Robot Joint Motor Communication Instructions

CAN Communication Protocol

HONPINE
鸿 磐

Version: 1.1.0

7. CAN Control Mode

Classic CAN messages support payload lengths of 1, 2, 3, 4, 5, 6, 7, and 8 bytes, whereas only FD-CAN messages can achieve payload lengths of up to 64 bytes. By categorizing CAN messages based on their data length, the following section describes the classification of motor-received CAN commands.

7.1. 1-Byte Command

Command Name	Motor Disable
Command Message (hex)	02
Command Reply(hex)	None
Command Description	Upon receiving this command, the motor will be disabled immediately. If a brake is present, it will engage at once, preventing any further movement of the motor.

Command Name	Get Motor Operation Mode
Command Message (hex)	03
Command Reply(hex)	03 08 00 00 00
Command Description	In a 5-byte reply, the first byte remains the same as the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. For example, "08 00 00 00" combines to form the decimal value 8, while "09 00 00 00" combines to form the decimal value 9. Reply status values: 8 represents Position Mode, 9 represents Speed Mode, 10 represents Current Mode, 1 represents Profiled Position Mode, 3 represents Profiled Speed Mode, and 4 represents Profiled Current Mode.

Command Name	Get Motor Feedback Current
Command Message (hex)	04
Command Reply(hex)	04 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer, where positive values indicate forward current and negative values indicate reverse current. The unit of the data is milliamperes (mA).

Command Name	Get Motor Feedback Filtered Current
Command Message (hex)	05
Command Reply(hex)	05 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer, where positive values represent forward current and negative values represent reverse current. The data unit is milliamperes (mA).
Command Name	Get Motor Speed
Command	06

Message (hex)	
Command Reply(hex)	06 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer, where positive values indicate forward speed and negative values indicate reverse speed. The data unit is 0.01 Hz (0.01 rps). Note that the speed refers to the motor speed on the high-speed side of the reducer.

Command Name	Get Motor Target Speed
Command Message (hex)	07
Command Reply(hex)	07 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer, where positive values indicate forward speed and negative values indicate reverse speed. The data unit is 0.01 Hz (0.01 rps). Note that the speed refers to the motor speed on the high-speed side of the reducer.

Command Name	Get Feedback Position
Command Message (hex)	08
Command Reply(hex)	08 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the motor's absolute position. The data unit is encoder counts. For single encoder configurations, this indicates the motor position on the high-speed side of the reducer; for dual encoder configurations, it indicates the output shaft position on the low-speed side of the reducer.

Command Name	Get Target Position
Command Message (hex)	09
Command Reply(hex)	09 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the motor's target position.

Command Name	Get Error Status
Command Message (hex)	0A
Command Reply(hex)	0A 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the motor's error code. A value of 0 indicates no error, while a non-zero value indicates an error. The specific error type is determined by which bit is set to 1: bit0: Internal software error (e.g., writing to FLASH during motor operation) bit1: Over-voltage bit2: Under-voltage bit4: Startup error bit5: Speed feedback error bit6: Over-current bit7: Software error (e.g., writing to FLASH during motor operation) bit16: Encoder communication error bit17: Motor overtemperature bit18: Circuit board overtemperature bit19: Driver chip error/Current overload bit20: EtherCAT communication error (Occurs when EC is connected then disconnected; resolved by restarting the master or power cycling) bit21: Position following error too large (Only in Position and Profiled Position modes; this function is disabled by default for CAN)

Command Name	Clear Error
Command Message (hex)	0B
Command Reply(hex)	None
Command Description	Upon receiving this command, the motor will clear the current error code and attempt to restore itself to a normal operational state. In some cases, if the error persists after clearing (e.g., when the temperature remains excessively high), the error may continue to be reported, preventing the motor from entering the operational state. In such situations, it is necessary to wait for the motor to cool down before it can resume operation.

Command Name	Restore Factory Settings
Command Message (hex)	0D
Command Reply(hex)	None
Command Description	Upon receiving this command, the motor will restore parameters from the Flash memory and save them back to the Flash. This operation must be executed while the motor is in the stopped state.

Command Name	Save Parameters to Flash
Command Message (hex)	0E
Command Reply(hex)	None
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. Upon receiving this command, the motor executes the action of saving parameters to Flash. This operation must be performed while the motor is in the stopped state.

Command Name	Restore Factory Settings
Command Message (hex)	0F
Command Reply(hex)	None
Command Description	Upon receiving the command, the motor will restore default parameters without saving them to Flash. This operation must be executed while the motor is in the stopped state. If saving to Flash is required, the 0x0E command must be used subsequently to save the parameters.

Command Name	Get Speed Loop Proportional Gain
Command Message (hex)	10
Command Reply(hex)	10 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the speed loop proportional gain.

Command Name	Get Speed Loop Integral Gain
Command Message (hex)	11
Command Reply(hex)	11 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the speed loop integral gain.

Command Name	Get Position Loop Proportional Gain
Command Message (hex)	12
Command Reply(hex)	12 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the position loop proportional gain.

Command Name	Get Position Loop Derivative Gain
Command Message (hex)	13
Command Reply(hex)	13 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the position loop derivative gain.

Command Name	Get Bus Voltage
Command Message (hex)	14
Command Reply(hex)	14 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the bus voltage value, with a unit of 1V.

Command Name	Get Maximum Acceleration
Command Message (hex)	16
Command Reply(hex)	16 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the maximum acceleration parameter. Note that this parameter is only valid in CAN-controlled speed mode.

Command Name	Get Minimum Acceleration
Command Message (hex)	17
Command Reply(hex)	17 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the minimum acceleration parameter. Note that this parameter is only valid in CAN-controlled speed mode.

Command Name	Get Maximum Speed
Command Message (hex)	18
Command Reply(hex)	18 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the maximum speed parameter. Under no circumstances can the command speed of the speed

	loop exceed the value set by this parameter. The setting unit is 0.01 Hz (0.01 rps).

Command Name	Get Minimum Speed
Command Message (hex)	19
Command Reply(hex)	19 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the minimum speed parameter. Under no circumstances can the command speed of the speed loop be less than the value set by this parameter. The setting unit is 0.01 Hz (0.01 rps).

Command Name	Get Maximum Position
Command Message (hex)	1A
Command Reply(hex)	1A 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the maximum position parameter. The position limit remains active at all times. To disable it, the parameter can be set to INT32_MAX and INT32_MIN. It is recommended to set a reasonable value.

Command Name	Get Minimum Position
Command Message (hex)	1B
Command Reply(hex)	1B 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the minimum position parameter. The position limit remains active at all times. To disable it, the parameter can be set to INT32_MAX and INT32_MIN. It is recommended to set a reasonable value.

Command Name	Get Motor Temperature
Command Message (hex)	31
Command Reply(hex)	31 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current temperature of the motor. The unit is 1°C.

Command Name	Get Driver Temperature
Command Message (hex)	32
Command Reply(hex)	32 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current temperature of the driver. The unit is 1°C.

Command Name	Get Speed Loop Derivative Gain
Command Message (hex)	33
Command Reply(hex)	33 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current value of the speed loop derivative gain.

Command Name	Get Position Loop Integral Gain
Command Message (hex)	34
Command Reply(hex)	34 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current value of the position loop integral gain.

Command Name	Get Maximum Current Limit
Command Message (hex)	35
Command Reply(hex)	35 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the maximum current limit parameter. The unit is milliamperes (mA). The default value is set to 75% of the absolute maximum current limit.

Command Name	Get Minimum Current Limit
Command Message (hex)	36
Command Reply(hex)	36 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the minimum current limit parameter. The unit is milliamperes (mA). The default value is set to 75% of the absolute maximum current limit.

Command Name	Get Maximum Absolute Current Limit
Command Message (hex)	37
Command Reply(hex)	37 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the value of the maximum absolute current limit parameter. The unit is milliamperes (mA). The absolute value of any current limit parameter should always be less than the value of this maximum absolute current limit parameter.

Command Name	Get Profiled Acceleration
Command Message (hex)	3B
Command Reply(hex)	3B 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current profiled acceleration value. This parameter is valid in Profiled Position Mode and Profiled Speed Mode. The unit is cnt/s².

Command Name	Get Profiled Jerk
Command Message (hex)	3C
Command Reply(hex)	3C 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current profiled jerk value. This parameter is valid in Profiled Position Mode and Profiled Speed Mode. The unit is cnt/s².

Command Name	Get Current, Speed, and Position Feedback
Command Message (hex)	41
Command Reply(hex)	01 00 02 00 03 00 00 00
Command Description	In the 8-byte reply: Bytes 1-2 represent the feedback current, Bytes 3-4 represent the feedback speed, Bytes 5-8 represent the feedback position. All values are assembled in low-byte first, high-byte last order. For example: "01 00" represents a feedback current of 1 mA, "02 00" represents a feedback speed of 0.02 Hz, "03 00 00 00" represents a feedback position of 3 cnt.

Command Name	Get Current Position Offset
Command Message (hex)	54
Command Reply(hex)	54 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current motor position offset value. When the position offset is 0 and the position feedback is 12345, setting the position offset to 12345 will change the position feedback to 0.

Command Name	Get Motor Model Information
Command Message (hex)	64
Command Reply(hex)	64 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the motor model identification code. The host computer can parse this code to generate a human-readable string or user-friendly description.

Command Name	Get Motor Software Version Number
Command Message (hex)	65
Command Reply(hex)	65 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the code of the current motor software version number.

Command Name	Get Motor Hardware Version Number
Command Message (hex)	66
Command Reply(hex)	66 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the code of the current motor hardware version number.

Command Name	Get Electrical Angle
Command Message (hex)	6F
Command Reply(hex)	6F 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current electrical angle value. A value of 0 indicates that electrical angle alignment has not been performed and the alignment procedure is required. Typically, a non-zero value is preset at the factory and saved to Flash.

Command Name	Get Encoder Operating Mode
Command Message (hex)	70
Command Reply(hex)	70 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current encoder operating mode.

Command Name	Get 8-Byte Command Control Mode
Command Message (hex)	71
Command Reply(hex)	71 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current 8-byte command control mode. A value of 0 indicates Force-Position Hybrid Mode, while a value of 1 indicates Profiled Position Mode.

Command Name	Get Encoder Battery Voltage
Command Message (hex)	78
Command Reply(hex)	78 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current encoder battery voltage, with a unit of 0.01 V.

Command Name	Get Inner Encoder Position
Command Message (hex)	79
Command Reply(hex)	79 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current inner encoder

	position value.

Command Name	Get Current Loop KP
Command Message (hex)	85
Command Reply(hex)	85 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current value of the current loop proportional gain (KP).

Command Name	Get Current Loop KI
Command Message (hex)	86
Command Reply(hex)	86 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current value of the current loop integral gain (KI).

Command Name	Get Maximum Voltage
Command Message (hex)	8A
Command Reply(hex)	8A 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current maximum voltage value. A voltage alarm will be triggered if the actual voltage exceeds this value.

Command Name	Get Minimum Voltage
Command Message (hex)	8C
Command Reply(hex)	8C 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current minimum voltage value. A voltage alarm will be triggered if the actual voltage falls below this value.

Command Name	Get Motor Maximum Temperature
Command Message (hex)	8F
Command Reply(hex)	8F 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current maximum temperature threshold of the motor, with a unit of 1°C. A temperature alarm will be triggered if the actual motor temperature exceeds this value.

Command Name	Get Driver Maximum Temperature
Command Message (hex)	93
Command Reply(hex)	93 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current maximum temperature threshold of the driver, with a unit of 1°C. A temperature alarm will be triggered if the actual driver temperature exceeds this value.

Command Name	Get Inner Encoder Counts Per Revolution
Command Message (hex)	94
Command Reply(hex)	94 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current counts per revolution value of the inner encoder.

Command Name	Get Inner Encoder Multi-Turn Value
Command Message (hex)	95
Command Reply(hex)	95 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current multi-turn value of the inner encoder.

Command Name	Get Outer Encoder Single-Turn Value
Command Message (hex)	96
Command Reply(hex)	96 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current single-turn value of the outer encoder.

Command Name	Get Outer Encoder Multi-Turn Value
Command Message (hex)	97
Command Reply(hex)	97 00 00 00 00
Command Description	In the 5-byte reply, the first byte remains unchanged from the received command message. The remaining 4 bytes form a 32-bit integer, assembled in low-byte first, high-byte last order. The reply data is a signed 32-bit integer representing the current multi-turn value of the outer encoder.

HONPINE
鸿 磐

7.2. Five-Byte Command

Command Name	Motor Disable
Command Message (hex)	02 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. In this command, the parameter is meaningless. Upon receiving the command, the motor will be disabled immediately. If a brake is present, it will also be powered off to lock the motor and prevent rotation.

Command Name	Set Target Current
Command Message (hex)	1C 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. Upon receiving the command, the motor enters current control mode. The command parameter is used to set the target current. The final current output is subject to internal limits.

Command Name	Set Target Speed
Command Message (hex)	1D 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. If the command parameter is within the set speed limit range, the motor will enter speed control mode upon receiving the command, and the parameter will be used to set the target speed. If the command parameter is outside the set speed limit range, the motor's command state will remain unchanged upon receiving the command (if it was already in speed mode before receiving the command, the motor will continue to operate in speed mode with the target speed unchanged).

Command Name	Set Target Position
Command Message (hex)	1E 00 00 00 00
Command Reply(hex)	None

Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. If the command parameter is within the set position limit range, the motor will enter position control mode upon receiving the command, and the parameter will be used to set the target position. If the command parameter is outside the set position limit range, the motor's command state will remain unchanged upon receiving the command (if it was already in position mode before receiving the command, the motor will continue to operate in position mode with the target position unchanged).

Command Name	Set Maximum Current
Command Message (hex)	20 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the maximum current limit value.

Command Name	Set Minimum Current
Command Message (hex)	21 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the minimum current limit value.

Command Name	Set Acceleration
Command Message (hex)	22 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the acceleration value. This parameter is only effective in speed mode.

Command Name	Set Deceleration
Command Message (hex)	23 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the deceleration value. This parameter is only effective in speed mode.

Command Name	Set Maximum Speed
Command Message (hex)	24 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the maximum speed limit value.

Command Name	Set Minimum Speed
Command Message (hex)	25 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the minimum speed limit value.

Command Name	Set Maximum Position
Command Message (hex)	26 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the maximum position limit value.

Command Name	Set Minimum Position
Command Message (hex)	27 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the minimum position limit value.

Command Name	Set Speed Loop Proportional Gain
Command Message (hex)	29 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the speed loop proportional gain coefficient.

Command Name	Set Speed Loop Integral Gain
Command Message (hex)	2A 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the speed loop integral gain coefficient.

Command Name	Set Position Loop Proportional Gain
Command Message (hex)	2B 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the position loop proportional gain coefficient.

Command Name	Set Position Loop Derivative Gain
Command Message (hex)	2D 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the position loop derivative gain coefficient.

Command Name	Set CAN Communication ID
Command Message (hex)	2E 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the CAN communication ID. The new ID takes effect immediately, but this command does not save the setting to Flash. To retain the setting after power-off, the 0x0E save command must be used.

Command Name	Set Motor Electrical Angle (If electrical angle writing is required, please consult the original manufacturer's technical staff for confirmation)
Command Message (hex)	3A 00 00 00 00
Command Reply(hex)	3A 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the motor electrical angle. It is generally used to clear the motor electrical angle (set to 0) or to reset the electrical angle after a software update (the actual electrical angle value must be recorded before the update). The following sequence must be sent in order: Send: 3A 00 00 00 00 → Reply: 3A 00 00 00 00 Send: 3A 01 00 00 00 → Reply: 3A 01 00 00 00 Send: 3A 02 00 00 00 → Reply: 3A 02 00 00 00 Send: 3A 03 00 00 00 → Reply: 3A 03 00 00 00 Send: 3A 00 00 00 00 → Reply: 3A 00 00 00 00 In the final command, a parameter of 0 clears the motor electrical angle, while a parameter set to the previously recorded electrical angle value resets the electrical angle. Note that arbitrary settings may cause the motor to fail to operate or result in high operating current and overheating.

Command Name	Set Profiled Acceleration
Command Message (hex)	3B 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the profiled acceleration value. This parameter is only valid in Profiled Position Mode and Profiled Speed Mode, with the unit of cnt/s^2.

Command Name	Set Profiled Jerk
Command Message (hex)	3C 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the profiled jerk value. This parameter is only valid in Profiled Position Mode and Profiled Speed Mode, with the unit of cnt/s^3.

Command Name	Set CAN Communication Baud Rate
Command Message (hex)	3F 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is set as the CAN communication baud rate. The new baud rate takes effect immediately but is not saved to non-volatile memory. To retain the setting after power-off, establish communication with the motor using the newly set baud rate after configuration, then use the 0x0E command to save parameters to Flash.

Command Name	Set Current and Get Current, Speed, Position Feedback
Command Message (hex)	42 00 00 00 00
Command Reply(hex)	01 00 02 00 03 00 00 00
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The motor will switch to current control mode, and the command parameter is used to set the target current. In the 8-byte reply: Bytes 1-2 represent the feedback current, in mA. Bytes 3-4 represent the feedback speed, in 0.01 Hz (0.01 rps). Bytes 5-8 represent the feedback position, in encoder counts. Example reply "01 00 02 00 03 00 00 00" indicates: Current: 1 mA Speed: 0.02 Hz Position: 3 encoder counts

Command Name	Set Speed and Get Current, Speed, Position Feedback
Command Message (hex)	43 00 00 00 00
Command Reply(hex)	01 00 02 00 03 00 00 00
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. If the command parameter is within the maximum and minimum speed range, the motor will switch to speed control mode, and the command parameter is used to set the target speed. In the 8-byte reply: Bytes 1-2 represent the feedback current in mA. Bytes 3-4 represent the feedback speed in 0.01 Hz (0.01 rps). Bytes 5-8 represent the feedback position in encoder counts. Example reply "01 00 02 00 03 00 00 00" indicates: Current: 1 mA Speed: 0.02 Hz Position: 3 encoder counts If the command parameter is outside the set speed limit range, the motor's command state will remain unchanged upon receiving the command (if it was already in speed mode before receiving the command, the motor will continue to operate in speed mode with the target speed unchanged).

Command Name	Set Position and Get Current, Speed, Position Feedback
Command Message (hex)	44 00 00 00 00
Command Reply(hex)	01 00 02 00 03 00 00 00
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. If the command parameter is within the maximum and minimum position range, the motor will switch to position control mode, and the command parameter is used to set the target position. In the 8-byte reply: Bytes 1-2 represent the feedback current in mA. Bytes 3-4 represent the feedback speed in 0.01 Hz (0.01 rps). Bytes 5-8 represent the feedback position in encoder counts. Example reply "01 00 02 00 03 00 00 00" indicates: Current: 1 mA Speed: 0.02 Hz Position: 3 encoder counts If the command parameter is outside the set position limit range, the motor's command state will remain unchanged upon receiving the command (if it was already in position mode before receiving the command, the motor will continue to operate in position mode with the target position unchanged).

Command Name	Set Position Loop Feedforward
Command Message (hex)	47 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the position loop feedforward value.

Command Name	Set Motor Position Offset
Command Message (hex)	53 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the motor position offset value, which adjusts the mechanical reference point corresponding to the motor's zero feedback position. When setting the offset, the original offset value is accumulated with the current position. It is recommended to set the offset value within ±180 degrees of a single rotation.

Command Name	Set Position Limit Function
Command Message (hex)	55 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to configure the position limit function. Set to 0 to disable the position limit function, set to 1 to enable the position limit function.

Command Name	Set Encoder Operating Mode
Command Message (hex)	70 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the dual encoder operating mode: 0: 0–360° unlimited mode (Whether multi-turn values are retained after power loss depends on whether the encoder supports battery installation. Additionally, if the motor rotation angle remains within 0–360°, no battery is needed to retain the current motor angle). 1: ±180° operating mode, default mode (Whether multi-turn values are retained after power loss depends on whether the encoder supports battery installation. Additionally, if the motor rotation angle remains within ±180°, no battery is needed to retain the current motor angle). After setting to ±180° mode, the position command range is ±131072 (exclusive). Considering PID parameters and load variations, it is recommended to limit the range to within ±130000.

Command Name	Set 8-Byte Command Control Mode
Command Message (hex)	71 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the 8-byte command control mode: 0: Force-Position Hybrid Mode 1: Profiled Position Mode

Command Name	Set Current Loop Proportional Gain
Command Message (hex)	83 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the current loop proportional gain (KP) coefficient.

Command Name	Set Current Loop Integral Gain
Command Message (hex)	84 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the current loop integral gain (KI) coefficient.

Command Name	Set Maximum Voltage
Command Message (hex)	87 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the maximum voltage value. A voltage over-limit alarm will be triggered when the actual voltage exceeds the set value.

Command Name	Set Minimum Voltage
Command Message (hex)	89 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the minimum voltage value. A voltage over-limit alarm will be triggered when the actual voltage falls below the set value.

Command Name	Set Motor Maximum Temperature
Command Message (hex)	8D 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the maximum allowable temperature for the motor. A temperature over-limit alarm will be triggered when the actual temperature exceeds the set value.

Command Name	Set Driver Maximum Temperature
Command Message (hex)	91 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code, and the following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. The command parameter is used to set the maximum allowable temperature for the driver. A temperature over-limit alarm will be triggered when the actual temperature exceeds the set value.

7.3. Six-Byte Command

Command Name	Set/Read Motor Reduction Ratio
Command Message (hex)	41 20 00 00 00 00 41 40 00 00 00 00
Command Reply(hex)	None 41 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit integer parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the motor reduction ratio. Valid reduction ratios are 51, 81, 101, 121. Setting any other value is not permitted.

Command Name	Set/Read Force-Position Hybrid Mode KP
Command Message (hex)	42 20 00 00 00 00 42 40 00 00 00 00
Command Reply(hex)	None 42 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the proportional gain (KP) for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Force-Position Hybrid Mode KD
Command Message (hex)	43 20 00 00 00 00 43 40 00 00 00 00
Command Reply(hex)	None 43 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the derivative gain (KD) for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Force-Position Hybrid Mode KT
Command Message (hex)	44 20 00 00 00 00 44 40 00 00 00 00
Command Reply(hex)	None 44 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the KT parameter for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Force-Position Hybrid Mode Tmin
Command Message (hex)	46 20 00 00 00 00 46 40 00 00 00 00
Command Reply(hex)	None 46 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the Tmin parameter for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Force-Position Hybrid Mode Tmax
Command Message (hex)	47 20 00 00 00 00 47 40 00 00 00 00
Command Reply(hex)	None 47 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the Tmax parameter for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Force-Position Hybrid Mode Imin
Command Message (hex)	48 20 00 00 00 00 48 40 00 00 00 00
Command Reply(hex)	None 48 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the Imin parameter for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Force-Position Hybrid Mode Imax
Command Message (hex)	49 20 00 00 00 00 49 40 00 00 00 00
Command Reply(hex)	None 49 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the Imax parameter for Force-Position Hybrid Mode. This parameter is only effective in Force-Position Hybrid Mode.

Command Name	Set/Read Motor Temperature Sensor Type
Command Message (hex)	96 20 00 00 00 00 96 40 00 00 00 00
Command Reply(hex)	None 96 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the motor temperature sensor type. Only values 1 or 2 can be set. Incorrect settings will cause abnormal motor temperature detection, and setting values other than 1 or 2 will forcibly disable the motor temperature detection function (displaying motor temperature as 0).

Command Name	Set/Read Position PID Dead Zone Function Switch
Command Message (hex)	97 20 00 00 00 00 97 40 00 00 00 00
Command Reply(hex)	None 97 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the position PID dead zone function switch: 0: Disables the function 1: Enables the function

Command Name	Set/Read Position PID Dead Zone Minimum Value
Command Message (hex)	98 20 00 00 00 00 98 40 00 00 00 00
Command Reply(hex)	None 98 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the minimum value for the position PID dead zone function.

Command Name	Set/Read Position PID Dead Zone Maximum Value
Command Message (hex)	99 20 00 00 00 00 99 40 00 00 00 00
Command Reply(hex)	None 99 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the maximum value for the position PID dead zone function.

Command Name	Set/Read EtherCAT Slave Address Declaration
Command Message (hex)	9A 20 00 00 00 00 9A 40 00 00 00 00
Command Reply(hex)	None 9A 40 00 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the EtherCAT slave address declaration. This can be understood as the EC slave address, but its usage differs significantly from CAN ID. The address will only be actually used when the master station is configured to utilize the slave address declaration, whereas with CAN, communication with the device becomes impossible without using the specific ID once it is set.

Command Name	Set/Read Position Control Mode
Command Message (hex)	9D 20 02 00 00 00 9D 40 00 00 00 00
Command Reply(hex)	None 9D 40 02 00 00 00
Command Description	The first byte of the received command is the instruction code, and the second byte is the read/write flag. The following 4 bytes are combined into a 32-bit floating-point parameter in low-byte first, high-byte last order. In the reply, the last 4 bytes represent the return value. When the second byte is 0x20: Writes the parameter. When the second byte is 0x40: Reads the parameter. The command parameter is used to read or set the position control mode: Write 0: Full-closed loop outer position with 262144 resolution Write 1: Inner position compatible with full-closed loop resolution (262144) Write 2: Inner position compatible with single encoder resolution (65536 × reduction ratio)

7.4. Seven-Byte Command

Command Name	Feedforward Current Position Mode
Command Message (hex)	10 01 00 00 00 02 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code. Bytes 2–5 represent the position command, and bytes 6–7 represent the current feedforward value. The unit of the feedforward current is 0.1 A.

Command Name	Feedforward Speed Position Mode
Command Message (hex)	11 01 00 00 00 02 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code. Bytes 2–5 represent the position command, and bytes 6–7 represent the speed feedforward value. The unit of the feedforward speed is 0.01 Hz / 0.01 rps.

Command Name	Speed Limited Position Mode
Command Message (hex)	58 00 00 00 00 00 00
Command Reply(hex)	None
Command Description	The first byte of the received command is the instruction code. Bytes 2–5 represent the position command, and bytes 6–7 specify the maximum speed limit. This command will modify the motor's maximum and minimum speed limits. The unit for the speed limit is 0.01 Hz (0.01 rps).

7.5. Eight-Byte Command

To use Profiled Position Mode or Force-Position Hybrid Mode, before sending 8-byte commands, it is necessary to first set and verify the current mode using the 5-byte command "Set 8-Byte Command Control Mode" and the 1-byte command "Get 8-Byte Command Control Mode".

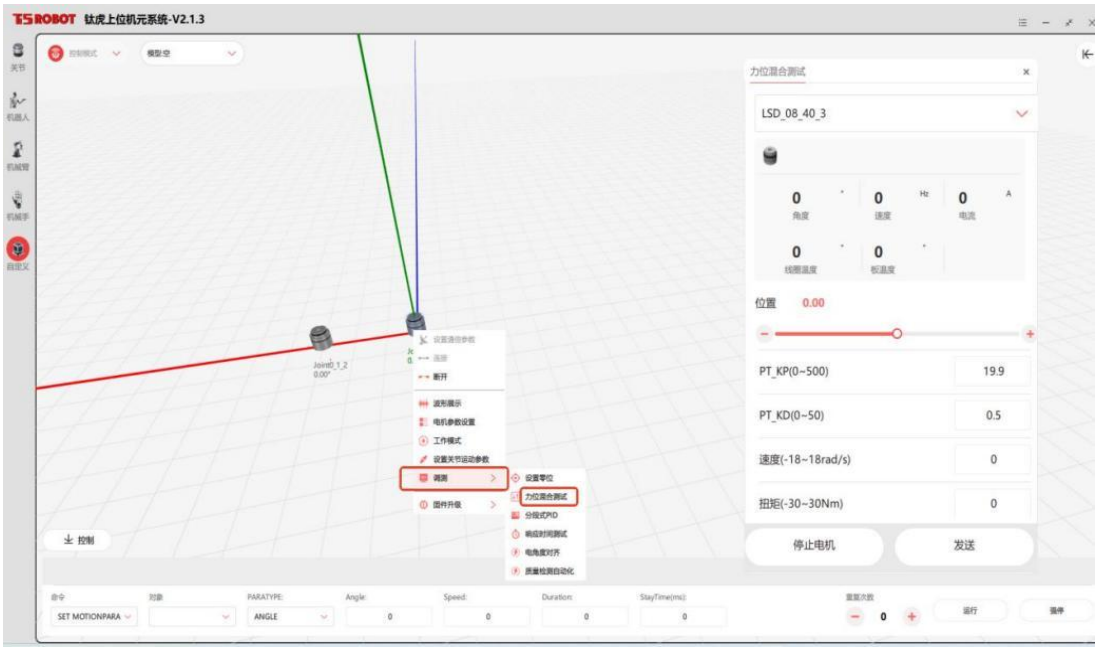
No reply or a return value of 0 indicates Force-Position Hybrid Mode.

A return value of 1 indicates Profiled Position Mode.

For the host computer to switch to Profiled Position Mode, set the 8-Byte Command Control Mode to 1 in the lower-left corner and press Enter to activate.



Force-Position Hybrid Mode requires debugging using the Meta System V2.1.3 software. This software is currently in the development and testing phase, with limited language support and documentation. If you need to use this feature, please contact us to obtain the software and the Force-Position Hybrid Mode Meta System operation manual.



7.5.1. Profiled Position Mode

In the profile position mode, both the sent command and the returned data are eight bytes, and the format is defined as follows:

Command Format (Transmission)

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Assembled in low-byte first, high-byte last order: Target Position				Profile Velocity		Control Word	

l Target Position is in encoder count units.

l Profile Velocity is in units of 0.01 Hz, calculated based on the position sensor speed (for dual encoders, it is calculated using the output side speed; for single encoders, it is calculated using the motor side speed).

l Control Word 0x003F operates in immediate mode (upon receiving a second position command during the motion of the first position, it immediately starts planning the trajectory for the second position).

Setting bit 0x0100 enables the halt function (decelerates to zero speed at the set acceleration).

Setting bit 0x0040 enables the relative position function.

Reply Data Format

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Assembled in low-byte first, high-byte last order: Feedback Position				Feedback Speed (Motor Speed)		Status Word	

l Feedback Position is in encoder count units

l Feedback Speed is in units of 0.01 Hz, calculated based on the speed sensor (always calculated using the motor side speed).

l When bit 0x1000 of the Status Word is set to 1, it indicates that a new target position has been received and trajectory planning has been updated.

When bit 0x0400 of the Status Word is set to 1, it indicates that the trajectory of the previous command has been completed.

Profiled Position Mode Control Parameters

l Profiled Acceleration (deceleration is fixed as negative acceleration):

Write parameter: 5-byte command 59 (0x3B)

Read parameter: 1-byte command 59 (0x3B)

Unit: cnt/s²

l Profiled Jerk (deceleration jerk is fixed as negative jerk):

Write parameter: 5-byte command 60 (0x3C)

Read parameter: 1-byte command 60 (0x3C)

Unit: cnt/s³

Additional Notes:

- The velocity profile follows a double S-curve with a 7-phase trajectory planning algorithm. (If conditions are not met, certain phases may be omitted based on the situation. The actual operation may not necessarily consist of seven phases, but the planned trajectory ensures continuous acceleration, resulting in minimal mechanical impact.)

- The following parameter pairs are fixed as opposites (A = B × (-1)):
  Acceleration and Deceleration,
  Jerk and Deceleration Jerk.

- The current motor software supports replanning functionality. This means that if the first position is not fully executed, the system can proceed to the second position. During replanning, the command position and velocity remain continuous. However, since the algorithm assumes an initial acceleration of 0 at the start of replanning, performing replanning when the acceleration is non-zero may result in a longer travel distance and potential acceleration discontinuities.

- If replanning occurs from a high speed to a nearby target point, it may be impossible to decelerate within the specified distance using the configured acceleration and jerk parameters. In such cases, the system will decelerate according to the configured acceleration and jerk (which may cause overshooting beyond the new setpoint), followed by an automatic secondary planning phase to reach the new target point.

HONPINE
鸿 磐

8. Alarm Codes and Handling Methods

8.1. Inspection and Handling of Alarms Encountered During EtherCAT Communication

When using EtherCAT communication and an alarm occurs, it is essential to focus on checking the 0x603F error code. When a motor fault occurs, a value will be retained in 0x603F until the error is cleared.

0x1001 Error Register

Bit Definition	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Meaning	Encoder Alarm	0	0	Communication Error	Temperature Out of Range	Voltage Out of Range	Overcurrent	General Error

0x603F Error Codes

Error Codes	Name	Meaning	Cause and Handling Method
0x2310	Continuous Over Current	Motor Overload	Sustained high current output at the motor side; Check for mechanical blockages / Select a motor model with higher power rating.
0x3120	Mains Under Voltage	Low Power Supply Voltage	Transient or prolonged low power supply voltage; Check if the supply voltage is normal / Connect a large capacitor in parallel at the power input / Select a power supply with higher current output capability / Choose a power supply with higher voltage output (must be within the rated voltage range) / Reduce the motor acceleration.
0x3110	Mains Over Voltage	High Power Supply Voltage	Transient or prolonged high power supply voltage; Check if the supply voltage is normal / Connect a large capacitor in parallel at the power input / Connect an energy discharge module in parallel at the power input / Select a power supply with lower voltage output (must be within the rated voltage range) / Reduce the motor deceleration.

0x4200	Temperature Device	Motor Temperature Alarm	Motor temperature too high or too low; Check if the motor temperature is normal / Check if the ambient temperature is normal / Consider whether heat dissipation or insulation measures are needed.
0x4300	Temperature Drive	Driver Temperature Alarm	Driver temperature too high or too low; Check if the motor temperature is normal / Check if the ambient temperature is normal / Consider whether heat dissipation or insulation measures are needed.
0x7300	Sensor Error	Encoder Alarm	Encoder data reading error; Check the encoder communication cable.
0x5220	Control Computing Circuit	Control Circuit CPU Error	PU operating load too high; Check if the communication cycle can be adjusted / Check if the PDO content can be streamlined by removing unnecessary items / Record detailed operating condition information and contact the manufacturer's technical support.
0x2320	Short Circuit Earth Leakage Motor Side	Output Short-Circuit Alarm	Instantaneous excessive output current on the motor side; Check for short circuits on the motor side / Inspect the motor for damage.
0x6100	Internal Software	Internal Software Error	Collect detailed operating condition information and contact the manufacturer's technical support.
0x7500	Communication	Communication Error	Abnormal transition from OP state to non-OP state; Check if the communication cable's shielding is properly connected / Verify the device is correctly grounded / Inspect the communication cable for damage or poor contact / Check for loose or poor contact at communication connectors / Consider whether it is necessary to add ferrite

			beads / Check for EtherCAT communication ALStatusCode errors.
0x8611	Following Error	Following Error Too Large	The difference between the position command and the actual position exceeds the set following error range and persists for the set following error duration; Check if the following error range and time settings are reasonable / Verify if the torque, speed, and position loop control gain parameters are appropriately set / Check for mechanical limitations preventing the motor from reaching the specified position / Verify if an inappropriate electronic gear ratio is set / Confirm the encoder resolution setting is correct.

HONPINE
鸿 磐

8.2. Checking CAN Communication Errors Using USB-CAN-TOOL

8.2.1. Checking Error Code 0x0A via CAN Communication

Enter the returned command value into a calculator and check the error by viewing the corresponding binary code.

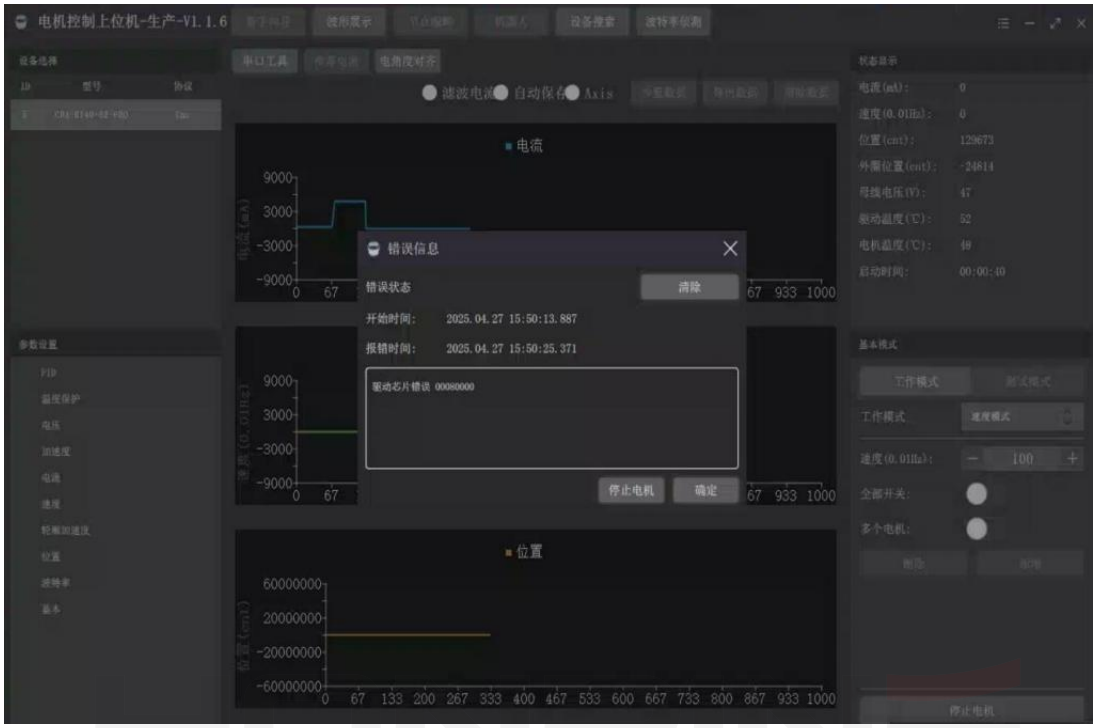
bit位	故障原因	造成故障的原因	解决方法
0	代表电机内部软件错误，如电机运行时写入FLASH等	电机在运行中保存参数	需停止运行再保存
1	代表过压	电压高于56V设定值	①将电源电压调到额定电压（建议36-48V） ②加电容或稳压模块
2	代表欠压	电压低于20V设定值	①将电源电压调到额定电压（建议36-48V） ②加电容
4	代表启动错误		检查线路，重新启动
5	代表速度反馈错误		暂无
6	代表过流	负载过大，短路	①减轻负载测试 ②寄回原厂需排查
7	代表软件错误，如电机运行时写入FLASH等	电机在运行中保存参数	电机在运行中保存参数，需停止运行再保存
16	代表编码器通讯错误	编码器损坏，编码器线损坏，控制板损坏	寄回原厂需排查
17	代表电机温度过高	电机温度超过100℃，过热，工况不好，长时间运行，温度传感器坏	①待温度下降后再运行测试 ②寄回原厂需排查
18	代表电路板温度过高	板子温度超过90℃，过热，工况不好，长时间运行，温度传感器坏	①待温度下降后再运行测试 ②寄回原厂需排查
19	代表驱动芯片错误	负载较大，急加速、减速，功率较大情况下引起报错	①加电容或稳压模块 ②不要突然速度过快或者过慢调试运行 ③寄回原厂排查

For example, the error shown in the figure is an encoder communication error.



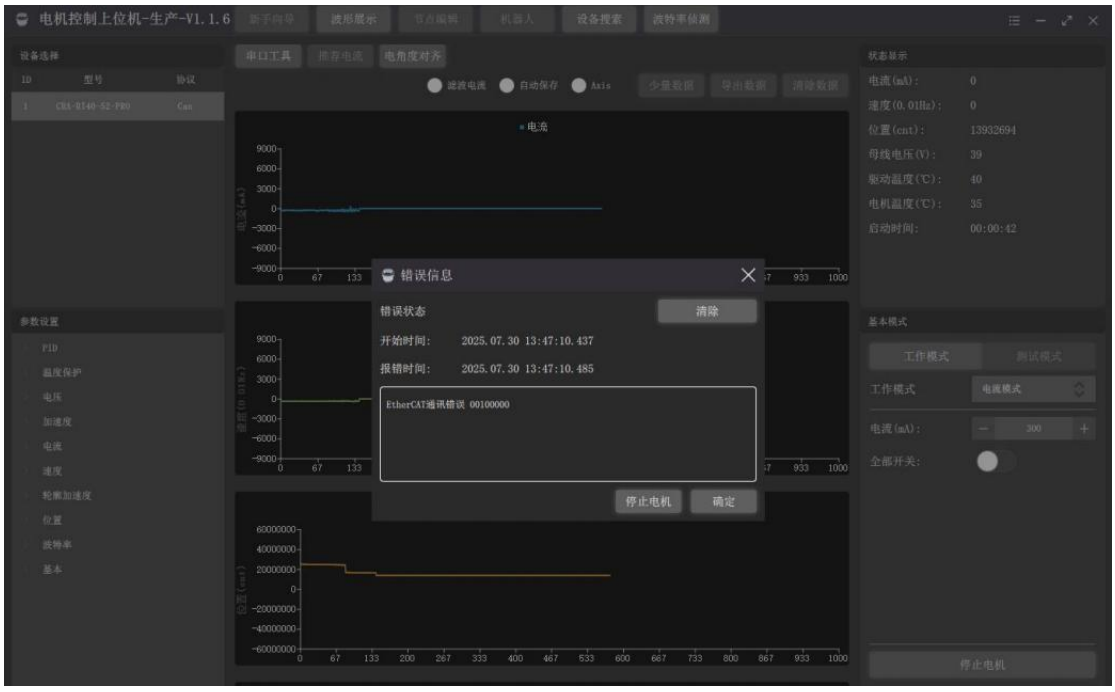
8.3.3. Driver Chip Error

Possible causes of the error: starting speed too fast, sustained overcurrent operation, emergency stop, collision, or motor stalling (ensure the program version is correct).



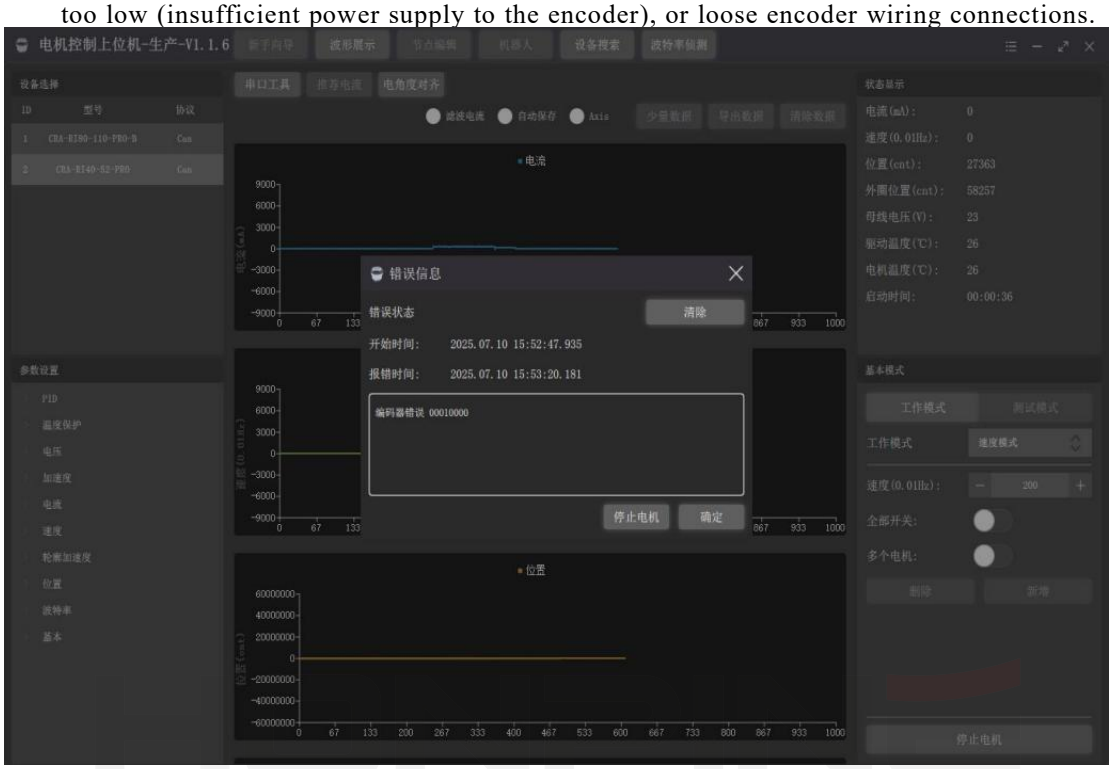
8.3.4. EtherCAT Communication Error

Possible causes: Switching between EtherCAT and CAN communication, or errors in EtherCAT communication. Solution: Clear the error. EtherCAT and CAN ports cannot be used to control the motor simultaneously. When controlling via EtherCAT, the CAN port can only be used to monitor the motor status.



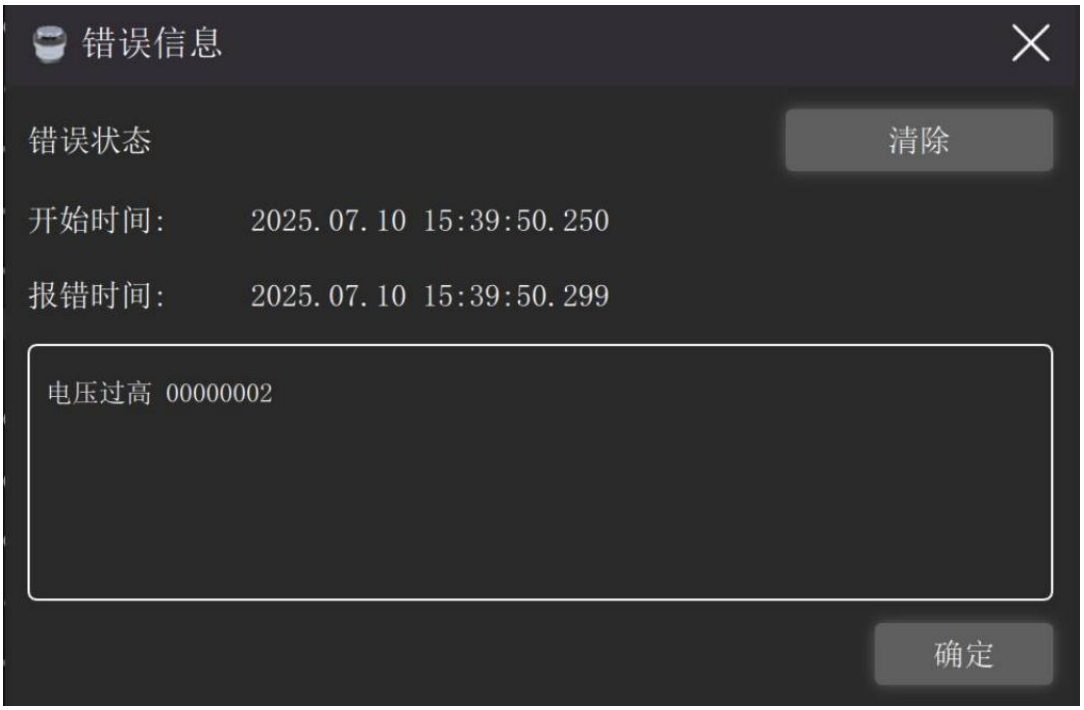
8.3.5. Encoder Error

The encoder did not receive a verification response. Possible causes include: multiple motors connected in series, current/voltage settings of the DC regulated power supply set too low (insufficient power supply to the encoder), or loose encoder wiring connections.



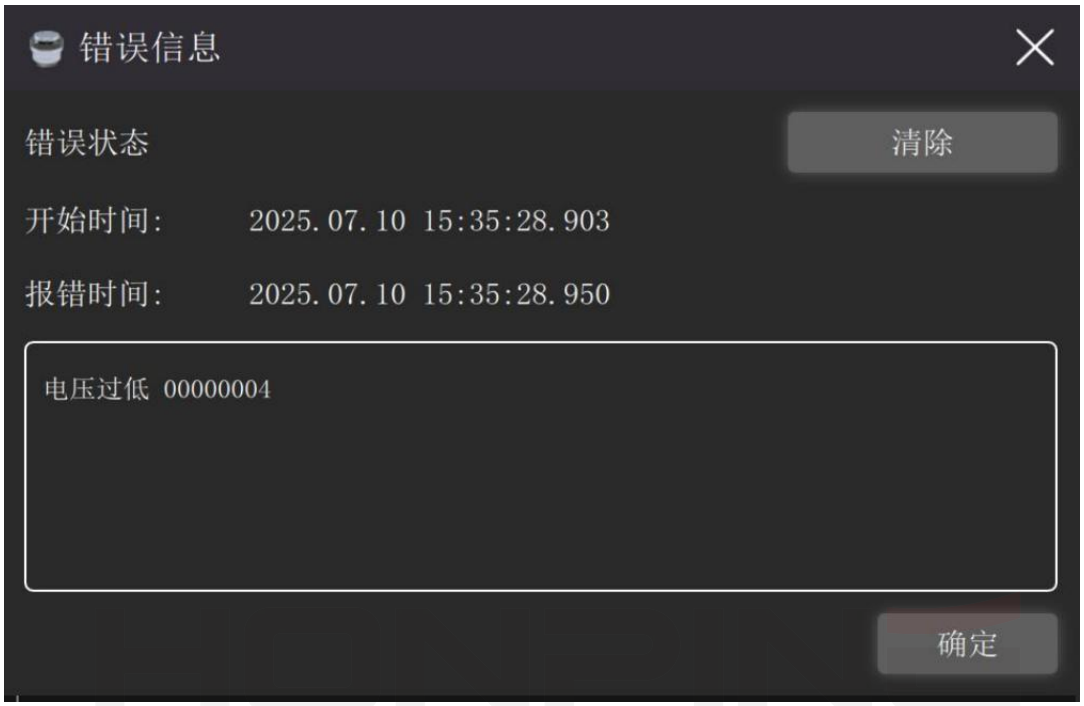
8.3.6. Overvoltage Error

When the host computer reports an overvoltage error while starting the motor:
① Check whether the power supply voltage is within the rated range (recommended 36V –48V).
② Connect an external braking resistor.



8.3.7. Undervoltage Error

When the host computer reports an undervoltage error while starting the motor:
① Check whether the power supply voltage is within the rated range (recommended 36V –48V).
② Connect an external capacitor.



鸿磐

8.4. Status Lights: Green & Blue

8.4.1. Green Light

The green light serves as the software status indicator.

- Slow Flash (500ms on, 500ms off): Indicates normal operation of both software and hardware.
- Fast Flash (100ms on, 100ms off): Indicates an error has occurred and requires handling.
- Double Flash (50ms on, 50ms off, 50ms on, 1850ms off): Indicates no operational error, but there is an ESC (EtherCAT Slave Controller) connection fault. A solid green light indicates firmware upgrade mode. ESC connection faults are generally caused by EEPROM errors or hardware malfunctions.

- Fast Flash (100ms on, 100ms off): Indicates that an error has occurred and requires handling.
- Double Flash (50ms on, 50ms off, 50ms on, 1850ms off): Indicates no operational error, but an ESC connection fault exists. A solid light indicates firmware upgrade mode. ESC connection faults are generally caused by EEPROM errors or hardware malfunctions.

8.4.2. Blue Light

The RUN indicator states are specified in Table 7.

Table 7: RUN Indicator States

Indicator states	Slave State	Description	Requirement
Off	INITIALISATION	The device is in state INIT	Mandatory
Blinking	PRE-OPERATIONAL	The device is in state PRE-OPERATIONAL	Mandatory
Single Flash	SAFE-OPERATIONAL	The device is in state SAFE-OPERATIONAL	Mandatory
On	OPERATIONAL	The device is in state OPERATIONAL	Mandatory
Flickering	INITIALISATION or BOOTSTRAP	The device is booting and has not yet entered the INIT state, or: The device is in state BOOTSTRAP. Firmware download operation in progress	Optional

9. FAQ Common Questions

(1) What are single-encoder motors and dual-encoder motors?

Motors are categorized into single-encoder and dual-encoder versions based on the encoders used.

Single-encoder motors have one encoder integrated on the motor side inside the joint actuator. There is no encoder on the reducer side (i.e., the output end).

Dual-encoder motors have two encoders inside the joint actuator: one on the motor side and one on the reducer side.

Single-encoder Details: Uses a magnetic encoder. Total encoder count: 1. Components: 1 high-speed magnetic ring + 1 encoder signal acquisition board.

Dual-encoder Details: Total encoder count: 1

For models 80-110 and below: Use magnetic encoders.

Components: 1 high-speed magnetic ring + 1 low-speed magnetic ring + 1 encoder signal acquisition board.

For other models: Use inductive encoders.

Components: 1 high-speed chopper disk + 1 low-speed chopper disk + 1 encoder signal acquisition board.

Note: The chopper disk acts as the rotor.

(2) What is Reduction Ratio?

The reduction ratio is a transmission ratio in speed reduction mechanisms, representing the ratio between the instantaneous input speed and output speed in a reducer system. For example, a reducer with a 50:1 reduction ratio means the input shaft rotates 50 times for every single rotation of the output shaft. This reduction mechanism increases output torque while decreasing output speed to meet various application requirements. (Note: Add 1 to the ratio during calculation).

For dual-encoder motors, it is essential to ensure that the reducer's physical reduction ratio matches the ratio set in the software. All motors can have their reduction ratio modified using a 6-byte Command Message with the following instruction:

指令名称	设置/读取电机减速比
指令报文(hex)	41 20 00 00 00 00 41 40 00 00 00 00
指令回复(hex)	无 41 40 00 00 00 00
指令说明	收到指令的第 1 字节为指令代码，第 2 字节为读写标志，后面 4 个字节按低位在前高位在后的顺序组合成一个 32 位整数指令参数，回复中后面 4 字节为返回值。第 2 字节为 0x20 时写入参数，0x40 时读取参数。 指令参数用于读取/设置电机减速比。实际减速比有 51/81/101/121，不允许设置其他值。

(3) What is Enable/Brake (Holding Brake)?

Enable: "Enable" refers to allowing the motor driver or controller to provide power and control signals to the motor, thereby enabling the motor to respond to commands and operate. In other words, enabling the motor activates the motor control system, preparing it to receive commands such as speed, position, or torque and execute actions accordingly.

Disable: The opposite of enable is "disable," which means stopping the supply of power or control signals to the motor, placing it in a standby state. In this state, the motor will not respond to any new commands.

Brake (Holding Brake): Some motor models are equipped with a brake device used to automatically lock the rotor when the motor loses power, preventing it from rotating freely. When the motor is enabled, the brake is released; when disabled, the brake is re-engaged to secure the motor position.

Note: To prevent the motor from dropping during start/stop moments, it is necessary to increase the motor position loop proportional gain or add feedforward.

Relationship between Enable and Brake (Holding Brake):
Power on the motor -> Enable -> Energize the brake -> Release the brake.
Power off the motor -> Disable -> De-energize the brake -> Motor is braked.

(4) Related Calculations in Position Mode:

Single-encoder motor position calculation uses 16-bit resolution and the actual reduction ratio. For example, 65,536 pulses represent one motor revolution. Multiplying this by the motor's reduction ratio gives the number of pulses for one output shaft revolution. If the motor has a 121:1 reduction ratio, then 65,536 * 121 = 7,929,856 pulses are required for the motor output shaft to complete one revolution.

Dual-encoder motor position calculation uses 18-bit resolution and does not require considering the reduction ratio. For example, 262,144 pulses represent one revolution of the motor output shaft.

(5) Multi-turn Position Retention After Power Loss:

Single-encoder motors can preserve the current absolute position during power loss by connecting a battery. Dual-encoder motors currently do not support connecting a battery to retain the current absolute position, but they can maintain the current output shaft angle after power loss without battery connection.

This means:

Single-encoder motors (with battery connected) can support applications requiring multi-turn rotation.

Dual-encoder motors cannot support applications requiring multi-turn rotation and are limited to applications where rotation stays within ±180 degrees.

(6) Speed Unit Conversion:

Both single-encoder and dual-encoder motor speed calculations use 16-bit resolution and the actual reduction ratio. For example, 65,536 pulses per second represents one motor revolution per second. Considering the actual motor reduction ratio, if the motor has a 101:1 reduction ratio, the output shaft will complete one revolution every 101 seconds.

Motor speed essentially represents the angular velocity of the motor, physically defined as angular displacement per unit time. Consequently, various commonly used units exist depending on the chosen unit of time and displacement. Common units for motor speed include revolutions per minute (RPM), pulses per second, degrees per second, radians per second, and 0.01 Hertz, etc. Conversion formulas between these units are listed below. Conversions not explicitly listed can be derived based on their physical meanings and the existing formulas.

1 radian per second (rad/s)
= (1/(2π)) * 360 degrees per second (°/s)
= (1/(2π)) Hertz (Hz) = (1/(2π)) * 100 units of 0.01 Hz
= (1/(2π)) * 60 revolutions per minute (RPM)
Note: π is approximately 3.1415926...

Motor-side speed in 0.01 Hz corresponds to (0.01 r/s). Multiply by 0.6 to get motor-side speed in RPM. For single-encoder motors, divide this by the reduction ratio to get the output shaft speed in RPM. Then, multiply by 360 and divide by 60 to get degrees per second (°/s), or multiply by π and divide by 180 to get radians per second (rad/s). Dual-encoder motors do not require considering the reduction ratio.

For specific conversions, please refer to our motor protocol: Conversion Formulas section.

速度转换

	电机端（0.01HZ）	减速比	电机端（RPM）	输出端（RPM）	输出端（度/秒）	输出端（弧度/S）
输入	2000	51	1200.0	23.5	141.2	2.5
		81		14.8	88.9	1.6
		101		11.9	71.3	1.2
		121		9.9	59.5	1.0
		161		7.5	44.7	0.8

	输出端（RPM）	减速比	电机端（0.01HZ）
输入	30	51	2,550
		81	4,050
		101	5,050
		121	6,050

下发速度*0.6=电机RPM

Note:

When using EtherCAT communication, the actual feedback speed unit is cnt/s, while the maximum output speed at the motor end is specified in RPM

索引	0x606C	子索引	0
比特长度	32	数据类型	有符号 32 位整型
读写权限	只读	名称	Velocity Actual Value
详细说明	实际速度反馈值。单位编码器脉冲数每秒 cnt/s。		

索引	0x6080	子索引	0
比特长度	32	数据类型	无符号 32 位整型
读写权限	读写	名称	Max Motor Speed
详细说明	电机端最大输出速度。单位圈每分 RPM。		

(7) Motor Speed Limiting - Restricting Maximum Motor Speed

Can be modified via SDO parameter 0x6080 or CAN message. Default is 1200, unit is RPM (revolutions per minute). If conversion to motor output speed is required, use the actual motor reduction ratio for calculation. Our position mode operates based on this maximum speed limit.

(8) If the motor output shaft does not reach the target XX RPM speed after sending the command

Check the power supply voltage and the motor-side speed limit. Higher speeds cannot be achieved with insufficient power supply voltage; try using 48V.
Multiply XX RPM by the motor's reduction ratio to get the required motor-side speed. Check if this motor-side speed is limited by SDO parameter 0x6080. For example, to achieve 30 RPM with a 101:1 reduction ratio, the motor-side speed needs to be 3030 RPM, so set 0x6080 to 3030.

(9) Method for Setting Maximum Current Limit

For drive software version 1.7.0.0 and above, use SDO parameters 0x60E0 or 0x60E1. Unit is per mil (‰) of the maximum current. Default value is 750, representing 75% of the maximum current. For example, setting it to 912 limits the current to 91.2% of the maximum. The maximum current value can be read from SDO parameter 0x6075, unit is milliamps (mA).

(10) Output Torque Calculation Method

For drive software version 1.7.0.0 and above, both the commanded torque (0x6071) and the actual torque (0x6077) are in units of per mil (‰) of the peak start/stop torque. For example, 100 corresponds to 10% of the peak torque, 751 corresponds to 75.1%. Refer to the selection guide table for the specific model's peak start/stop torque value.

(11) Method for Saving Parameters After Power Loss

When using EtherCAT communication, all parameter writes are not saved after power loss by default. To save parameters persistently, write 1 to SDO parameter 0x2000, then read it back. A value change to 0 indicates the save operation is complete. Note: Parameters cannot be saved when the motor is enabled or in a fault state.

(12) Support for Incremental Encoders

The joint motor uses built-in encoders. Only single-encoder or dual-encoder models with absolute encoders are available. Incremental encoders are not supported. There is also no incremental encoder output interface (e.g., ABZ interface).

(13) PID Parameter Tuning

1. P is the proportional gain, I is the integral gain, D is the derivative gain.
2. A larger P parameter increases response speed but can cause oscillation if too high; too low results in slow response. The I parameter eliminates steady-state error and improves response speed but can cause oscillation if too high. The D parameter suppresses overshoot and oscillation; too small causes overshoot, too large affects response speed and can introduce noise.
3. The control system consists of three loops: from inner to outer, they are the current loop, speed loop, and position loop. Generally, the current loop is not tuned; start tuning from the speed loop.
4. Metrics for evaluating loop tuning include steady-state error (error between actual and target values after stabilization), dynamic following error (error during motion), and overshoot (amount the actual value exceeds the target before settling).

5. If there is oscillation when reaching the target in position mode, try increasing the position loop D parameter.

6. If there is constant oscillation during constant velocity phases in position mode, try increasing the speed loop P and decreasing the speed loop I parameter.

(14) Motor Abnormal Noise

Motor abnormal noise can be caused by various factors, such as a loose reducer, foreign objects in the brake friction, shaft misalignment, or foreign objects rubbing against the magnets. For joint motor operational noise issues, try reducing the speed loop proportional and integral gains, and the position loop proportional gain. Refer to the initial data in the host computer software for modification to reduce friction. First, check if abnormal noise occurs when running in position mode via the host computer. Then, investigate whether it is caused by program interpolation issues or non-smooth position commands.

HONPINE
鸿 磐