or +1), phase order calibration
0x3011	3	I08	RW	Pole count (pole pairs * 2)
0x3011	4	U16	RW	Torque constant (mNm/A)
0x3011	8	I16	RW	Max continuous current (mA)
0x3011	9	I16	RW	Peak current (mA)
0x3011	10	I16	RW	Peak current duration (ms)
0x3012	1	I16	RO	Raw encoder reading (cts)
0x3013	1	U32	RW	Encoder resolution (cts/rev)
0x3013	5	U32	RW	Encoder lag factor (calibrated)

Amplifier Parameters

Table 7: Amplifier Parameters

Index	SubIdx	Type	Access	Description
0x6078	0	I16	RO	Actual current (/1000 of Peak Current), Q-axis current derived from sensor
0x3000	1	U16	RO	Sensed bus voltage (units TBD)
0x3000	2	U16	RO	Temperature (units TBD)
0x3001	1	U32	RW	PWM frequency (Hz), control rate = PWM freq / 5
0x3001	2	U16	RW	Dead time (ns)
0x3001	3	U16	RW	Minimum gate driver propagation delay (ns)
0x3001	4	U16	RW	Delta propagation delay (ns), max delay – min delay
0x3001	5	U16	RW	Maximum settling time of current sensor (ns), calibrated
0x3001	6	U16	RW	ADC sampling time (ns)
0x3001	7	U16	RW	ADC conversion time (ns)
0x3001	8	I16	RW	Maximum current (mA)
0x3001	9	I16	RW	Maximum phase voltage (mV), bus voltage / sqrt(3)
0x3022	1	U16	RO	Aux ADC 1 (0-4095, corresponding to 0-3.3 V)
0x3022	2	U16	RO	Aux ADC 2 (0-4095, corresponding to 0-3.3 V)

Motion Control

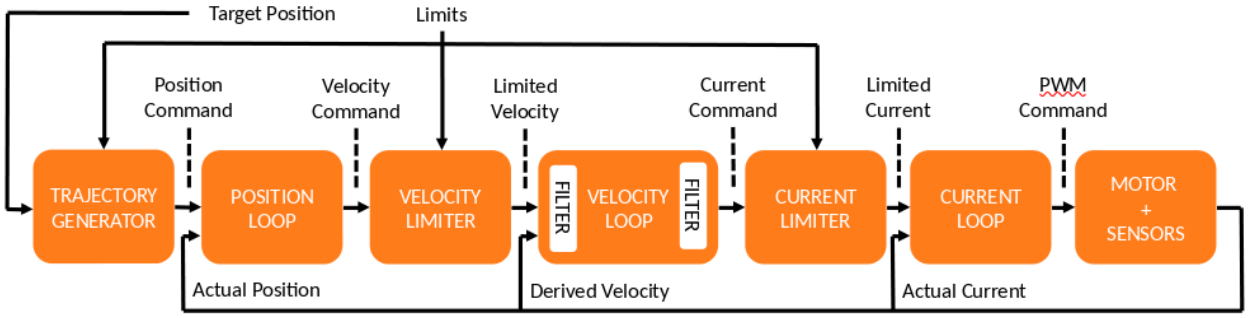


Figure 3: Nested Control Architecture

Barrett's P4 motion controller uses up to three nested control loops: current, velocity, and position- to control a motor, depending on the operating mode.

In position mode, all three control loops are enabled. As shown above, the position loop drives the nested velocity loop, which drives the nested current loop.

In velocity mode, the velocity loop drives the current loop. And in current mode, the current loop is driven directly.

In each mode, a target (along with associated limits) is given to a profile generator which outputs a demand. There are 6 different profile generators:

Table 8: Profile Generators

#	Mode	Description
1	1	Profile Position (PP)
2	3	Profile Velocity (PV)
3	4	Profile Torque (PT)
4	8	Cyclic Synchronous Position (CSP)
5	9	Cyclic Synchronous Velocity (CSV)
6	10	Cyclic Synchronous Torque (CST)

The demand is sent to a controller (with feedback) which outputs an effort. There are 3 different controllers:

Table 9: Control Loops

Input	Control Loop	Gains	Output
Position demand	Position controller	Kp, Kvff, Kaff, Kout	Velocity effort/demand
Velocity demand	Velocity controller	Kp, Ki	Torque effort/demand
Torque demand	Current controller	Kp, Ki	Phase voltages

Profile Position Mode

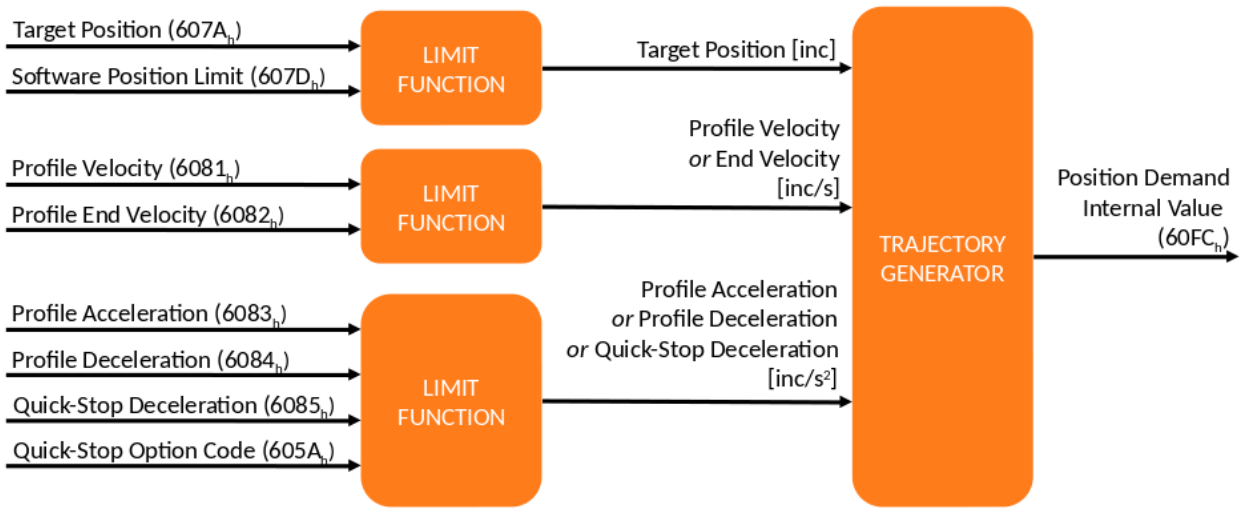


Figure 4: Profile Position Mode

In this mode, a profile position generator accepts a waypoint and outputs a series of intermediate positions (0x6062,0; 0x60FC,0), velocities (0x2250,0), and accelerations (0x2251,0) to reach the commanded waypoint.

Table 10: Object Dictionary Entries for a Waypoint

Index	SubIdx	Type	Access	Description
0x607A	0	I32	RW	Target position (cts)
0x6081	0	U32	RW	Profile (cruise) Velocity (cts/s)
0x6082	0	U32	RW	Endpoint (final) Velocity (cts/s)
0x6083	0	U32	RW	Acceleration (cts/s^2)
0x6084	0	U32	RW	Deceleration (cts/s^2)

The Target Position may be Absolute (with respect to zero), or Relative (an offset from the present commanded position).

Control Word Bits for Profile Position

9	8	7	6	5	4	3	2	1	0
CSP	H	-	ABS/REL	BUF/IMM	NEW	-	-	-	-

Key	Meaning
NEW	Set to ‘1’ to accept a new waypoint, set to ‘0’ after ACK
BUF/IMM	0: Buffer new waypoints, 1: Immediately replace active profile
ABS/REL	0: Target is Absolute, 1: Target is relative to present position
H	Halt
CSP	Change on Set Point

Status Word Bits for Profile Position

13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE	ACK	ILA	TR	-	-	-	-	-	-	-	-	-	-

Key	Meaning
FE	Following Error
ACK	Set-point acknowledge
ILA	Internal Limit Active
TR	Target Reached

There are four ways to use the Profile Position Mode:

	Buffered (ControlWord[5] = 0)	**Immediate (ControlWord[5] = 1)**
Absolute (ControlWord[6] = 0)	If a profile is active, the new absolute waypoint is added to a circular buffer, otherwise it is executed immediately	The active profile, if any, is replaced by the new absolute waypoint
Relative (ControlWord[6] = 1)	If a profile is active, the new relative waypoint is added to a circular buffer, otherwise it is executed immediately	The active profile, if any, is replaced by the new relative waypoint

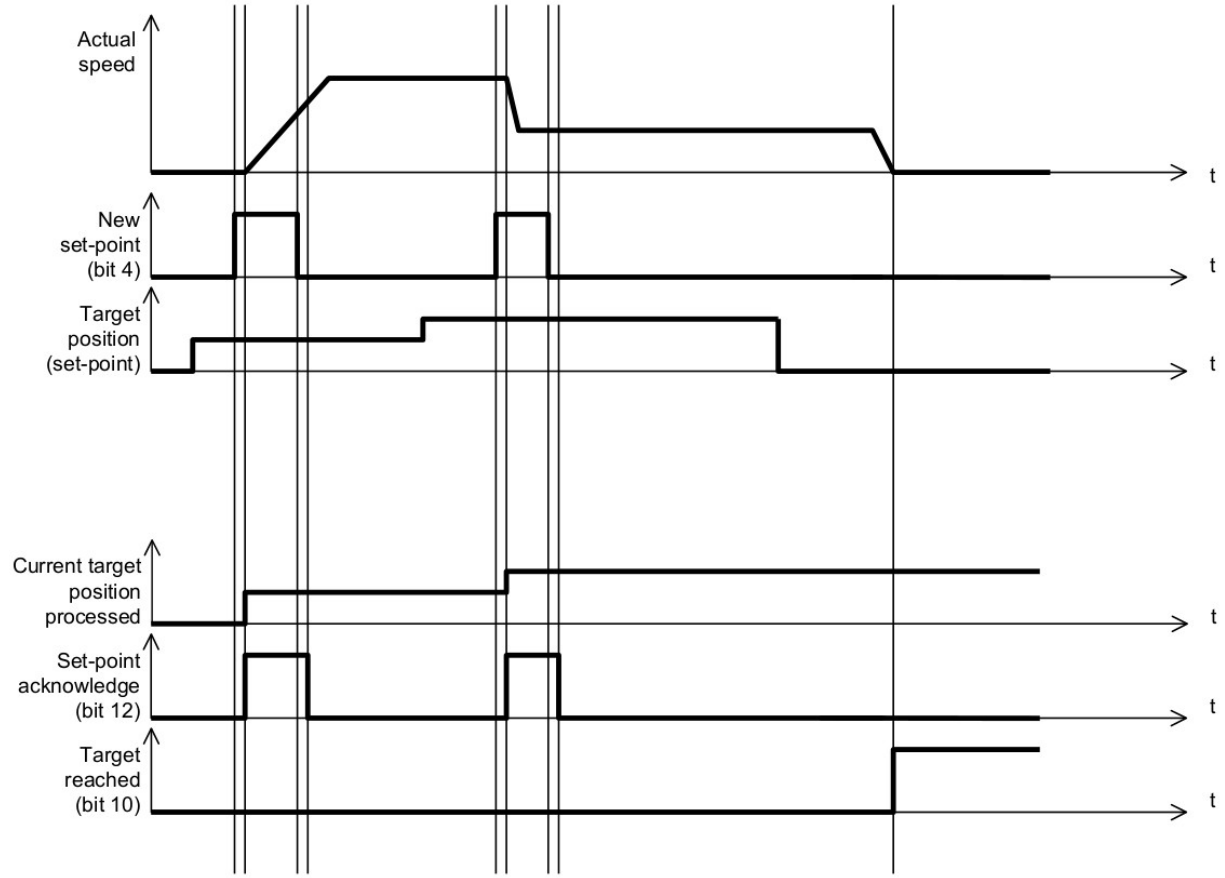


Figure 5: Profile Position, Immediate

Setting the “Immediate” bit (bit 5) of the ControlWord clears any waypoints in the circular buffer.

Profile Position, Buffered

You can store up to eight waypoints in the buffer for sequential execution.

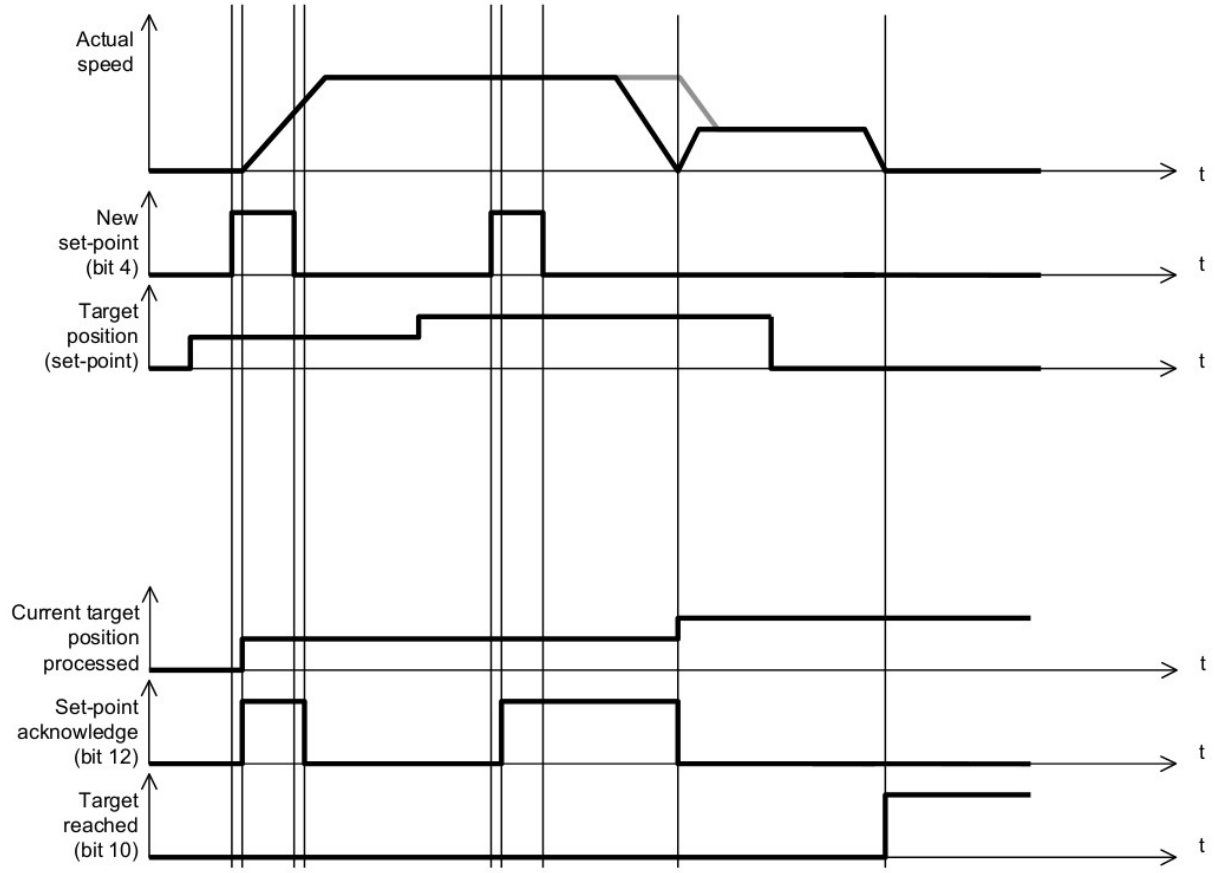


Figure 6: Profile Position, Buffered

If you are using buffered waypoints, and ControlWord[9] “Change on Set Point” = 1, then the profile generator will attempt to blend the waypoints by replacing the Endpoint Velocity of the active profile with the Profile (cruise) Velocity of the next waypoint. This is represented by the gray line, above.

Table 11: Position Control & Feedback Parameters

Index	SubIdx	Type	Access	Description
0x607A	0	I32	RW	Target position (cts), modes 1 & 8
0x60FC	0	I32	RO	Internal position demand (cts)
0x2250	0	I32	RO	Internal velocity demand (cts/s)
0x2251	0	I32	RO	Internal acceleration demand (cts/s^2)
0x6064	0	I32	RO	Actual position (cts)
0x60F4	0	I32	RO	Position following error (cts), demand - actual

Table 12: Profile Position Configuration Parameters

Index	SubIdx	Type	Access	Description
0x607D	1	I32	RW	Position minimum (cts)
0x607D	2	I32	RW	Position maximum (cts)
0x6081	0	I32	RW	Profile (cruise) velocity (cts/s)
0x6082	0	U32	RW	Profile Endpoint (final) velocity (cts/s)
0x6083	0	U32	RW	Acceleration (cts/s^2)
0x6084	0	U32	RW	Deceleration (cts/s^2)
0x6085	0	U32	RW	Quick-stop deceleration (cts/s^2)
0x2382	1	U16	RW	Position Proportional Gain (Kp)
0x2382	2	U16	RW	Velocity Feed-forward Gain (Kvff)
0x2382	3	U16	RW	Acceleration Feed-forward Gain (Kaff)
0x2382	4	U16	RW	Output effort multiplier (Kout, 100 = 1.0)
0x6065	0	U32	RW	Position tracking warning window (cts)
0x6066	0	U16	RW	Position tracking warning timeout (ms)
0x2120	0	U32	RW	Position tracking fault window (cts)
0x2121	0	U16	RW	Position tracking fault timeout (ms)

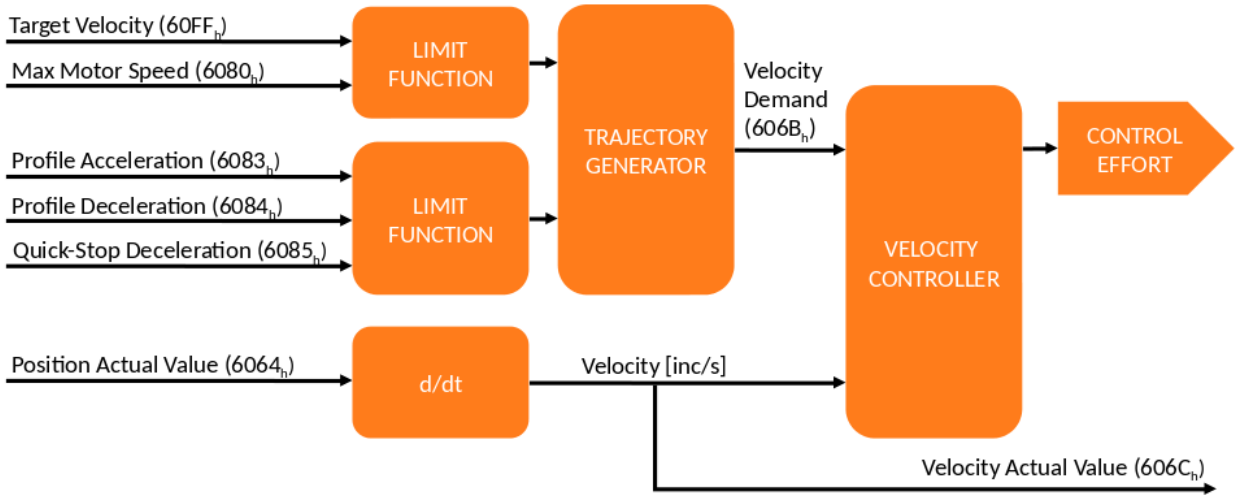


Figure 7: Profile Velocity Mode

In this mode, a target velocity is specified, along with acceleration and deceleration limits. At each time step, the controller applies some acceleration or deceleration (if required) to approach the target velocity.

Table 13: Velocity Control & Feedback Parameters

Index	SubIdx	Type	Access	Description
0x60FF	0	I32	RW	Target velocity (cts/s), modes 3 & 9
0x606B	0	I32	RO	Internal velocity command (cts/s)
0x606C	0	I32	RO	Actual velocity (cts/s), calculated from encoder history
0x210E	0	I32	RO	Velocity error (cts/s)
0x2100	1	U16	RW	Actual velocity filter, low-pass cutoff frequency (Hz)
0x2100	2	U16	RW	Velocity controller input filter, cutoff frequency (Hz)
0x2100	3	U16	RW	Velocity controller output filter, cutoff frequency (Hz)

Table 14: Profile Velocity Configuration Parameters

Index	SubIdx	Type	Access	Description
0x6080	0	U32	RW	Max velocity (cts/s)
0x6083	0	U32	RW	Acceleration (cts/s^2)
0x6084	0	U32	RW	Deceleration (cts/s^2)
0x6085	0	U32	RW	Quick-stop deceleration (cts/s^2)
0x2381	1	U32	RW	Velocity proportional gain (Kp)
0x2381	2	U32	RW	Velocity integral gain (Ki)
0x606D	0	U16	RW	Velocity tracking warning window (cts/s)
0x606E	0	U16	RW	Velocity tracking warning timeout (ms)
0x2104	0	U16	RW	Velocity tracking fault window (cts/s)
0x2105	0	U16	RW	Velocity tracking fault timeout (ms)

Profile Torque Mode

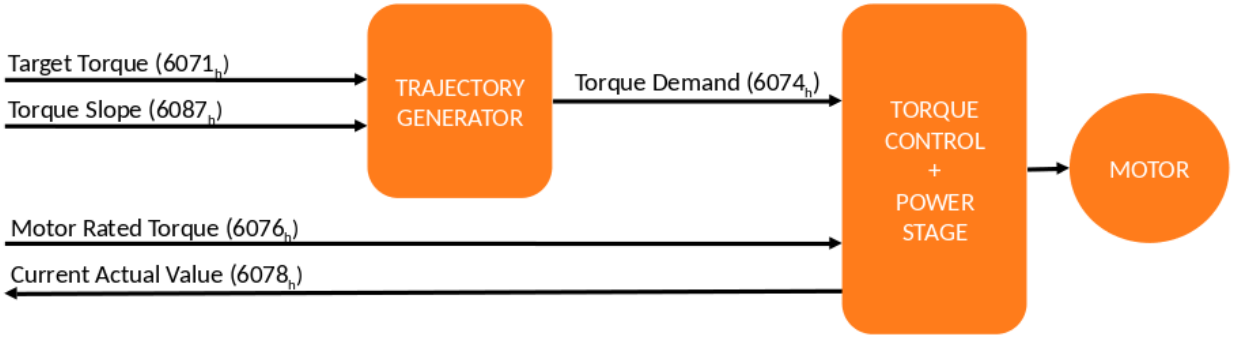
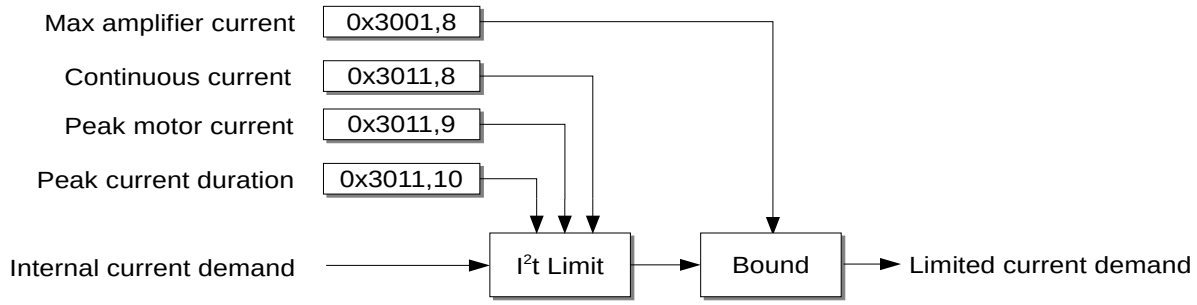


Figure 8: Profile Torque Mode

In this mode, a target torque is specified, along with a torque slope (0x6087,0). At each time step, the controller increases or decreases the applied torque based on the torque slope (if required) to approach the target torque.

This Current Demand is then limited based on the following diagram:



Current Limiting

Finally, this current demand is sent to a PI controller to generate the required motor currents.

Table 15: Torque Control & Feedback Parameters

Index	SubIdx	Type	Access	Description
0x6071	0	I16	RW	Target torque (/1000 of rated torque)
0x6074	0	I16	RO	Internal torque demand (/1000 of rated torque)
0x6087	0	U32	RW	Torque slope (/1000 of rated torque / sec)

Current Control Parameters

Index	SubIdx	Type	Access	Description
0x3011	8	U16	RW	Max continuous current (mA)
0x3011	9	U16	RW	Peak current (mA)
0x3011	10	U16	RW	Peak current duration (ms)
0x6078	0	I16	RO	Q-axis actual current (/1000 of peak current)
0x3010	6	I16	RO	D-axis actual current (/1000 of peak current)
0x2380	1	U32	RW	Current proportional gain (Kp)
0x2380	2	U32	RW	Current integral gain (Ki)

Cyclic Synchronous Configuration

The following two parameters are required to generate the “Cycles per SYNC” value shown in the diagrams below.

Table 16: Cyclic Synchronous Mode Configuration Parameters

Index	SubIdx	Type	Access	Description
0x60C2	1	U08	RW	Interpolation time period value
0x60C2	2	I08	RW	Interpolation time period scale (10^n), use -3 for ms

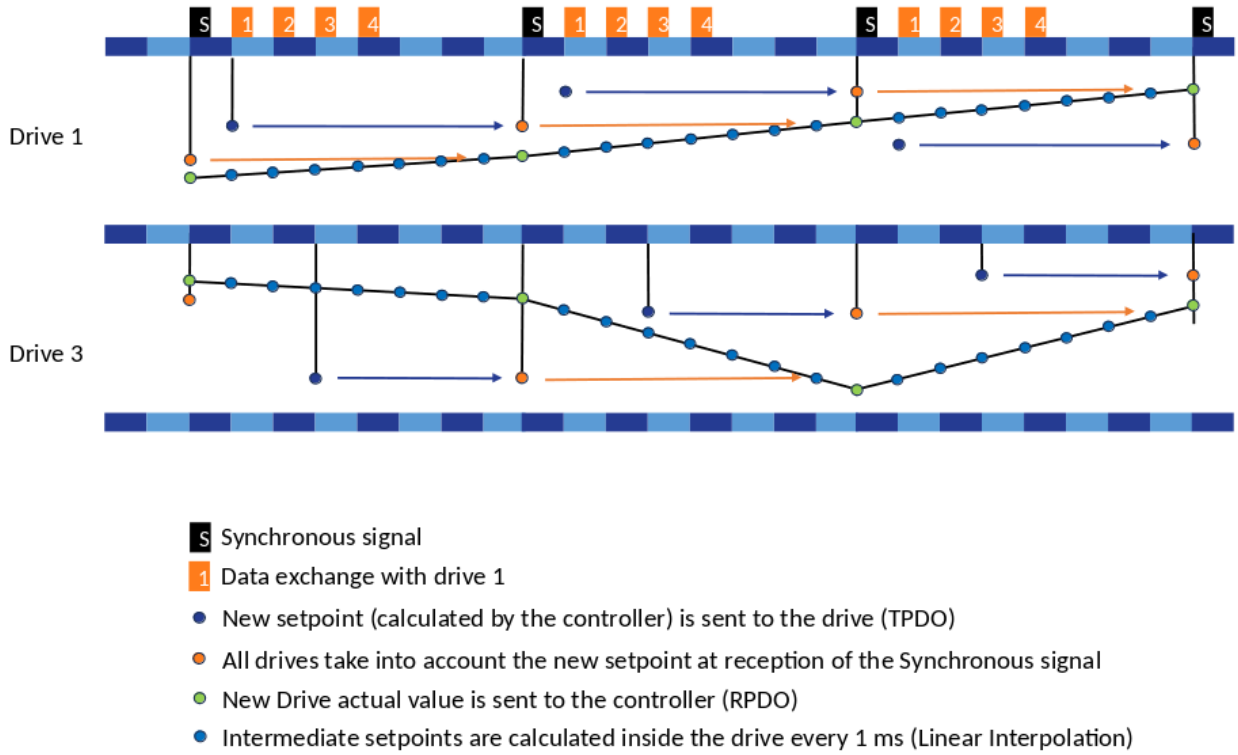
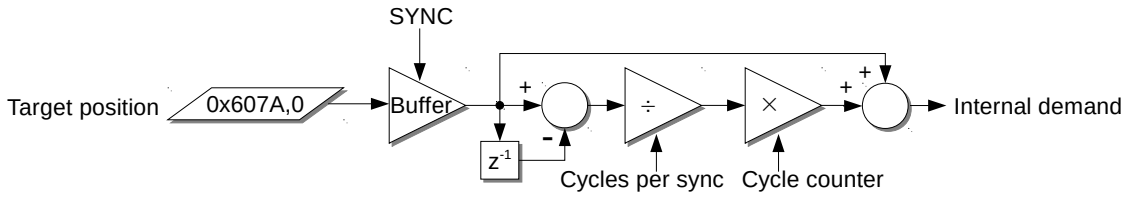


Figure 9: How Cyclic Synchronous Control Works

Cyclic Synchronous Position Mode

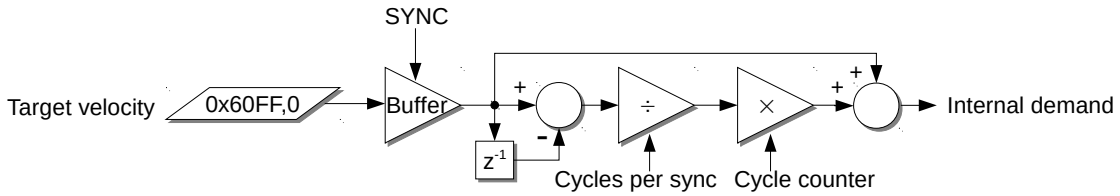
In this mode, the Target Position (0x607A,0) is used as an input to generate an Internal Position Demand which is then fed to the Profile Position controller (described above). A Position Offset (0x60B0,0), if set, will be added to the Target Position.



Cyclic Synchronous Position (CSP), Linear Interpolation

Cyclic Synchronous Velocity Mode

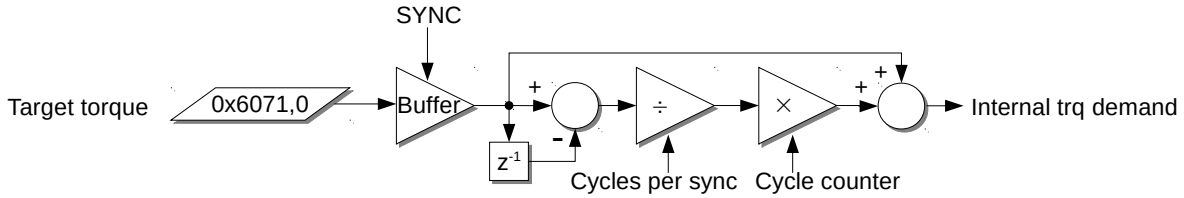
In this mode, the Target Velocity (0x60FF,0) is used as an input to generate an Internal Velocity Demand which is then fed to the Profile Velocity controller (described above). A Velocity Offset (0x60B1,0), if set, will be added to the Target Velocity.



Cyclic Synchronous Velocity (CSV), Linear Interpolation

Cyclic Synchronous Torque Mode

In this mode, the Target Torque (0x6071,0) is used as an input to generate an Internal Torque Demand which is then fed to the Profile torque controller (described above). A Torque Offset (0x60B2,0), if set, will be added to the Target Torque.



Cyclic Synchronous Torque (CST), Linear Interpolation

Motion Control State Machine

The CANopen Finite State Automaton (FSA) for motion control is a state machine designed to provide safe and consistent operation across all drives in a system. Transitions between states are governed by the Control Word (0x6040,0) and any active faults (0x603F,0). The present state of the drive is available in the Status Word (0x6041,0).

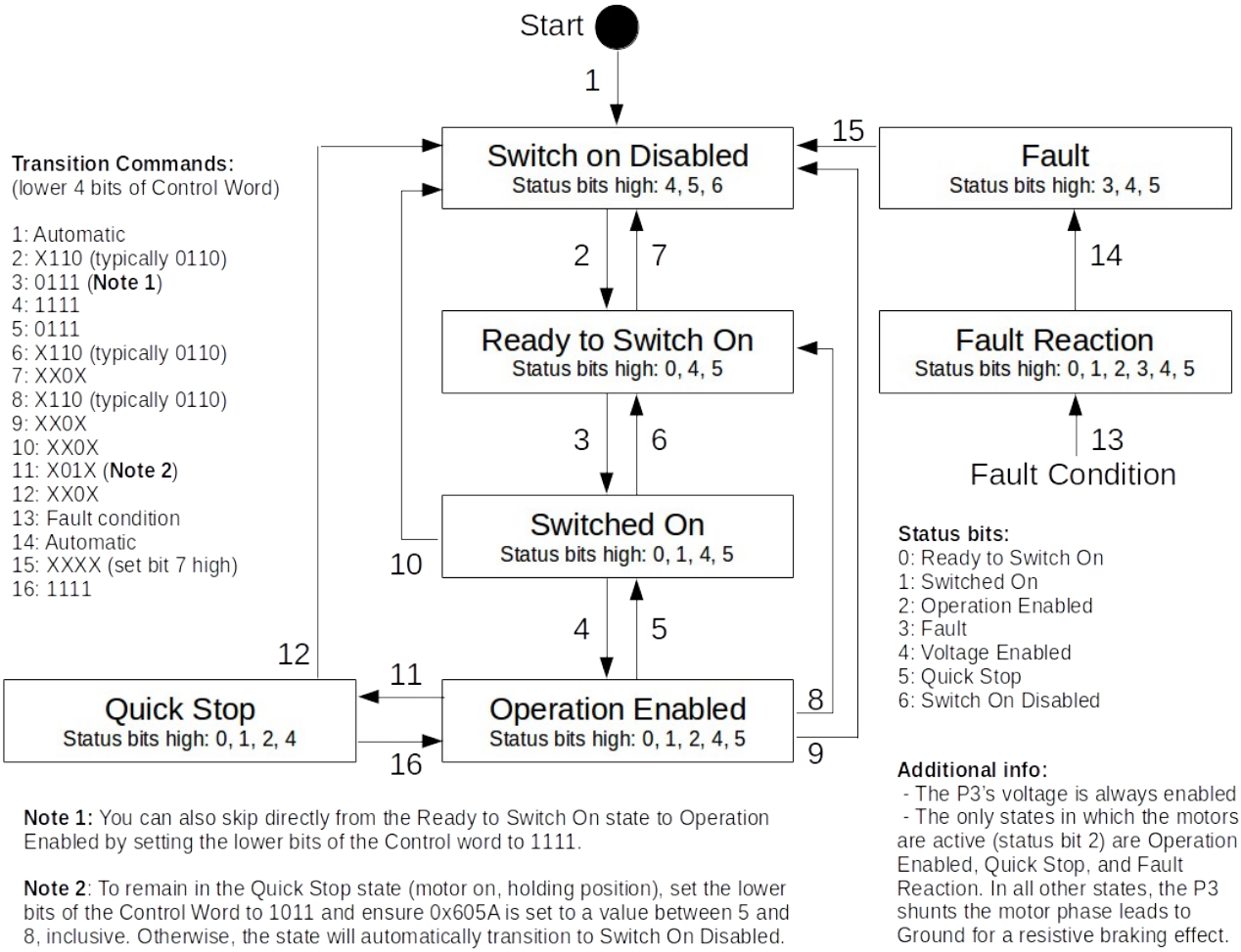


Figure 10: CANopen Finite State Automaton (FSA) for Motion Control

Homing

P4 supports Homing Method (0x6098,0) 37 (Homing on Current Position), only. In this method, the present position sensor information is taken to be the home position.

When the Control Mode (0x6060,0) is set to Homing (6), and the ControlWord's bit 4 "Initiate Homing" is set to 1, the Actual Position(0x6064,0) is reset to the Homing Offset (0x607C,0). By default, the Homing Offset is set to zero during boot-up. While the Homing Offset is writable, it is not saved across power cycles.

In this mode, the status word (0x6041) contains bits (13,12,10) indicating the homing status .

Bit 13	Bit 12	Bit 10	Definition
0	0	0	Homing procedure is in progress
0	0	1	Homing procedure is interrupted or not started
0	1	0	Homing is attained, but target is not reached
0	1	1	Homing procedure is completed successfully
1	0	0	Homing error occurred, velocity is not 0
1	0	1	Homing error occurred, velocity is 0
1	1	X	Reserved

Table 17: Homing Mode Configuration Parameters

Index	SubIdx	Type	Access	Description
0x607C	0	I32	RW	Homing offset (cts)
0x6098	0	I08	RW	Homing method

Faults and Warnings (EMCY)

Error bitfield (0x1001,0):

Bit	Meaning
0	Generic error. Plus, if any higher bit is set, this bit is also set.
1	Motor current error
2	Motor voltage error
3	Temperature error
4	Communication error
5	Unused
6	Unused
7	Motion control error

Manufacturer Status Register (0x1002,0)

This object contains more detailed information about the error(s) that occurred. See the table below.

Last error (0x603F,0) & Error history (0x1003,n)

The last emitted (non-zero) EMCY code is stored in 0x603F,0 (U16) and in the lower 16 bits of 0x1003,1 (U32). Any previous code in 0x1003,1 is pushed down the stack into 0x1003,2, and so on. Any previous code in 0x1003,8 is lost.

Table of Faults and Warnings

Error Description	EMCY Code	OD Warn	OD Fault	Type	Units	0x1001 bit	0x1002 bit	States		Notes
Watchdog reset	0x6010		x	U08	Boolean	0	0	All		Emitted after boot-up message if reset was due to watchdog
Parameter error	0x6320		x	U08	Boolean	0	1	All		Checks for out-of-range and valid configuration hash
Encoder feedback error	0x7320		x	U08	Boolean	0	2	All		Invalid encoder feedback. Set to 1 to generate EMCY message & fault when error occurs.
Encoder magnet distance	0x7321		0x2403,0	U08	Boolean	0	3	All		Set to 1 to generate EMCY message & fault when error occurs
Current limit active	0x2310	0x2111,0		U16	0.01 A	1	4	OpEn		User Continuous Current. Also requires 0x2110,0 U16 User Peak Current (0.01 A) and 0x2112,0 U16 User Peak Current Time (ms)
Short-circuit detected	0x2320		x	U08	Boolean	1	5	OpEn		Peak Current exceeded. Set to 1 to generate EMCY message & fault when error occurs.
Bus over voltage	0x3210		0x2384,6	U16	0.1 V	2	6	All		Amplifier Max Voltage
Bus under voltage	0x3220		0x2384,7	U16	0.1 V	2	7	All		Amplifier Min Voltage
Amplifier over temperature	0x4210		0x2384,9	U08	Dec C	3	8	OpEn		Amplifier Max Temperature
Motor over temperature	0x4310		0x220A,0	U08	Deg C	3	9	OpEn		Motor Max Temperature
CAN overrun (frames lost)	0x8110		0x2404,0	U08	Boolean	4	10	All		Set to 1 to generate EMCY message & fault
SYNC loss fault timeout	0x8180		0x2406,0	U16	ms	4	11	All		The drive will fault if it does not hear periodic SYNCs
Recovered from bus-off	0x8140		0x2405,0	U08	Boolean	4	12	All		Set to 1 to generate EMCY message & fault
Motor phasing error	0x7122		x	U08	Boolean	7	13	OpEn		Set to 1 to generate EMCY message & fault
Positive limit switch	0x7380		x	U08	Boolean	7	14	OpEn		Set to 1 to generate EMCY message & fault. Ignored in Idle/Homing modes.
Negative limit switch	0x7381		x	U08	Boolean	7	15	OpEn		Set to 1 to generate EMCY message & fault. Ignored in Idle/Homing modes.
Homing switch	0x7382		x	U08	Boolean	7	16	OpEn		Set to 1 to generate EMCY message & fault. Ignored in Idle/Homing modes.
Positive soft stop	0x7383	x		U08	Boolean	7	17	OpEn		Set to 1 to generate EMCY message
Negative soft stop	0x7384	x		U08	Boolean	7	18	OpEn		Set to 1 to generate EMCY message
Position wrapped	0x73A0		x	U08	Boolean	7	19	All		Set to 1 to generate EMCY message & fault

Velocity tracking warning	0x8411	0x606D,0		U16	cts/sec	7	20	OpEn		Velocity Tracking Warning Window. Also requires 0x606E,0 U16 Velocity Tracking Warning Timeout (ms)
Velocity tracking fault	0x8414		0x2104,0	U16	cts/sec	7	21	OpEn		Velocity Tracking Fault Window. Also requires 0x2105,0 U16 Velocity Tracking Fault Timeout (ms)
Velocity limit active	0x8418	0x6046,1		U32	cts/sec	7	22	OpEn		Max Velocity Limit. Use 0x6046,2 U32 for Min Velocity Limit (same EMCY message)
Acceleration limit active	0x8480	0x60C5,0		U32	cts/sec/sec	7	23	OpEn		Acceleration Limit. Use 0x60C6,0 U32 for Deceleration Limit (same EMCY message)
Position tracking warning	0x8611	0x6065,0		U32	cts	7	24	OpEn		Position Tracking Warning Window. Also requires 0x6066,0 U16 Position Tracking Warning Timeout (ms)
Homing error	0x8613		x	U08	Boolean	7	25	OpEn		Failed to achieve minimum homing range. Set to 1 to generate EMCY message & fault when error occurs.
Position tracking fault	0x8614		0x2120,0	U32	cts	7	26	OpEn		Position Tracking Fault Window. Also requires 0x2121,0 U16 Position Tracking Fault Timeout (ms)

Important Notes About Faults

Only the **BOLD** errors in the table above have been implemented.

Warnings are auto-reset when the condition clears.

Faults are latched, the amplifier transitions to fault state, and motor voltage is disabled.

Transitioning from Fault to Switch On Disabled clears all latched faults, but if the fault condition still exists, it will re-trigger the fault.

EMCY message format:

- MsgID = 0x80| NodeID
- DLC = 8
- Payload = [CodeLow, CodeHigh, 0x1001, 0, 0, 0, ResetCodeLow, ResetCodeHigh]

When an error condition is cleared, an EMCY message is emitted with code = 0x0000, an updated 0x1001, and the code of the corresponding cleared error.

To disable a warning/fault, set its value to zero.

Detailed Fault Descriptions

Fault	Description
Encoder magnet distance	If enabled (set to 1), and if the magnetic field generated by encoder magnet is too weak or too strong for the encoder sensor, a fault will occur. This may be due to improper spacing between the encoder magnet and the sensor, or the magnet may have lost its field strength due to excessive heat, or the magnet may be the wrong type.
Current limit active	When the i2t power limit is active, this warning will be emitted. This warning will be reset automatically when the i2t power limit becomes inactive.
Bus over voltage	If enabled (set > 0), and if the bus voltage exceeds the amplifier max voltage for more than 450 ms, a fault will occur.
Bus under voltage	If enabled (set > 0), and if the bus voltage falls below the amplifier min voltage for more than 450 ms, a fault will occur.
Amplifier over temperature	If enabled (set > 0), and the 18 Hz low-pass filtered temperature exceeds the max temperature, a fault will occur.
CAN overrun (frames lost)	If enabled (set to 1), and the CAN input buffer of the actuator is overrun, a fault will occur. The actuator will automatically attempt to re-initialize its CAN subsystem to re-enable the receipt of CAN messages, but one or more CAN message may be lost.
SYNC loss fault timeout	If enabled (set > 0), and the time elapsed since the last SYNC message exceeds the timeout, a fault will occur.
Recovered from bus-off	If enabled (set to 1), and the actuator recovers from a bus-off condition, a fault will occur. The actuator will automatically attempt to re-initialize its CAN subsystem to re-enable the receipt of CAN messages, but one or more CAN message may be lost.
Velocity tracking warning	If enabled (set > 0), and the velocity tracking error exceeds the warning threshold for longer than the warning timeout, this warning will be emitted. This warning will be reset automatically when the velocity tracking error falls below the warning threshold.
Velocity tracking fault	If enabled (set > 0), and the velocity tracking error exceeds the fault threshold for longer than the fault timeout, a fault will occur.
Velocity limit active	If enabled (set either min or max limit > 0), and the actuator velocity does not fall between the min/max limits, this warning will be emitted. This warning will be reset automatically when the actuator velocity is between the min/max limits.
Position tracking warning	If enabled (set > 0), and the position tracking error exceeds the warning threshold for longer than the warning timeout, this warning will be emitted. This warning will be reset automatically when the position tracking error falls below the warning threshold.
Position tracking fault	If enabled (set > 0), and the position tracking error exceeds the fault threshold for longer than the fault timeout, a fault will occur.

Appendix

I^2T Power Limiting

The puck uses an I^2t algorithm to ensure that the integral of the power dissipated by the motor in the form of thermal energy does not exceed its thermal limits. The algorithm uses three parameters in its thermal model:

1) Continuous Current – this nominal current that can flow through the motor is determined by the power it can dissipate continuously without exceeding its thermal limits.
2) Peak Current – a transient current above the Continuous Current level that can be tolerated.
3) Peak Current Time – the amount of time the system can tolerate operating at the Peak Current.

It is important to note that these parameters are heavily dependent on the thermal conductivity and thermal capacity of the frame in which the actuator is mounted and the surrounding air temperature.

The excess energy limit of the system is calculated by:

$$Energy_{Limit} = Peak_{Current} * Peak_{Current_{Time}}$$

If the commanded current exceeds the Peak Current parameter, it is limited at Peak Current. The controller keeps a running sum of the system's excess energy by:

$$Energy = Energy + (Command^2 - Continuous^2) * dt$$

Whenever this Energy exceeds the Energy_Limit, the commanded current is limited to the Continuous Current parameter. This results in the following behavior:

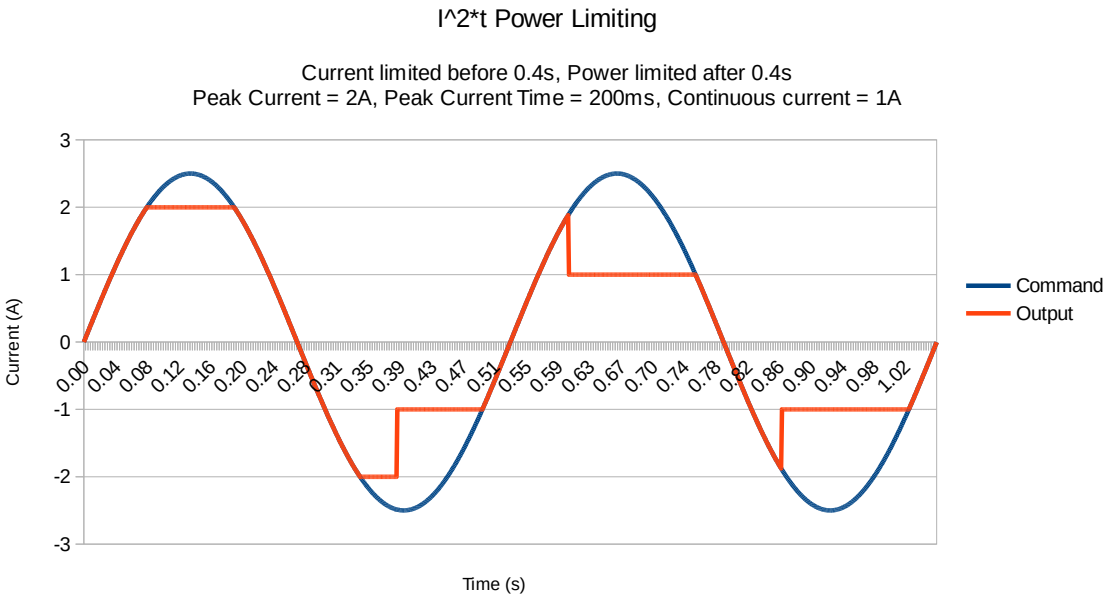


Figure 11: I2T Power Limiting

CAN Physical Layer

Bus Topology and Termination

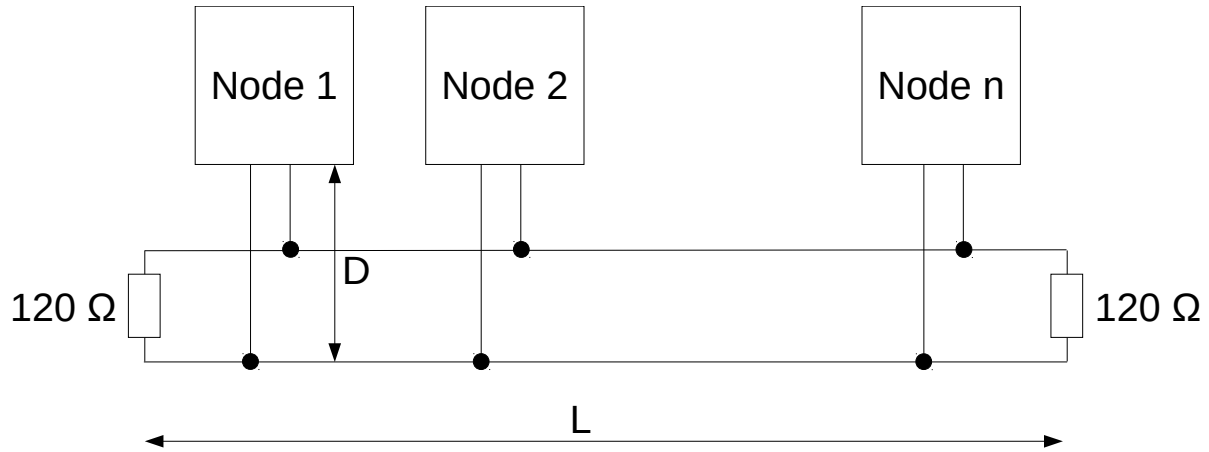


Figure 12: CAN Bus Topology and Termination

The CAN bus must be terminated at each end with a 120-Ohm resistor to prevent signal reflections.

The maximum bus length (L) for a 1 Mbps bus is 40 meters.

The maximum drop length (D) for a 1 Mbps bus is 30 cm.

Nominal Bus Voltage Levels

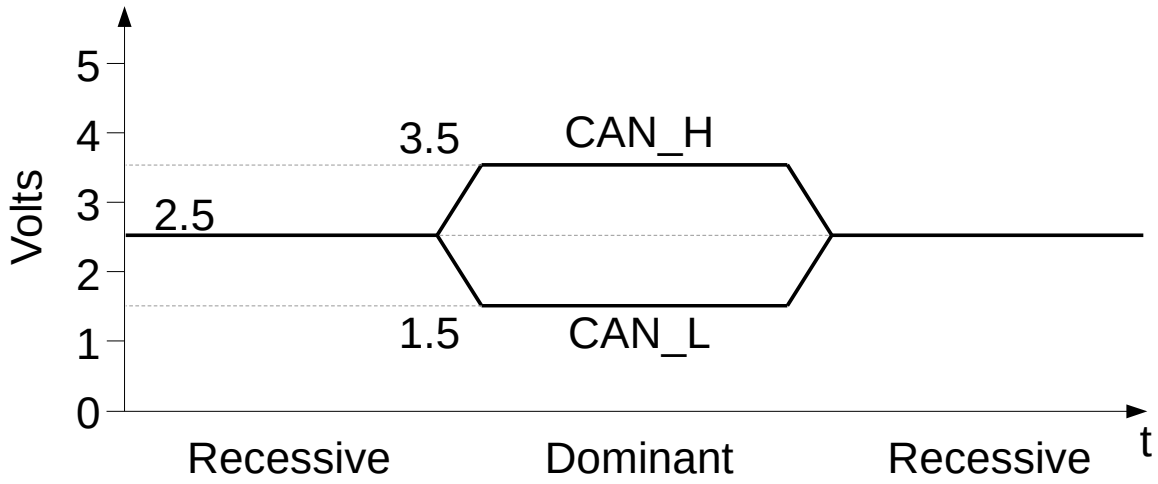


Figure 13: Nominal CAN Bus Voltage Levels

Bit Rates vs. Line Lengths

Use the following recommended settings to configure the timing parameters of your CAN bus.

Rate (kbps)	Max L (m)	Bit Time (us)	TQ	Q (ns)	Sample Point (tq)
1000	40	1	8	125	6
800	50	1.25	10	125	8
500	100	2	16	125	14
250	250	4	16	250	14
125	500	8	16	500	14

TQ = Time Quanta

Q = Quantum duration

SJW = Synchronization Jump Width: 1 TQ

Frame Format

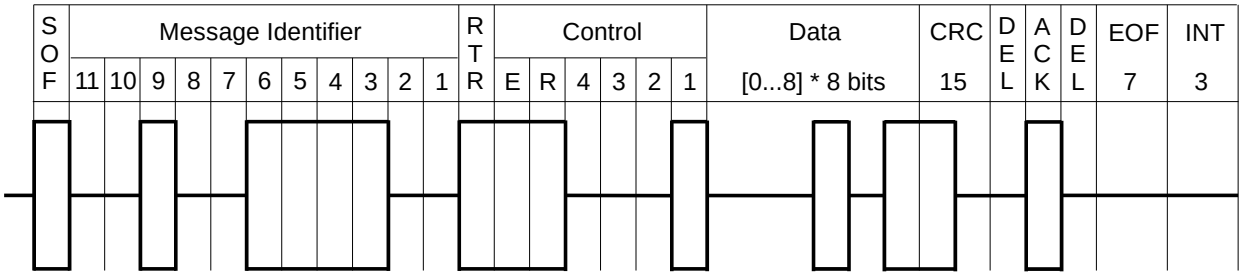


Figure 14: Frame Format

SOF = Start-of- Frame bit, dominant

Message Identifier = 11 bits (CAN 2.0A)

RTR = Remote Transmission Request bit: dominant = Data Frame, recessive = Request Frame

Control Field

E = Extended Frame: dominant = 11 bit identifier, recessive = 29-bit identifier

R = Reserved, always dominant

[4 3 2 1] = Data Length Code (DLC), specifies number of bytes in Data Field [0...8]

Data Field = 0 to 8 bytes of data

CRC = Cyclic Redundancy Check: 15-bit checksum of SOF, Message ID, RTR, Control, and Data fields

DEL = Acknowledgment Delimiter, always recessive

ACK = Acknowledgment bit, transmitted recessively. Any node which receives this message correctly will assert a dominant value here to indicate a successful transmission to the sending node.

EOF = End-of-Frame, a series of 7 recessive bits

INT = Intermission, a minimum of 3 bit times between successive frames

NOTE about data interpretation:

To interpret data bits, Dominant = 0, Recessive = 1 (somewhat counter-intuitive)

NOTE about Bit-Stuffing:

Transmissions are bit-stuffed 5:1 to aid in receiver synchronization. When the input stream contains 5 consecutive bits of the same value, a complementary bit is inserted (and considered part of the input stream, in case the following 4 bits match the stuffed bit). This applies to all bits from SOF up to, but not including, the CRC's DELimiter bit. To recover the original bit stream, the receiver always strips out the bit following a sequence of five bits of equal value, up to the CRC's DELimiter.

CANopen Protocol

Message Types

- **Network Management (NMT)** - Includes commands to manipulate the device communication state machine between Stopped, Pre-Operational, and Operational, as well as commands to reset the device's communications peripheral or reboot the device.
- **Heartbeat** - When a CANopen slave device boots up, it alerts the CANopen master about its presence with a heartbeat message. Devices can also be configured to send these heartbeat messages periodically during operation so the CANopen master can ensure all devices are operational.
- **Emergency** - These high-priority messages alert the CANopen nodes about any critical system errors.
- **Timestamp** - A CANopen device can be configured to emit periodic Timestamp messages to ensure all nodes are synchronized with a reference time (and date).
- **Service Data Object (SDO)** - Read or write any single value in the OD.
- **Process Data Object (PDO)** - Devices can be configured to pack up to 8 bytes of OD data into a single CAN frame. This helps conserve CAN bandwidth when sending a variety of control data or receiving a variety of feedback data during a control loop. CANopen slave devices listen for Receive Process Data Objects (RPDOs) sent from the CANopen master, and they send Transmit Process Data Objects (TPDOs) back to the CANopen master.
- **SYNC** - The CANopen master sends a CAN frame with a message ID of 0x80 and no payload. This SYNC message prompts all CANopen slaves to 1) apply the most recently received RPDO control setpoint(s) and 2) respond with their latest TPDO process feedback data.

Message Formats

- **Network Management (NMT)**
  - MsgID = 0x000 (heard by all nodes)
  - DLC = 2
  - Payload = [Commanded_State] [NodeID]
    - Commanded_State
      - 0x01 = Operational
      - 0x02 = Stopped

                - 0x80 = Pre-operational
                - 0x81 = Reset Node (Performs a full power-on reset)
                - 0x82 = Reset Communications (Resets the CAN device then enters Boot-up state)
            - NodeID
                - 0 = All nodes
                - 0x01 – 0x7F = Single node specified by NodeID
    - Note that the commanded state codes in an NMT message do not match the state codes reported in a Heartbeat message- this is in accordance with the CiA-301 specification.
- **Heartbeat**
    - MsgID = 0x700 | NodeID
    - DLC = 1
    - Payload = [State]
        - State
            - 0x00 = Boot-up
            - 0x04 = Stopped
            - 0x05 = Operational
            - 0x7F = Pre-operational
- **SYNC**
    - MsgID = 0x080
    - DLC = 0
- **Emergency (EMCY)**
    - MsgID = 0x080 | NodeID
    - DLC = 8
    - Payload = [ErrLow] [ErrHigh] [ErrRegister] [Mfg-Specific]…
        - [ErrLow] [ErrHigh] = 16-bit CANopen error code
        - [ErrRegister] = copy of OD 1001,00
        - [Mfg-Specific] = 0-5 bytes of error data (optional)
- **Timestamp**
    - MsgID = 0x100
    - DLC = 4
    - Payload = 32-bit timestamp value (LSB)
- **SDO Write (Expedited)** – Write this data to the Object Dictionary (OD)
    - MsgID = 0x600 | NodeID
    - DLC = 8
    - Payload = [0010 nn e s] [lowIdx] [highIdx] [subIdx] [data]…
        - 0010 = SDO Write
        - nn = number of bytes w/o data (iff s == 1)
        - e = Expedited
        - s = size indicated in nn
        - [lowIdx] [highIdx] = 16-bit OD index
        - [subIdx] = OD entry sub-index
        - [data] = 1-4 bytes of data (little-endian)
- **SDO Write Ack** – Data was written to the OD

    - MsgID = 0x580 | NodeID
    - DLC = 8
    - Payload = [0110 0000] [lowIdx] [highIdx] [subIdx] 0x00 0x00 0x00 0x00
- **SDO Read** – Read data from OD
    - MsgID = 0x600 | NodeID
    - DLC = 8
    - Payload = [0100 0000] [lowIdx] [highIdx] [subIdx] 0x00 0x00 0x00 0x00
- **SDO Read Response (Expedited)** – Here is the data you requested
    - MsgID = 0x580 | NodeID
    - DLC = 8
    - Payload = [0100 nn e s] [lowIdx] [highIdx] [subIdx] [data]…
        - 0100 = SDO Read
        - nn = number of bytes w/o data (iff s == 1)
        - e = Expedited
        - s = size indicated in nn
        - [lowIdx] [highIdx] = 16-bit OD index
        - [subIdx] = OD entry sub-index
        - [data] = 1-4 bytes of data (little-endian)
- **Receive PDO** – process data object to be received by the device
    - MsgID = 0x200 | NodeID, 0x300 | NodeID, etc.
    - DLC = n (byte count)
    - Payload = [data]…
        - [data] = little-endian data
- **Transmit PDO** – process data object to be transmitted by the device
    - MsgID = 0x180 | NodeID, 0x280 | NodeID, etc.
    - DLC = n (byte count)
    - Payload = [data]…
        - [data] = little-endian data

Document Change History

Revision	Date	Description	Originator
AA	2016-01-18	Original Issue	B. Zenowich
AB	2016-01-18	Fixed Scaled Feedback OD Index	B. Zenowich
AC	2016-01-19	Added more detail to Example Operation, added change history.	B. Zenowich
AD	2016-01-19	Restored change history after accidental deletion.	B. Zenowich
AE	2016-02-10	Added documentation for the “set mode” entry in the OD (index = 0x6060).	C. Woodall
AF	2016-02-24	Added documentation for the “Error” entry in the OD (index = 0x1001) Added example of how to check for an over-temperature fault.	C. Woodall
AG	2016-03-01	Added information on CANopen OD entry 0x3F00 for accessing BarrettCAN properties Added examples for using property 0x3F00	C. Woodall
AH	2016-04-13	Added Appendix B on supporting multi-field TPDOs	C. Woodall
AI	2016-04-22	Added safe position information.	C. Woodall
AJ	2016-05-31	Added Trapezoidal Trajectory Mode. Added over current error to Error register.	C. Woodall
AK	2016-06-21	Updated some fields which were marked with the wrong Read-Write permissions	C. Woodall
AL	2017-03-24	Added Target Velocity, Target Position, and more documentation on the BarrettCAN Link including an example of writing to Auto-Homing and Saving the Change To EEPROM	C. Woodall

AM	2017-07-19	Added LOAD, control word, status word, quick stop option code, raw position command, raw position feedback, actual current feedback, bitfield of supported drive modes. Updated 0x6060 modes to match DSP-402 standard. Corrected velocity target units. Added a Notes section, Finite State Automaton (FSA) flow chart, and a “How it was tested” section. Reorganized Appendix B: Examples	B. Zenowich
AN	2017-07-20	Added Motor Is Homed flag (0x2402,0)	B. Zenowich
AO	2017-09-21	Added new Velocity, CSV, CSP modes. Added Faults/Warnings/EMCY codes.	B. Zenowich
AP	2017-09-22	Corrected Control Mode data type	B. Zenowich
AQ	2017-09-27	Added SYNC loss fault timeout	B. Zenowich
AR	2017-11-03	Added velocity feedforward parameters	C. Woodall
AS	2017-12-08	Updated control diagrams to match firmware r147. Current feedback range changed from rated (peak) current to continuous current. Removed Appendix B- Example CAN messages.	B. Zenowich C. Woodall
AT	2018-01-09	Added detailed fault descriptions, a note about temperature faults, and Appendix B- Algorithm Descriptions. Matches firmware r147.	B. Zenowich
AU	2018-11-20	Updated to match rewritten P4 firmware.	B. Zenowich
AV	2019-10-22	Added position control and wxp3 software.	B. Zenowich
AW	2020-02-27	Added notes about CAN frame bit interpretation and bit-stuffing.	B. Zenowich
AX	2024-04-10	Reorganized and updated to match latest P4 firmware	B. Zenowich
AY	2025-03-13	Added velocity filters and homing method	B. Zenowich