Robot Articulation

Mercury technical manual

March

v.1

2019

Mercury technical manual

v.1

1. Introduction

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

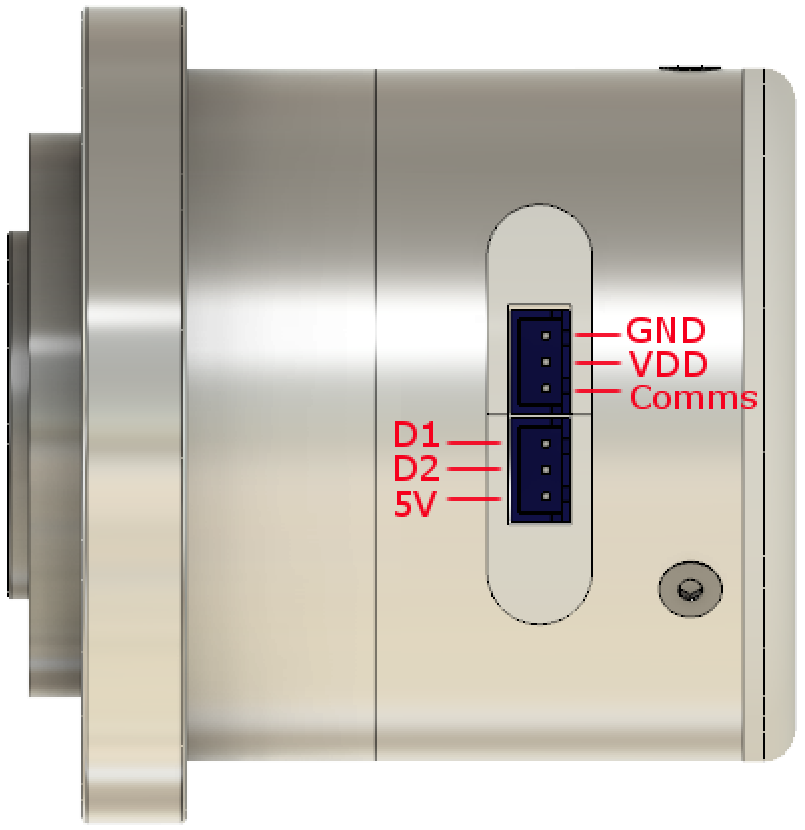
2. Connection details

2.1 Pin assignments

Each Mercury servo has 6 input pins.

The pinouts are as follows:

PIN1: GND
PIN2: VDD (15-24V DC)
PIN3: Comms (half-duplex, TTL levels)
PIN4: Digital input 1 (opto-isolated)
PIN5: Digital input 2 (opto-isolated)
PIN6: 5V common supply (to optocouplers)



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.

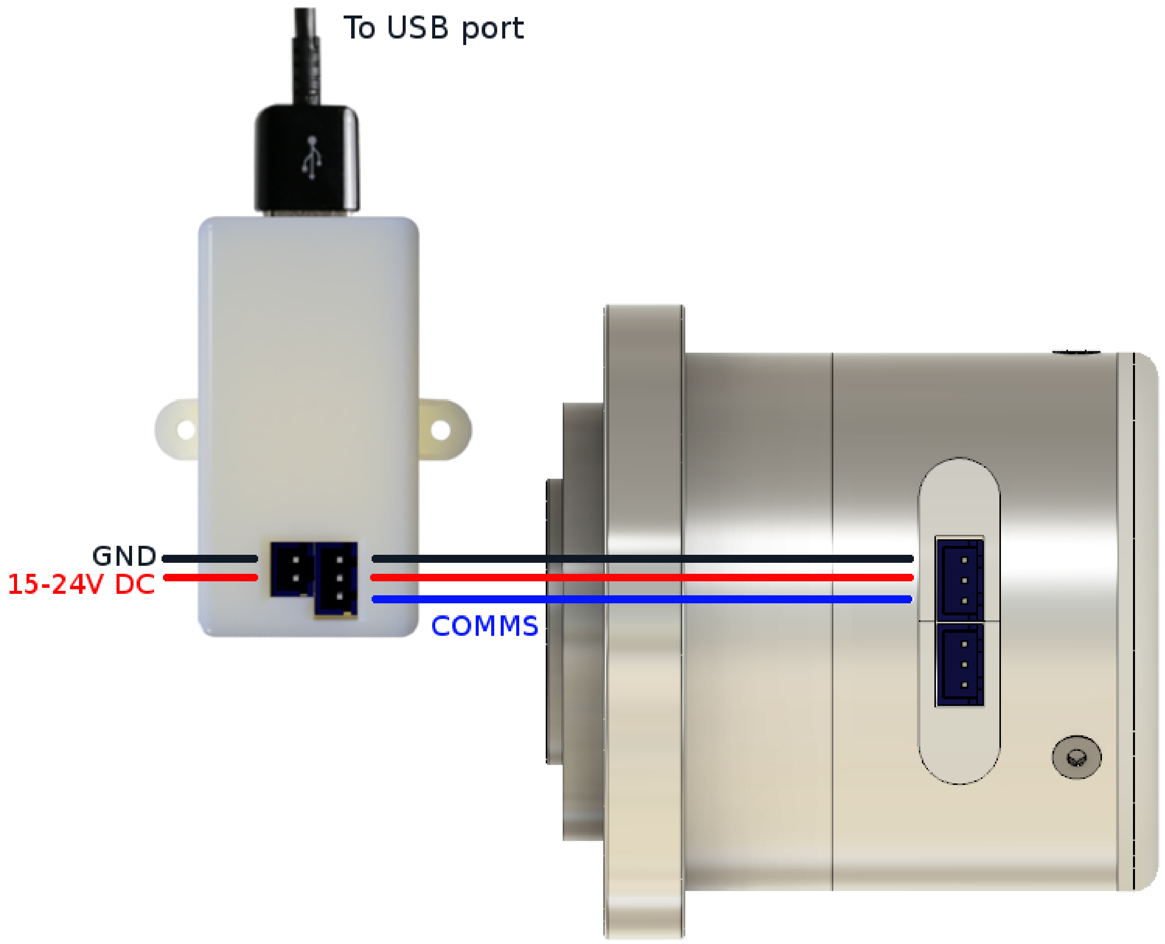
2.3 USB2Mercury

In order to control (or configure) Mercury digital servos from a PC, a USB2Mercury unit should be used.

The USB2Mercury unit contains:

- a USB 2.0 type A connector which connects to a USB port on a PC
- a single 2-way XH-series header for connection to an external 15-24V DC power source
- a single 3-way XH-series header to connect to a Mercury servo
- reverse voltage and surge protection

The USB2Mercury unit is connected in the following way:



Please see the USB2Mercury document for more details.

2.4 Mercury power hub

Mercury digital servos may be connected in parallel to form a network of Mercury servos. To facilitate such parallel operations, a Mercury Power Hub should be used.

The Mercury Power Hub has the following capabilities:

- contains an in-build USB2Mercury 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 USB2Mercury 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 USB2Mercury unit to the Mercury servos. Acknowledgement packets are sent from Mercury servos to the USB2Mercury unit.

3.2 Protocol

Communication between Mercury servos and the USB2Mercury 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 a 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 M30	
2 (0x02)	1	Firmware version	R			
3 (0x03)	1	ID	RW	0→252	1 (0x01)	Note that the USB2Mercury 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	250 (0xFA)	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 (0x0FFF) (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	M30: 0→1,800 M50: 0→3,000 M65: 0→4,000	M30: 600 (0x258) M50: 1,000 (0x3E8) M65: 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)	4	Home position offset	RW	Single-turn: -1024→1535 Multi-turn: -1,044,479 → 1,044,479	0	See details below.
24 (0x18)	2	Moving threshold	RW	0 → 2,000	200 (0x00C8)	Moving velocity threshold in milli-rad/s. See details below.
26 (0x1a)	1	Reserved				
...	21	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	0	See the angular velocity profile details below.
70 (0x46)	2	Acceleration profile	RW	0→Acceleration limit	0	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)	4	Target position	RW	Single-turn: 0→4095 (-π to π)	Value from actual position register.	See servo output position section below

				Multi-turn: -1,048,575 → 1,048,575		
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)	4	Actual position	R	Single-turn: 0→4095 (-π to π) Multi-turn: -1,048,575 → 1,048,575	-	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	Trajectory status	R			Indicates the current trajectory status. See [details](#)below.
96 (0x60)	2	Calculated angular velocity profile trajectory	R			
98 (0x62)	2	Calculated position profile trajectory	R			
100 (0x64)	1	Registered instruction	RW	0→1	0	1 = pending SHADOW_WRITE instruction.
101 (0x65)	1	Reserved...				
...	5	Reserved				
107 (0x6B)	1	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	4
7 (0x07)	Clockwise (CW) angle limit	0	4095 (0x0FFF)
9 (0x09)	Counter-clockwise (CCW) angle limit	0	4095 (0xFFF)
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	M30: 1,800 (0x708) M50: 3,000 (0xBB8) M65: 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: -1535 Multi-turn: -32,767	Single-turn: 1535 Multi-turn: 32,767
23 (0x18)	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)	Pending shadow instruction	0	1

4.3 Register details

4.3.1 Control enable

This register controls the following:

Value	Description
0 (default)	● unlocks all non-volatile registers ● cuts all power to the motor
1 (0x01)	● locks all non-volatile registers ● enables the motor

4.3.2 Baud rates

The baud rate of the Mercury servo is calculated as follows:
Baud rate (BPS) = 2000000 / (value + 1)

Standard baud rates:

Value	Baud Rate
1 (0x01)	1000000
3 (0x03)	500000
4 (0x04)	400000
7 (0x05)	250000
9 (0x09)	200000
16 (0x10)	115200
34 (0x22)	57600
103 (0x67)	19200
207 (0xcf)	9600

The baud rate formula is: Speed (BPS) = 2000000 / (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

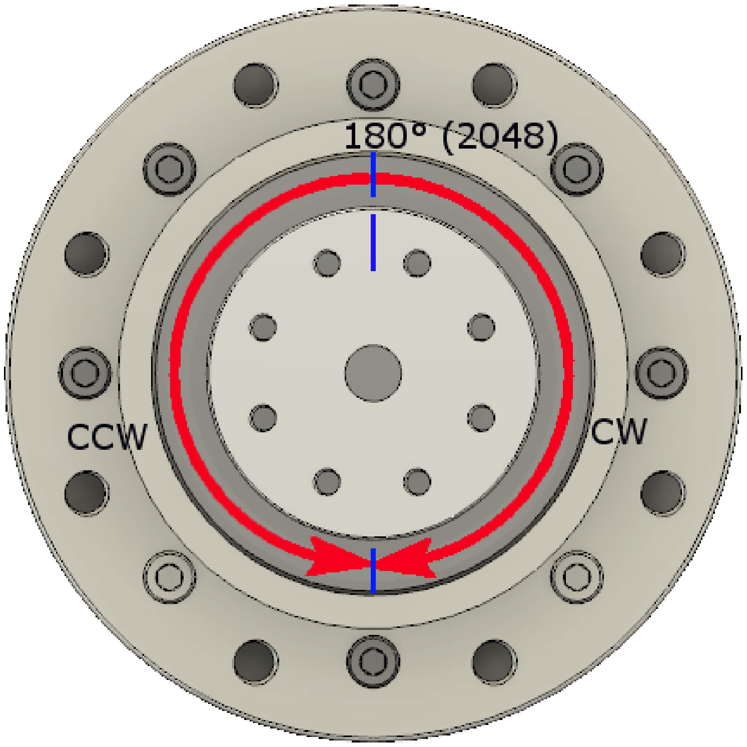
Mode	Value	Description
Torque mode	0	Controls the output torque of the Mercury servo. Makes use of: ● Acceleration limit ● Acceleration profile ● Torque limit ● Target torque ● Angular velocity limit
Continuous rotation mode	1	Controls the angular velocity of the Mercury servo. ● Rotates the servo at the target angular velocity. ● The direction bit (15) controls the direction of rotation. Makes use of: ● Acceleration limit ● Acceleration profile ● Angular velocity limit ● Target angular velocity ● Velocity proportional gain (Kp) ● Velocity integral gain (Ki)
Single-turn position mode	2	Moves to the target position based on the specified velocity and acceleration profiles. Makes use of: ● 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	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. Makes use of: ● 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

Stepper mode	4	In this mode, digital pins D1 and D2 are used as a STEP and DIRECTION inputs respectively. Each step will advance the output horn (CW or CCW depending on the polarity of the DIRECTION input pin D2) by 1/4096 degrees. Makes use of: ● Home position offset ● Acceleration limit ● 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) joint mode, the values in brackets represent the **actual position** register values at π radians (180°) and with a **Home position offset** register value of 0.

Clockwise (CW) and counterclockwise (CCW) angle limits

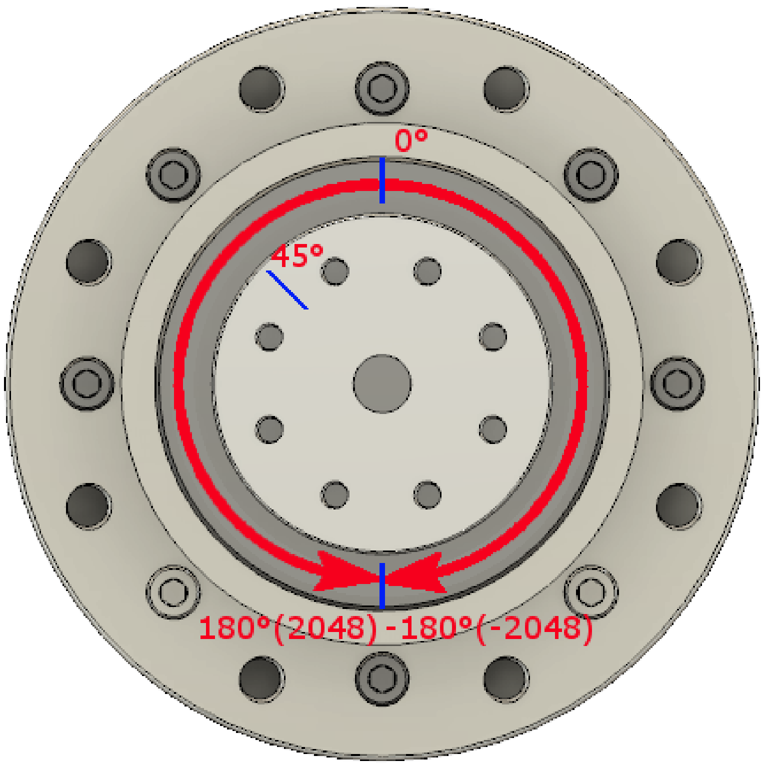
CW and CCW angle limits register are only relevant in joint mode. In multi-turn mode, the CW and CCW angle limits are ignored..

Home position offset

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

In joint (single-turn) mode (where the maximum rotation is 360°), the valid range is -1024→1024. 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 -1,044,479 → 1,044,479. This is the equivalent of +/- 255 turns.



In the above diagram:

- The real position is 45° (512)
- The home position offset is -512
- The actual position register value is 0.

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

PID position control

The PID control function is as follows:

$$u(t) = K_{\text{p}} e(t) + K_{\text{i}} \int_0^t e(\tau)\, d\tau + K_{\text{d}} \frac{de(t)}{dt}$$

where:

Kp = proportional gain

Ki = integral gain

Kd = 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 (Kp) determines the ratio of output response to the error signal. In general, increasing Kp will increase the speed of the servo's response. However, if Kp is too large, oscillations could occur. A very large Kp 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 and goal position. If the integral gain (Ki) is too small, the servo response will be sluggish. If Ki 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 (Kd) 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 Kd 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, feedforward gains 1 & 2 represent velocity and acceleration gains respectively. The PWM signal that is sent to the motor is therefore the sum of the feedforward and PID components.

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.

Angular velocity profile is used only in **Position control** mode.

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 (~57 rpm/min^2). Valid values are 0 → 100, representing a maximum profile acceleration of 10 rad/s^2 (~5700 rpm/min^2) .

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

Acceleration profile is used in both **Angular velocity control** mode and **Position control** mode.

Trajectory profile

The trajectory profile register provides information relating to the current trajectory state.

Status	Description
0x80	Unused
0x40	Trapezoidal angular velocity profile
0x20	Triangular angular velocity profile
0x10	Rectangular angular velocity profile
0x08	Step angular velocity profile
0x04	Target position trajectory error

0x02	Trajectory in progress
0x01	Target position reached

Dead zone

The dead zone register allows a 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→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 position profile control mode

In position control mode, 4 different angular velocity profile control schemes are available:

Mode	Angular velocity profile value	Acceleration profile value	Notes
Step	0	X	Angular velocity == maximum achievable
Rectangle	> 0	0	Acceleration == maximum achievable
Triangle	> 0	> 0	Angular velocity profile value <= calculated angular velocity profile trajectory
Trapezoidal	> 0	> 0	Angular velocity profile value > calculated angular velocity profile trajectory

1. Step

   In step mode, the acceleration and angular velocity profiles are not used. Instead, the horn accelerates at a maximum rate to the maximum angular velocity before decelerating at a maximum rate to the goal position.
2. Rectangle

   In Rectangular mode, the horn accelerates at a maximum rate to the calculated angular velocity profile trajectory value. The horn then maintains the calculated angular velocity profile trajectory value until reaching the goal position. The horn then decelerates at the maximum rate.
3. Triangle

   In Triangular mode, the horn accelerates at the calculated acceleration profile trajectory value. The velocity of the horn at the point where the horn starts to decelerate towards the position setpoint is less (or equal) to the calculated angular velocity profile trajectory. The profile is thus triangular.
4. Trapezoidal

In Trapezoidal mode, the horn accelerates at the calculated acceleration profile trajectory value until the calculated angular velocity profile trajectory is reached. This velocity is then maintained before decelerating to the position setpoint.

4.3.7 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 milli-rad/s ⇔ 4.4 rad/s ⇔ 42 rpm.

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

- 0 ⇔ counterclockwise (CCW)
- 1⇔ 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 (~57 rpm/min^2). Valid values are 0 → 100, representing a maximum profile acceleration of 10 rad/s^2 (~5700 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 (Kp) determines the ratio of output response to the error signal. In general, increasing Kp will increase the speed of the servo's response. However, if Kp is too large, oscillations could occur. A very large Kp 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 and target velocity. If the integral gain (Ki) is too small, the servo response will be sluggish. If Ki is set too high, oscillation may occur.

4.3.8 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 ⇔ counterclockwise (CCW)
- 1⇔ 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 (~57 rpm/min^2). Valid values are 0 → 100, representing a maximum profile acceleration of 10 rad/s^2 (~5700 rpm/min^2) .~~

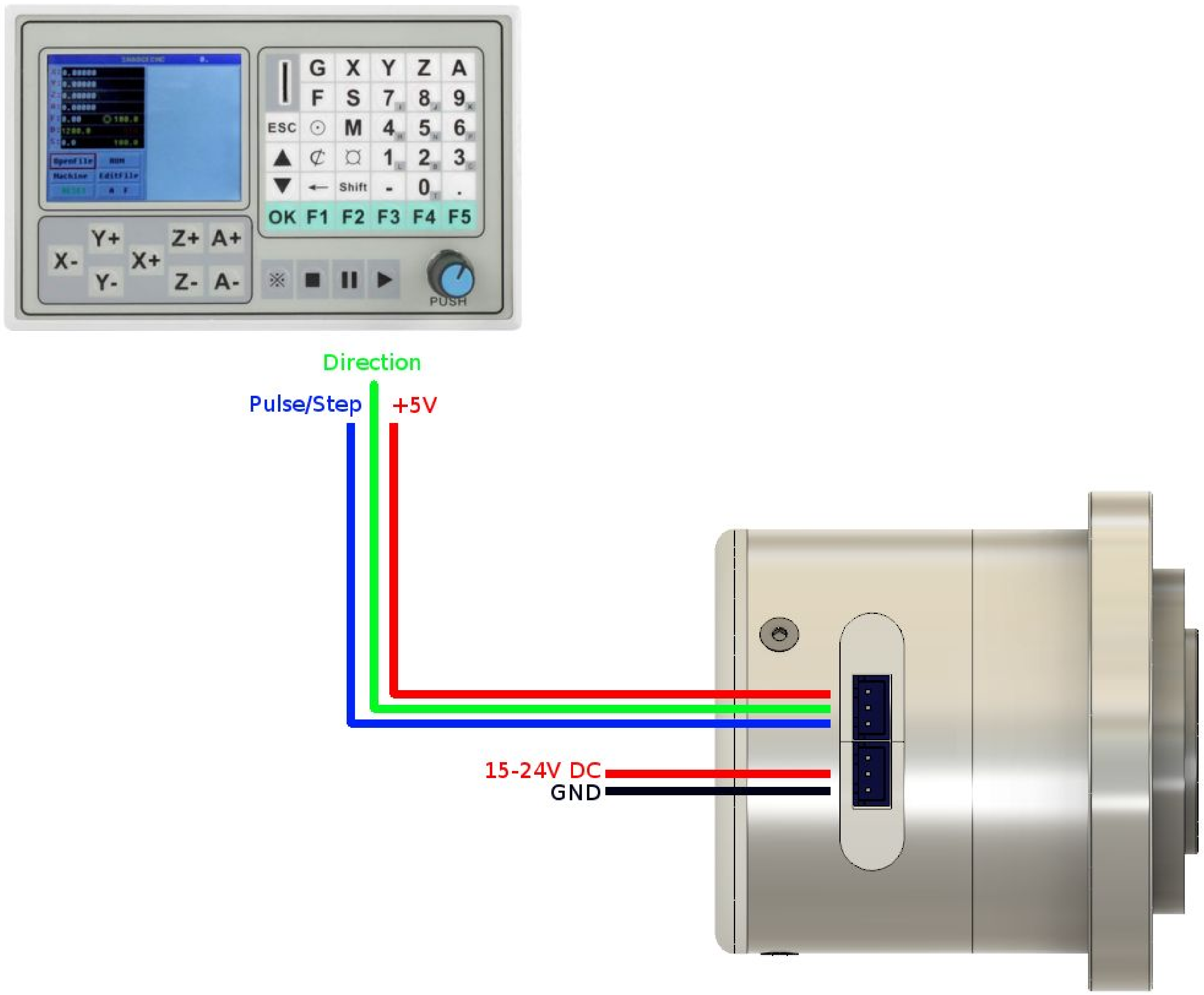
~~When set to 0, the applied acceleration corresponds to the maximum acceleration of the motor, but is limited by the torque limit setting.~~

4.3.9 Stepper mode

This control mode allows for the straightforward connection of Mercury digital servos to CNC-type controllers with STEP and DIRECTION outputs. Unlike conventional open-loop stepper motors, the Mercury servo's control electronics will ensure the correct output response based on input steps and chosen control parameters.

The two opto-isolated digital inputs (D1 & D2) are **active low**. See the Connection details diagram for more information. Please note that most CNC controllers allow the output polarity of their STEP and DIRECTION pins to be specified.

Input D1 (pin4) must be connected to the STEP output of the CNC controller.
Input D2 (pin 5) must be connected to the DIRECTION output of the CNC controller.
The 5v common (pin 6) must be connected to the 5 volt output of the CNC controller.



The CNC controller motor limits should ideally be set to the following settings:

- 3600 steps per 360° (0.1° degrees per step)
- 1000 steps per second
- Max acceleration/deceleration pulse period <= 50ms

For optimal tracking of the incoming step pulses, ensure that acceleration & deceleration are not too high.

4.3.10 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. Instructions

5.1 Instructions table

The Mercury servo range supports the following instructions:

Instruction	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 instruction 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 Instruction 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 instruction.

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

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 instruction if its Pending shadow instruction register has a value of 1.

The Pending shadow instruction register is reset to 0 following a COMMIT_SHADOW instruction.

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