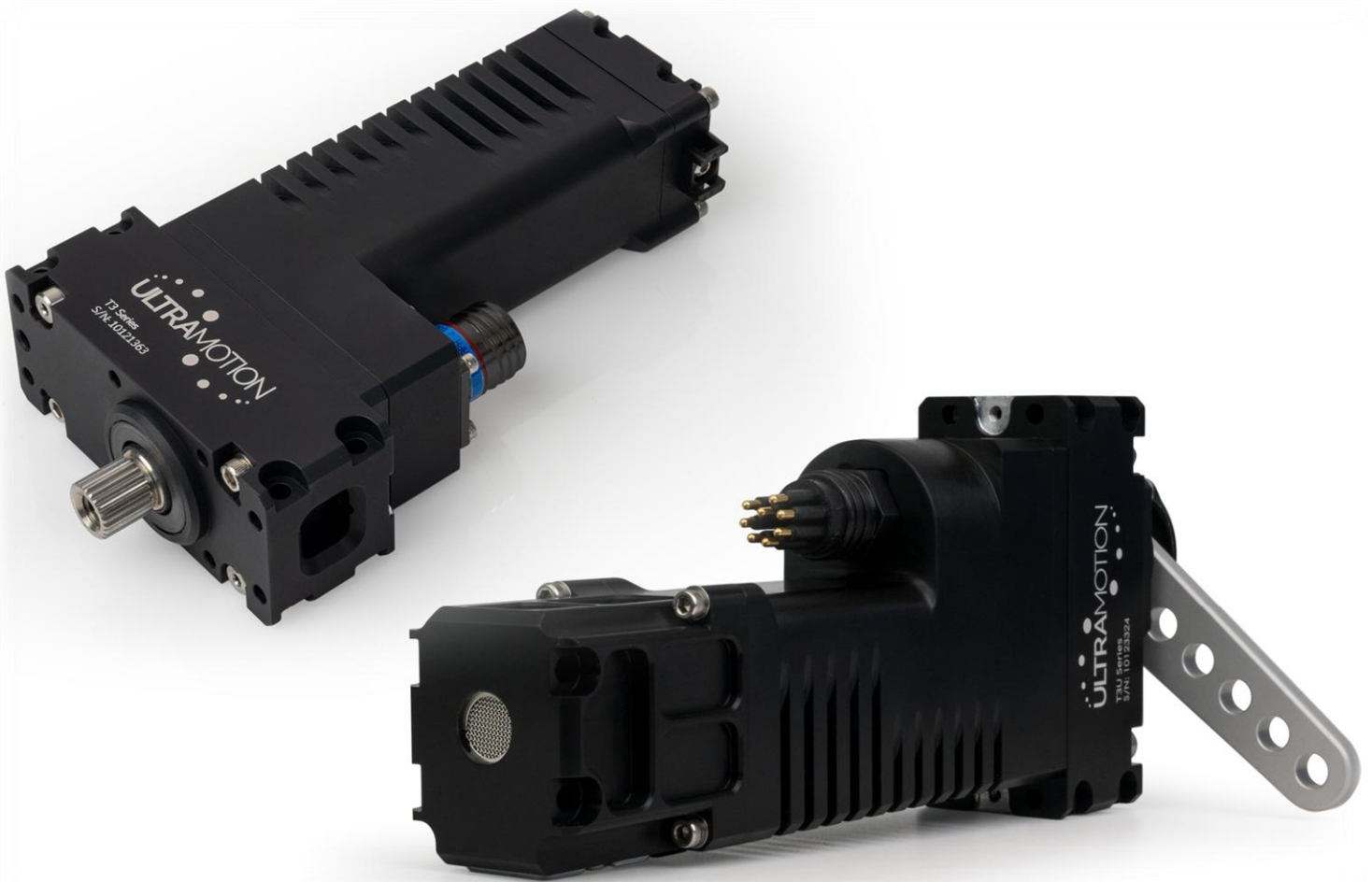




T-Series Rotary Servo Actuator



Important Information When Using Our Products

Please note that Ultra Motion's commercial off-the-shelf (COTS) products are not intended for use in critical applications where failure of the product may cause bodily harm or death. Please consider the following information when designing our products into your system.

Performance

All commercial off-the-shelf products manufactured by Ultra Motion are designed to meet the performance specifications we publish in the product's manual. All life related data is provided as reference only and does not take into account application specific factors that can have significant impacts to the overall life of the product. Application specific factors can include: design loads, transient loads (shock, vibration, inertia), speed, environmental stresses (temperature, contamination), etc. Due to the fact that application specific factors can greatly affect the product's life, it is not possible to provide a generalized Mean Time Before Failure (MTBF). It is the customer's responsibility to determine the suitability of the product for their particular application.

Software

T-Series actuators have a built-in controller and are shipped with the latest release of controller firmware. The controller firmware is changed from time to time to add features, fix undesired behavior, or change how the controller operates. We do our best to thoroughly test each firmware release, but we do not guarantee that the controller firmware will be free of software problems that may cause undesirable or unpredictable behavior. It is extremely important that you test the actuator for your application and do not use the actuator in applications where the failure or unpredictable operation of the actuator may result in injury or death.

Change Control

Commercial off the shelf products are subject to changes that do not affect form, fit, or function. These changes can include the use of different PCB components, internal part revisions, suppliers, firmware, coloration, etc. Ultra Motion has the ability to track and manufacture version locked designs if your project has specific change control requirements. In a version locked design, the customer will be notified before any changes are made to their product.

Quality Control

Ultra Motion actuators are manufactured under our internal quality management system. 100% of the product we manufacture goes through a complete performance QC inspection before leaving our facility. Documented results of QC records are available to all customers.

Safety Information

IMPORTANT: Read this manual before installing and operating the Ultra Motion T-Series Servos. Failure to read this section can result in personal harm or damage to the product.

Safety Disclaimer

The T-Series is intended to be a subcomponent of a larger piece of machinery or automated system. This section is not intended to provide the safety guidelines for the entire machine or system that the servo is installed into. It is the responsibility of the purchaser or system designer to assess the risks and safety requirements of the end application they are designing.

Safety Warnings

Once powered, the servo is capable of rapid motion and can produce large amounts of torque. Always ensure that safe clearances from people and equipment are maintained before applying power.

While the servo operates on low voltage (10 to 36 VDC recommended), you must still use caution when handling and working around the actuator to avoid electrical shock.

The motor of the actuator can become very hot, especially at high current draws. Take adequate time to cool before handling, and provide adequate ventilation for cooling of this device.

Safety Notifications



As you read through the manual, you will notice certain safety notifications that indicate other important safety related information.

Content

IMPORTANT INFORMATION WHEN USING OUR PRODUCTS	2
CONTENT	3
LIST OF TABLES	4
LIST OF FIGURES	4
REVISION HISTORY	4
T-SERIES OVERVIEW	5
<i>Typical applications:</i>	5
<i>Features:</i>	6
<i>Product Number Structure</i>	7
<i>T3 Specifications:</i>	10
<i>T3 Series Performance and Power Requirement Graphs</i>	11
<i>T3 Series Electrical Interfaces</i>	12
<i>T3 Dimensions</i>	14
<i>Temperature Derating</i>	15
T-SERIES CONTROLLER OVERVIEW	16
<i>Configuration Overview</i>	16
<i>T-Series Control Overview</i>	17
T-SERIES CAN 2.0B PROTOCOL	23
<i>T-Series CAN Behavior/Configuration</i>	23
<i>T-Series CAN Telemetry</i>	24
<i>T-Series Event Based Asynchronous CAN Messages</i>	27
T-SERIES BINARY SERIAL CONTROL PROTOCOL	29
<i>T-Series BSC Behavior/Configuration</i>	29
T-SERIES RC PWM MODE	33
T-SERIES RS-485 SERIAL COMMAND LINE INTERFACE	34
<i>T-Series Serial CLI Command Details</i>	36
<i>T-Series Configuration Variables</i>	41
GENERAL ACTUATOR DESIGN GUIDELINES	54
T-SERIES FIRMWARE UPDATE	55
UNIT CONVERSIONS	56
<i>Definitions</i>	56
<i>Equations</i>	56
OPTIONS AND ACCESSORIES	57
<i>Control Arm</i>	57
<i>Rotational Hardstop</i>	58
<i>Ultra Motion Power Supplies</i>	58
<i>CBL-T3-DEV Cable Set</i>	58
<i>CBL-T3U-1M Cable Set</i>	58
<i>CBL-T3U-5M Cable Set</i>	58
<i>Grounding Strap</i>	59
CONTACT INFORMATION	60

List of Tables

Table 1: Part number option descriptions	8
Table 2: T3 Specifications.....	10
Table 3: Pin numbers, functionality, and standard cable wire colors for T3M series actuators.	12
Table 4: T3U Controller Code 2 Pinout	13
Table 5: T3U Controller Code 3 Pinout – Opto1	13
Table 6: Command options for T-Series Actuator Control.....	18
Table 7: Control Word.....	18
Table 8: System Errors Register bit descriptions	20
Table 9: System Warnings Register bit descriptions.....	21
Table 10: Status Register bit descriptions.....	22
Table 11: Telemetry options for the CAN 2.0B protocol used in txNData and “RR” command	26
Table 12: Asynchronous Message Format	27
Table 13: Warning event codes, causes, and data fields	28
Table 14: Miscellaneous event codes, causes, and data fields.....	28
Table 15: BSC Command Frame Format	29
Table 16: BSC Command Code List	29
Table 17: BSC Response Frame Format	30
Table 18: BSC Response Frame Error Code List	30
Table 19: Serial CLI Commands List.....	35
Table 20: Configuration variable reference list.....	42
Table 21: Firmware update command line options.....	55
Table 22: Unit conversion definitions	56
Table 23: Unit conversion equations	56

List of Figures

Figure 1: T3M mechanical cutaway	6
Figure 2: T3 torque vs. speed performance graph.....	11
Figure 3: Power Requirements (approximate)	11
Figure 4: D38999/20ZB35PN T3M Series Receptacle.	12
Figure 5: MCBH8M T3U Series Receptacle	13
Figure 6: T3M mechanical interface control drawing.....	14
Figure 7: T3U mechanical interface control drawing.....	14
Figure 8: Derating factor for maximum continuous load CT as a function of temperature.	15
Figure 9: Optically isolated digital input wiring schematic	33

Revision History

Revision	Date	Details
A.01	11/19/2025	Initial Release

T-Series Overview

The T-Series Rotary Servo Actuators are an advanced electromechanical actuator equipped with integrated brushless DC control electronics, CAN 2.0B and RS-485 serial communication protocols, and a single-turn, 12-bit, contactless absolute position sensor. Their design focuses on dynamic operation in the most demanding environments, utilizing durable mechanical elements, top-tier materials, and best-in-class components.

At its core, the T-Series Servo Actuators are powered by a high-power density Brushless DC (BLDC) motor, which is paired with a planetary gearhead to generate high torques in a compact package. This motor and gearhead assembly links to the servo's output via a large spur geartrain for increased reliability.

Unlike the common industry practice of using compound spur gear trains, our unique gearing setup enhances the actuator's capability to handle high transient loads without succumbing to structural failures.

A pressure compensated oil-filled version of the T-Series is available for submerged applications to 6000+ meters.

Typical applications:

- Uncrewed Aerial Vehicles
 - Swashplate and tail-rotor control
 - Fixed wing control surfaces
 - Utility Actuation
- Uncrewed Surface Vehicles
 - Rudder control
 - Accessory deployment
- Subsea vehicle control
- Subsea valves and tools
- Applications requiring dynamic and robust servo actuation in harsh environments

Features:

- Contactless single turn position feedback of the output shaft, 12-bit.
- Built-in brushless DC control electronics
 - Control options including CAN 2.0B, RS-485 serial, RC PWM (optically isolated).
 - Feedback includes absolute position, motor phase current, temperature, bus voltage, humidity, and actuator health
 - 10 VDC to 36 VDC max operating voltage range (65 VDC absolute max)
- -40°C to +100°C operating temperature range
- Robust mechanical components for high peak load tolerance and reliability
- High power density with extremely dynamic motion capabilities
- D38999/20ZB35PN or SubConn MCBH8MAS connector standard
- IP67 of fully submersible versions
- Fully conductive hardcoat anodized aluminum housing for EMI/EMC protection
 - 316SS housing options available
- 17-4PH H1100 output shaft
- Continuous rotation output shaft, electronically limited travel

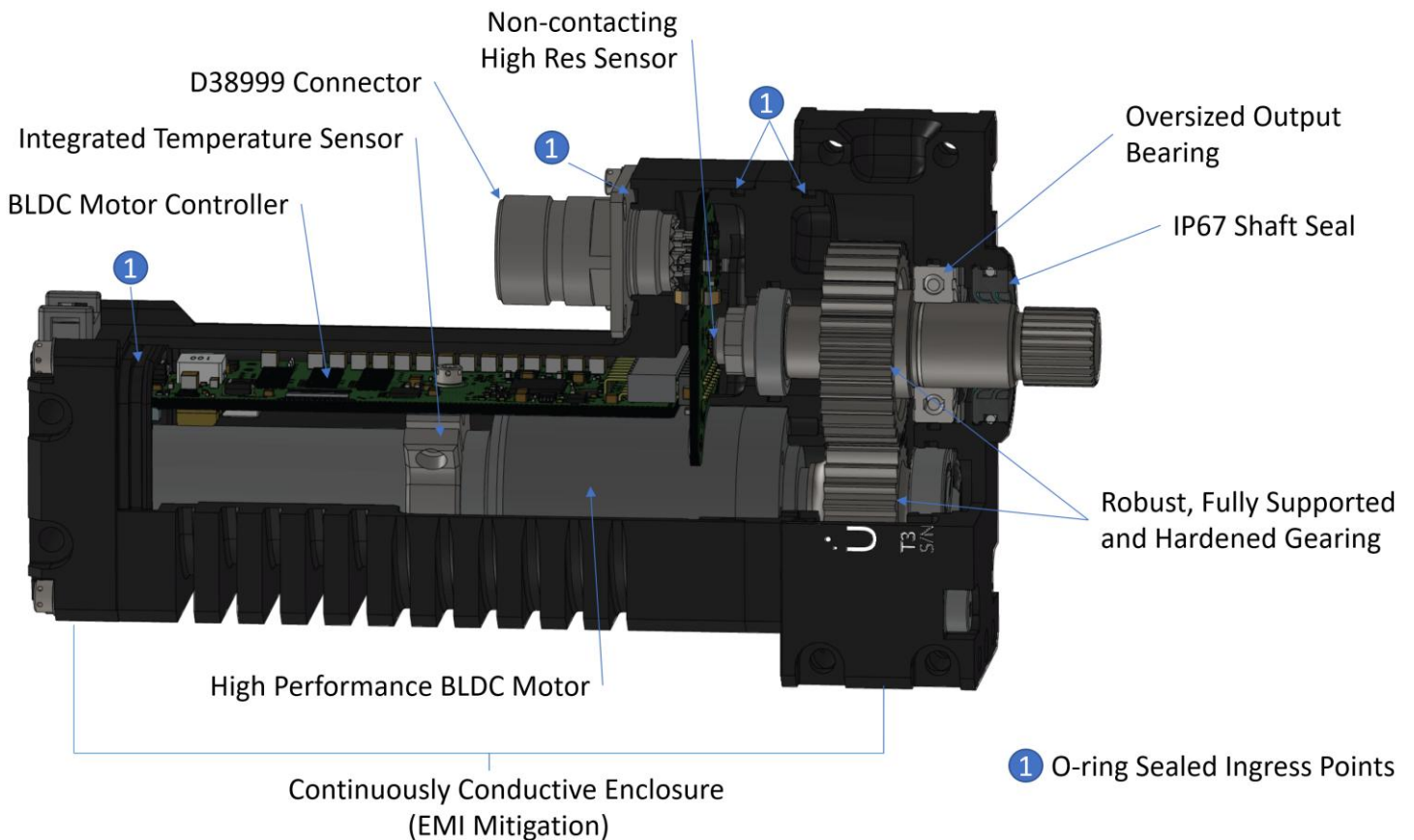


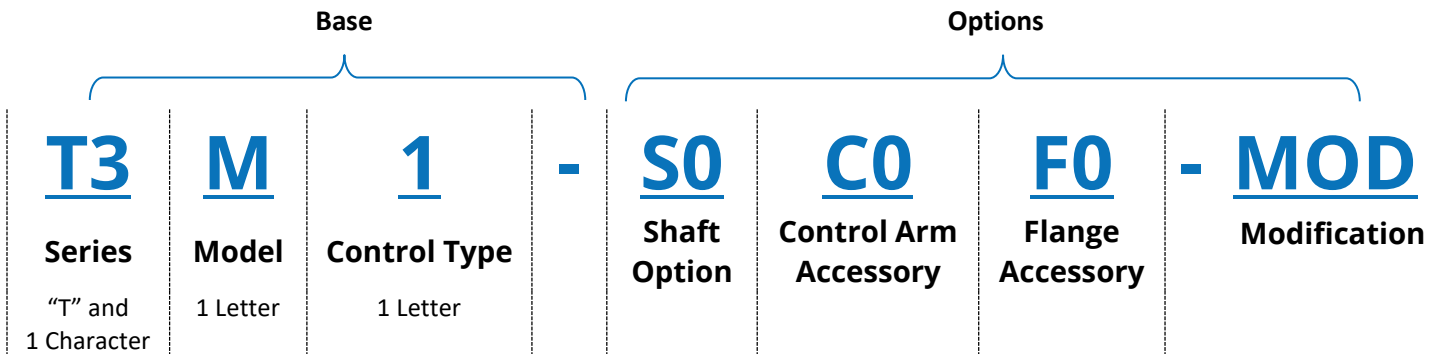
Figure 1: T3M mechanical cutaway

Product Number Structure

Use the following diagram to help determine the individual features called out within the ordering code. Further in this section, a full description of all available options can be found further in this section.

This example ordering code given here describes an T3-Series “Rotary Servo”, a M type model variant, CAN control, and the S0, C0, and F0 options.

Example Ordering Code:



Series

Code	Characteristics
T3	108 in-lbf continuous/172 in-lbf peak static 1.6" thickness Standard, Submersible, and High Torque Model Variations

Models

Code	Characteristics
M	Standard model for most applications
U	Fully Submersible to 6,000+ meters (pressure compensated, oil filled)
X	High Torque Model (in development)

Controller Type

Code	Control	Limitations
1	CAN 2.0B, RS-485 serial, RC PWM	Standard “M” model
2	CAN 2.0B and RS-485 serial	Submersible “U” model
3	RS-485 serial and RC PWM	Submersible “U” model

Output Shaft Options

Code	Description	Limitations
S0	25 Tooth Involute Spline Output	

Control Arm Accessory

Code	Description	Limitations
C0	No control arm accessory	
C1	Single ended 6061-T6 control arm	

Flange Accessory

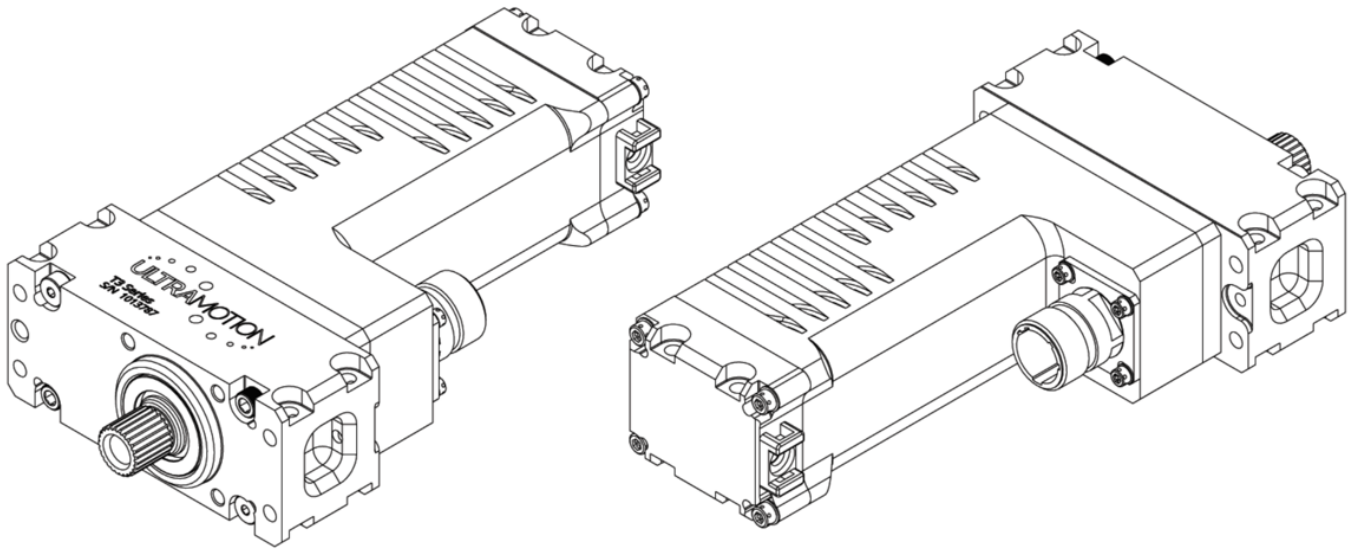
Code	Description	Limitations
F0	No flange accessory	
F1	100° hardstop accessory	

Modification Code

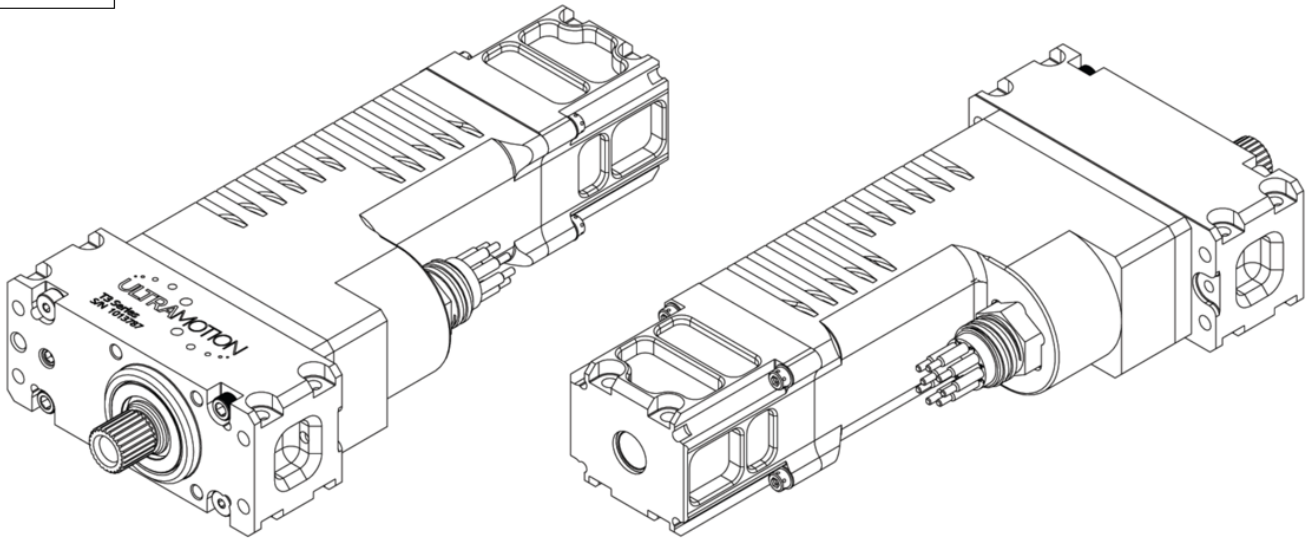
Code	Description	Limitations
N/A	No modifications	
316SS	All 316 stainless steel housing	

Table 1: Part number option descriptions

Model "M"



Model "U"



T3 Specifications:

Operating Voltage	10 to 36 VDC
Absolute Maximum Voltage	65 VDC
Operating Temperature Range	-40°C to +100°C
Mass (T3M)	2.65 lbm
Mass (T3U)	3.10 lb dry / 2.5 lb submerged
Allowable Rotation	360° electronically limited Continuous rotation modes possible
Peak No Load Speed @ 28 VDC	500 °/s
Continuous No Load Speed @ 28 VDC	310 °/s
Continuous Torque	108 in-lbf
Intermittent Torque	132 in-lbf
Peak Static Holding Torque	172 in-lbf
Unpowered Backdrive Torque	10 in-lbf
Nominal Backlash	0.6°
Static Radial Load Rating (0.5" from the front mount face)	232 lbf
Dynamic Radial Load Rating (0.5" from the front mount face)	453 lbf
Axial Load Rating**	47 lbf
Maximum Axial Motion**	0.05"
Communication Protocols	CAN 2.0B, RS-485 Serial, RC PWM
Environmental Sealing	IP67 OR 10,000 PSI ON SUBMERSIBLE MODELS

Table 2: T3 Specifications

**Tensile loads above 16 lbf will start to compress an internal spring and you will get axial motion. The spring rate is 0.025"/16 lbf. Consult engineering about higher loads or reference Axial Loads in the General Actuator Design Guidelines section.

T3 Series Performance and Power Requirement Graphs

PERFORMANCE (Torque vs. Speed)

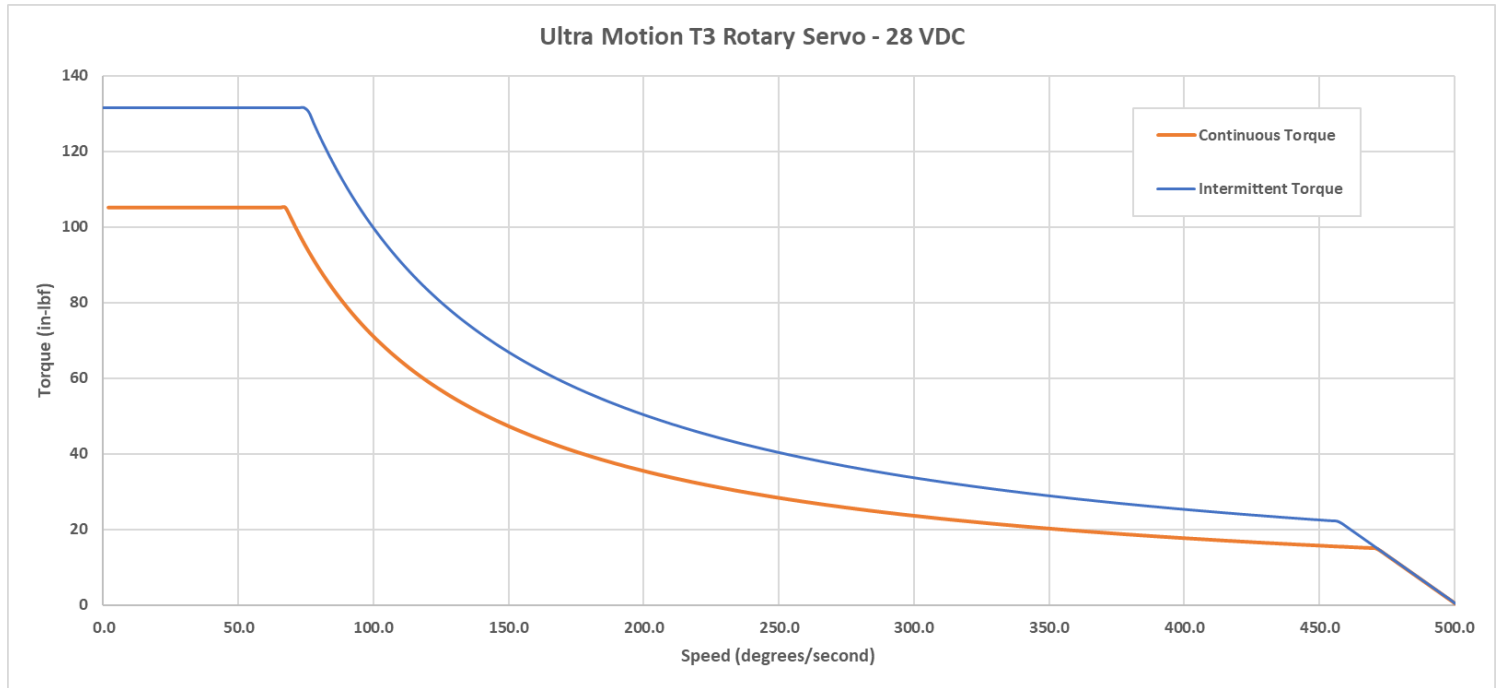


Figure 2: T3 torque vs. speed performance graph

Continuous Torque: 100% duty cycle assuming 25°C ambient and natural convection

Intermittent Torque: 20% duty cycle

POWER

T3 Series Power Consumption Estimate

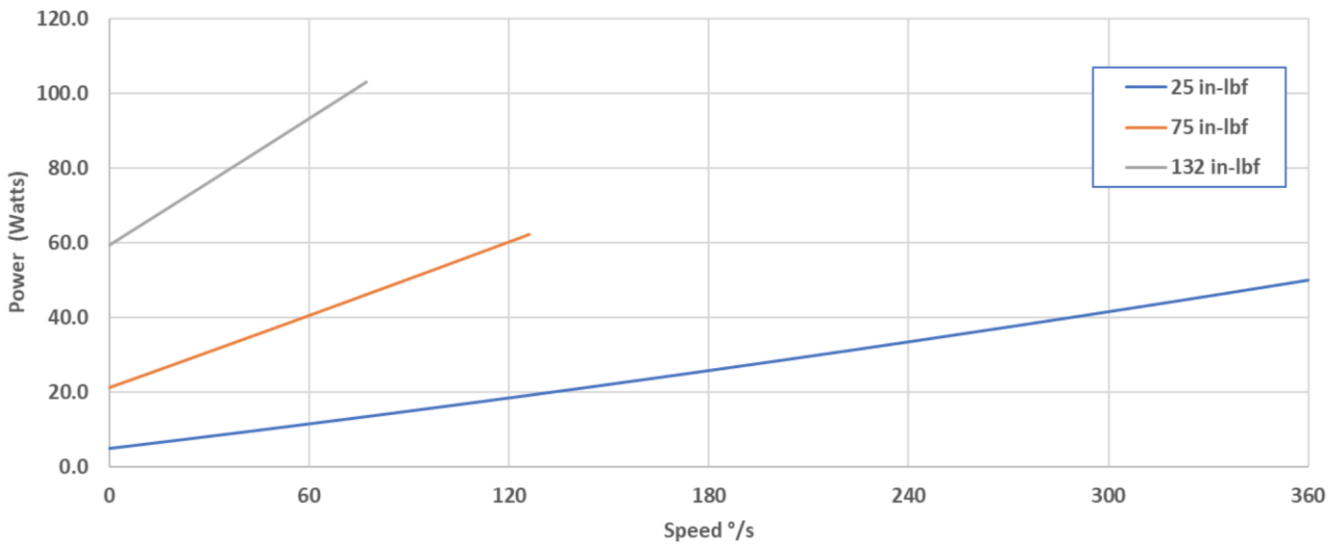


Figure 3: Power Requirements (approximate)

Note: Shown at 1 atmosphere pressure, 22°C ambient. This is for dynamic movement, holding position statically against a load requires significantly less power due to the internal friction and inefficiencies helping hold the position.

T3 Series Electrical Interfaces



WARNING: The electrical interface and pinouts vary between the actuator controller models. Using an improperly wired cable **will** damage to the actuator. Always be aware of your controller and its electrical interface before using the actuator.

T3M Electrical Interface

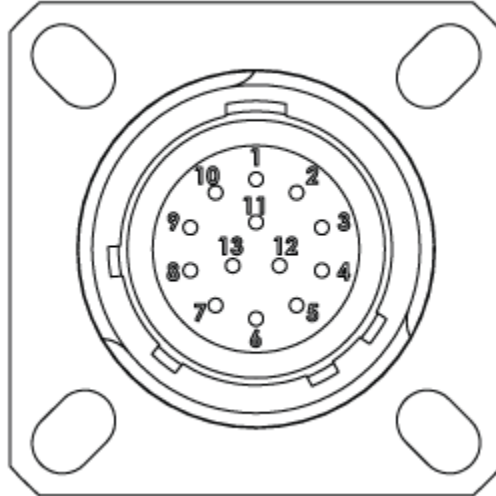


Figure 4: D38999/20ZB35PN T3M Series Receptacle.

	Pin No.	Function	Flying Lead Color (CBL-T3-DEV)
D38999/20ZB35PN	1	CAN_L	BLUE
	2	GND	TAN
	3	GND	SLATE
	4	V+	PINK
	5	V+	VIOLET
	6	GND	ORANGE
	7	RS-485 A	RED
	8	RS-485 B	BLACK
	9	GND	WHITE
	10	CAN_H	GREEN
	11	OPTO COMMON CATHODE	BROWN
	12	OPTO 1 ANODE	YELLOW
	13	OPTO 2 ANODE	N/C
	SHELL	CHASSIS	DRAIN

Table 3: Pin numbers, functionality, and standard cable wire colors for T3M series actuators.

Note that the “SHELL” pin refers to the metal connector casing.



Figure 5: MCBH8M T3U Series Receptacle

	No.	Pin Function	Flying Lead Color (CBL-T3U)
MCBH8M	1	GND	BLACK
	2	GND	WHITE
	3	V+	RED
	4	V+	GREEN
	5	RS485 A	ORANGE
	6	RS485 B	BLUE
	7	CAN_H	WHITE/BLACK
	8	CAN_L	RED/BLACK

Table 4: T3U Controller Code 2 Pinout

	No.	Pin Function	Flying Lead Color (CBL-T3U)
MCBH8M	1	GND	BLACK
	2	GND	WHITE
	3	V+	RED
	4	V+	GREEN
	5	RS-485A	ORANGE
	6	RS-485B	BLUE
	7	OPTO1_ANODE	WHITE/BLACK
	8	OPTO1_COMMON	RED/BLACK

Table 5: T3U Controller Code 3 Pinout – Opto1

T3U Bulkhead Connector Details

The standard T3U SubConn connector must be lubricated with Molykote 44 Medium on every installation. A layer of grease corresponding to a minimum of 1/10 socket depth should be applied to the female connector in dry-mate conditions. A layer of grease corresponding to 1/3 socket depth should be applied to the female connector in wetmate situations. Angular loads on the cable and mated cable assembly must be minimized. Debris and mud on the male connectors and within the female connector sockets must be cleaned thoroughly before mating with liquid soap, hot water, and isopropyl alcohol. The orientation of the T3U receptacles on the housing will be as shown with a tolerance of $\pm 30^\circ$.

T3 Dimensions

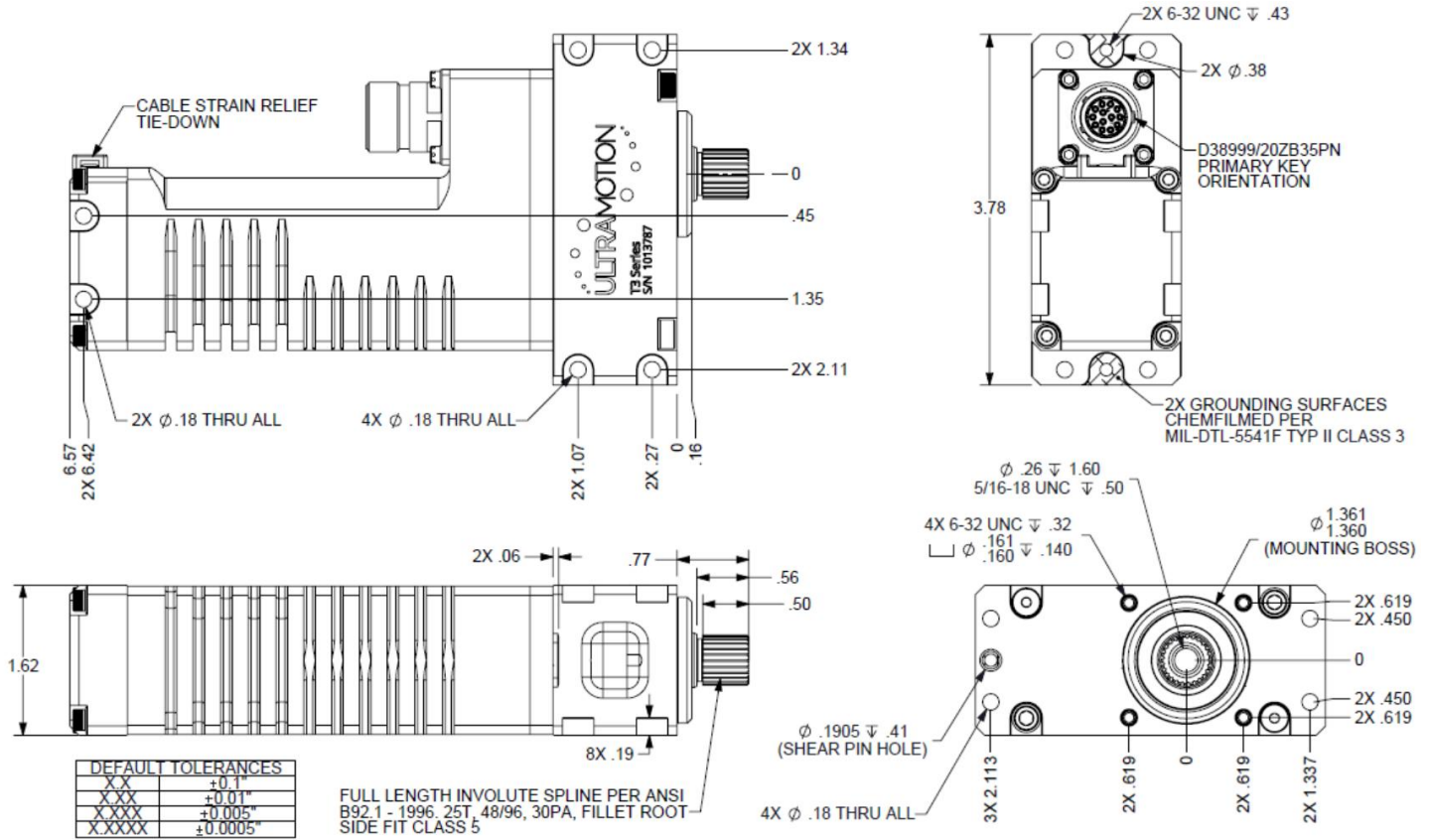


Figure 6: T3M mechanical interface control drawing

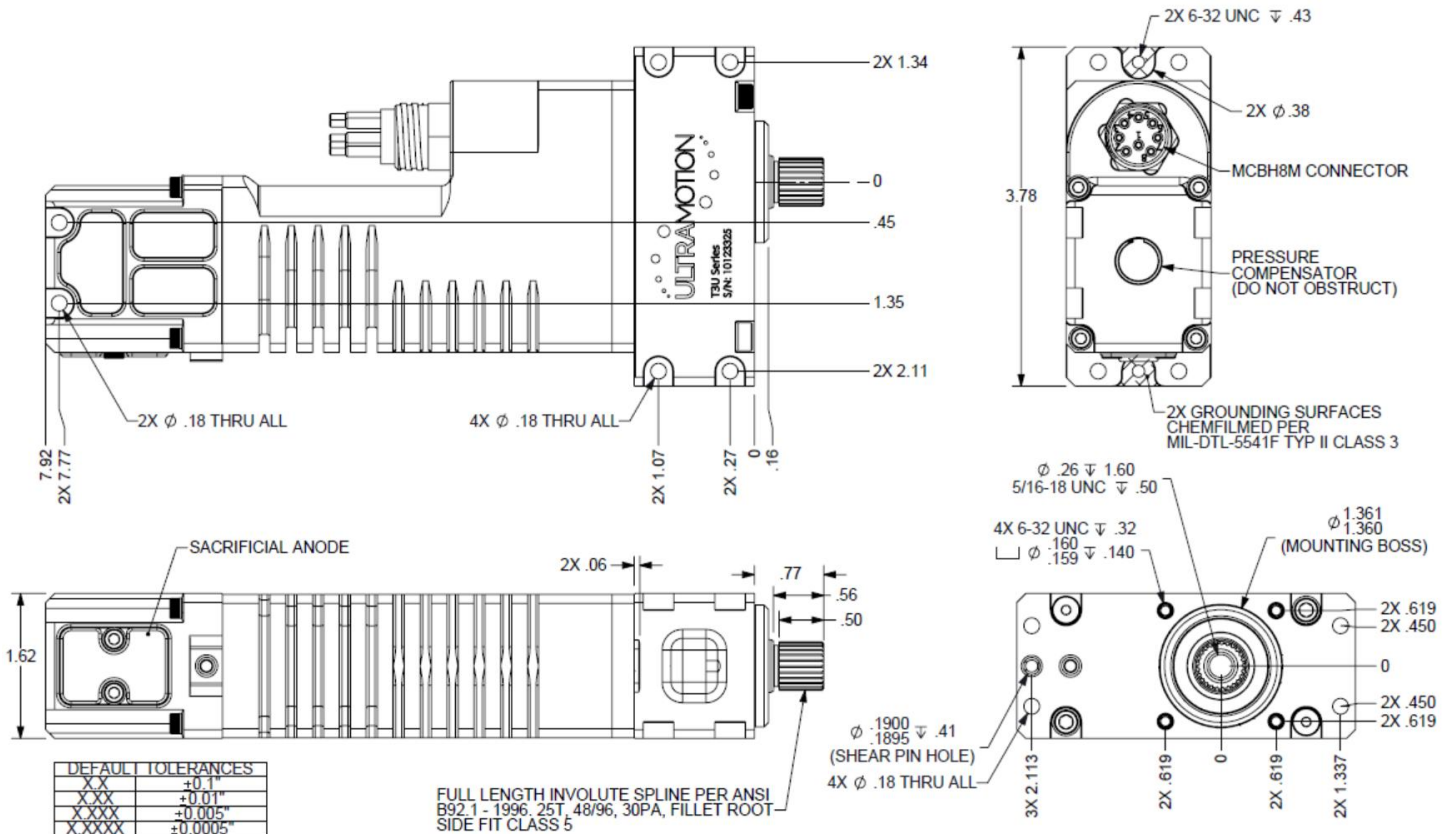


Figure 7: T3U mechanical interface control drawing

Temperature Derating

The maximum continuous load rating of the T-Series is related to the rate of heat generation by the motor and the maximum operating temperature of the actuator components. As such, when environmental temperature increases, the maximum continuous load rating of the actuator must decrease accordingly. A derating factor C_T is applied to the value for the nominal temperature maximum continuous load rating at (F_{nom}) to determine the maximum continuous load rating (F_T) for the actual operational ambient temperature:

$$F_T = F_{nom} \times C_T$$

The derating factor C_T is a function of the nominal temperature at which the maximum continuous load rating actuator is defined ($T_{nom} = 22^\circ\text{C}$ for the T-Series), the maximum permissible winding temperature of the actuator ($T_{max} = 150^\circ\text{C}$), and the ambient temperature of the application to which you are derating ($T_{ambient}$).

$$C_T = \sqrt{\frac{T_{max} - T_{ambient}}{T_{max} - T_{nom}}} = \sqrt{\frac{150^\circ\text{C} - T_{ambient} (^\circ\text{C})}{150^\circ\text{C} - 22^\circ\text{C}}} = \sqrt{\frac{302^\circ\text{F} - T_{ambient} (^\circ\text{F})}{302^\circ\text{F} - 71.6^\circ\text{F}}}$$

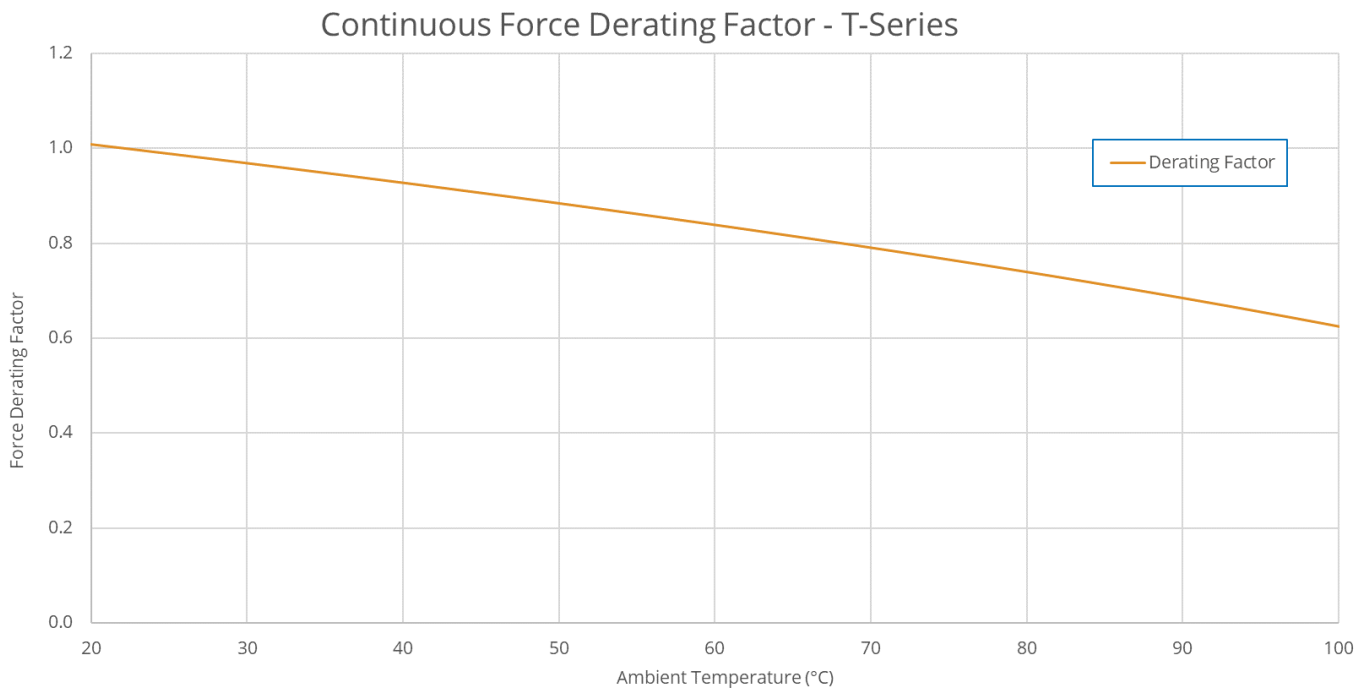


Figure 8: Derating factor for maximum continuous load C_T as a function of temperature.

T-Series Controller Overview

This section provides an overview of how to configure the T-Series controller. All configuration variables can be read or written via the RS-485 interface and saved to non-volatile memory.

The RS-485 command line interface is half-duplex and non-addressable. The Binary Serial Control protocol is addressable.

Configuration Overview

Read Variable

The current value in RAM for all configuration variables can be read via the RS-485 interface with the 'RV' command followed by the case-sensitive configuration variable name. E.g. reading the operating mode can be accomplished with the command "RV opMode". Issuing the "RV" command with no argument will show all the available configuration variables.

Write Variable

All configuration variables can be overwritten via the RS-485 interface with the 'WV' command followed by the case-sensitive configuration variable and the desired setting. E.g., setting the operating mode to 1 can be accomplished with the command "WV opMode 1"

Save Configuration

The WV command will only change the variable in RAM, the configuration must be saved to non-volatile memory with the "CW321" command to maintain the changes after power-cycle or reset.

T-Series Control Overview

This section provides information on the configuration and structure of commands used to control the actuators motion through either CAN 2.0B, RS-485 Binary Serial Control protocol (BSC), and RC PWM control (RCP). User must set operating mode (`opMode`) to define how the actuator behaves and control source (`cntlSrc`) to define how you would like to communicate with the actuator. The actuator's serial interface is available when communicating via CAN 2.0B or RC PWM for configuration and/or telemetry.

Operating Mode

The actuator firmware supports several operating modes, including single-turn absolute position, velocity, and torque control, as well as a command line interface (CLI) mode accessible via RS-485 serial communication. The operating mode is selected using the configuration variable `opMode`. The sections below provide a high-level overview of each mode. Detailed configuration and usage instructions are provided in later chapters of this manual.

Command Line Interface Control Mode:

`opMode` = 0

This mode enables actuator position control through the RS-485 Command Line Interface, using commands such as TA (trajectory move to absolute position) or PA (set absolute position).

CLI mode provides a simple, direct interface for initial configuration, familiarization with actuator behavior, and basic functional testing. It is primarily intended for configuration, development, debugging, or bench-level control rather than real-time operational use.

****Note:** The control source variable (`cntlSrc`) is ignored when `opMode` = 0

Position Control Mode:

`opMode` = 1

In this mode, the actuator operates in 12-bit single turn absolute position control. All motion commands sent to the actuator will map to a single-turn absolute position using the output encoder. The actuator's speed, acceleration, and deceleration, will be controlled by the change in position per update interval sent by the user. This mode is used for traditional servo applications requiring precise, dynamic, and repeatable position control within one mechanical revolution of the output shaft such as control surface and valve control applications.

Velocity Control Mode:

`opMode` = 2

Velocity control mode allows continuous rotation at a user-defined target velocity. The actuator will rotate continuously at the requested velocity, maintaining a relative position counter using an independent 32-bit position encoder. The user can set the encoder feedback values via RS-485 serial to establish a known position based on external reference. The relative position feedback from the actuator can be monitored via telemetry for example.

Torque Control Mode:

`opMode` = 3

In torque control mode, the actuator's internal position and velocity loops are bypassed, and the user commands torque (current) directly to the motor's current controller. This mode is intended for integration into systems where an external controller closes the position or velocity loop.

Use caution when operating in this mode—since internal motion limiting is disabled, system-level safety and feedback control must be implemented externally.

Control Source

The control sources for this firmware include CAN 2.0B, RS-485 Binary Serial Control protocol, and RC PWM. The control source can be set via the `cntlSrc` configuration variable.

`cntlSrc` = 0 CAN 2.0B

`cntlSrc` = 1 RS-485 Binary Serial Control (BSC)

`cntlSrc` = 2 RC PWM (RCP), (1-2ms pulse)

Serial Mode

Serial mode defines whether the actuator's serial interface is using the binary serial control protocol or the command line interface when `opMode` is not equal to 0 and `cntlSrc` is not equal to 1. The serial mode can be set via the `sMode` configuration variable.

`sMode` = 0 Command Line Interface (CLI)

`sMode` = 1 Binary Serial Control (BSC)

Control Frame Data Format

The T-Series can receive up to 8 data bytes in each control frame from the master controller. The `rxData` variable is a string with a maximum length of 8 characters where each character designates a byte of data in the control frame. The actuator must be configured to receive a position command update.

The data format for received position command messages is configured with the `rxData` configuration variable.

Control Frame Data Format

Char	Description
<	Position command LSB
>	Position command MSB
(Max motor current LSB
)	Max motor current MSB
*	Control Word
X	Ignore byte
x	Ignore byte

Table 6: Command options for T-Series Actuator Control

The default value of `rxData` = "<>" requires the user to send a message with 2 data bytes containing the LSB of the position command as the first data byte in the message, and the MSB of the position command as the second data byte. Note that setting the max motor current to a low value does not lead to a freely backdriveable actuator with the T-Series and will instead cause a dynamic braking behavior. To allow the actuator to backdrive freely, the Control Word bit 0 must be set high, or the startup/timeout action set to COAST the motor.

Control Word

The user has control over several functions through the Control Word if `rxData` includes the "*" designator.

Control Word

Bit	Bit Description
0	COAST the motor when active (0 = normal, 1 = coast) *overrides dynamic brake function
1	Dynamic Brake the motor (0 = normal, 1 = brake)
2	Reserved
3	Zero 32bit secondary position encoder
4 to 7	Reserved

Table 7: Control Word

Position Interpolation

A position interpolation function provides smooth motion between each position update. The smoother motion provided by the interpolator increases mechanical reliability and the actuator response.

Interpolation can be enabled/disabled with the `inEna` configuration variable.

`inEna` = 0 disables interpolation

`inEna` = 1 enables interpolation

Interpolation Period

The interpolation period for the CAN, BSC, and RCP control modes can be set with the `canIvl`, `bscIvl`, and `rcpIvl` variables. This variable is set in 1 ms increments and should be set equal to the expected position update interval from the master controller. If the update rate cannot be matched exactly, setting the interpolation interval slightly larger than the expected update interval is recommended. The default value is 50, which is equal to a 50 ms update interval (20 Hz update frequency).

Default Start Position

The T-Series can be configured to drive to a default position on startup before receiving a valid command message, or on failure of the communication bus leading to a timeout event. These events can execute a trajectory move to the position defined by `defPos` with trajectory parameters defined by the `maxSpeed` and `accel` configuration variables. The default position can be set with the `defPos` configuration variable with units are in absolute position encoder counts.

Received Message Timeout

The received message timeout period for the communication protocols can be set with the `canTO`, `bscTO`, and `rcpTO` configuration variable. This variable is set in 1 ms increments. The default value is 1250 (1.25 seconds). When a timeout occurs, the appropriate status register bit (bit 40) will become active and the actuator's timeout behavior for the currently active `opMode` (set by `pTOact`, `vTOact`, or `tTOact` for position, velocity, or torque operating modes) will initiate. The actuator will not execute the timeout action if no valid command frames have been received after powerup.

Timeout Actions

The T-Series can be configured for different timeout behaviors depending on the active operating mode (position, velocity, or torque control mode). These behaviors are defined by the `pTOact`, `vTOact`, and `tTOact` variables for position, velocity and torque operating modes respectively.

Position Mode Timeout Actions:

`pTOact` = 0 Holds the last valid position with the last valid max motor current command

`pTOact` = 1 Executes a trajectory move to the default position `defPos` with the last valid motor current command

`pTOact` = 2 COAST the motor

`pTOact` = 3 Dynamic brake the motor

`pTOact` = 4 Holds the last valid position with the configured `maxCurr` setting

`pTOact` = 5 Executes a trajectory move to the default position `defPos` with the configured `maxCurr` setting

Velocity Mode Timeout Actions:

`vTOact` = 0 COAST the motor

`vTOact` = 1 Dynamic brake the motor

`vTOact` = 2 Reset max current to configured `maxCurr` setting, stop immediately and hold position

`vTOact` = 3 Stop immediately and hold position

`vTOact` = 4 Reset max current to configured `maxCurr` setting, decelerate to a stop and hold position

`vTOact` = 5 Decelerate to a stop and hold position

Torque Mode Timeout Actions:

tTOact = 0 COAST the motor

tTOact = 1 Dynamic brake the motor

tTOact = 2 Holds the current position in position control mode

The timeout action will not occur if no valid command frames has been received after powerup. At least one valid command frame must be received for the timeout action to take place.

Startup Actions

The T-Series can be configured for different behaviors at startup before receiving the first valid command frame. These behaviors are defined by the pSUact, vSUact, and tSUact variables for position, velocity and torque operating modes respectively.

Position Mode Startup Actions:

pSUact = 0 Holds the last valid position with the last valid max motor current command

pSUact = 1 Executes a trajectory move to the default position defPos with the maxCurr setting

pSUact = 2 COAST the motor

pSUact = 3 Dynamic brake the motor

Velocity Mode Startup Actions:

vSUact = 0 COAST the motor

vSUact = 1 Dynamic brake the motor

vSUact = 2 Hold the current position in position control mode with maxCurr setting

Torque Mode Startup Actions:

tSUact = 0 COAST the motor

tSUact = 1 Dynamic brake the motor

tSUact = 2 Holds the current position in position control mode using the maxCurr setting

System Errors Register

The System Errors Register contains error status bits that result in automatic entry into safe-mode (motor OFF) when set.

System Error Register

Error Bit	Bit Description
0	Configuration not loaded
1	Invalid Configuration in NVM
2	Bridge driver IC initialization failure
3	Encoder initialization failure
4 to 7	Reserved

Table 8: System Errors Register bit descriptions

System Warning Register

The System Warning Register contains latching error status bits that do NOT result in automatic safe-mode (motor OFF).

System Warning Register	
Warning Bit	Bit Description
0	Bad configuration block in NVM
1	Configuration error
2	Erroneous reset
3	Bridge Fault
4	Hall sensor error
5	Supply voltage High (> 55 VDC)
6	Supply voltage Low (< 9 VDC)
7	Temperature High
8	UART error
9	Switch Voltage Low
10	Humidity sensor comm error
11	Encoder sensor comm error
12	Encoder mismatch error
13	Encoder diagnostic error
14	CAN module init error
15	CAN TX error

Table 9: System Warnings Register bit descriptions

System Status Register

The status register bytes contain bits that represent different aspects of the T-Series health and state. The configurable telemetry options include latched and non-latched versions of the individual status bytes.

Status Byte	Bit	Status Bit #	Description
0	0	0	Nominal 3.3 V bus
0	1	1	Nominal 5.0 V bus
0	2	2	Switching power supply output < 6.0V
0	3	3	Supply voltage less than minimum (9 VDC)
0	4	4	Supply voltage greater than maximum (55 VDC)
0	5	5	Safe Mode, MOTOR OFF
0	6	6	Bridge is Active
0	7	7	Bridge driver IC fault
1	0	8	State of Optically Isolated Input 2
1	1	9	Direction of travel
1	2	10	Motor current demand capped at maxCurr
1	3	11	Motor current feedback greater than ovCurr
1	4	12	Trajectory move is active
1	5	13	PWM output is set at 0
1	6	14	Hall sensor zero velocity
1	7	15	Hall sensor direction
2	0	16	System error register is non-zero
2	1	17	System warning register is non-zero
2	2	18	Temperature at PCB is greater than ovTemp value
2	3	19	Temperature at PCB is less than unTemp value
2	4	20	Humidity at PCB greater than ovHumd value
2	5	21	Encoder health status (0 = healthy)
2	6	22	RC PWM Timeout
2	7	23	Reserved
3	0	24	RC PWM input capped by rMin
3	1	25	RC PWM input capped by rMax
3	2	26	Absolute position less than spMin
3	3	27	Absolute position greater than spMax
3	4	28	Absolute position less than posLe
3	5	29	Absolute position greater than posGr
3	6	30	Encoder feedback zero velocity
3	7	31	Encoder direction
4	0	32	CAN RX error counter > 0 and < 127
4	1	33	CAN TX error counter > 0 and < 127
4	2	34	CAN RX Warning
4	3	35	CAN TX Warning
4	4	36	CAN RX error passive state
4	5	37	CAN TX error passive state
4	6	38	CAN TX Bus Off
4	7	39	CAN TX error flag
5	0	40	CAN RX Timeout Flag
5	1	41	BSC RX Timeout Flag
5	2	42	CAN position command capped low at pMin
5	3	43	CAN position command capped high at pMax
5	4	44	BSC position command capped low at pMin
5	5	45	BSC position command capped high at pMax
5	6	46	Overvolt braking is active
5	7	47	Dynamic braking off (0 = brake on, 1 = brake off)

Table 10: Status Register bit descriptions

T-Series CAN Behavior/Configuration

This section provides an overview of the T-Series CAN protocol configuration options and behavior. Note that there are no built-in terminating resistors.

CAN Baud Rate

The T-Series' CAN baud rate is definable with the `CANspd` variable. Values of 0 to 7 are acceptable and defined as follows:

<code>CANspd</code> =	
0:	1 Mbps
1:	500 Kbps
2:	250 Kbps
3:	125 Kbps
4:	100 Kbps
5:	50 Kbps
6:	20 Kbps
7:	10 Kbps

CAN Message Type

The firmware will support either 11-bit (standard) or 29-bit (extended) identifiers, settable with the `CANext` variable.

`CANext` = 0: Standard message type (11-bit identifier)

`CANext` = 1: Extended message type (29-bit identifier)

rxID

`rxID` is the actuator's network address and is used along with `rxMask` to filter incoming CAN command messages. For acceptance, the message's identifier must match `rxID` where each corresponding bit in `rxMask` is set to '1'. If `CANext` = 0, only the lower 11 bits of `rxID` and `rxMask` are examined (standard message type). If `CANext` is 1, the lower 29 bits of `rxID` and `rxMask` are examined (extended message type). Configuration is accomplished with the `rxID` variable. `rxID` can be any value from 0x00000000 to 0x1FFFFFFF.

rxMask

`rxMask` is used along with `rxID` to filter incoming CAN command messages. For acceptance, the message's identifier must match `rxID` where each corresponding bit in `rxMask` is set to '1'. If a bit in `rxMask` is set to '0', the corresponding bit of an incoming can message is matched for acceptance regardless of its value. If `rxMask` is set to 0x00000000, CAN messages with any identifier will be accepted. If `rxMask` is set to 0x1FFFFFFF, only CAN messages with an identifier that exactly matches the relevant bits of `rxID` will be accepted. If `CANext` = 0, only the lower 11 bits of `rxID` and `rxMask` are examined (standard message type). If `CANext` is 1, the lower 29 bits of `rxID` and `rxMask` are examined (extended message type). Configuration is accomplished with the `rxMask` variable. `rxMask` can be any value from 0x00000000 to 0x1FFFFFFF.

T-Series CAN Telemetry

The T-Series can send user-configurable telemetry messages over the CAN bus with a configurable ID. There are three independent telemetry messages that can be configured with separate identifiers, data, and update rates.

Telemetry Enable

The `txEna` configuration variable is used to turn on/off the T-Series CAN telemetry transmit functionality. The least significant 3 bits correspond to the 3 telemetry messages with bit0 as `tx1Ena`, bit1 as `tx2Ena`, and bit2 as `tx3Ena`.

Example settings for `txEna`:

`txEna` = 0 disables telemetry

`txEna` = 6 (0b110) activates telemetry messages 2 and 3

`txEna` = 7 (0b111) activates all 3 telemetry messages

Telemetry Interval

The rate at which the T-Series transmits the telemetry CAN message is configurable and set by editing the `txNlvl` variable. The setting is in milliseconds and has a valid range of 2 to 10,000. Care must be taken to not overload the CAN bus by setting the interval value too low. The default values are `tx1lvl` = 1000, `tx2lvl` = 2500, and `tx3lvl` = 5000.

Telemetry Message ID

The `txNID` variable sets the CAN identifier for the transmitted CAN telemetry message. This represents the destination address of the telemetry message. Either the lower 11 bits (standard) or the lower 29 bits (extended) of `txNID` are used depending on CAN bus message type setting (`CANext`). The user can specify the ID in decimal or hex if preceded with "0x". `txNID` can be any value from 0x00000000 to 0x1FFFFFFF, the default values are `tx1ID` = 0x0000007F, `tx2ID` = 0x0000027F, `tx3ID` = 0x0000037F.

Telemetry Message Data Format

The data bytes of each telemetry message are configurable with the `txNData` configuration variables. The `txNData` variables are strings where each character designates a variable in the telemetry message. `txNData` may be 1 to 8 characters in length, but the total number of resulting data bytes cannot exceed 8 for a single message.

Telemetry Message Data Format

Char	Description	Type	Num Bytes	Available via RS485
A	ID Byte	UINT8	1	Yes
B	Critical Errors Byte	UINT8	1	Yes
C	Warning Byte 0	UINT8	1	Yes
D	Warning Byte 1	UINT8	1	Yes
E	Warning Register – Clear on Broadcast	UINT16	2	Yes
F	CAN Command Value	UINT16	2	Yes
G	Position Demand	UINT16	2	Yes
H	Motor Current Demand	INT16	2	Yes
I	Motor Current Demand Limit	UINT16	2	Yes
J	Duty Cycle Out	INT16	2	Yes
K	Encoder Position Feedback	UINT16	2	Yes
L	Hall Position Feedback	UINT16	2	Yes
M	Encoder Velocity Count	INT16	2	Yes
N	Encoder Velocity Interval	UINT32	4	Yes
O	Motor Current – Instantaneous	INT16	2	Yes

P	Motor Current – Average over interval	INT16	2	No
Q	Motor Current – Minimum over interval	INT16	2	No
R	Motor Current – Maximum over interval	INT16	2	No
S	Switch Voltage – Instantaneous	UINT16	2	Yes
T	Switch Voltage – Average over interval	UINT16	2	No
U	Switch Voltage – Minimum over interval	UINT16	2	No
V	Switch Voltage – Maximum over interval	UINT16	2	No
W	Relative position – 32 bits	UINT32	4	Yes
X	RC PWM Command	UINT16	2	No
Y	RC PWM raw input	UINT32	4	No
Z	RC PWM interval	UINT16	2	No
a	Supply Voltage – Instantaneous	UINT16	2	Yes
b	Supply Voltage – Average over interval	UINT16	2	No
c	Supply Voltage – Minimum over interval	UINT16	2	No
d	Supply Voltage – Maximum over interval	UINT16	2	No
e	Status Register Byte 0	UINT8	1	Yes
f	Status Register Byte 1	UINT8	1	Yes
g	Status Register Byte 2	UINT8	1	Yes
h	Status Register Byte 3	UINT8	1	Yes
i	Status Register Byte 4	UINT8	1	Yes
j	Status Register Byte 5	UINT8	1	Yes
k	Status Register Byte 0 – Latched High	UINT8	1	No
l	Status Register Byte 1 – Latched High	UINT8	1	No
m	Status Register Byte 2 – Latched High	UINT8	1	No
n	Status Register Byte 3 – Latched High	UINT8	1	No
o	Status Register Byte 4 – Latched High	UINT8	1	No
p	Status Register Byte 5 – Latched High	UINT8	1	No
q	Status Register Byte 0 – Latched Low	UINT8	1	No
r	Status Register Byte 1 – Latched Low	UINT8	1	No
s	Status Register Byte 2 – Latched Low	UINT8	1	No
t	Status Register Byte 3 – Latched Low	UINT8	1	No
u	Status Register Byte 4 – Latched Low	UINT8	1	No
v	Status Register Byte 5 – Latched Low	UINT8	1	No
w	Core Temperature (2 Hz update)	UINT8	1	Yes
x	PCB Relative Humidity (0.25 Hz update)	UINT8	1	Yes
y	PCB Temperature2 – Byte	UINT8	1	Yes
z	Core Temperature (°C)– 32 bits (2 Hz update)	FLOAT32	4	Yes
0	Hall Position Counter – 32 bits	UINT32	4	Yes
1	Millisecond Counter	UINT64	8	Yes
2	BSC Command Interval	UINT16	2	Yes
3	PCB Humidity (%) – 32 bits (0.25 Hz update)	FLOAT32	4	Yes
4	PCB Temperature (°C)	FLOAT32	4	Yes
5	CAN Interval	UINT16	2	Yes
6	cntlSrc	UINT8	1	Yes
7	Serial Number	UINT32	4	Yes
8	UART Status Register	UINT16	2	Yes
9	CAN Errors (MSB = TX errors, LSB = RX errors)	UINT16	2	Yes
+	BSC Command Value	UINT16	2	Yes
^	BSC Raw Input	UINT16	2	Yes
&	CAN Raw Input	UINT16	2	Yes
#	BSC Control Word	UINT8	1	Yes

~	CAN Control Word	UINT8	1	Yes
@	BSC CRC Errors	UINT16	2	Yes
\$	BSC Timeouts	UINT16	2	Yes
%	Serial TX Dropped	UINT16	2	Yes
!	opMode	UINT8	1	Yes
=	Supply Voltage (VDC)	FLOAT32	4	Yes
:	Velocity (RPM)	FLOAT32	4	Yes
	Position PID I_term	FLOAT32	4	Yes
.	Velocity PID I_term	FLOAT32	4	Yes

Table 11: Telemetry options for the CAN 2.0B protocol used in txNData and “RR” command

The telemetry message may include anywhere from 1 to 8 bytes of data and they can be any combination of the above options. Latched status bytes are reset after they are transmitted. Latched low bytes are reset high, and latched high bytes are reset low. Values that are average or min/max over the telemetry interval are also reset after they are transmitted. Values that are reset after they are transmitted should only be accessed from one telemetry message; either [tx1Data](#), [tx2Data](#), or [tx3Data](#), but never more than one at the same time.

Since status bits are tested one time every millisecond, a status bit event lasting less than one millisecond may not be latched.

The default values are: [tx1Data](#) = “GKHO”, [tx2Data](#) = “klmno”, and [tx3Data](#) = “wxy”.

T-Series Event Based Asynchronous CAN Messages

The T-Series can be configured to transmit CAN messages on the occurrence of asynchronous events.

The asynchronous event message frames will all have 8 data bytes with the following format:

CAN Frame Data Byte	Byte Description
0	IDbyte (configuration variable, should be set to a unique value for each actuator on the bus)
1	Event Code
2-7	Event Data Bytes 0-5

Table 12: Asynchronous Message Format

Configuring Event Messages

The actuator will broadcast event messages on the CAN bus with an address defined by `evntID`

There are two separate configuration variables used to enable event message broadcasts, one for system warning events, and one for non-critical “miscellaneous” events.

Warning events occur when a warning bit transitions from low to high. After a warning message is sent for a particular warning bit, a timer with the value of `evntIvl` milliseconds must expire before another message for the same bit can be re-broadcast.

Warning event messages are enabled with the `evntWrn` configuration variable. Setting a bit high in `evntWrn` enables event messages for the corresponding bit in the system warning register. The warning bit will clear after the warning event message is broadcast if the corresponding “clear” bit in the `evntWrnC` mask is high. If the clear bit is low, the warning bit will remain unchanged by the event message broadcast. If the warning bit is not cleared, the warning event will not reoccur.

The `evntMsc` configuration variable works the same way as `evntWrn` with each bit in the mask enabling a pre-defined event message. The events in `evntMsc` are not constrained by a minimum rebroadcast time as the warning events are.

Table 13 and Table 14 detail the pre-defined event messages that can be enabled with `evntWrn` and `evntMsc`:

System Event Details

The following tables detail the possible asynchronous event messages. Please contact Ultra Motion engineering for details regarding the various event data.

Event Code	evntWrn bit	Warning Event Cause	Warning Event Data Bytes (6 bytes)
0	0	Bad configuration block in NVM	byte 0-5: unused
1	1	Configuration error	byte 0-1: 16bit Config Content Error Code
2	2	Erroneous reset	byte 0-1: 16bit Reset Cause byte 2-5: 32bit Error Address
3	3	Bridge Fault	byte 0-1: Bridge Fault Count byte 2-3: Bridge Fault Register byte 4-5: Bridge Fault Register
4	4	Hall sensor error	byte 1-2: 16bit Invalid Hall State Count byte 3-4: 16bit Invalid Hall Sequence Count
5	5	Voltage High (> 55 VDC)	byte 0-1: 16bit Supply Voltage Sense
6	6	Voltage Low (< 9 VDC)	byte 0-1: 16bit Supply Voltage Sense
5	5	FRAM error	byte 0-5: unused
6	6	UART error	byte 0-1: 16bit UART Status byte 2-3: 16bit UART Error Count
7	7	Temperature High	byte 0: Core Temperature byte 1: PCB Temperature
8	8	UART Error	byte 0-1: UART Status byte 1-2: UART Error Count
9	9	Switch Voltage Low	byte 0-1: 16bit Switching Voltage Sense
10	10	Humidity sensor comm error	byte 0-1: 16bit Sensor Error Code
11	11	Encoder sensor comm error	byte 0-1: 16bit Primary Parity Error Count byte 2-3: 16bit Primary Error Bit Count
12	12	Encoder mismatch error	byte 0-1: 12 bit SPI absolute position byte 2-3: 12 bit QEI absolute position
13	13	Encoder diagnostic error	byte 0-1: 16bit Primary Diagnostic Register
14	14	CAN initialization error	N/A
15	15	CAN transmit error	N/A

Table 13: Warning event codes, causes, and data fields

Event Code	evntMsc bit	Miscellaneous Event Cause	Miscellaneous Event Data Bytes (6 bytes)
16	0	Normal Reset (power-on or user commanded)	byte 0: 8bit System Error Register byte 1: 8bit System Status Byte1 byte 2-5: 32bit Actuator Serial Number
17	1	System Error (motor in safe mode)	byte 0: 8bit System Error Register byte 1: 8bit System Status Byte1 byte 2-3: 16bit System Warning Register byte 4-5: 16bit Config Content Error Code
18	2	Opto Active	byte 0-5: Status Bytes 0-5
19	3	Opto Inactive	byte 0-5: Status Bytes 0-5

Table 14: Miscellaneous event codes, causes, and data fields

T-Series Binary Serial Control Protocol

The T-Series' Binary Serial Control protocol (BSC) is a CRC protected, addressable RS-485 serial protocol. The following section will detail the BSC's usage and behavior.

T-Series BSC Behavior/Configuration

A BSC network is setup as a multi-drop half-duplex RS-485 network with 1 controller and 1 or more actuators. The controller issues command frames to the actuators and the actuators respond with response frames. Only the controller may issue command frames. The controller may have only one outstanding command frame at a time. After the controller issues a command frame to an actuator it must receive a valid response frame within the **response timeout period** or it should consider the command frame failed. While transmitting a command frame, the controller should not pause more than 30 serial bit times between bytes. A pause of greater than 30 bit times between bytes of the command frame will cause the receiving actuator to timeout and drop the command frame. Actuators will only respond to command frames with an address byte that matches their `bscAddr` configuration variable. Actuators will not respond to invalid command frames or command frames addressed to the group address (address byte = 0).

bscAddr

In BSC mode, the configuration variable `bscAddr` is the actuator's address on the shared RS-485 bus. Each actuator on a shared RS-485 bus must have a unique address. The actuator will only accept command frames that match the address set in its `bscAddr` configuration variable. `bscAddr` can be any value from 1 to 255.

Command Frame Format

Each command frame sent to an actuator must be configured as follows:

BSC Command Frame Format

Byte	Description
0	Start Byte [0xAA]
1	Actuator Address
2	Command Code
3	Data Length Byte
4...N	Data
N+1	CRC LSB
N+2	CRC MSB

Table 15: BSC Command Frame Format

Command Code

The BSC protocol currently supports the following commands:

BSC Command Codes

Command Code	Description
0x01	Command Line Interface Passthrough
0x02	BSC Control Update Command
0x03	Set Operating Mode - <code>opMode</code>
0x04	Read Runtime Variable
0x05	Set Control Source – <code>cntlSrc</code>

Table 16: BSC Command Code List

Command Line Interface Passthrough:

The CLI passthrough will act the same as sending commands through the command line interface when `opMode` = 0. For example, the data being sent with command code 0x01 can be used to write configuration variables "wv `maxCurr` 8000" or issue a reset command "zc321". Trajectory motion commands are not valid in this mode.

BSC Control Command:

The BSC control command is used to send commands, motor phase current, and Control Word updates to the actuator based on the value of `rxData` (see Table 6: Command options for T-Series Actuator Control). These commands should be sent to each actuator at a regular rate, defined by the BSC protocol's configured interpolation interval `bsclvl`.

Set Operating Mode:

Command code 0x03 is used to adjust the operating mode while actively in BSC mode (`cntrlSrc = 1` or `sMode = 1`)

Read Runtime Variable:

Command code 0x04 mimics the "RR x" functionality found in the CLI and is used for querying actuators for various telemetry and health information. The response data will be in little endian format (least significant byte first). The 0x04 command can be sent with zero data which will return zero data.

Set Control Source:

Command code 0x05 is used to adjust the control source while actively in BSC mode (`cntrlSrc = 1` or `sMode = 1`)

Response Frame Format

The actuator will respond to each valid command frame as follows:

BSC Response Frame Format

Byte	Description
0	Start Byte [0x55]
1	Actuator Address
2	Response Code
3	Data Length Byte
4...N	Data
N+1	CRC LSB
N+2	CRC MSB

Table 17: BSC Response Frame Format

Response Code

The actuator's response code will include the command frame's command code in the upper 4 bits along with an error code in the lower 4 bits. As an example, an invalid argument sent with the CLI passthrough code (0x01) would have the response code 0x16

BSC Response Frame Error Codes

Error Code	Error Description
0	CMD_OK
1	CMD_ERROR_INVALID_CMD
2	CMD_ERROR_LEN_ZRO
3	CMD_ERROR_INTERNAL
4	CMD_ERROR_ARG_TOOMANY
5	CMD_ERROR_ARG_TOOFEW
6	CMD_ERROR_ARG_INVALID
7	CMD_ERROR_ARG_RANGE
8	CMD_ERROR_STRING_LONG
9	CMD_ERROR_PERMISSION_DENIED
10	CMD_ERROR_NOT_ALLOWED
11	CMD_ERROR_NOT_FOUND
12	CMD_ERROR_COND_STATUS
13	CMD_ERROR_COND_STATE
14	CMD_ERROR_CLI_LOCKED
15	CMD_ERROR_BUFFER_FULL

Table 18: BSC Response Frame Error Code List

BSC Interpolation Period

The interpolation period can be set with the `bscIvl` variable. This variable is set in 1 ms increments and should be set equal to the expected position update interval from the master controller. If the update rate cannot be matched exactly, setting the interpolation interval slightly larger than the expected update interval is recommended. The default value of `bscIvl` is 50. This value is equal to a 50 ms update interval (20 Hz update frequency).

Revert to Command Line Interface

A special command is provided to allow a user to manually revert back to `opMode` = 0 through a terminal program when the actuator is in BSC mode. The special command “[opmodezero]” (including the brackets) will change the operating mode to zero and return the user to a CLI prompt.

Group Messages:

A value of 0x00 for the actuator address in a command frame will be accepted by all actuators on the serial bus. Not every BSC command will execute as a group command. Currently only command 0x02, “BSC Control Update Command”, will execute as a group command. Any other command addressed to the group address will be quietly dropped. The actuator will not issue a response frame after executing a group command.

Receiving Actuator Command Frame Timeout:

While the controller is transmitting a command frame to an actuator on the serial bus, the maximum delay between bytes of the command frame is 30 serial bit times. If the controller pauses for longer than the 30-bit-time timeout period, the actuator will quietly drop the command frame.

Cyclic Redundancy Check:

The CRC algorithm ensures data integrity by calculating a checksum based on a specific polynomial. The following steps outline the CRC-16-CCITT algorithm with the polynomial 0x1021 and an initial value of 0xFFFF used by the T-Series BSC protocol.

1. Exclude the Start Byte (0xAA).
2. Compute CRC for Address Byte, Command Code, Data Length Byte, and Data bytes.

CRC16:

```
uint16_t softCRC16( uint8_t data[], uint16_t data_len )
{
    uint16_t CRC_result = 0xFFFFu;
    uint16_t polynomial = 0x1021u;

    uint8_t data_width = 8;
    for( uint16_t word = 0; word < data_len; word++ )
    {
        CRC_result ^= (data[word] << 8);
        for( uint8_t count = 0; count < data_width; count++ )
        {
            if( CRC_result >= 0x8000u )
            {
                CRC_result = (CRC_result << 1) ^ polynomial;
            }
            else
            {
                CRC_result = (CRC_result << 1);
            }
        }
    }
    return CRC_result;
}
```

Example Command and Response Frames:

Read Runtime Variable 'K' - Encoder Position Feedback (2 bytes):

Command Frame: AA 80 04 01 4B A6 4F

Start Byte: 0xAA, Address: 0x80, Command Code: 0x04, Command Length: 0x01, Command Data : 0x4B ['K'], Checksum: 0x4FA6

Received Response: 2048d

Response Frame: 55 80 40 02 00 08 28 B2

Start Byte: 0x55, Address: 0x80, Response Code: 0x40, Response Length: 0x02, Response Data: 0x0008, Checksum: 0xB228

CLI Passthrough "vv ovTemp 40.0"

Command Frame: AA 80 01 0E 77 76 20 6F 76 54 65 6D 70 20 34 30 2E 30 FB 56

Start Byte: 0xAA, Address: 0x80, Command Code: 0x01, Command Length: 0x0E, Command Data:

77 76 20 6F 76 54 65 6D 70 20 34 30 2E 30 ["vv ovTemp 40.0"], Checksum: 0x56FB

Received Response: 40.0

Response Frame: 55 80 10 04 34 30 2E 30 B2 F9

Start Bytes: 0x55, Address: 0x80, Response Code: 0x10, Response Length: 0x04, Response Data: 34 30 2E 30, Checksum: 0xF9B2

Position Command Update to 3210:

Command Frame: AA 80 02 02 8A 0C 0B 85

Start Byte: 0xAA, Address: 0x80, Command Code: 0x02, Command Length: 0x02, Command Data: 8A 0C, Checksum: 0x850B

Response Frame: 55 80 20 00 20 F1

Start Byte: 0x55, Address: 0x80, Response Code: 0x20, Response Length: 0x00, Response Data: none, Checksum: 0xF120

T-Series RC PWM Mode

The T-Series controller includes optically isolated digital inputs that can be used to operate via an RC PWM (1 to 2 millisecond pulse) control source. The RC PWM control signal must be wired to IN1+/IN1-

Optically Isolated Inputs

The T-Series' digital inputs are optically isolated, meaning that they are electrically isolated from the rest of the circuit and therefore do not share a common ground. This is to provide optimal noise rejection. It is acceptable to connect IN1- or IN2- to actuator ground to simplify wiring, but this will remove the benefit of the noise rejection from optical isolation.

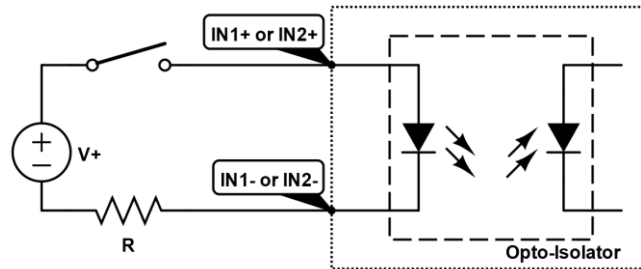


Figure 9: Optically isolated digital input wiring schematic

Using Higher Voltages for Optically Isolated Inputs on the T-Series Controller

The digital inputs need to be energized with between 5 mA and 15 mA of forward current to be activated. This can be achieved with a voltage between 2.5 and 6.6 volts (DC). To avoid intermittent signal issues, ensure your signal source is capable of sourcing the required activation current without sagging. A voltage in excess of 6.6 volts can be used if it is in conjunction with an appropriately sized current limiting resistor. The range of resistor sizes can be calculated as a function of voltage using the equations below. Be sure to choose appropriate wattage ratings for the resistors.

$$R_{min}[\text{ohms}] = \frac{V[\text{volts}] - 1.25}{.015} - 356, \quad R_{max}[\text{ohms}] = \frac{V[\text{volts}] - 1.85}{.005} - 440$$

RC PWM Interpolation Period

The interpolation period can be set with the `rcplvl` variable. This variable is set in 1 ms increments and should be set equal to the expected position update interval from the master controller. If the update rate cannot be matched exactly, setting the interpolation interval slightly larger than the expected update interval is recommended. The default value of `rcplvl` is 50. This value is equal to a 50 ms update interval (20 Hz update frequency).

T-Series RS-485 Serial Command Line Interface

The T-Series RS-485 serial interface is half-duplex, 8N1. A command line is parsed and executed after a carriage return byte is received by the UART. A command line consists of the following: 0 or more space or tab bytes, 2-byte ASCII command, 0 or more ASCII arguments preceded by 1 or more space or tab bytes, 0 or more space or tab bytes, carriage return byte.

The CLI will respond to a command line with the command's returned data, "OK" if the command was executed successfully but has no data to return, or "ERROR: " + *error type* if a command was not executed successfully.

Command Line Interface Lock

The Command Line Interface (CLI) is designed with a safety feature to protect the actuator from executing erroneous commands due to noise or user error. The T-Series will start-up with the CLI unlocked by default, because the configuration variable `cliLock` is set to 1.

If the variable `cliLock` is set to 0, the CLI will be locked on startup, and no serial commands will be executed. In this locked state, you must use the special command "LK" with the argument "unlock" (i.e., LK unlock\r) to activate the interface.

The configuration variable `cliLock` controls the power-cycle behavior: setting `cliLock` to 1 permanently enables the CLI on startup, while setting it to 0 causes the CLI to default to the locked state after every power cycle."

T-Series Serial Commands Quick Reference Table

Command	Description	Group	Argument Num
RV	Read configuration variable	Configuration	1 or 0
WV	Write configuration variable	Configuration	2
WF	Write any configuration variable to flash only	Configuration	2
CW	Save configuration settings to non-volatile memory	Configuration	1
CC	Copy configuration block from inactive to active partition	Configuration	1
RR	Read runtime variable	Read	1 or 0
SR	Read status register bits	Read	2 or 0
RI	Read reset information	Read	0
ZD	Read bridge driver FAULT register log	Read	0
FS	Show serial number, part number, firmware version, target PCB	Read	0
HE	Show all available serial commands	Read	0
TA	Trajectory move to absolute position	Motion Control	1
TO	Trajectory move to offset position	Motion Control	1
TR	Trajectory move to fully retracted position (<i>spMin</i>)	Motion Control	0
TM	Trajectory move to mid-stroke $(spMin+spMax)/2$	Motion Control	0
TE	Trajectory move to fully extended position (<i>spMax</i>)	Motion Control	0
TD	Trajectory move to the default position (<i>defPos</i>)	Motion Control	0
T1	Trajectory move to tPos1	Motion Control	0
T2	Trajectory move to tPos2	Motion Control	0
TK	Interrupt current trajectory move	Motion Control	0
TS	Stop trajectory at maximum acceleration	Motion Control	0
PC	Set target position to current position	Motion Control	0
PA	Set absolute target position	Motion Control	1
PO	Set target position with an offset	Motion Control	1
SE	Set 32-bit relative encoder position	Motion Control	1
ZR	Restart actuator controller	System	1
LK	Lock or unlock serial command-line-interface	System	1 or 0
FW	Show firmware installed on active and inactive partitions	System	0
SA	Swap active/inactive firmware partition	System	1
ZC	Run actuator calibration utility	System	1
ZU	Run serial firmware update utility	System	1
BR	Read partition BSEQ number	System	0
BW	Write partition BSEQ number	System	1
FC	Check for inactive partition configuration block	System	0
BC	Check for active partition configuration block	System	0
VC	Display size of configuration block	System	0

Table 19: Serial CLI Commands List

T-Series Serial CLI Command Details

RV – Read Configuration Variable(s)

This command returns the value of any of the configuration variable. Sending this command with no argument returns the value of all configuration variables.

Usage: RV

Usage: RV arg1

arg1 Description: Variable name (case sensitive)

arg1 Data Type: String

WV –Write Configuration Variable

This command sets any configuration variable in active ram. To save the current configuration to non-volatile memory, use the **CW** command.

Usage: WV arg1 arg2

arg1 Description: Configuration variable name (case sensitive)

arg1 Data Type: String

arg2 Description: Value to write

arg2 Data Type: Configuration variable's data type

WF –Write Configuration Variable to Flash Only

This command sets any configuration variable in flash only. This command does not change the configuration variable in active ram, only non-volatile memory. It is primarily used for configuring the operating mode to opMode = 2.

Usage: WF arg1 arg2

arg1 Description: Configuration variable name (case sensitive)

arg1 Data Type: String

arg2 Description: Value to write

arg2 Data Type: Configuration variable's data type

CW – Write Configuration Settings to Non-Volatile Memory

This command will take all current configuration settings and write them to non-volatile memory. *This must be done if a setting was changed via a serial command, and you wish to retain this change after a restart or power-cycle.*

Usage: CW arg1

arg1 Description: Passcode to execute command

arg1 Data Type: Unsigned Integer

arg1 Range: "321"

CC – Copy Configuration Block from inactive to active partition

The CC command will copy the actuator configuration from the inactive partition to the active partition. This will overwrite the existing configuration block on the active partition. The command will only execute if the configuration block on the inactive partition is the same size as active partition's configuration block. This command is useful for copying the existing configuration to new firmware after a firmware update. It will only work if the configuration block layout has not changed between old firmware and new. A reset is required to load the copied configuration block after this command is executed.

Usage: CC arg1

arg1 Description: Passcode to execute command

arg1 Data Type: Unsigned Integer

arg1 Range: "321"

RR – Read Run-Time variable

Returns the value of value of any run-time variable. Running this command with no argument shows all of the available run-time variables. The value of some variables can only be read via CAN telemetry. See *Table 11: Telemetry options for the CAN 2.0B protocol used in txNData* for telemetry options accessible via serial. Sending multiple characters will return the requested variables in comma-separated-value format (e.g. RR ABC). Some variables can only be read via CAN telemetry.

Usage: RR

Usage: RR arg1

arg1 Description: One or more run-time variable character designators (see Table 5)

arg1 Data Type: String

SR – Read Status Register Bits

Returns the current value of all status register bytes when the command is sent with no arguments. Returns a particular bit within the status register selected by argument 1 (status byte number) and argument 2 (bit number).

Usage: SR

Usage: SR arg1 arg2

arg1 Description: Status Byte Number

arg1 Data Type: Unsigned Integer

arg1 Range: 0 to 5

arg2 Description: Status Bit Number

arg2 Data Type: Unsigned Integer

arg2 Range: 0 to 7

FS – Read Actuator Details

Return the actuator's serial number, part number, firmware version ID, and target PCB.

Usage: FS

HE – Show All Serial Commands

Return a list of all available serial commands and their descriptions.

Usage: HE

RI – Read Reset Information

Read information about the last CPU reset.

Usage: RI

ZD – Read Bridge Driver Fault Register Log

Returns the fault register log of the bridge driver IC.

Usage: ZD

TA – Trajectory Move to Absolute Position

Sending this command result in a trajectory move an absolute encoder position. Trajectory moves obey the [maxSpeed](#) (maximum profile speed) and [accel](#) (profile acceleration) settings.

Usage: TA arg1

arg1 Description: Absolute encoder position

arg1 Data Type: Unsigned Integer

arg1 Range: [spMin](#) to [spMax](#)

TO – Trajectory Move to Offset Position

Trajectory move to the current position + offset. The resultant position must be within the bounds defined by **spMin** and **spMax**

Usage: TO arg1

arg1 Description: Relative position offset

arg1 Data Type: Signed Integer

arg1 Range: **spMin** < Target Position < **spMax**

TR – Trajectory Move to Fully Retracted Position (spMin)

Trajectory move to **spMin** - "Software Position Minimum" with user defined speed and acceleration.

Usage: TR

TM – Trajectory Move to Mid-Stroke

Trajectory move to midpoint with user defined speed and acceleration. The midpoint is defined by $\frac{(spMax+spMin)}{2}$

Usage: TM

TE – Trajectory Move to Fully Extended Position (spMax)

Trajectory move to **spMax** - "Software Position Maximum" with user defined speed and acceleration.

Usage: TE

T1 – Preset Position 1 Trajectory Move

Initiate a trajectory move to **tPos1**

Usage: T1

T2 – Preset Position 2 Trajectory Move

Initiate a trajectory move to **tPos2**

Usage: T2

TD – Trajectory Move to the default Position

Trajectory move to **defPos** – "Default Position" with user defined speed and acceleration.

Usage: TD

TS – Stop Trajectory Move at Maximum Acceleration

Stop the current trajectory at the maximum acceleration rate.

Usage: TS

TK – Interrupt Current Trajectory Move

Halt the current trajectory motion at current position.

Usage: TK

PA – Set Absolute Target Position

Sending this command with a valid argument will set the current position demand register resulting in an immediate move to the new position. This command writes directly to the position demand register, bypassing the trajectory generator. This leads to a full acceleration, full speed move to the new target position which ignores maximum speed and maximum acceleration settings. Users will typically use this command for maximum dynamic performance in conjunction with an external trajectory generator. Use this command with caution.

Usage: PA arg1

arg1 Description: Absolute encoder position

arg1 Data Type: Unsigned Integer

arg1 Range: spMin to spMax

PO – Set Target Position with an Offset

Set the target position to the current position + the offset. The resultant position must be within the bounds defined by spMin and spMax. This command writes directly to the position demand register, bypassing the trajectory generator. This leads to a full acceleration, full speed move to the new target position and ignores maximum speed and maximum acceleration settings. Use this command with caution.

Usage: PO arg1

arg1 Description: Relative position offset

arg1 Data Type: Signed Integer

arg1 Range: spMin < Target Position < spMax

PC – Set PID Target to Current Position

Set the current position setpoint to the current absolute position value.

Usage: PC

SE – Set Relative Encoder Position

Sending this command with a valid argument will set the relative encoder counter to the commanded value.

Usage: SE arg1

arg1 Description: Relative Encoder Counter

arg1 Data Type: Signed 32-bit Integer

arg1 Range: 0 to 0xFFFFFFFF

LK – Lock or Unlock Serial Command-Line-Interface

Lock or unlock the serial command line interface.

Usage: LK arg1

arg1 Description: Command string

arg1 Data Type: String

arg1 Range: “lock”, “unlock”

FW – Show Firmware Installed on Active and Inactive Partitions

Show any firmware loaded onto the active and inactive flash memory partitions.

Usage: FW

SA – Swap Active Firmware Partition

Swap the active and inactive flash memory partitions. The CPU must be reset in order to run the new active partition.

Usage: SA arg1

arg1 Description: Passcode to execute command

arg1 Data Type: Unsigned Integer

arg1 Range: “321”

ZC – Run Actuator Calibration Utility

Run the actuator’s calibration utility. The actuator must be uncoupled from any mechanism or load and free to move through its entire mechanical stroke. If successful, the **ZC** command will overwrite certain configuration variables and save all of the configuration settings to non-volatile memory. The actuator must be restarted after this command is issued using the **ZR** command. Consult Ultra Motion engineering before issuing this command.

Usage: ZC arg1
arg1 Description: Passcode to execute command
arg1 Data Type: Unsigned Integer
arg1 Range: “321”

ZR – Reboot Actuator

Restart the actuator controller. Upon restart the actuator will load all configuration settings from non-volatile memory.

Usage: ZR arg1
arg1 Description: Passcode to execute command
arg1 Data Type: Unsigned Integer
arg1 Range: “321”

ZU – Run Serial Firmware Update Utility

The ZU command closes the serial CLI and runs the serial firmware update utility.

Usage: ZU arg1
arg1 Description: Passcode to execute command
arg1 Data Type: Unsigned Integer
arg1 Range: “321”

BR – Read Partition BSEQ Number

Show any firmware loaded onto the active and inactive flash memory partitions.

Usage: BR

BW – Write Partition BSEQ Number

The ZU command closes the serial CLI and runs the serial firmware update utility.

Usage: BW arg1
arg1 Description:
arg1 Data Type:
arg1 Range:

FC – Check for Inactive Partition Configuration Block

Show any firmware loaded onto the active and inactive flash memory partitions.

Usage: FC

BC – Check for Active Partition Configuration Block

Show any firmware loaded onto the active and inactive flash memory partitions.

Usage: BC

VC – Display Size of Configuration Block

Show any firmware loaded onto the active and inactive flash memory partitions.

Usage: VC

T-Series Configuration Variables

This section details configuration variables used by this firmware. These variables are stored within the controller's non-volatile memory and are configurable via the RS-485 serial interface. All configuration variables can be set with the **WV** command. For example, "wv **opMode** 1". To save all configuration variables to non-volatile memory, use the command "cw 321".

T-Series Configuration Variables Quick Reference Table

Variable	Description	Group	Default Value	Notes
opMode	Operating Mode Selector	Actuator Operation	0	Default: Serial CLI Mode
cntISrc	Control Source Selector	Actuator Operation	0	Default: CAN control
sMode	Serial Mode Selector	Actuator Operation	0	Default: CLI
op2Sel	Opto Input 2 Behavior	Actuator Operation	0	Default: No Function
spMin	Software Position Minimum	Actuator Travel Limits	1536	Units: Encoder Counts
spMax	Software Position Maximum	Actuator Travel Limits	2560	Units: Encoder Counts
maxCurr	Max Motor Current Demand	Motion Control	10000	
pkp	Position PID gain kp	Motion Control	1200.0	
pki	Position PID gain ki	Motion Control	2.0	
pkd	Position PID gain kd	Motion Control	500.0	
vkp	Velocity PI gain kp	Motion Control	240.0	
vki	Velocity PI gain ki	Motion Control	45.0	
maxSpeed	Max Speed for Trajectory Moves	Motion Control	15000	
accel	Accel/Decel for Trajectory Moves	Motion Control	200	
ovbOn	Overvoltage Dynamic Braking	Motion Control	1	Default: Enabled
rxData	Position Command Data Format	Motion Control	<>	Default: Position LSB, Position MSB
pMin	Minimum Position Command	Motion Control	0	
pMax	Maximum Position Command	Motion Control	65535	
pInvert	Invert Response to CAN/BSC Command	Motion Control	0	Default: Does Not Invert
vMin	Minimum Velocity Command	Motion Control	-30.0	
vMax	Maximum Velocity Command	Motion Control	30.0	
vZdb	Zero Velocity Deadband	Motion Control	0.01	
rMin	RC PWM Minimum Signal	Motion Control	100000	Units: 10 µs
rMax	RC PWM Maximum Signal	Motion Control	200000	Units: 10 µs
rInvert	RC PWM Invert	Motion Control	0	
inEna	Interpolation Enable Flag	Motion Control	1	Default: Enabled
rcpTO	RC PWM Command Timeout	RC PWM	1250	Units: Milliseconds
rcpIvl	RC PWM Interpolation Interval	RC PWM	50	Units: Milliseconds
defPos	Default Position	CAN Mode Operation	2048	Units: Encoder Counts
CANspd	Can Bus Baud Rate Selector	CAN Bus	0	Default: 1 Mbps
CANext	Can Bus Message Type	CAN Bus	1	Default: Extended Message Type (29-bit ID)
rxID	CAN Position Command RX ID	CAN Mode Operation	0x00000003	
rxMask	Mask for CAN RX ID	CAN Mode Operation	0x1FFFFFFF	
canIvl	Interpolation Interval	CAN Mode Operation	50	Units: Milliseconds
canTO	Position Command Timeout	CAN Mode Operation	1250	Units: Milliseconds
txEna	Telemetry Enable Flags	CAN Telemetry	0	Default: Disabled

Variable	Description	Group	Default Value	Notes
tx1Data	Telemetry Data Format 1	CAN Telemetry	GKPCD	
tx1ID	Telemetry Message Destination ID 1	CAN Telemetry	0x0000007F	
tx1Ivl	Telemetry Interval 1	CAN Telemetry	1000	Units: Milliseconds
tx2Data	Telemetry Data Format 2	CAN Telemetry	klmnpb	
tx2ID	Telemetry Message Destination ID 2	CAN Telemetry	0x000027F	
tx2Ivl	Telemetry Interval 2	CAN Telemetry	2500	Units: Milliseconds
tx3Data	Telemetry Data Format 3	CAN Telemetry	wxy	
tx3ID	Telemetry Message Destination ID 3	CAN Telemetry	0x000037F	
tx3Ivl	Telemetry Interval 3	CAN Telemetry	5000	Units: Milliseconds
evntID	Event Message Destination ID	CAN Event Messages	0x0000001F	
evntIvl	Event Message Min Interval	CAN Event Messages	4000	Units: Milliseconds
evntWrn	Warning Event Mask	CAN Event Messages	0x0000	
evntWrnC	Warning Event Mask - Clear on Broadcast	CAN Event Messages	0x3FFF	
evntMsc	Miscellaneous Event Mask	CAN Event Messages	0x0000	
IDbyte	Identification Byte	CAN Event Messages	255	
sBaud	Serial Baud Rate	RS485 Serial CLI	115200	Units: Baud
cliLock	CLI Lock on Startup	RS485 Serial CLI	1	Default: Unlocked
cliBanr	Startup Banner	RS485 Serial CLI	1	Default: Enabled
tPos1	Trajectory Position 1	RS485 Serial CLI	2048	Units: Encoder Counts
tPos2	Trajectory Position 2	RS485 Serial CLI	2049	Units: Encoder Counts
posLe	Position Less than Threshold	Status Bits	0	Units: Encoder Counts
posGr	Position Greater than Threshold	Status Bits	50175	Units: Encoder Counts
stplvl	Stopped Speed Threshold	Status Bits	10	Units: Milliseconds @ 0 velocity count
ovCurr	Motor Current Threshold	Status Bits	8200	
ovTemp	Over Temperature Threshold	Status Bits	60.0	Units: Celsius
unTemp	Under Temperature Threshold	Status Bits	-30.0	Units: Celsius
ovHumi	Over Humidity Threshold	Status Bits	85.0	Units: Percent
bscAddr	BSC Actuator Address	RS-485 BSC Protocol	128	
bscTO	BSC Position Command Timeout	RS-485 BSC Protocol	1250	Units: Milliseconds
bscIvl	BSC Interpolation Interval	RS-485 BSC Protocol	50	Units: Milliseconds
pSUact	Position Mode Startup Action Selector	Timeout/Startup Behavior	0	Default: Hold Position
pTOact	Position Mode Timeout Action Selector	Timeout/Startup Behavior	0	Default: Hold Last Position, Last Current Command
vSUact	Velocity Mode Startup Action Selector	Timeout/Startup Behavior	0	Default: Coast
vTOact	Velocity Mode Timeout Action Selector	Timeout/Startup Behavior	0	Default: Coast
tSUact	Torque Mode Startup Action Selector	Timeout/Startup Behavior	0	Default: Coast
tTOact	Torque Mode Timeout Action Selector	Timeout/Startup Behavior	0	Default: Coast

Table 20: Configuration variable reference list

opMode – Operating Mode

This setting determines whether the actuator is set to control absolute position, velocity, or torque, or use the RS-485 command line interface.

`opMode` = 0 enables motion control through the RS-485 serial Command Line Interface

`opMode` = 1 Position control mode

`opMode` = 2 Velocity control mode

`opMode` = 3 Torque control mode

Data Type: Integer

Valid Range: 0 to 3

Default: 0

cntlSrc – Control Source

This setting determines whether the actuator is set to receive commands via CAN 2.0B, RS-485 BSC, or RC PWM

`cntlSrc` = 0 CAN 2.0B

`cntlSrc` = 1 RS-485 Binary Serial Control (BSC)

`cntlSrc` = 2 RC PWM

Data Type: Integer

Valid Range: 0 to 3

Default: 0

sMode – Serial Mode

Serial mode defines whether the actuator's serial interface is using the binary serial control protocol or the command line interface when `opMode` is not equal to 0 and `cntlSrc` is not equal to 1.

`sMode` = 0 Command Line Interface (CLI)

`sMode` = 1 Binary Serial Control (BSC)

Data Type: Integer

Valid Range: 0 to 3

Default: 0

sBaud – Serial Baud Rate

This setting sets the Baud rate for serial communication. Serial is accessible in all control modes and is used for diagnostics, configuration, initial setup, and control with `opMode` = 0 and `opMode` = 2. Default Baud rate is 115200. Lower baud rates will be more tolerant to noise and crosstalk at the expense of data bandwidth.

Data Type: Integer

Valid Range: 4800 to 460800

Default: 115200

cliLock – Command Line Interface Lock

This setting determines if the CLI will startup locked or unlocked. A setting of 1 is unlocked on startup.

Data Type: Integer

Valid Range: 0 to 1

Default: 1

cliBanr – CLI Power-On Banner

This setting determines if the serial banner that includes the actuator's firmware and hardware revisions, serial number, etc. is transmitted on reset or power-cycle. A setting of 0 disables the banner on startup.

Data Type: Integer

Valid Range: 0 to 1

Default: 1

pkp, pki, and pkd – Position Control Loop PID Gains k_p , k_i , and k_d (Respectively)

These three values represent the gains for the proportional, integral, and derivative terms of the position PID control loop. Internal scaling of these gains is unique to this PID algorithm. Default factory values used represent typical stable gains. Adjusting these values is not recommended unless first discussed with Ultra Motion engineering.

Data Type: Float

Valid Range: -

Default: $pkp = 1200.0$, $pki = 2.000$, $pkd = 500.0$

vkp, vki– Velocity Control Loop PI Gains k_p , k_i (Respectively)

These two gains are used to modify the velocity control loop behavior in $opMode = 2$.

Internal scaling of these gains is unique to this PID algorithm. Default factory values used represent typical stable gains. Adjusting these values is not recommended unless first discussed with Ultra Motion engineering.

Data Type: Float

Valid Range: -

Default: $vkp = 250.0$, $vki = 45.0$

maxSpeed - Maximum Speed

This variable sets the top speed for trapezoidal profile trajectory moves. Note that it is possible to set **maxSpeed** higher than the maximum physical speed capability of the actuator. This can lead to control instability. The maximum speed that the actuator can achieve depends on the operating voltage, load, gear reduction, and other factors. Exceeding the maximum physical speed with the trajectory generator can result in integral windup, target position overshoot, and deviation from the defined trajectory profile.

Data Type: Integer

Valid Range: 1 to 10000000

Default: 15000

maxCurr - Max Motor Current Demand

This setting limits the motor current demand signal that commands the current loop, thereby limiting the force produced by the T-Series. The value represents a percentage of full current output where 32767 equals 100%. The relationship is linear with a slight offset do to unloaded running friction of the system. Contact Ultra Motion engineering for more detailed information. Care must be taken if exceeding the continuous torque rating of the actuator or if there's a potential to drive into a hardstop. Mechanical and thermal damage is possible in these situations.

Note that setting **maxCurr** low does not lead to a freely backdriveable actuator with the T-Series and will instead cause a dynamic braking behavior. The actuator's control word must be commanded to COAST, or the startup/timeout action set to COAST the motor in order to have the actuator backdrive freely.

Data Type: Integer

Valid Range: 0 to 32767

Default value: 10000 (approximate continuous limit at 25°C with natural convection)

accel - Acceleration and Deceleration Rate

This setting defines the acceleration and deceleration the T-Series will use in profile trajectory moves. Note: the acceleration and deceleration will be equal. Setting the acceleration value greater than what's physically achievable by the actuator for a given **maxCurr** setting and load can lead to control loop instability.

Data Type: Integer

Valid Range: 0 to 131071

Default: 200

ovbOn – Overvoltage Dynamic Braking

When set to 1 value, the actuator will dynamically brake when the sensed bridge voltage is greater than 55.0 VDC and will disable dynamic braking when the sensed voltage drops below 52.0 VDC. This helps to prevent the actuator's regenerated energy from spiking the bridge voltage to levels that can damage the actuator.

Setting this value to zero disables this dynamic braking functionality and should only be done if the power supply can handle the regenerated energy from a backdriving or deceleration event, or if power shunt electronics are used to prevent the bus voltage from spiking to an unsafe level. The overvoltage dynamic braking will not engage if the enable input is inactive.

Data Type: Integer

Valid Range: 0 to 1

Default: 1

CANspd – CAN Bus Baud Rate

This variable defines the baud rate of the CAN interface. There are 8 options designated by the integers 0 to 7.

0 = 1 Mbps, 1 = 500 Kbps, 2 = 250 Kbps, 3 = 125 Kbps, 4 = 100 Kbps, 5 = 50 Kbps, 6 = 20 Kbps, 7 = 10 Kbps

Data Type: Integer

Valid Range: 0 to 7

Default: 0

CANext – CAN Bus Message Type

This variable defines whether the T-Series is configured for CAN 2.0B standard message type (11-bit identifiers), or extended message type (29-bit identifiers). **CANext** = 0 configures standard message type (11-bit ID). A value of 1 in **CANext** sets the actuator to extended message type (29-bit ID).

Data Type: Integer

Valid Range: 0 to 1

Default: 1

rxID – CAN RX Identifier

rxID is the actuator's network address and is used along with **rxMask** to filter incoming CAN command messages. For acceptance, the message's identifier must match **rxID** where each corresponding bit in **rxMask** is set to '1'. If **CANext** = 0, only the lower 11 bits of **rxID** and **rxMask** are examined (standard message type). If **CANext** is 1, the lower 29 bits of **rxID** and **rxMask** are examined (extended message type).

Data Type: Integer

Valid Range: 0x00000000 to 0x1FFFFFFF

Default: 0x00000003

rxMask – Mask for CAN RX Identifier

rxMask is used along with **rxID** to filter incoming CAN command messages. For acceptance, the message's identifier must match **rxID** where each corresponding bit in **rxMask** is set to '1'. If a bit in **rxMask** is set to '0', the corresponding bit of an incoming can message is matched for acceptance regardless of its value. If **rxMask** is set to 0x00000000, CAN messages with any identifier will be accepted. If **rxMask** is set to 0x1FFFFFFF, only CAN messages with an identifier that exactly matches the relevant bits of **rxID** will be accepted. If **CANext** = 0, only the lower 11 bits of **rxID** and **rxMask** are examined (standard message type). If **CANext** is 1, the lower 29 bits of **rxID** and **rxMask** are examined (extended message type).

Data Type: Integer

Valid Range: 0 to 0x1FFFFFFF

Default: 0x1FFFFFFF

rxData – CAN/BSC Control Frame Data Format

This variable configures the data format of received position command messages. **rxData** is a string with a length from 1 to 8 characters. Each character in **rxData** designates a function for the corresponding data byte in the command message frame. The number of data bytes in the command message frame must match the number of characters in **rxData**. For example, if the master is sending out command messages with 8 data bytes, then **rxData** should be padded with 'X' or 'x' like this: "xxxx<>xx", where '<' is the position low byte, '>' is the position high byte, and 'x' means ignore this byte. The characters '(' and ')' can also be used to set the max motor current demand. '(' is max current demand low byte, and ')' is max current demand high byte. The acceptable range for the position command value is **pMin** to **pMax**. The acceptable range of the max current demand value is 0 to 32,767. The max current demand value is not updated if '(' and ')' characters are not included in **rxData**.

Data Type: String

Valid Range: <>()*Xx

Default: <>

pSUact – Position Control Startup Action

This variable defines the behavior of the actuator after start-up and before receiving a valid position command message.

pSUact = 0 Holds the last valid position with the last valid max motor current command

pSUact = 1 Executes a trajectory move to the default position defPos with the maxCurr setting

pSUact = 2 COAST the motor

pSUact = 3 Dynamic brake the motor

Data Type: Integer

Valid Range: 0 to 3

Default: 0

pTOact – Position Control Timeout Action

This variable defines the behavior of the actuator after not receiving a valid position command message within the timeout period for the active control source. The actuator must receive at least 1 valid position command message before a timeout can occur.

pTOact = 0 Holds the last valid position with the last valid max motor current command

pTOact = 1 Executes a trajectory move to the default position defPos with the last valid max motor current

pTOact = 2 COAST the motor

pTOact = 3 Dynamic brake the motor

pTOact = 4 Holds the last valid position with the configured maxCurr setting

pTOact = 5 Executes a trajectory move to the default position defPos with the configured maxCurr setting

Data Type: Integer

Valid Range: 0 to 5

Default: 0

vSUact – Velocity Control Startup Action

This variable defines the behavior of the actuator after start-up and before receiving a valid velocity command message.

vSUact = 0 COAST the motor

vSUact = 1 Dynamic brake the motor

vSUact = 2 Hold the current position in position control mode with maxCurr setting

Data Type: Integer

Valid Range: 0 to 2

Default: 0

vTOact – Velocity Control Timeout Action

This variable defines the behavior of the actuator after not receiving a valid velocity command message within the timeout period for the active control source. The actuator must receive at least 1 valid velocity command message before a timeout can occur.

vTOact = 0 COAST the motor

vTOact = 1 Dynamic brake the motor

vTOact = 2 Reset max current to configured maxCurr setting, stop immediately and hold position

vTOact = 3 Stop immediately and hold position

vTOact = 4 Reset max current to configured maxCurr setting, stop immediately and hold position

vTOact = 5 Decelerate to a stop and hold the current position in position control mode

Data Type: Integer

Valid Range: 0 to 5

Default: 0

tSUact – Torque Control Startup Action

This variable defines the behavior of the actuator after start-up and before receiving a valid torque command message.

tSUact = 0 COAST the motor

tSUact = 1 Dynamic brake the motor

tSUact = 2 Hold the current position in position control mode with maxCurr setting

Data Type: Integer

Valid Range: 0 to 2

Default: 0

tTOact – Torque Control Timeout Action

This variable defines the behavior of the actuator after not receiving a valid torque command message within the timeout period for the active control source. The actuator must receive at least 1 valid torque command message before a timeout can occur.

tTOact = 0 COAST the motor

tTOact = 1 Dynamic brake the motor

tTOact = 2 Holds the current position in position control mode using the last valid torque command

Data Type: Integer

Valid Range: 0 to 2

Default: 0

canTO – CAN Command Timeout Period

This variable defines the CAN command message timeout period in increments of 1 ms.

Data Type: Integer

Valid Range: 0 to 65535

Default: 1250

bscTO – BSC Command Timeout Period

This variable defines the RS-485 Binary Serial Control Protocol command message timeout period in increments of 1 ms.

Data Type: Integer

Valid Range: 0 to 65535

Default: 1250

rcpTO – RC PWM Command Timeout Period

This variable defines the RC PWM command message timeout period in increments of 1 ms.

Data Type: Integer

Valid Range: 0 to 65535

Default: 1250

pMin, pMax – Position Command Range

This variable defines the valid command range that will be sent to the actuator via command messages. The value of pMin/pMax will be mapped to the actuator's travel range spMin/spMax in position control modes, and vMin/vMax in velocity control modes. If plnvr is set to 1, pMax will map to spMin/vMin, and pMin will map to spMax/vMax.

Data Type: Integer

Valid Range: 0 to 65535

Default: pMin = 0, pMax = 65535

pInvert – Inversion of Response to command Frame

Inverts the response of the actuator with respect to the command range pMin/pMax in BSC and CAN control modes. A value of 0 does not invert. A value of 1 inverts.

Data Type: Integer

Valid Range: 0 to 1

Default: 0

rInvert – Inversion of Response to RC PWM command

Inverts the response of the actuator with respect to the RC PWM command. A value of 0 does not invert. A value of 1 inverts.

Data Type: Integer

Valid Range: 0 to 1

Default: 0

rMin – Minimum RC PWM command

This variable defines the minimum RC PWM command.

Data Type: Unsigned 32 bit integer

Valid Range: 90000 to 110000

Default: 100000

rMax – Maximum RC PWM command

This variable defines the maximum RC PWM command.

Data Type: Unsigned 32 bit integer

Valid Range: 190000 to 210000

Default: 200000

spMin, spMax - Software Position Minimum, Software Position Maximum

These two configuration variables set the minimum and maximum allowable position command in the T-Series actuator. The values are expressed in encoder counts.

Data Type: Integer

Valid Range: 1 to 4095

Default: 1536 and 2560

vMin, vMax – Velocity Command Range

This variable defines the valid command range for velocity commands in units of RPM

Data Type: FLOAT32

Valid Range: -100.0 to 100.0

Default: vMin = -30.0, vMax = 30.0

vZdb – Velocity Command Zero Deadband

This variable defines the window around the commanded zero velocity point that will be interpreted by the controller as a command of zero velocity. Units are in RPM

Data Type: FLOAT32

Valid Range: -100.0 to 100.0

Default: vZdb = 0.01

op2Sel – Optically Isolated Input 2 Function Select

This variable defines the functionality of optically isolated digital input 2 (IN2+/IN2-)

op2Sel = 0 No Function

op2Sel = 1 Enable Input – active input enables motor

op2Sel = 2 Disable Input – active input disables motor

Data Type: Integer

Valid Range: 0 to 2

Default: 0

defPos – Default Position

defPos is the absolute encoder value of the default position. The actuator can move to defPos upon startup if no valid position command messages have been received, or in the event of a position command message not being received in the command timeout period. The behavior of the actuator in these situations is defined by the startup and timeout action variables.

Data Type: Integer

Valid Range: spMin to spMax

Default: 2048

canIvl – CAN Interpolation Interval

This variable defines the interpolation period in units of 1 ms for the received CAN commands. The default value is 50 (50 milliseconds), which is equivalent to a 20 Hz position update rate.

Data Type: Integer

Valid Range: 1 to 65535

Default: 50

bscIvl –BSC Interpolation Interval

This variable defines the interpolation period in units of 1 ms for the received BSC commands. The default value is 50 (50 milliseconds), which is equivalent to a 20 Hz position update rate.

Data Type: Integer

Valid Range: 1 to 65535

Default: 50

rcpIvl – RC PWM Interpolation Interval

This variable defines the interpolation period in units of 1 ms for the received RC PWM commands. The default value is 50 (50 milliseconds), which is equivalent to a 20 Hz position update rate.

Data Type: Integer

Valid Range: 1 to 65535

Default: 50

inEna –Interpolation Enable Flag

Setting inEna to 0 disables position interpolation, setting inEna to 1 enables position interpolation.

Data Type: Integer

Valid Range: 0 to 1

Default: 1

txEna – Telemetry Enable Flags

Setting **txEna** to 0 disables all telemetry messages from the actuator. Setting to a value from 1 to 7 (0b001 to 0b111) will activate the corresponding telemetry message. For example, **txEna**=3 (0b011) would enable the broadcasting of **tx1Data** and **tx2Data**

Data Type: Integer

Valid Range: 0 to 7

Default: 0

txNID – CAN Telemetry Message ID

These variables are the 29-bit or 11-bit CAN 2.0B identifiers for the telemetry messages. They represent the destination address of each telemetry message. If **CANext**=0, only the lower 11 bits are used.

Data Type: Integer

Valid Range: 0 to 0x1FFFFFFF

Default: **tx1ID** = 0x0000007F, **tx2ID** = 0x000027F, **tx3ID** = 0x000037F

txNData – Telemetry Data Selection

The specific telemetry data bytes to be broadcast are selected with these configuration variables. The string may have a length from 1 to 8 characters. Each character in **txNData** designates one variable in the telemetry message. Variables may be one or more bytes in length each. The total number of data bytes cannot exceed 8 for a CAN 2.0B telemetry message.

Data Type: String

Valid Range: All chars in Table 11

Default: **tx1Data** = GKPCD, **tx2Data** = klmnbp, **tx3Data** = wxy

txNlvl – Telemetry Interval

The telemetry intervals define the broadcast period of each telemetry message from the actuator in milliseconds. Care must be taken to ensure the CAN bus is not overloaded by too rapid a transmission rate.

Data Type: Integer

Valid Range: 2 to 10000

Default: **tx1lvl** = 1000, **tx2lvl** = 2500, **tx3lvl** = 5000

bscAddr – BSC Address

bscAddr is the actuator's RS-485 serial address and is used to filter incoming BSC command messages. For acceptance, the message's Actuator ID byte must match **bscAddr**

Data Type: Integer

Valid Range: 1 to 255

Default: 128

ovCurr – Motor Current Threshold

Defines the motor current threshold that causes the "Over Current" status bit to go high (when Motor Current > **ovCurr**). This variable is only used to define the behavior of the related status bit.

Data Type: Integer

Valid Range: 0 to 32766

Default: 8200

stpLvl – Stopped Speed Threshold

Defines the number of milliseconds with a 0 velocity count before the actuator is considered to be “stopped” and the “stopped” status bit is set high. This variable is only used to define the behavior of the related status bit.

Data Type: Integer
Valid Range: 1 to 65535
Default: 10

ovTemp – Over Temperature Threshold

Defines the temperature threshold that causes the “Over Temperature” status bit to go active (when Temp > ovTemp). This variable is only used to define the behavior of the related status bit. Units are in °C

Data Type: Float
Valid Range: -50.0 to 149.0
Default: 60.0

unTemp – Under Temperature Threshold

Defines the temperature threshold that causes the “Under Temperature” status bit to go active (when Temp < unTemp). This variable is only used to define the behavior of the related status bit. Units are in °C

Data Type: Float
Valid Range: -49.0 to 150.0
Default: -30.0

ovHumi – Over Humidity Threshold

Defines the relative humidity threshold that causes the “Over Humidity” status bit to go active (when RH > ovHumi). This variable is only used to define the behavior of the related status bit. Units are in %

Data Type: Float
Valid Range: 0 to 100.0
Default: 85.0

tPos1 – Trajectory Position 1

Default position that can be used to command trajectory moves “T1”

Data Type: Integer
Valid Range: spMin to spMax
Default: 2048

tPos2 – Trajectory Position 2

Default position that can be used to command trajectory moves “T2”

Data Type: Integer
Valid Range: spMin to spMax
Default: 2049

posLe – Position less than threshold

Used as an additional position threshold that controls the posLe status bit. The status bit will go high if absolute position is less than the posLe threshold. This variable is only used to define the behavior of the related status bit.

Data Type: Integer
Valid Range: 0 to 4095
Default: 1024

posGr – Position greater than threshold

Used as an additional position threshold that controls the `posGr` status bit. The status bit will go high if absolute position is greater than the `posGr` threshold. This variable is only used to define the behavior of the related status bit.

Data Type: Integer
Valid Range: 0 to 4095
Default: 2048

evntID – Event ID

This is the CAN ID that will be used for asynchronous event messages. If `CANext=0`, only the lower 11 bits are used.

Data Type: Integer
Valid Range: 0 to 0x1FFFFFFF
Default: 0x0000001F

evntMsc– Miscellaneous Event Mask

Bit mask for miscellaneous event enable

Data Type: Integer
Valid Range: 0 to 65535
Default: 0x0000

evntWrn– Warning Event Mask

Bit mask for warning event enable

Data Type: Integer
Valid Range: 0 to 16383
Default: 0x0000

evntWrnC– Warning Event Mask – Clear on Broadcast

Bit mask for warning bit is cleared on event message broadcast.

Data Type: Integer
Valid Range: 0 to 16383
Default: 0x3FFF

evntIvl – Minimum Event Message Interval

The minimum delay between successive warning event messages for the same warning bit in milliseconds.

Data Type: Integer
Valid Range: 100 to 65535
Default: 4000

IDbyte – Identification Byte

An 8-bit value that can be used to identify actuators on the CAN bus. The byte is used in asynchronous event messages and can be added to telemetry messages. This can be useful for identification if multiple actuators are sending messages to a single address.

Data Type: Integer
Valid Range: 0 to 255
Default: 255

General Actuator Design Guidelines

Below are some common pitfalls and guidelines to consider when using Ultra Motion servo actuators. Contact Ultra Motion Engineering for advice regarding the successful integration of the T-Series into an application.

Mechanical vs. Software Position Limits

Users must configure the software defined rotational limits to ensure compatibility in their mechanical system. There should be clearance to any hardstops in the system to accommodate for mechanical tolerances and control system overshoot.

Overloading and Mechanical Stops

When using an actuator with a BLDC motor, take care that you do not run the actuator into a physical “hardstop”, such as a machine element or any physical body which the actuator impacts on its travel and which disallows movement. If this happens, the control system will attempt to obey position commands by powering through the physical block, setting the motor current command to the maximum. This can result in the motor over-heating and burning out, especially with higher maximum motor current settings or extended overload durations. Mechanical damage is possible with high maximum current settings and repeated overload events.

The exact continuous thermal limit is difficult to accurately predict due to the unique heat transfer environments from application to application. A **maxCurr** setting of 10,000 is a rough approximation of the continuous capability of the actuator assuming a 25°C ambient environment and natural convection.

Regenerative Energy

The actuator can push regenerated energy back into the power supply during deceleration or backdriving events. The power supply being used in the application must be sized adequately to prevent unwanted bus spikes during use. Regulated power supplies are particularly susceptible to bus spikes due to regenerated energy. Consider adding shunt electronics to the power system if significant amounts of regenerative energy are expected, a regulated power supply is being used, or if the bus is shared with other sensitive electronics.

Solid Particle and Liquid Exposure

The T-Series includes a highly engineered shaft seal, and O-ring seals on all static mechanical joints to provide robust IP67 sealing. Care should be taken to avoid the ingress of solid particles into an unmated connector by always using a connector cap when unmated.

Hot Plugging

The actuator should not be connected/disconnected to the power supply while the power supply is active.

Axial Loads

Ideally, the actuator is subjected to pure torque loads and axial loads are transmitted elsewhere. In cases where this is not practical, the actuator is capable of handling axial loads up to ~47 lbf in tension/compression. Loads above 16 lbf in tension start to create axial movement in the output shaft up to a maximum ~0.05”.

Absolute axial limits where brinelling can occur:

Compression: 94 lbf

Tension: 220 lbf

Absolute Single Turn Position Encoder Roll

When operating the actuator in single-turn absolute position mode, the user must prevent motion near the absolute position limits at 0 and 4095. Approaching either endpoint increases the risk of encoder roll-over, which can cause discontinuities in reported position and may lead to unexpected motion, control instability, and failure. The user should mechanically limit the actuator’s travel using external hardstops with the center of the allowable mechanical stroke aligning with absolute position 2048. The full range of actuator motion must remain well away from positions 0 and 4095 at all times.

T-Series Firmware Update

The T-Series can be updated to new firmware in the field by using Ultra Motion's Firmware update utility. `fw_loader_v1_0.exe` is a lightweight Windows console application for loading firmware onto Ultra Motion T-Series actuators over a serial (COM) port. It reads an Intel HEX formatted file and writes to the flash memory of the actuator's control unit using a proprietary serial protocol. The actuator must be running its firmware update utility to interact with this program. Windows 7 or later is required.

Firmware Updating Procedure

The firmware updating utility and the most up to date firmware file can be received from Ultra Motion Engineering. The T-Series actuator must receive the "ZU321" command via serial to begin running its firmware update utility. After navigating to the `fw_loader_v1_0.exe` directory, the following command line instruction can be sent:

```
fw_loader_v1_0.exe -f <file.hex> -p <COMx> [-b <baud>] [-q] [-n]
```

Firmware Update Command Line Options

Option Code	Description	Optionality
-f, --file	Path to the Intel HEX file	Required
-p, --port	COM port (e.g. COM3)	Required
-b, --baud	Baud rate (4800 to 460800)	Optional, default: 115200
-q, --quiet	Verbose output OFF	Optional
-n, --noexec	Skip automatic jump to application	Optional

Table 21: Firmware update command line options

After successfully loading the firmware, the actuator may be sent the "CC321" command to copy the previous calibration and configuration variables to the updated firmware.

Example:

```
C:\>fw_loader_v1_0.exe -p COM3 -b 115200 -f firmware.hex
```

Firmware Update Tool Verification

To verify the integrity of the `fw_loader_v1_0.exe` file, compare its SHA-256 hash with the value received from Ultra Motion Engineering or available on our website at the following location:

https://www.ultramotion.com/downloads/SHA256_list.html

The following commands can be used to compute the local hash:

- Using Command Prompt: `certutil -hashfile fw_loader_v1_0.exe SHA256`
- Using PowerShell: `Get-FileHash .\fw_loader_v1_0.exe -Algorithm SHA256`

Verify the output of the local hash and the hash received from Ultra Motion engineering or Ultramotion.com are equal
e.g: `0xcc756ef5ec8dbd215508a37c9f729d621cca76e7b4dd9ad824ce05378c79b58c`

Unit Conversions

Definitions

Variable	Definition	Units	Note
N	Sensor Counts	–	N denotes sensor counts for the specific data type in question
P	Position	<i>degrees</i>	Rotational position with respect to spMin. Clockwise is positive.
a	Rotational Acceleration	<i>deg/s²</i>	Rotational acceleration, which is always positive (scalar) for setting 'accel'
v	Velocity	<i>deg/s</i>	Rotational velocity of the actuator shaft
V_{bus}	Bus Voltage	V	Supply voltage to the actuator (before bridge enable switch)
T	Temperature	$^{\circ}C$	Internal temperature, as measured at the controller PCB
H_R	Relative Humidity	%	Relative humidity, actual values will be between 0 and 100%
I_{motor}	Motor Current	A	Current through the BLDC motor

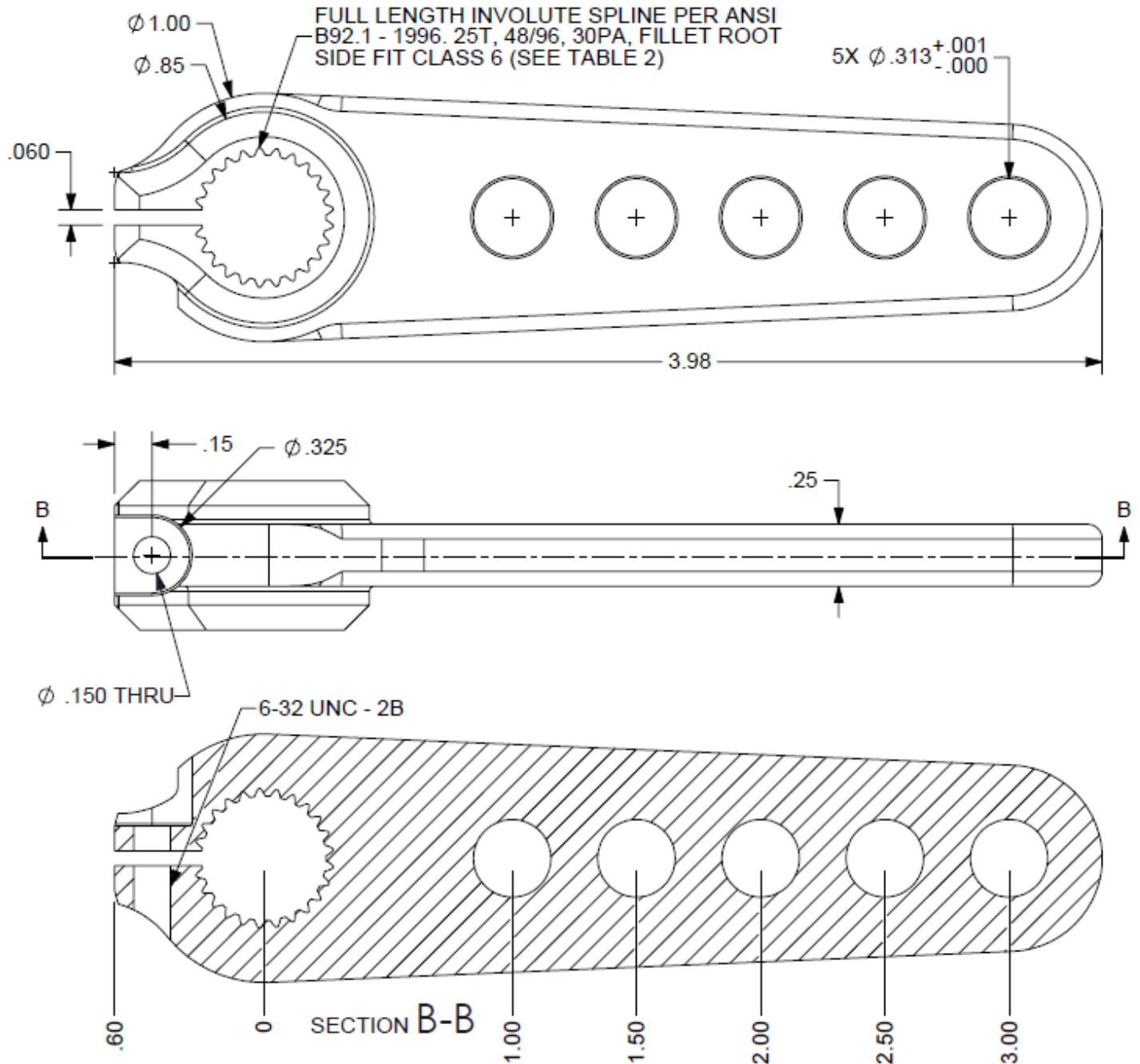
Table 22: Unit conversion definitions

Equations

Variable	Conversion to physical units
Position	$P = \frac{(N - spMin) * 360}{4096}$
Speed	$v = maxSpeed * 1000 * \frac{360}{65535 * 4096}$
Accel	$a = accel * 1000^2 * \frac{360}{65535 * 4096}$
Bus Voltage	$V_{bus} = VinSenADC * \frac{55.0}{16380}$
Motor Current	$I_{motor} = motorCurrent * \frac{20.0}{32752}$
8-bit Temperature	$Temperature (^{\circ}C) = Feedback - 50$

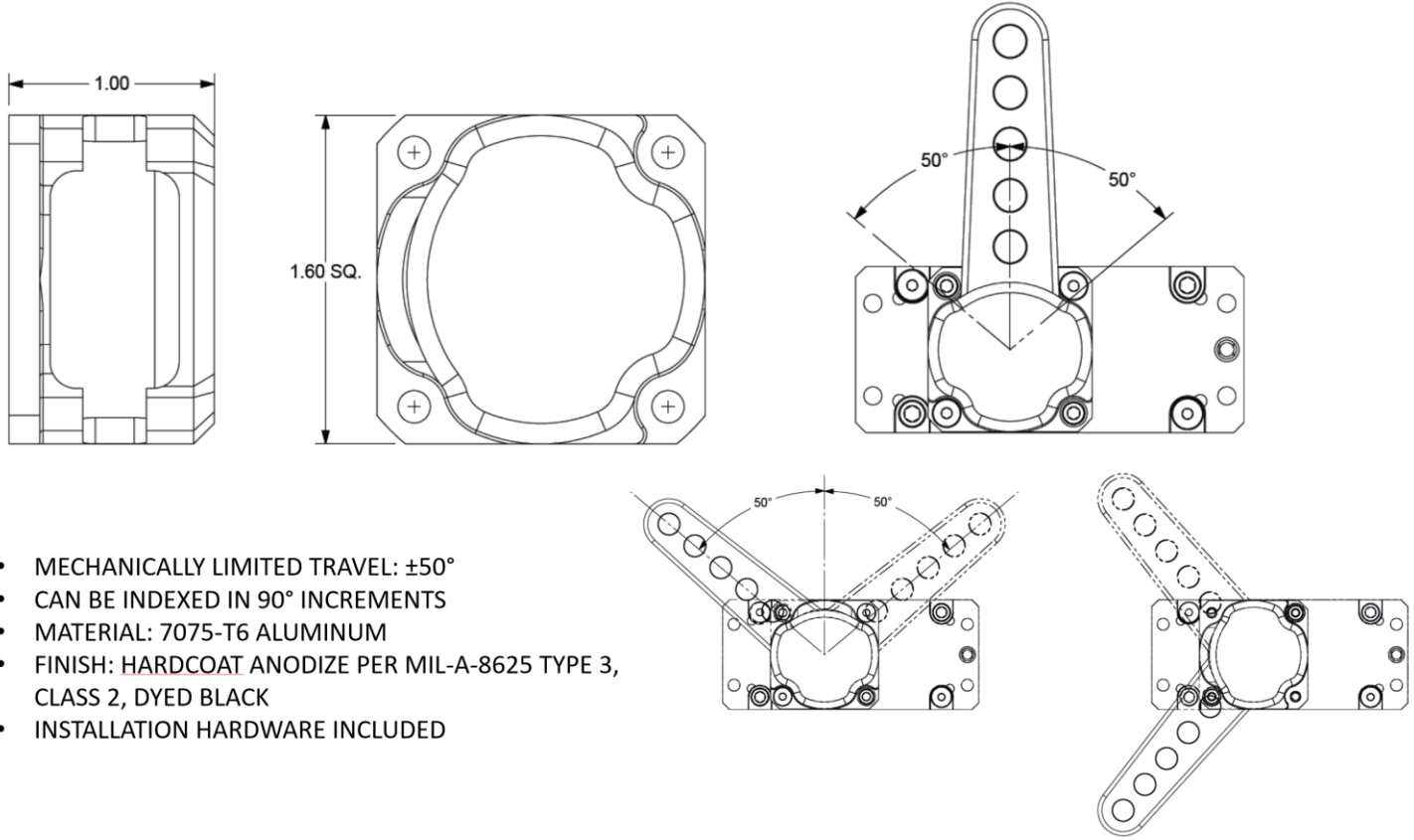
Table 23: Unit conversion equations

Control Arm



- MATERIAL: 6061-T6 ALUMINUM
- FINISH: HARDCOAT ANODIZE PER MIL-A-8625 TYPE 3, CLASS 1
- INSTALLATION HARDWARE INCLUDED

Rotational Hardstop



Ultra Motion Power Supplies

The PS-1X0A and PS-1X2A power supplies

CBL-T3-DEV Cable Set

Development cable set, 10 foot length PVC cable terminated with 6" flying leads, D38999/26FB35SN with M85049/38-11N strain relief backshell

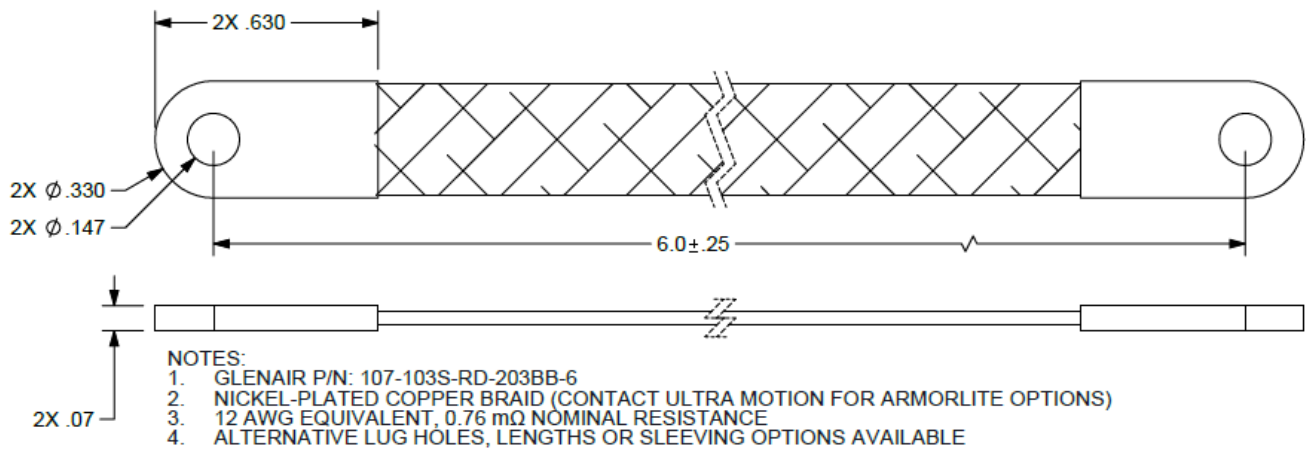
CBL-T3U-1M Cable Set

Series T3U power & communication cable, 1-meter length, 8-socket connector with locking sleeve, neoprene jacket, no shield, unterminated flying leads on power & control end

CBL-T3U-5M Cable Set

Series T3U power & communication cable, 5-meter length, 8-socket connector with locking sleeve, neoprene jacket, no shield, unterminated flying leads on power & control end

Grounding Strap



Contact Information

If you have any questions about the T-Series or any of our other products, contact us by one of the following methods:



INQUIRY

Leave a web inquiry (to be replied to within one business day):
ultramotion.com/contact



LIVE CHAT

Live Chat directly with one of our engineers:
ultramotion.com



EMAIL

Email (to be replied to within one business day):
info@ultramotion.com



CALL

PH: 888-321-9178
Fax: 631-298-6593



ADDRESS

Ultra Motion
22355 CR 48, #21
Cutchogue, NY 11935



HOURS

Our Business Hours:
Monday-Friday
9AM – 5PM EST

