

Mercury technical manual

September
2017

v.1

Mercury technical manual

[Mercury technical manual](#)

[1. Introduction](#)

[2. Connection details](#)

[2.1 Pin assignments](#)

[2.2 Connecting multiple units](#)

[2.3 Mercury Link](#)

[2.4 Mercury power hub](#)

[3. Communication protocol](#)

[3.1 Overview](#)

[3.2 Protocol](#)

[3.3 Request packet](#)

[3.3.1 Checksum description](#)

[3.3.2 Unique servo ID](#)

[3.4 Acknowledgement packet](#)

[3.4.1 Error description](#)

[4. Control registers](#)

[4.1 Register table](#)

[4.2 Register limits](#)

[4.3 Register details](#)

[4.3.1 Control enable](#)

[4.3.2 Baud rates](#)

[4.3.3 Acknowledgement packet response time](#)

[4.3.4 Operating modes table](#)

[4.3.5 Mercury position modes](#)

[Single and multi-turn position modes](#)

[Clockwise \(CW\) and counterclockwise \(CCW\) angle limits](#)

[Home position offset](#)

[PID position control](#)

[Feedforward control](#)

[Angular velocity profile](#)

[Acceleration profile](#)

[Dead zone](#)

[Further reading](#)

[4.3.6 Mercury continuous rotation mode](#)

[Target angular velocity](#)

[Acceleration profile](#)

[PI velocity control](#)

[4.3.7 Torque mode](#)

[Acceleration profile](#)

[4.3.8 Is moving](#)

[5. Operations](#)

[5.1 Operations table](#)

[5.2 Operation details](#)

[5.2.1 PING](#)

[5.2.2 READ_DIRECT](#)

[5.2.3 WRITE_DIRECT](#)

[5.2.4 WRITE_SHADOW](#)

[5.2.5 COMMIT_SHADOW](#)

[5.2.6 WRITE_COMPOSITE](#)

[5.2.7 RESET](#)

1. Introduction

This document provides detailed technical information on the Mercury range of digital servos from Robot Articulation.

2. Connection details

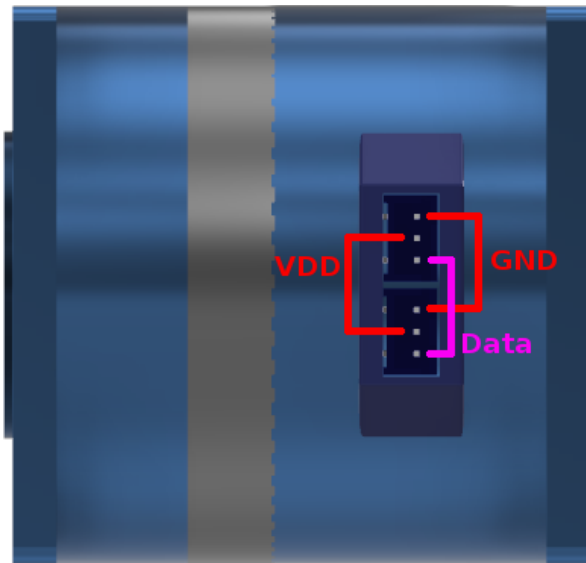
2.1 Pin assignments

Each Mercury servo has two three-pin connectors. The two connectors are wired in parallel internally. The pinouts are as follows:

PIN1: GND

PIN2: VDD (15-24V DC)

PIN3: Data (TTL levels)



2.2 Connecting multiple units

Multiple Mercury digital servos may be connected in parallel. Each servo must be configured to have its own unique address.

2.3 Mercury Link

In order to control (or configure) Mercury digital servos from a PC, a Mercury Link unit should be used. The Mercury-Link unit contains:

- a USB 2.0 type A connector which connects to a USB port on a PC
- a PCB-mounted terminal block for connection to an external 15-24V DC power source
- a single XH-series header to connect to a Mercury servo
- reverse voltage and surge protection

Please see the Mercury Link document for more details.

2.4 Mercury power hub

A Mercury digital servo's two three-pin connectors have a maximum current rating of 3A. This maximum current rating limits the number of Mercury servos that can be directly daisy-chained in a network of Mercury servos.

To overcome this limitation, a Mercury Power Hub should be used.

The Mercury Power Hub has the following capabilities:

- contains an in-build Mercury Link to interface between a PC's USB port and a network of Mercury servos
- accepts a single USB 2.0 type A connection from a PC
- accepts an external DC power source of between 15 and 24V
- provides 5 (in-parallel) XH-series headers to connect directly to Mercury servos.
- can supply a maximum of 15A to the connected Mercury servos - assuming that the external DC power source can supply this amount of current.
- reverse voltage and surge protection

Please see the Mercury Power Hub document for more details.

3. Communication protocol

3.1 Overview

The Mercury-Link communicates with the connected Mercury digital servos, by sending and receiving a series of data packets. There are two types of packets - Request (RQT) and Acknowledgement (ACK) packets. Request packets are sent from the Mercury-Link unit to the Mercury servos. Acknowledgement packets are sent from Mercury servos to the Mercury-Link unit.

3.2 Protocol

Communication between Mercury servos and the Mercury-Link unit is performed using an asynchronous serial protocol of 8 bits, with 1 stop bit and no parity.

See the [table](#) (below) for details on supported baud rates.

3.3 Request packet

The structure of the Request Packet is as follows:

Start bytes		Unique servo ID / Broadcast ID (0xFE)	Length (ParamN + 2)	Instruction	Param1	...	ParamN	Checksum (see description below)
0xFF	0xFF	ID	Length	Instruction	Param1	...	ParamN	Checksum

3.3.1 Checksum description

The checksum is calculated as follows:

Check Sum = ! (ID + Length + Instruction + Param1 + ... ParamN)

If the calculated value is greater than 255 (0xFF), the lower byte is defined as the checksum value. '!' represents the NOT logic operation.

3.3.2 Unique servo ID

A servo ID in the range of 0→252 may be specified in the request packet.

Alternatively, a broadcast ID (0xFE) may be specified where applicable. This has the effect of sending the instruction packet to all Mercury servos on the network. Please note that no acknowledgement packet is returned when the broadcast ID is specified.



3.4 Acknowledgement packet

The Acknowledgement Packet is sent by a Mercury servo in response to a Request Packet. The structure of the Acknowledgement packet is as follows:

Start bytes		Unique servo ID	Length (ParamN + 2)	Error (see description below)	Param1	...	ParamN	Checksum (see description)
0xFF	0xFF	ID	Length	Error	Param1	...	ParamN	Checksum

3.4.1 Error description

A non-zero error byte value indicates that an error condition has been encountered. Each bit in the Error byte indicates a specific error.

Bit	Bit Name	Description
0	Input Voltage Error	Set if the input voltage exceeds the minimum or maximum voltages specified in the Register Table.  The Mercury servo will remove all power from the motor when the input voltage is outside of the specified limits.
1	Angle Limit Error	Set if the Goal Position exceeds the CW and CCW limits specified in the Register Table.
2	Overheating Error	Set if the internal temperature of the Mercury Servo exceeds the limit specified in the Register Table.  The Mercury servo will remove all power from the motor in the event of an overheating error.
3	Instruction Range Error	Set if the request instruction exceeds the defined range of instructions

4	Checksum Error	Set if the Checksum specified in the Request Packet is incorrect.
5	Overload Error	Set if the torque required to perform the request exceeds the maximum torque specified in the Register Table.
6	Instruction Error	Set if an undefined instruction is specified, or an action request is specified with a REG_WRITE instruction.
7	0	Not used.

4. Control registers

4.1 Register table

Non-volatile registers						
Address	Size	Description	R/W	Range	Default Value	Notes
0 (0x00)	1	Model number minor	R		E.g. 0x01 for model minor version 1	
1 (0x01)	1	Model number major	R		E.g. 30 (0x1E) for Mercury T30	
2 (0x02)	1	Firmware version	R			
3 (0x03)	1	ID	RW	0→252	1 (0x01)	Note that the Mercury Link reserves ID 253 (0xFD)
4 (0x04)	1	Baud rate	RW	1→254	1 (0x01) (1,000,000 BPS)	See table below
5 (0x05)	1	Acknowledgement packet response time	RW	0→254	254 (0xFE)	See details below
6 (0x06)	1	Operating mode	RW	0-3	2 (single-turn mode)	See table below
7 (0x07)	2	Clockwise (CW) angle limit	RW	0→4095 (- π to π)	0 (- π)	See servo output position details below
9 (0x09)	2	Counter-clockwise (CCW) angle limit	RW	0→4095 (- π to π)	4,095 (0xFFFF) (π)	
11 (0x0B)	1	Upper temperature limit	RW	0→254	55 (0x37)	Value in °C
12 (0x0C)	1	Lower input voltage limit	RW	0→254	150 (0x96)	Voltage = register value / 10.

						200 ⇔ 20V
13 (0x0D)	1	Upper input voltage limit	RW	0→254	240 (0xF0)	As per above. 250 ⇔ 25V
14 (0x0E)	2	Torque limit	RW	T30: 0→1,800 T50: 0→3,000 T65: 0→4,000	T30: 600 (0x258) T50: 1,000 (0x3E8) T65: 1,500 (0x5DC)	 To avoid damage to your Mercury servo, do not routinely exceed the rated torque figure.
16 (0x10)	2	Angular velocity limit	RW	0 → 5000	5000 (0x1388)	5000 equates to 5000 milli-rad/s.
18 (0x12)	2	Acceleration limit	RW	0→100	0	0 indicates the maximum possible acceleration.
20 (0x14)	2	Home position offset	RW	Single-turn: -1535→1535 Multi-turn: -32,767→ 32,767	0	See details below.
22 (0x16)		Reserved				
23 (0x17)	2	Moving threshold	RW	0 → 2,000	200 (0x00C8)	Moving velocity threshold in milli-rad/s. See details below.
25 (0x19)	1	Reserved				
...		Reserved				
47 (0x2F)	1	Reserved				

Volatile registers						
Address	Size	Description	R/W	Range	Default Value	Notes
48 (0x30)	1	Control enable	RW	0→1	0	See details below.
49 (0x31)	1	Reserved				
50 (0x32)	2	Position derivative gain (Kd)	RW	0→16,383	0	See PID position details below.
52 (0x34)	2	Position integral gain (Ki)	RW	0→16,383	0	
54 (0x36)	2	Position proportional gain (Kp)	RW	0→16,383	1000 (0x3E8)	
56 (0x38)	2	Position feedforward gain 1	RW	0→16,383	0	See feedforward details below.
58 (0x3A)	2	Position feedforward gain 2	RW	0→16,383	0	
60 (0x3C)	2	Velocity integral gain	RW	0→16,383	1500 (0x5DC)	See PI velocity details below.
62 (0x3E)	2	Velocity proportional gain	RW	0→16,383	200 (0xC8)	
64 (0x40)	2	Reserved				
66 (0x42)	2	Reserved				
68 (0x44)	2	Angular velocity profile	RW	0→Angular velocity limit		See the angular velocity profile details below.
70 (0x46)	2	Acceleration profile	RW	0→Acceleration limit		See acceleration profile details below.
72 (0x48)	2	Dead zone	RW	0→1024	0	Position control only. See dead zone details below.
74 (0x4A)	2	Reserved				
76 (0x4C)	2	Reserved				
78 (0x4E)	2	Target position	RW	Single-turn: 0→4095 (-π to π)	Value from actual position register.	See servo output position section below

				Multi-turn: -32,767→32,767		
80 (0x50)	2	Target angular velocity	RW	0→10000	0	Target angular velocity in milli-rad/s. See details below. Only used in continuous rotation mode.
82 (0x52)	2	Target torque	RW	0→value from Torque limit register.	0	Model dependant target torque in units of 10 mNm. See torque mode details below.
84 (0x54)	2	Actual position	R	Single-turn: 0→4095 (-π to π) Multi-turn: -32,767→32,767	-	See servo output position section below
86 (0x56)	2	Actual angular velocity	R			Angular velocity in milli-rad/s. e.g. 4400 milli-rad/s ⇔ 4.4 rad/s ⇔ 42 rpm
88 (0x58)	2	Actual torque	R			Torque in units of 10 mNm 100 ⇔ 1Nm
90 (0x5A)	1	Actual voltage	R			Voltage = register value / 10. e.g. 200 ⇔ 20V
91 (0x5B)	1	Actual current	R			1 ⇔ 1mA
92 (0x5C)	1	Actual temperature	R			Value in °C
93 (0x5D)	1	Reserved				
94 (0x5E)	1	Moving	R	0→1		Indicates if the servo is moving. See details below.
95 (0x5F)	1	Reserved				

96 (0x60)		Registered instruction	RW	0→1	0	1 = pending shadow write operation.
97 (0x61)		Reserved...				
...		Reserved				
200 (0xC8)		Reserved				

4.2 Register limits

Each writable register has an associated minimum and maximum value. Write instructions made outside of valid ranges will return an out-of-range status error, and no update will take place.

The following table details the data range for each register. 16 bit registers must be written atomically within the same instruction packet.

Write address	Description	Min value	Max value
3 (0x03)	ID	0	252 (0xFC)
4 (0x04)	Baud rate	1	254 (0xFE)
5 (0x05)	Acknowledgement packet response time	0	254 (0xFE)
6 (0x06)	Operating mode	0	3
7 (0x07)	Clockwise (CW) angle limit	0	4095 (0xFFFF)
9 (0x09)	Counter-clockwise (CCW) angle limit	0	4095 (0xFFFF)
11 (0x0B)	Upper temperature limit	0	55 (0x37)
12 (0x0C)	Lower input voltage limit	150 (0x64)	254 (0xFE)
13 (0x0D)	Upper input voltage limit	150 (0x64)	254 (0xFE)
14 (0x0E)	Torque limit	0	T30: 1,800 (0x708) T50: 3,000 (0xBB8) T65: 4,000 (0xFA0)
16 (0x10)	Angular velocity limit	0	5000 (0x1388)
18 (0x12)	Acceleration limit	0	100 (0x64)
20 (0x14)	Home position offset	Single-turn:	Single-turn:

		-1535 Multi-turn: -32,767	1535 Multi-turn: 32,767
23 (0x17)	Moving threshold	0	2,000 (0x7D0)
48 (0x30)	Control enable	0	1
50 (0x32)	Position derivative gain	0	16,383 (0x3FFF)
52 (0x34)	Position integral gain	0	16,383 (0x3FFF)
54 (0x36)	Position proportional gain	0	16,383 (0x3FFF)
56 (0x38)	Position feedforward gain 1	0	16,383 (0x3FFF)
58 (0x3A)	Position feedforward gain 2	0	16,383 (0x3FFF)
60 (0x3C)	Velocity integral gain	0	16,383 (0x3FFF)
62 (0x3E)	Velocity proportional gain	0	16,383 (0x3FFF)
68 (0x44)	Angular velocity profile	0	Angular velocity limit
70 (0x46)	Acceleration profile	0	Acceleration limit
72 (0x48)	Dead zone	0	1024
78 (0x4E)	Target position	Single-turn: CW angle limit Multi-turn: -32,767	Single-turn: CCW angle limit Multi-turn: 32,767 (0x7FFF)
80 (0x50)	Target angular velocity	0	Angular velocity limit
82 (0x52)	Target torque	0	Torque limit
96 (0x60)	Registered instruction	0	1

4.3 Register details

4.3.1 Control enable

This register controls the following:

Value	Description
0 (default)	<ul style="list-style-type: none"> • unlocks all non-volatile registers • cuts all power to the motor
1 (0x01)	<ul style="list-style-type: none"> • locks all non-volatile registers • enables the motor

4.3.2 Baud rates

The baud rate of the Mercury servo is calculated as follows:

$$\text{Baud rate (BPS)} = 2000000 / (\text{value} + 1)$$

Standard baud rates:

Value	Baud Rate
1 (0x01)	1000000
2 (0x02)	666666
3 (0x03)	500000
4 (0x04)	400000
7 (0x05)	250000
9 (0x09)	200000
16 (0x10)	117647
34 (0x22)	57600

The baud rate formula is: $\text{Speed (BPS)} = 2000000 / (\text{code} + 1)$

The baud rate margin of error is set at < 3%.

4.3.3 Acknowledgement packet response time

This register controls the (approximate) elapsed time between the reception of the request packet and the transmission of the acknowledgement packet. The elapsed time is given by 2u seconds * the register value.

4.3.4 Operating modes table

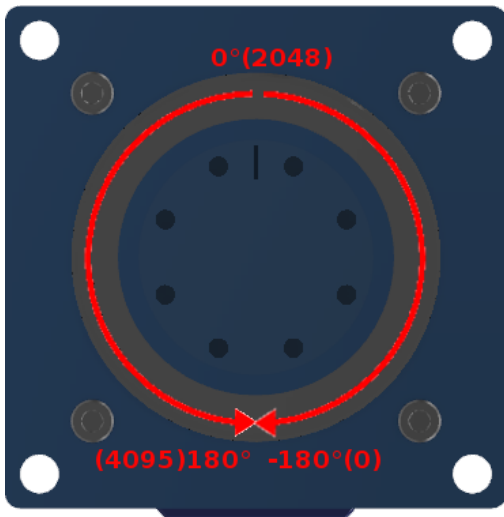
Mode	Value	Description
Torque mode	0	Controls the output torque of the Mercury servo. Makes use of: <ul style="list-style-type: none">• Acceleration limit• Acceleration profile• Torque limit• Target torque
Continuous rotation mode	1	Controls the angular velocity of the Mercury servo. <ul style="list-style-type: none">• Rotates the servo at the target angular velocity.• The direction bit (15) controls the direction of rotation. Makes use of: <ul style="list-style-type: none">• Acceleration limit• Acceleration profile

		<ul style="list-style-type: none"> • Angular velocity limit • Target angular velocity • Velocity proportional gain (Kp) • Velocity integral gain (Ki)
Single-turn position mode	2	<p>Moves to the target position based on the specified velocity and acceleration profiles.</p> <p>Makes use of:</p> <ul style="list-style-type: none"> • Home position offset • CW & CCW angle limits • Acceleration limit • Acceleration profile • Angular velocity limit • Angular velocity profile • Position proportional gain (Kp) • Position integral gain (Ki) • Position derivative gain (Kd) • Position feedforward gains 1 & 2
Multi-turn position mode	3	<p>Moves to the target position based on the specified velocity and acceleration profiles. Allows a (real) target angle to be specified that is greater than 360°. Maximum turns are -16→16.</p> <p>Makes use of:</p> <ul style="list-style-type: none"> • Home position offset • Acceleration limit • Acceleration profile • Angular velocity limit • Angular velocity profile • Position proportional gain (Kp) • Position integral gain (Ki) • Position derivative gain (Kd) • Position feedforward gains 1 & 2

4.3.5 Mercury position modes

Single and multi-turn position modes

A front-facing view of a Mercury servo is shown below.



In the (default) single-turn position mode, the values in brackets represent the **actual position** register values at $-\pi$ radians (-180°), 0 and π radians (180°) - with a **Home position offset** register value of 0.

Clockwise (CW) and counterclockwise (CCW) angle limits

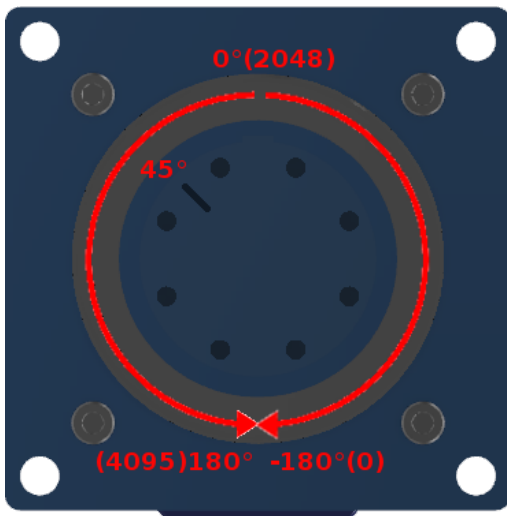
CW and CCW angle limits register are only relevant in the single-turn mode. In multi-turn mode, the CW and CCW angle limits are ignored and a fixed range of $-32,767 \rightarrow 32,767$ is applied instead.

Home position offset

This 2's complement figure represents the home position offset.

In single-turn mode (where the maximum rotation is 360°), the valid range is $-1535 \rightarrow 1535$. A value outside of this range will be considered an error condition, and a value of zero will be assumed.

In multi-turn position mode, the range is $-32,767 \rightarrow 32,767$.



In the above diagram:

- The real position is 45° (3072)
- The home position offset is -1024

- The actual position register value is 2048.

That is, the actual position register value = real position + home position offset.

PID position control

The PID control function is as follows:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

where:

K_p = proportional gain

K_i = integral gain

K_d = derivative gain

Proportional (position) gain

The proportional component depends only on the difference between the goal position and the actual position. This difference is referred to as the Error term. The proportional gain (K_p) determines the ratio of output response to the error signal. In general, increasing K_p will increase the speed of the servo's response. However, if K_p is too large, oscillations could occur. A very large K_p may cause the servo output to oscillate out of control.

Integral (position) gain

The integral component sums the error term over time. The result is that even a small error term will cause the integral component to increase slowly. The integral response will continually increase over time unless the error is zero, so the effect is to drive the steady-state error to zero. Steady-state error is the final difference between the servo's actual position and goal position. If the integral gain (K_i) is too small, the servo response will be sluggish. If K_i is set too high, oscillation may occur.

Derivative (position) gain

The derivative component is inversely proportional to the rate of change of the servo position error term. Increasing the derivative gain (K_d) parameter will cause the control system to react more strongly to changes in the error term and will increase the speed of the overall control system response. In practice, only a very small derivative K_d should be used, as the derivative response is highly sensitive to noise in the position sensor. That is, if the sensor feedback signal is noisy or if the control loop rate is too slow, the servo output may become unstable.

Feedforward control

In a feedforward system, knowledge about the system is used to calculate an output component that is added to the output of the PID controller.

With Mercury digital servos, the feedforward gains (1 & 2) scale the target torque setting. The PWM signal that is sent to the motor is therefore the sum of the feedforward components and the PID component.

Angular velocity profile

The angular velocity profile register maintains the velocity profile for the relevant profile type. Profile angular velocity is represented in milli-rad/s. The maximum angular velocity varies slightly between models, but is approximately 42 rpm. That is, a register value of 4400 equates to 4400 milli-rad/s ⇔ 4.4 rad/s ⇔ 42 rpm.

Acceleration profile

The acceleration profile register maintains the acceleration profile for the relevant profile type. Profile acceleration is represented in units of 0.1 rad/s^2 ($\sim 57 \text{ rpm/min}^2$). Valid values are $0 \rightarrow 100$, representing a maximum profile acceleration of 10 rad/s^2 ($\sim 5700 \text{ rpm/min}^2$).

When set to 0, the applied acceleration corresponds to the maximum acceleration of the motor.

Dead zone

The dead zone register allows the setpoint error tolerance to be defined. With a (default) value of 0, power to the Mercury servo will only be cut when the exact target position is reached. With a non-zero dead zone value, the dead zone value is applied symmetrically to the target position.

For example:

- Dead zone tolerance: 5
- Target position: 2048

Motor power will be cut between actual positions: 2043 \rightarrow 2053

Further reading

A detailed description of the Mercury control algorithms can be found in the accompanying document **Mercury control algorithms explained**.

4.3.6 Mercury continuous rotation mode

In *Continuous rotation* mode, the servo will always attempt to maintain the target angular velocity. The velocity PI parameters are used to maintain the actual angular velocity at the target angular velocity. The profile acceleration register value is used to control the acceleration to the target velocity.

Target angular velocity

The target angular velocity register maintains the target velocity in milli-rad/s. The maximum angular velocity varies slightly between models, but is approximately 42 rpm. That is, a register value of 4400 equates to $4400 \text{ milli-rad/s} \Leftrightarrow 4.4 \text{ rad/s} \Leftrightarrow 42 \text{ rpm}$.

Bit 15 of the target angular velocity register maintains the direction of rotation:

- 0 \Leftrightarrow counterclockwise (CCW)
- 1 \Leftrightarrow clockwise (CW)

Note that target angular velocity register is distinct from the profile angular velocity profile register.

Acceleration profile

The acceleration profile register maintains the acceleration profile for in continuous rotation mode. Profile acceleration is represented in units of 0.1 rad/s^2 ($\sim 57 \text{ rpm/min}^2$). Valid values are $0 \rightarrow 100$, representing a maximum profile acceleration of 10 rad/s^2 ($\sim 5700 \text{ rpm/min}^2$).

When set to 0, the applied acceleration corresponds to the maximum acceleration of the motor.

PI velocity control

Proportional (velocity) gain

The proportional component depends only on the difference between the target velocity and the actual velocity. This difference is referred to as the Error term. The proportional gain (K_p) determines the ratio of output response to the error signal. In general, increasing K_p will increase the speed of the servo's response. However, if K_p is too large, oscillations could occur. A very large K_p may cause the servo output to oscillate out of control.

Integral (velocity) gain

The integral component sums the error term over time. The result is that even a small error term will cause the integral component to increase slowly. The integral response will continually increase over time unless the error is zero, so the effect is to drive the steady-state error to zero. Steady-state error is the final difference between the servo's actual velocity and target velocity. If the integral gain (K_i) is too small, the servo response will be sluggish. If K_i is set too high, oscillation may occur.

4.3.7 Torque mode

In Torque mode, the target angular velocity is ignored. Instead, the servo will attempt to maintain a target torque. The result of this is that the angular velocity will depend only on the target torque setting and the load on the servo.

In Torque mode, the following parameters are ignored:

- Target angular velocity
- CW and CCW limits
- Goal position
- PID (position and velocity) parameters

Bit 15 of the target torque register maintains the direction of rotation:

- 0 \Leftrightarrow counterclockwise (CCW)
- 1 \Leftrightarrow clockwise (CW)

Acceleration profile

The acceleration profile register maintains the acceleration profile for in torque mode. Profile acceleration is represented in units of 0.1 rad/s^2 ($\sim 57 \text{ rpm/min}^2$). Valid values are $0 \rightarrow 100$, representing a maximum profile acceleration of 10 rad/s^2 ($\sim 5700 \text{ rpm/min}^2$).

When set to 0, the applied acceleration corresponds to the maximum acceleration of the motor, but is limited by the torque limit setting.

4.3.8 Is moving

The Moving register is set to 1 if the Mercury digital servo is deemed to be moving. Otherwise the Moving register is set to zero indicating that the servo is stationary.

Determining whether the servo is moving or is stationary is done by comparing the actual velocity with the moving threshold value. If the (absolute) actual velocity exceeds the threshold value, the servo is deemed to be moving.

5. Operations

5.1 Operations table

The Mercury servo range supports the following operations:

Operation	Description	Value	Number of parameters
PING	Returns a status servo from the targeted servo. No servo update is performed.	0x01	0
READ_DIRECT	Direct read of values from the register table.	0x02	2
WRITE_DIRECT	Direct write to the active register table.	0x03	2+
WRITE_SHADOW	Write to the shadow register table. This operation does not affect the current operation of the servo.	0x04	2+
COMMIT_SHADOW	Commit the previously-written shadow register table to the active register table.	0x05	0
WRITE_COMPOSITE	Used for the simultaneous control of multiple Mercury servos	0x83	4~
RESET	Reset the servo to the default (factory) settings	0x06	0

5.2 Operation details

5.2.1 PING

The PING command is used to request a status packet from a particular Mercury servo specified by an id.
Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0→0xFC (252)	ID of the targeted servo
4	0x02	Length
5	0x01	PING instruction
6	~	Calculated checksum

5.2.2 READ_DIRECT

The READ_DIRECT command is used to read data directly from the register table of a Mercury servo

Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0→0xFC (252)	ID
4	0x04	Length
5	0x02	READ_DIRECT instruction
6	0+	Register table start address
7	1+	Number of bytes to read, starting from the (above) start address
8	~	Calculated checksum

5.2.3 WRITE_DIRECT

The WRITE_DIRECT command is used to write data directly to the register table of a Mercury servo.

No status packet is returned if the broadcast ID is used.

Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0→0xFC (252)	ID. The broadcast ID (0xFE) can also be specified.
4	N+3	Length - the number of data bytes to be written + 3
5	0x03	WRITE_DIRECT instruction
6	0+	Register table start address
7	0→0xFE (254)	Data byte 1
...		Data bytes 2→ N-1
7 + Number of data bytes	0→0xFE (254)	Data byte N
Length+4	~	Calculated checksum

5.2.4 WRITE_SHADOW

The WRITE_SHADOW command is used to write data to the shadow register table of a Mercury servo. This has no direct effect on the operation of the Mercury servo, as the active register table is not updated following a WRITE_SHADOW operation.

The Registered instruction register is set to 1 following a WRITE_SHADOW operation.

No status packet is returned if the broadcast ID is used.

Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0→0xFC (252)	ID. The broadcast ID (0xFE) can also be specified.
4	N+3	Length - the number of data bytes to be written + 3
5	0x04	WRITE_SHADOW instruction
6	0+	Register table start address
7	0→0xFE (254)	Data byte 1
...		Data bytes 2→N-1
7 + Number of data bytes	0→0xFE (254)	Data byte N
Length+4	~	Calculated checksum

5.2.5 COMMIT_SHADOW

The COMMIT_SHADOW command is used to commit the data held in the shadow register table of a Mercury servo to the active register table. A Mercury servo will only execute a COMMIT_SHADOW operation if its Registered instruction register has a value of 1.

The Registered instruction register is reset to 0 following a COMMIT_SHADOW operation.

No status packet is returned if the broadcast ID is used.

Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0→0xFC (252)	ID. The broadcast ID (0xFE) can also be specified.
4	0x02	Length
5	0x05	COMMIT_SHADOW instruction

6	~	Calculated checksum
---	---	---------------------

5.2.6 WRITE_COMPOSITE

The WRITE_COMPOSITE command is used to commit data blocks to multiple Mercury servos. The data start address and data length are common to all the specified data blocks. However, the data itself may vary for each addressed servo.

No status packet is returned as the broadcast ID is used.

Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0xFE (254)	The broadcast ID.
4	0x02	Length:- ((Data block length + 1)*Number of Mercury servos)+4
5	0x83	WRITE_COMPOSITE instruction
6	0+	Register table start address
7	1+	Data block length
8	0→0xFC (252)	ID of the first Mercury servo
9	~	First data byte for the first Mercury servo
...	~	Nth data byte for the first Mercury servo
...	0→0xFC (252)	ID of the second Mercury servo
9	~	First data byte for the second Mercury servo
...	~	Nth data byte for the second Mercury servo
...	0→0xFC (252)	ID of the Nth Mercury servo
...	~	First data byte for the Nth Mercury servo
...	~	Nth data byte for the Nth Mercury servo
Length+4	~	Calculated checksum

5.2.7 RESET

The RESET command is used to reset the register table of a Mercury servo to the original factory defaults.

Packet details:

Byte	Value	Description
1	0xFF	Start byte 1
2	0xFF	Start byte 2
3	0→0xFC (252)	ID of the targeted Mercury servo
4	0x02	Length
5	0x06	RESET instruction
6	~	Calculated checksum