Communication Protocol for the onero Actuator

1. Communication Packet Structure

📌 Endianness Mode

Data transmission uses **little-endian mode**. For example, for 0xB562, 0x62 is sent first, followed by 0xB5.

🎹 CAN Parameters: Baud rate 1Mbps; DLC is the data length, and data without a specified number of bytes defaults to 1 Byte

CANId: Node ID

MstId: Feedback Frame ID

2. Communication Frame

1. Feedback Frame

<table>
<tr><th>ID</th><th colspan="8">Data</th><th>D</th></tr>
<tr><td>MstId</td><td>ID | ERR<<4</td><td>POS [15:8]</td><td>POS [7:0]</td><td>VEL [11:4]</td><td>VEL[3:0] | T[11:8]</td><td>T[7:0]</td><td>T_MOS</td><td>T_Coil</td><td></td></tr>
</table>

- After each control frame is sent, there will be 1 feedback frame response, and the feedback frame ID is MstId.
- ID represents controller ID, taking the lower 8 bits of CANId.
- ERR indicates the status, corresponding to the following status types: 0 - Disabled; 1 - Enabled; 8 - Overvoltage; 9 - Undervoltage; A - Overcurrent; B - MOSFET Over-temperature; C - Motor Coil Over-temperature; D - Communication Loss; E - Overload;
- POS: motor position
- VEL: motor speed

- T: motor torque
- T_MOS: MOSFET temperature. Indicates the temperature of the power MOSFETs on the driver board, in °C, with a data type of int8_t.
- T_Coil: Motor winding temperature. Indicates the temperature measured at the motor’ s internal windings, in °C, with a data type of int8_t.

Position, velocity, and torque values are linearly mapped from floating-point to signed fixed-point data. Position uses 16 bits, while velocity and torque each use 12 bits.

Taking position as an example: if the motor's current position is 5.0 rad and the configured position range is ±12.5 rad (i.e., PMAX = 12.5 rad), the transmitted value is calculated as follows: `POS = 5.0 / (12.5 - (-12.5)) * (2^16 - 1) = 13107 = 0x3333`

This conversion is implemented using the `uint_to_float()` and `float_to_uint()` functions, as detailed below:

```c
int float_to_uint(float x, float x_min, float x_max, int bits){
  /// Converts a float to an unsigned int, given range and number of bits ///
    float span = x_max - x_min;
    float offset = x_min;
    return (int) ((x-offset)*((float)((1<<bits)-1))/span);
}

float uint_to_float(int x_int, float x_min, float x_max, int bits){
  /// converts unsigned int to float, given range and number of bits ///
    float span = x_max - x_min;
    float offset = x_min;
    return ((float)x_int)*span/((float)((1<<bits)-1)) + offset;
}
```

2. Control Frame

2.1 MIT Mode

ID	Data								DLC
CANId	P_des [15:8]	P_des [7:0]	V_des [11:4]	V_des[3:0] | Kp[11:8]	Kp [7:0]	Kd [11:4]	Kd[3:0] | T_ff[11:8]	T_ff[7:0]	8

- Frame ID equals the set CANID
- P_des: Position setpoint
- V_des: Speed setpoint
- Kp: Position proportional gain
- Kd: Position derivative gain
- T_ff: Feedforward torque

Each parameter follows the same scaling (linear mapping) convention as defined in **1. Feedback Frame.** The ranges for `P_des`, `V_des`, and `T_ff` are configurable via the Write Parameter command. The Position Proportional Gain (Kp) has a fixed range of [0, 500], and the Position Derivative Gain (Kd) has a fixed range of [0, 5].

A standard CAN (Classic CAN) data frame is limited to 8 bytes. The MIT control command format bit-packs the five parameters—Position, Velocity, Kp, Kd, and Feedforward Torque (T_ff)—into these 8 bytes with the following bit allocation:

- Position: 16 bits
- Velocity: 12 bits
- Kp: 12 bits
- Kd: 12 bits
- T_ff: 12 bits
  (Total: 64 bits = 8 bytes)

The MIT command encodes floating-point values into integers using a proportional scaling factor for transmission over the CAN bus. The actuator drive then decodes the received integers back into floating-point values using the identical inverse scaling.

2.2 Position Velocity Mode

ID	Data		DLC
CANId + 0x100	P_des(4 bytes)	V_des(4 bytes)	8

- Frame ID: The configured CAN ID value plus `0x100`.
- P_des: Position setpoint.
  - Data Type: Floating-point
  - Byte Order: Little-endian
- V_des: Velocity setpoint.
  - Data Type: Floating-point

- Byte Order: Little-endian
- Definition: Under trapezoidal velocity profile motion, this represents the target velocity during the constant-velocity phase (i.e., the maximum planned speed).

2.3 Velocity Mode

ID	Data	DLC
CANId + 0x200	V_des(4 bytes)	4

- Frame ID: The configured CAN ID value plus `0x200`.
- V_des: Velocity setpoint.
  - Data Type: Floating-point
  - Byte Order: Little-endian

3. Parameter Read/Write

3.1 Read Parameters

Command Transmission

ID	Data					DLC
0x7FF	CANId_L	CANId_H	0x33	RID	Arbitrary Value (4 bytes)	8

RID is the Register ID (address). Refer to the appendix *Register List and Ranges*.

Command Response

ID	Data					DLC
MstId	CANId_L	CANId_H	0x33	RID	Data (4 bytes)	8

The least significant byte of the returned data is Data[4], and the most significant byte is Data[7]. The same convention applies to the following descriptions.

3.2 Write Parameters

Command Transmission

ID	Data					DLC
0x7FF	CANId_L	CANId_H	0x55	RID	Data (4 bytes)	8

RID is the Register ID (address). Refer to the appendix *Register List and Ranges*.

Command Response

ID	Data					DLC
MstId	CANId_L	CANId_H	0x55	RID	Data (4 bytes)	8

The parameter change takes effect immediately after being written but is not retained in non-volatile memory. It will be lost upon power cycle. To permanently save the modified parameter(s), a Store Parameters command must be sent to write all changes to Flash memory.

3.3 Storage Parameters

Command Transmission

ID	Data				DLC
0x7FF	CANId_L	CANId_H	0xAA	0x01	4

Command Response

ID	Data				DLC
MstId	CANId_L	CANId_H	0xAA	0x01	4

Upon successful execution of this command, all parameters will be written to Flash memory in a single operation; no individual parameter-by-parameter saving is required.

3.4 Reset Parameters

Command Transmission

ID	Data				DLC
0x7FF	CANId_L	CANId_H	0xAF	0x01	4

Command Response

ID	Data				DLC
MstId	CANId_L	CANId_H	0xAF	0x01	4

Upon successful execution of this command, all parameters (except CANId and MstId) will be restored to their default settings and written to flash.

4. Other Commands

4.1 Get ID

Command Transmission

ID	数据				DLC
0x7FF	0x55	0x00	0x00	0xAA	4

Command Response

ID	Data				DLC
0x7FF	0x55	CANId	MstId	0xAA	4

4.2 Set ID

ID	Data				DLC
0x7FF	0xAA	CANId	MstId	0x55	4

After changing the ID, the motor should be immediately powered on again to apply the new ID.

4.3 Enable Motor

ID	Data								DLC
CANId	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFC	8

The actuator can only be controlled after passing the power-on self-test and receiving an Enable command.

The Enable Frame is a type of Control Frame. Its Frame ID can be any valid Control Mode ID. Upon successful transmission, the motor will begin transmitting Status Frames.

4.4 Disable Motor

ID	Data								DLC
CANId	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFD	8

Disabled is the default state of the motor upon power-up. In this state, the voltage waveforms across all three motor terminals are identical.

The Disable Frame is a type of Control Frame. Its Frame ID is the same as that of the Enable Frame.

4.5 Save Position Zero

ID	Data								DLC
CANId	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFE	8

The Save Position Zero frame is a type of Control Frame. Its Frame ID is identical to the ones mentioned above (e.g., Enable/Disable). Upon successful transmission, the motor drive will save its current position as the new position zero.

4.6 Clear Error

ID	Data								DLC
CANId	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFE	8

When a motor error (e.g., overtemperature) occurs, the error state can be cleared by sending a Clear command. The Clear Frame is a type of Control Frame, and its Frame ID is identical to those of the other control frames mentioned previously.

4.7 Get Motor Feedback

ID	Data				DLC
0x7FF	CANId_L	CANId_H	0xCC	0x00	4

After successful transmission, the motor driver will send a feedback frame.

Appendix: Register List and Ranges

Register Address (Decimal)	Variable	Description	Read/Write	Range	Data Type
0	UV_Value	Low voltage protection value	RW	(10.0,3.4E38]	float
2	OT_Value	Over Temperature Protection Value	RW	[80.0,200)	float
3	OC_Value	Overcurrent protection value	RW	(0.0,1.0)	float
4	ACC	Acceleration	RW	(0.0,3.4E38)	float
5	DEC	Deceleration	RW	[-3.4E38,0.0)	float
6	MAX_SPD	Maximum Speed	RW	(0.0,3.4E38]	float
7	MST_ID	Feedback ID	RW	[0,0x7FF]	uint32
8	ESC_ID	Receive ID	RW	[0,0x7FF]	uint32
10	CTRL_MODE	Control Mode	RW	[1,4]	uint32
13	hw_ver	Retain	RO	/	uint32
14	sw_ver	Software Version Number	RO	/	uint32
15	SN	Retain	RO	/	uint32
16	NPP	Number of pole pairs of the motor	RO	/	uint32
17	Rs	Motor phase resistance	RO	/	float
18	Ls	Motor phase inductance	RO	/	float
19	Flux	Motor flux linkage value	RO	/	float

20	Gr	Gear reduction ratio	RO	/	float
21	PMAX	Position Mapping Range	RW	(0.0,3.4E38]	float
22	VMAX	Speed Mapping Range	RW	(0.0,3.4E38]	float
23	TMAX	Torque Mapping Range	RW	(0.0,3.4E38]	float
25	KP_ASR	Speed Loop Kp	RW	[0.0,3.4E38]	float
26	KI_ASR	Speed Loop Ki	RW	[0.0,3.4E38]	float
27	KP_APR	Position Loop Kp	RW	[0.0,3.4E38]	float
28	KI_APR	Position Loop Ki	RW	[0.0,3.4E38]	float
29	OV_Value	Overvoltage protection value	RW	TBD	float
36	sub_ver	Subversion Number	RO	/	uint32

RW: Read and Write.

RO: Read-only.