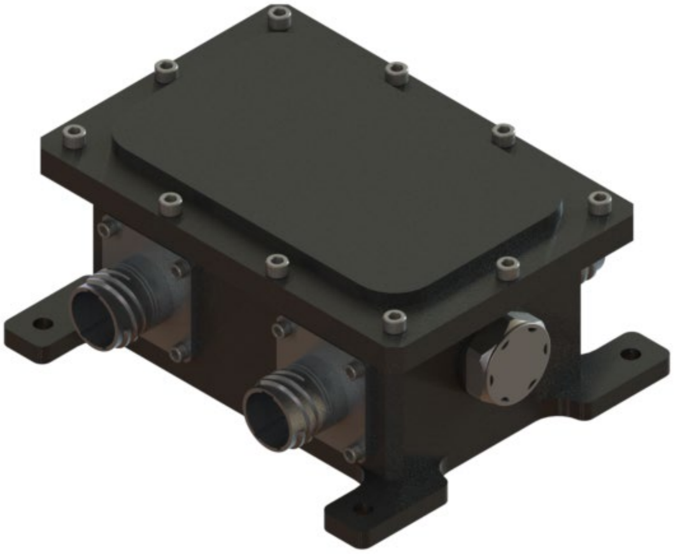
2250-A Union Place, Simi Valley, CA 93065
Tel: 800.823.3235 – sales@esimotion.com
Product of USA

Rugged Servo Drives & Control Systems for Extreme Environments

Safety Critical RS422 Protocol Specification

Document Number: 101718-00A
Date of Doc Rev: 7/30/2020



ESI MOTION CONFIDENTIAL & PROPRIETARY

WARNING: INFORMATION SUBJECT TO U.S. EXPORT CONTROL LAWS. These items are controlled by the U.S. government and authorized for export only to the country of ultimate destination for use by the ultimate consignee or end-user(s) herein identified. They may not be resold, transferred, or otherwise disposed of, to any other country or to any person other than the authorized ultimate consignee or end-user(s), either in their original form or after being incorporated into other items, without first obtaining approval from the U.S. government or as otherwise authorized by U.S. law and regulations. EAR ECCN 9E610 Applicable.

Revision History

Version	Date	Changed by	Items Changed
A	7/30/2020	Jim Jollota	Initial Release

Important Information:

ESI MOTION makes no warranty, either express or implied, including but not limited to any implied warranties of merchantability and fitness for a particular purpose, regarding any marketing materials and makes such materials available solely on an "as-is" basis. In no event shall ESI MOTION be liable to anyone for special, collateral, incidental, or consequential damages in connection with or arising out of the purchase or use of these materials, and the sole and exclusive liability of ESI MOTION, regardless of the form of action, shall not exceed the purchase price of this product. Moreover, ESI MOTION shall not be liable for any claim of any kind whatsoever against the use of these materials by any other party.

Table of Contents

1 Purpose

This document describes the software requirements of the Safety Critical RS422 protocol for ESI Motion servo drives and servo drive modules. This command interface can be used for general control of ESI Motion Servo drives and for status reporting during common operation.

2 Background

ESI Motion servo drives, modules and control systems are a complete ruggedized, off-the-shelf motor control solution which include ESI's rugged controller and power driver boards, an integrated EMI filter, military-grade submersible case, controller software, and user-friendly GUI. These systems are ideal for military, aviation, automotive or other heavy industrial applications operating in outdoor, high temperature, high vibration, or other extreme environmental conditions.

3 Support Information

- ESI Motion Controller User Manual, document number 100266

4 RS422 Overview:

4.1 Message Format

The RS422 message is composed of the following three parts:

1.	Header	(2 bytes)
2.	Body	(6 bytes)
3.	CRC	(2 bytes)
	Total	(10 bytes)

All data content is assumed to be little endian byte order.

4.1.1 Header

All messages start with a Header. Each message begins with the following 2 bytes hexadecimal sequence: 0xF0F0.

4.1.2 Body

The Body follows the Header. The body is either a 6-byte command packet (transmitted to the servo drive) or a 8-byte status packet (transmitted from the servo drive). Note that the field definitions are different in the status and command packet.

4.1.3 CRC

Following the message body is a 16-bit CRC of the message body (the 2 header bytes are not part of the CRC result). The CRC is defined as the CRC-16/CCITT-FALSE (implementation). The C-implementation is shown below and the CRC table referred to is in *Appendix A*. If the CRC calculated from the message doesn't match the CRC received, the message is discarded and ignored. For a single message CRC calculation, the start crc (starting seed) used shall be 0xFFFF.

```
/*******************************************************************************
 * @fn      crc16()
 *
 * @brief   Implements the "CRC-16/CCITT-FALSE" algorithm. The identification
 *          is shown by ASCII "123456789" = 0x29B1.
 *
 * @param   starting CRC, pointer to the buffer, and the len
 *
 * @return  The CRC-16 result
*******************************************************************************/
U16 crc16(const U16 start_crc, Uchar8 *buf_ptr, U16 len)
{
        U16 crc16 = start_crc;
        while (len--)
        {
                crc16 = crc16_table[(crc16 >> 8) ^ *buf_ptr++] ^ (crc16 << 8);
        }
        return(crc16);
}
```

4.2 Command Packet

The Command Packet is received by the ESI Controller. All commands start with the 2-byte header and end with the 2-byte CRC defined above. The Controller must be configured via HiDS as a current, velocity, or position command; see section 4.4. The command packet information is shown below.

Note all bytes shown below are in the order of transmission (Byte 0 is transmitted first)

4.2.1 Single-axis Controller (i.e. Stinger)

Signal name	Byte	Bit	Size (bits)	Data Type	Coding
Header	0-1	B[15..0]	16	Unsigned	Always 0xF0F0
Motor A Enable	2	Bit0	1	unsigned	0 - Motor Disable 1 - Motor Enable
Clear Motor A errors	2	Bit1	1	unsigned	0 - Do Not Clear Error 1 - Clear All Errors.
Command Mode A	2	Bit2-3	2	unsigned	0 – Torque Mode 1 – Velocity Mode 2 – Position Mode
Unused	2	Bits 4-7	4	N/A	Reserved
Command Motor A	3-4	B[15..0]	16	Signed	Command Mode and Scaling defined via HiDS Mode 0 – Current Command Mode 1 – Velocity Command Mode 2 – Position Command

4.2.2 Dual-axis Controller (i.e. Nova)

Signal name	Byte	Bit	Size (bits)	Data Type	Coding
Header	0-1	B[15..0]	16	Unsigned	Always 0xF0F0
Motor A Enable	2	Bit0	1	unsigned	0 - Motor Disable 1 - Motor Enable
Clear Motor A errors	2	Bit1	1	unsigned	0 - Do Not Clear Error 1 - Clear All Errors.
Command Mode A	2	Bit2-3	2	unsigned	0 – Torque Mode 1 – Velocity Mode 2 – Position Mode
Unused	2	Bits 4-7	4	N/A	Reserved
Command Motor A	3-4	B[15..0]	16	Signed	Command Mode and Scaling defined via HiDS Mode 0 – Current Command Mode 1 – Velocity Command Mode 2 – Position Command

Motor B Enable	5	Bit0	1	unsigned	0 - Motor Disabled 1 - Motor Enabled
Clear Motor B errors	5	Bit1	1	unsigned	0 - Do Not Clear Error 1 - Clear All Errors.
Command Mode B	5	Bit2-3	2	unsigned	0 – Torque Mode 1 – Velocity Mode 2 – Position Mode
Unused	5	Bits 4-7	4	N/A	Reserved
Command Motor B	6-7	B[15..0]	16	Signed	Command Mode and Scaling defined via HiDS Mode 0 – Current Command Mode 1 – Velocity Command Mode 2 – Position Command
CRC	8-9	B[15..0]	16	NA	CRC

4.2.3 Current-loop Packet Example

For example, to enable motor-A, with a +5 Amp MotorA command and to enable Motor-B with a -5Amp command, on a dual-axis controller the 10-byte packet contents would be: **F0 F0 01 88 13 01 78 EC 54 61**

Note, this example assumes the current-loop scale-factor, CommandCurrentScale = 0.001. The actual controller current command (Ma.IqPacketCommand) is the received RS422 command-word * CommandCurrentScale. The CommandCurrentScale variable can be configured to maximize your command-resolution for your torque application. Note each processor has its own CommandCurrentScale configuration.

If the Controller-configuration, Serial-packet, baud-rate, and data-bits are all valid, on the MotorAHSL page, you should see Ma.IqPacketCommand and Ma.IqCmd = 5 (or less if the MaxCurrentCommand is < 5). The Motor B current command should be -5. See trouble-shooting section below to debug.

4.2.4 Velocity-loop Packet Example

For example, to enable motor-A, with a +1000 RPM MotorA command and to enable Motor-B with a -1000 RPM command, on a dual-axis controller the 10-byte packet contents would be: **F0 F0 05 E8 03 05 18 FC 55 A0**

Note, this example assumes the velocity-loop scale-factor, CommandVelocityScale = 1. The actual controller velocity command (MaVL.RPMPacketCommand) is the received RS422 command-word * CommandVelocityScale. The CommandVelocityScale variable can be configured to maximize your command-resolution for your velocity application. Note each processor has its own CommandVelocityScale configuration.

If the Controller-configuration, Serial-packet, baud-rate, and data-bits are all valid, on the VelocityLoop page, you should see MaVL.RPMPacketCommand and MaVL.RPMCommand = 1000. The Motor B velocity command should be -1000. See trouble-shooting section below to debug.

4.2.5 Position-loop Packet Example

For example, to enable motor-A, with a +3.12 radian MotorA command and to enable Motor-B with a -3.12 Radian command, on a dual-axis controller the 10-byte packet contents would be: **F0 F0 09 8E 3F 09 72 C0 6B 7A**

Note, this example assumes the position-loop scale-factor, CommandPositionScale = 0.000191753. The actual controller position command (MaPL.RadiansPacketCommand) is the received RS422 command-word * CommandPositionScale. The CommandPositionScale variable can be configured to maximize your command-resolution for your position application. Note each processor has its own CommandPositionScale configuration.

If the Controller-configuration, Serial-packet, baud-rate, and data-bits are all valid, on the Position page, you should see MaPL.RadiansPacketCommand and MaPL.RadiansCommand = 3.119821. See trouble-shooting section below to debug.

Note the CommandPositionScale default value of 0.000191753 provides a RS422 command range of +/-2π.

4.3 Status Packet

The Status Packet is transmitted by the Controller. All Status Packets start with the 2-byte header and end with the 2-byte CRC defined above. As with the Command Packet, within each byte, bit7 is transmitted first and bit0 is transmitted last. The Status Packet provides the run-time and error information about the drive. The body of the status is shown below.

4.3.1 Single-axis Controller (i.e. Stinger)

Signal name	Byte	Bit	Size (bits)	Data Type	Coding
Header	0-1	B[15..0]	16	Unsigned	Always 0xF0F0
Motor Enabled A	2	Bit0	1	Unsigned	0 - Motor Disabled 1 - Motor Enabled
Error Condition Motor A	2	Bit1	1	Unsigned	0 – No Errors 1 – Any Error present
CPUA Overvoltage	2	Bit2	1	Unsigned	0 – No Error 1 – Over Voltage Error
CPUA Undervoltage	2	Bit3	1	Unsigned	0 – No Error 1 – Under Voltage Error
CPU Warning	2	Bit4	1	Unsigned	0 – No Error 1 – Any warning present
Error-IPC-CAN	2	Bit5	1	Unsigned	0 – No Error 1 – Inter-processor CAN
Error-command RS422	2	Bit6	1	Unsigned	0 – No Error 1 – RS422 packet error or timeout

Reserved	2	Bit7	1	NA	Reserved
Feedback Motor A	3-4	B[15..0]	16	Signed	Command Mode and Scaling defined via HiDS Mode 0 – Current IQ Mode 1 – Velocity RPM Mode 2 – Position Radians
Motor A Error-Overcurrent	5	Bit0	1	Unsigned	0 – No Error 1 – Over Current Error
Motor A Error-LossOfFeedback	5	Bit1	1	Unsigned	0 – No Error 1 – Loss Of Feedback Error
Motor A Error-Overspeed	5	Bit2	1	Unsigned	0 – No Error 1 – Over Speed Error
Motor A Error-MotorTemp	5	Bit3	1	Unsigned	0 – No Error 1 – Motor Temperature Error
Motor A Error-InternalTemp	5	Bit4	1	Unsigned	0 – No Error 1 – Internal Temperature Error
Motor A Error-Brake	5	Bit5	1	Unsigned	0 – No Error 1 – Brake over/under current
Motor A Error-Current-cal	5	Bit6	1	Unsigned	0 – No Error 1 – Phase sensors failed to calibrate
Motor A Logic Fault	5	Bit7	1	Unsigned	0 – No Error 1 – Logic voltage out of range

4.3.2 Dual-axis Controller (i.e. Nova)

Signal name	Byte	Bit	Size (bits)	Data Type	Coding
Header	0-1	B[15..0]	16	Unsigned	Always 0xF0F0
Motor Enabled A	2	Bit0	1	Unsigned	0 - Motor Disabled 1 - Motor Enabled
Error Condition Motor A	2	Bit1	1	Unsigned	0 – No Errors 1 – Any Error present
CPUA Overvoltage	2	Bit2	1	Unsigned	0 – No Error 1 – Over Voltage Error
CPUA Undervoltage	2	Bit3	1	Unsigned	0 – No Error 1 – Under Voltage Error

CPUA Warning	2	Bit4	1	Unsigned	0 – No Error 1 – Any warning present
CPUA Error-IPC-CAN	2	Bit5	1	Unsigned	0 – No Error 1 – Inter-processor CAN
Error-command RS422	2	Bit6	1	Unsigned	0 – No Error 1 – RS422 packet error or timeout
Reserved	2	Bit7	1	NA	Reserved
Feedback Motor A	3-4	B[15..0]	16	Signed	Command Mode and Scaling defined via HiDS Mode 0 – Current IQ Mode 1 – Velocity RPM Mode 2 – Position Radians
Motor A Error-Overcurrent	5	Bit0	1	Unsigned	0 – No Error 1 – Over Current Error
Motor A Error-LossOfFeedback	5	Bit1	1	Unsigned	0 – No Error 1 – Loss Of Feedback Error
Motor A Error-Overspeed	5	Bit2	1	Unsigned	0 – No Error 1 – Over Speed Error
Motor A Error-MotorTemp	5	Bit3	1	Unsigned	0 – No Error 1 – Motor Temperature Error
Motor A Error-InternalTemp	5	Bit4	1	Unsigned	0 – No Error 1 – Internal Temperature Error
Motor A Error-Brake	5	Bit5	1	Unsigned	0 – No Error 1 – Brake over/under current
Motor A Error-Current-cal	5	Bit6	1	Unsigned	0 – No Error 1 – Phase sensors failed to calibrate
MotorA Logic Fault	5	Bit7	1	Unsigned	0 – No Error 1 – Logic voltage out of range
Motor Enabled B	6	Bit0	1	Unsigned	0 - Motor Disabled 1 - Motor Enabled
Error Condition Motor B	6	Bit1	1	Unsigned	0 – No Errors 1 – An Error Occurred
CPUB Overvoltage	6	Bit2	1	Unsigned	0 – No Error 1 – Over Voltage Error
CPUB Undervoltage	6	Bit3	1	Unsigned	0 – No Error 1 – Under Voltage Error
CPUB Warning	6	Bit4	1	Unsigned	0 – No Error 1 – CPU self-test or other configured warning

Error-IPC	6	Bit5	1	Unsigned	0 – No Error 1 – Inter-processor CAN
Reserved	6	Bit6-7	2	NA	Reserved
Feedback Motor B	7-8	B[15..0]	16	Signed	Command Mode and Scaling defined via HiDS Mode 0 – Current IQ Mode 1 – Velocity RPM Mode 2 – Position Radians
Motor B Error-Overcurrent	9	Bit0	1	Unsigned	0 – No Error 1 – Over Current Error
Motor B Error-LossOfFeedback	9	Bit1	1	Unsigned	0 – No Error 1 – Loss Of Feedback Error
Motor B Error-Overspeed	9	Bit2	1	Unsigned	0 – No Error 1 – Over Speed Error
Motor B Error-MotorTemp	9	Bit3	1	Unsigned	0 – No Error 1 – Motor Temperature Error
Motor B Error-InternalTemp	9	Bit4	1	Unsigned	0 – No Error 1 – Internal Temperature Error
Motor B Error-Brake	9	Bit5	1	Unsigned	0 – No Error 1 – Brake over/under current
Motor B Error-Current-cal	9	Bit6	1	Unsigned	0 – No Error 1 – Phase sensors failed to calibrate
MotorB Logic Fault	9	Bit7	1	Unsigned	0 – No Error 1 – Logic voltage out of range
CRC	10-11	B[15..0]	16	NA	CRC

4.4 HiDS Configuration Variables

The HIDS tool is used to configure the serial interface. The following sections define the modifiable parameters.

4.4.1 Enabling the Serial Interface

The RS422-control interface can be enabled by setting variable SerialInterfaceEnable = 1 on the Parameter Data Item page under the HiDS Advanced tab.
The RS422-status interface is enabled if the RS422 Command interface is enabled.

4.4.2 Command Scaling

Each motor includes three variables for command scaling, one for torque, velocity, and position mode. There three variables are as follows:

- CommandCurrentScale
- CommandVelocityScale
- CommandPositionScale

Example scaling:
If CommandCurrentScale = 0.001, the range is -32.768 to +32.767 amps.
If CommandVelocityScale = 1, the range is -32,768 to +32,767 RPM.
If CommandPositionScale = 0.000191753, the range is -2π to +2π radians.

These variables are user-configurable and can be set to accommodate the range of motion of the motor while maximizing the resolution of the serial interface.

On a received RS422 command, the controller calculates the drive command as:

Command = (Signed 16-Bit Serial Input) * Command Scale

When transmitting the status packet, the measured-value is similarly scaled as:

Signed 16-Bit Serial Input = MeasuredValue / Command Scale

4.5 RS422 Interface

4.5.1 Command Rate

The serial interface is polled at 1Khz, and the status-packet is transmitted independent of the command packet, so the CommandRate chosen together with the configured baud-rate must allow for the packets to fit within this timing.

4.5.2 Baud Rate

The standard baud-rate selections are limited to 115200, 230400, and 1000000, and is configured by setting the SerialBaud variable. This baud rate could be modified for custom software applications. Note the serial parameters are 8 data bits, no parity, and 1 stop bit (8,N,1) and are fixed.

4.6 Trouble-shooting

Much can go wrong with a serial-interface, and basically everything (serial-wiring, number of bytes, baud-rate, serial-configuration, CRC, timing, and data-content) must be correct for the interface to work. If the desired command is not getting through to the loop-command, on the HiDS Serial page, check:

- SerialCharactersReceived should increment by 10 for each packet received. Note unplugging and plugging the serial-cable can sometimes cause a spurious single character to be received.

- SerialNumReportedErrors indicates received-data errors from the serial-transceiver and should not be changing. Note this flag can sometimes be set to 1 on power-up or on plugging in the serial-cable.

- SerialChecksumPassCount indicates the packet CRC matched, and this variable should increment by 1 for every received packet.
- SerialChecksumErrorCount indicates the correct number of bytes were received, but the packet CRC did not match, and this variable should be 0.
- LastSerialRxPacket[0]-[9] indicates the last 10 bytes received on the serial interface.