
Table of Contents

Section	Description	Page #
11.	Multispense Protocol Converter	2
11.1	Description	2
11.1.1	Service Connector	2
11.1.2	Industrial Ethernet Protocol Connectors	2
11.1.3	Multispense Interface Connector	3
11.1.4	Protocol Converter Operation Indicators	3
11.1.5	Industrial Ethernet Operation Indicators	3
11.2	Mounting	4
11.3	Setup	4
11.3.1	Home Page	4
11.3.2	Service Port Network Configuration Page	4
11.3.3	Industrial Ethernet Protocol Configuration Page	5
11.3.4	Industrial Ethernet Protocol Hardware Status Page	6
11.4	Software Interface	6
11.4.1	Message Packet Structure	6
11.4.2	Message Packet Processing Time	9
11.4.3	Message Packet Synchronization	9
11.4.4	Message Packet: Enable	9
11.4.5	Message Packet: Message Id	10
11.4.6	Message Packet: Command	10
11.4.7	Message Packet: Address	11
11.4.8	Message Packet: Value Quantity	11
11.4.9	Message Packet: Value 1, Value 2, Value 3	12
11.4.10	Message Packet: Warning Number	13
11.4.11	Message Packet: Value Quantity Channel [1..32]	13
11.4.13	Message Packet: Warning Number Channel	14
11.4.14	Industrial Ethernet Protocol: EthernetIP	15
11.4.15	Industrial Ethernet Protocol: Profinet	16
11.4.16	Industrial Ethernet Protocol: Modbus TCP	17
11.4.17	Message Packet Examples	17

11. MULTISPENSE PROTOCOL CONVERTER

The Multispense Protocol Converter provides a mechanism to exchange messages between a Terminal (PLC, touchscreen, PC, etc.) using an industrial Ethernet protocol (ModbusTCP, EthernetIP, Profinet RT) and an IVEK Multispense Controller (MS900 or MS2000). The Protocol Converter converts message packets into the RS232 command strings described in IVEK's Multispense manuals. The Protocol Converter controls the RS232 communication and parsing of ASCII strings, providing numeric results in mapped registers. The Protocol Converter also provides a mechanism that allows multiple channels within the IVEK controller to be individually commanded with a single message.

11.1 DESCRIPTION

The Protocol Converter is available with or without an enclosure. The optional enclosure contains a clear lid which allows the indicators on the Protocol Converter to be visible during operation. The enclosure also holds the connectors necessary to interface the Protocol Converter.

Figure 11.1 shows the following connectors and indicators; 1) Service Connector, 2) Industrial Ethernet Protocol Connectors (quantity 2) 3) Multispense Interface Connector 4) Protocol Converter Operation Indicators 5) Protocol Converter Industrial Ethernet Operation Indicators

11.1.1 Service Ethernet Connector

The Service Ethernet Connector provides an Ethernet port (shielded RJ-45) for use when commissioning the Industrial Ethernet ports. The Ethernet port is 10/100 Mbit/s, 10BASE-T/100BASE-TX, half/full duplex, with Autonegotiation and Auto-MDI/MDIX. The maximum allowable cable length between stations is 100m. A Modbus TCP interface is also available on this port to use as the Industrial Ethernet Protocol.

When the optional enclosure is included, the lid must be removed to physically access this port. When using Modbus TCP on a Protocol Converter with the optional enclosure, move one of the RJ-45 cable extenders from the Industrial Ethernet Protocol Connectors to this connector.

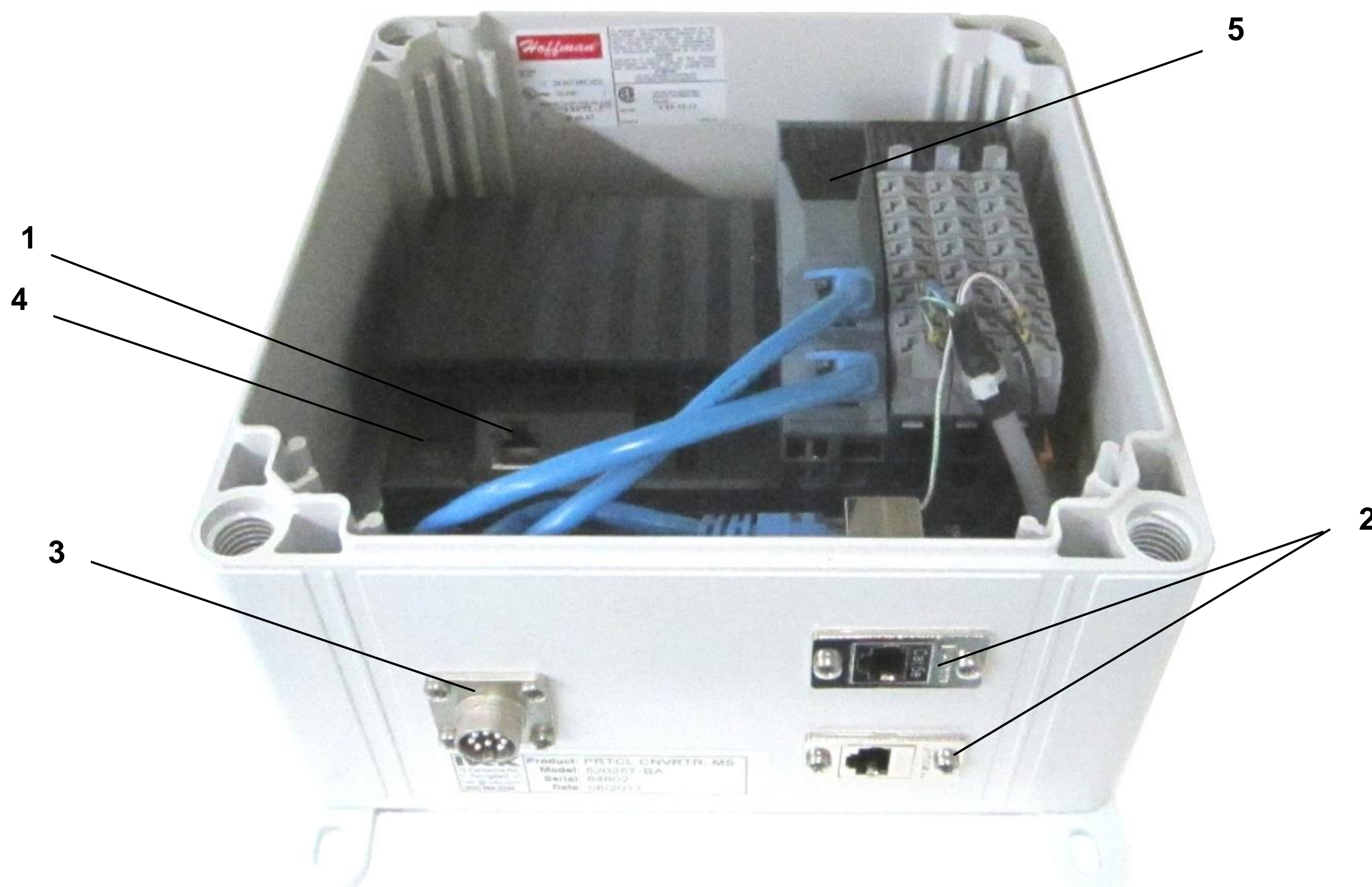


Figure 11.1 Multispense Protocol Converter

11.1.2 Industrial Ethernet Protocol Connectors

The Industrial Ethernet Protocol Connectors provide 2 ports with an integrated switch for connecting to the configured Industrial Ethernet Protocol. The Ethernet ports (shielded RJ-45) are compliant with the specific Industrial Ethernet Protocol and are 10/100 Mbit/s, 10BASE-T/100 BASE-TX, half/full duplex, with Autonegotiation and Auto-MDI/MDIX. The maximum allowable cable length between stations is 100m.

When the optional enclosure is included, the RJ45 ports are extended to the outside of the enclosure using shielded RJ-45 cable extenders.

11.1.3 Multispense Interface Connector

The Multispense Interface Connector provides a connection point between the Remote Terminal connector, located on the Multispense controller, and the Protocol Converter. The Multispense provides power (24VDC) to the Protocol Converter and also the necessary RS232 signals. The interconnecting cable (540135-##) must be ordered separately.

11.1.4 Protocol Converter Operation Indicators

The Protocol Converter contains many visual indicators. The following indicators are most relevant to operation of the Protocol Converter:

LED	Color	Status	Description
R	Green	On	Protocol Converter operating
		Blinking	Protocol Converter booting
ET	Green	On	A link to the Service Ethernet Port is established
		Blinking	A link to the Service Ethernet Port is established and activity is taking place
DC	Yellow	On	24V power to the Protocol Converter ok
S	Yellow	On	Protocol Converter exchanging RS232 data with the Multispense
RF	Yellow	Any	Error
E	Red	Any	Error

11.1.5 Industrial Ethernet Operation Indicators

The portion of the Protocol Converter which contains the RJ45 connectors for the Industrial Ethernet Protocol has many visual indicators. The following indicators are most relevant to operation of the Industrial Ethernet Protocol.

LED	Color	Status	Description
READY/RUN	Green	On	Module ready
		Off	No power to module
	Red	Blinking	Error
		On	Booting
Mod Status	Green	Blinking	Not yet configured
		On	Adapter (slave) is operational
		Off	No power to module
	Red	Any	Error
	Green/Red	Blinking	Initialization / Self-test
Net Status	Green	Blinking	No active connection
		On	At least one active connection
	Red	Blinking	Timeout on at least one connection
		On	Duplicate IP Address
	Green/red	Blinking	Initialization / Self-test
		Off	No IP address assigned or no power to module
L/A IF1/IF2	Green	On	Link to remote station
		Flickering	A link to the remote station has been established and activity is taking place
		Off	No link to remote station

11.2 MOUNTING

The Protocol Converter must be mounted in either a horizontal or vertical orientation. Units with an optional enclosure include 4 mounting feet for mounting to a wall or surface.

Units without an enclosure have mounting tabs for mounting to a EN 60715 standard (TH35-7.5) DIN rail. There must be 35mm of free space above the modules, 10mm of free space to the left and right of the modules, and 35mm of free space below the modules. The DIN rail must be conductive and connected to functional earth ground.

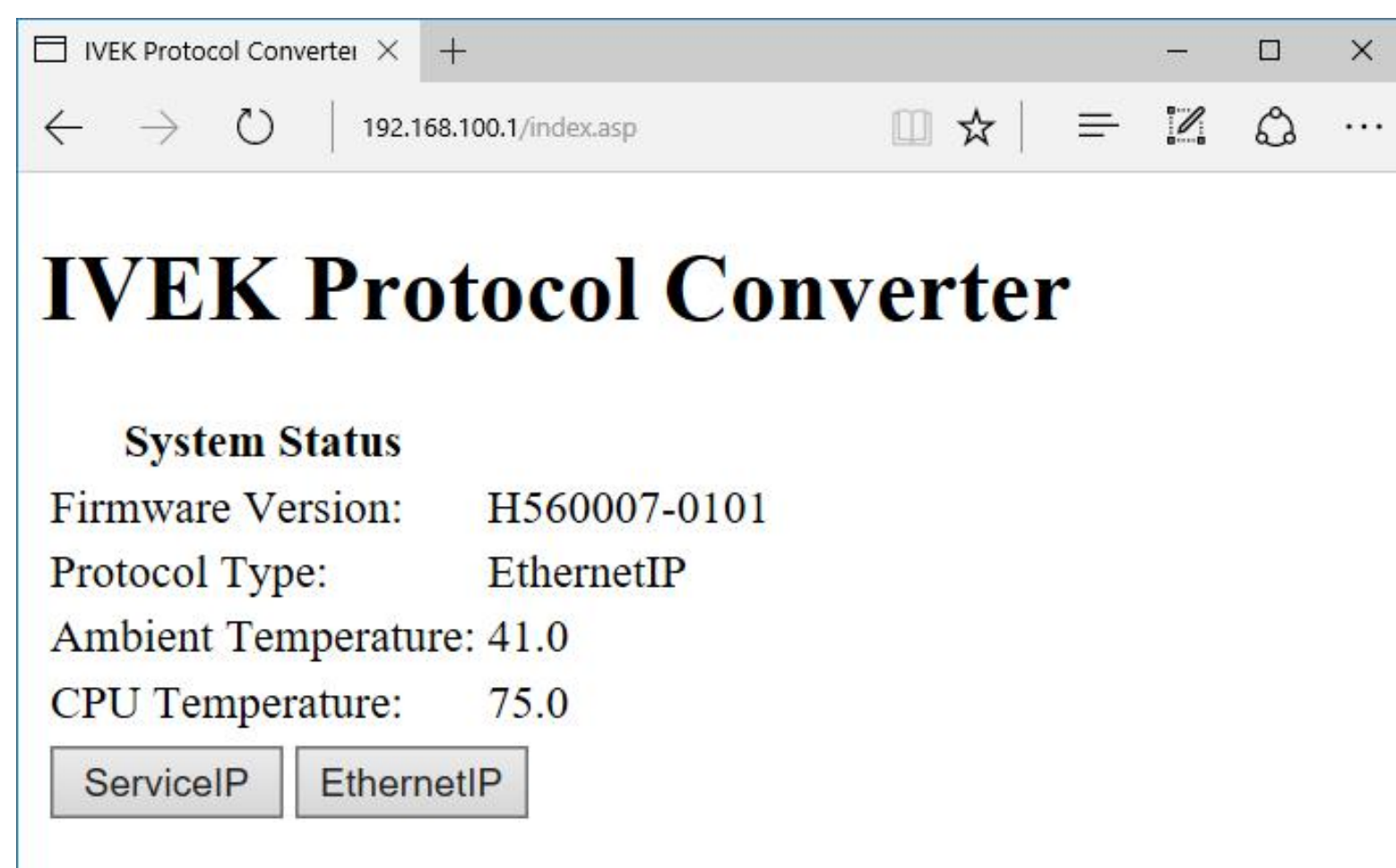
11.3 SETUP

The Service Ethernet Port is used for configuration/commission of the Protocol Converter. The Service Ethernet Port offers a web server which serves web pages to configure both the Service port and the Industrial Ethernet ports.

The Service Port has a factory default TCP/IP address of 192.168.100.1. To configure the Protocol Converter, connect to the Service Port with a PC, open a standard HTTP browser, and browse to 192.168.100.1 (port 80). If there is trouble connecting, verify the PC's Ethernet Port is on the same subnet.

11.3.1 Home Page

The home page displays the firmware version of the Protocol Converter, the type of the Industrial Ethernet Protocol, the Ambient, and the CPU temperature of the Protocol Converter. There are also 2 buttons that redirect to a webpage to either configure the Service Port or the Industrial Ethernet Protocol port.



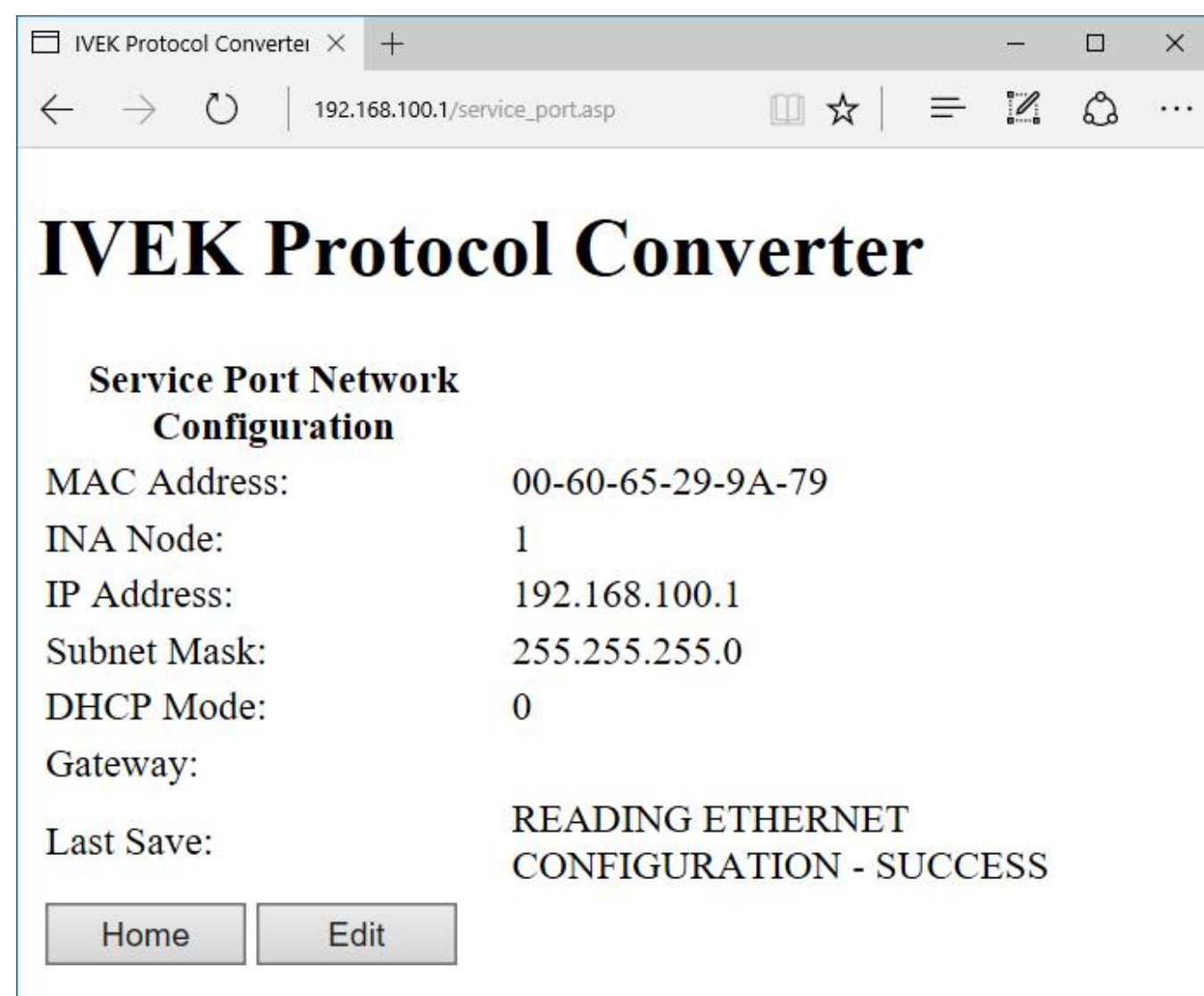
11.3.2 Service Port Network Configuration Page

The Service Port Network Configuration Page displays the parameters for the Service Port. The following values are displayed: MAC Address, INA Node Number, IP Address, Subnet Mask, DHCP Mode, Gateway Address, and Last Save Status.

The page includes 2 buttons: Home, and Edit. Pressing the Home button returns to the Home page. Pressing the Edit button allows the following values to be modified: INA Node Number, IP Address, Subnet Mask, DHCP Mode, and Gateway Address.

If editing the values, configure these values for the intended network, or leave as is if used for a single time setup only. If these values are changed, it is possible to return these values to factory default, insert any USB memory stick in the USB port and cycle power to the unit. When in Editing mode, two buttons allow either saving or canceling. The Last Save Status indicates the success or failure of the last time the Save button was pressed.

After pressing the Save button, a link appears indicating that the IP address may have been modified. Press the link button to redirect to the home page at the new IP address.

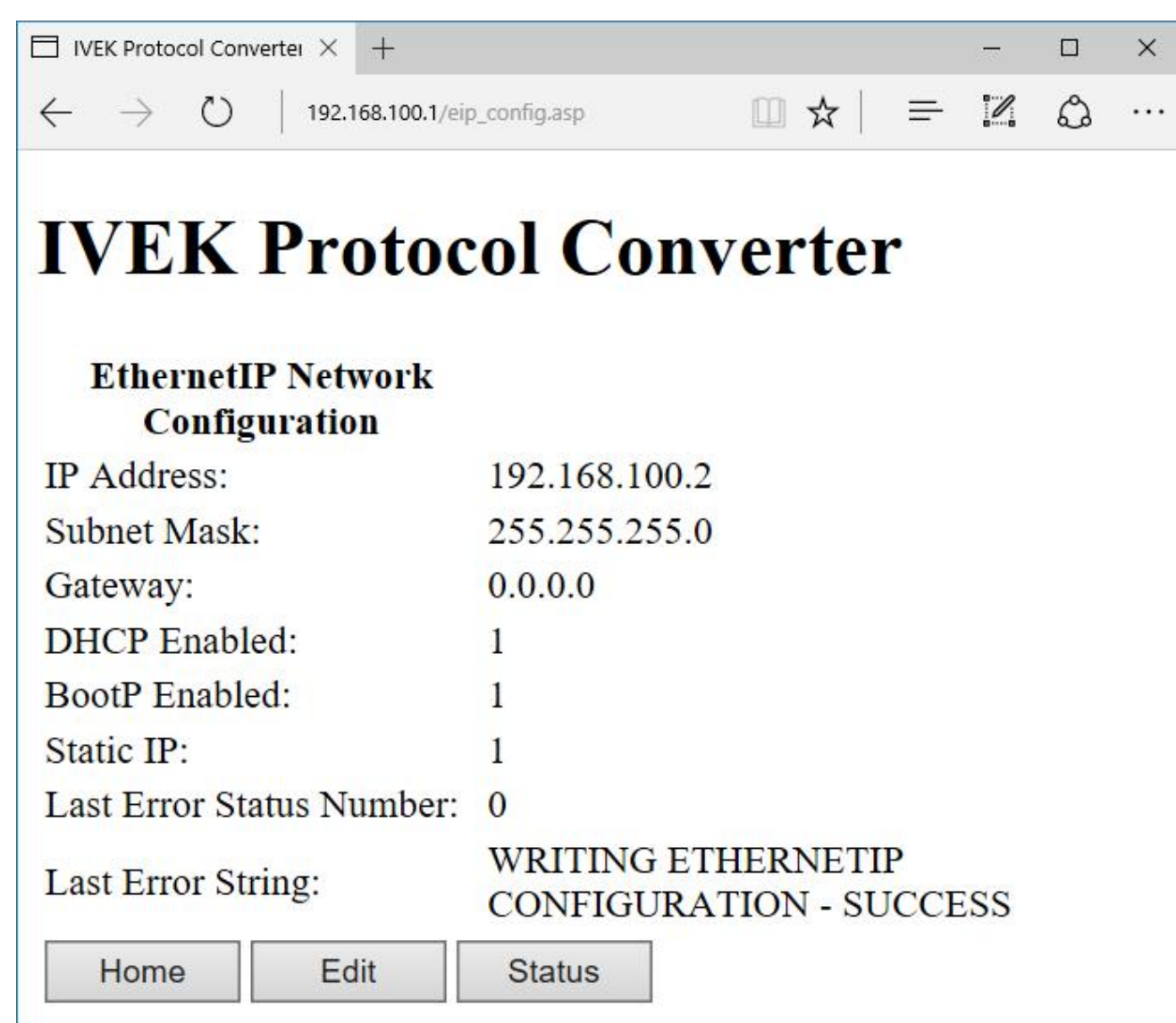


11.3.3 Industrial Ethernet Protocol Configuration Page

The Industrial Ethernet Protocol Configuration Page shows the values for the Industrial Ethernet Protocol. The following parameters are available for EthernetIP: IP Address, Subnet Mask, Gateway, DHCP Enabled, BootP Enabled, Static IP Mode Enabled, Last Save Error Number, Last Save Error String.

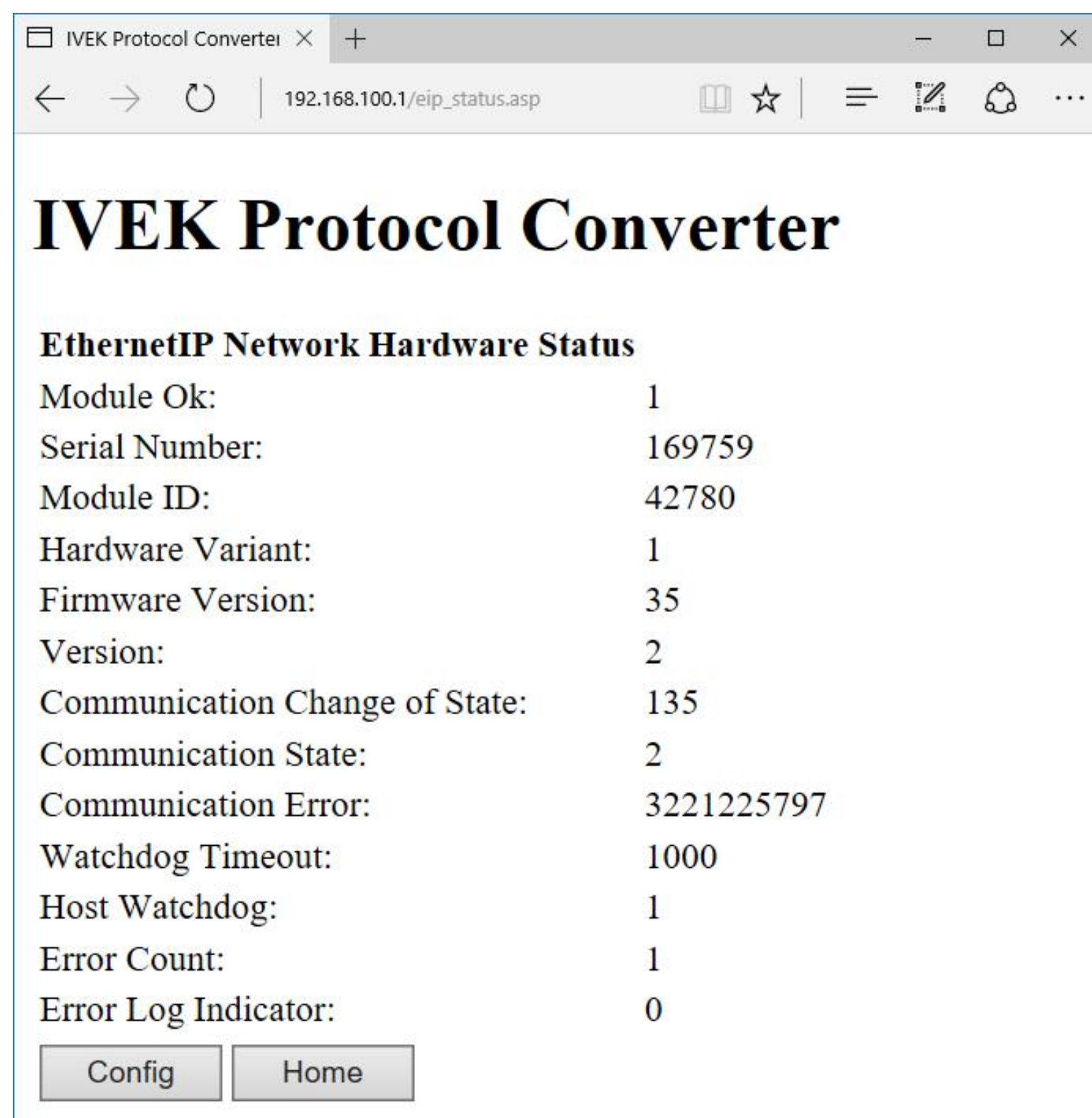
The page includes 3 buttons: Home, Edit, and Status. Pressing the Home button redirects to the Home page. Pressing the Edit button allows some of the values to be edited. Pressing the Status page redirects to a page which shows the present status value of the Industrial Ethernet Protocol.

When editing these values, configure the parameters based on the intended network. Most often, a static IP address will be used. In this case, disable DHCP mode and BootP mode. When an address is not static, DHCP mode has priority, followed by BootP mode, and finally the Converter will default to the static IP address if there is not either a DHCP server or BootP sever available. Once the values are edited, press the Save button to keep the new network parameters or Cancel to ignore the changes. To verify the changes, power-cycle the Protocol Converter after the values have been successfully saved, then return to this page to verify the values.



11.3.4 Industrial Ethernet Protocol Hardware Status Page

The Industrial Ethernet Protocol Hardware Status Page shows internal values useful for troubleshooting the connection. The two most important values when troubleshooting a connection are: Communication State and Communication Error. When the Protocol Converter Industrial Ethernet Protocol module is properly configured, the Communication State will be 2 and the Communication Error state will be 0. When two devices are successfully communicating, the Communication State will be 4, and the Communication Error will be 0. All other values may be requested for when calling IVEK Technical Support.



The screenshot shows a web browser window with the title 'IVEK Protocol Converter' and the URL '192.168.100.1/eip_status.asp'. The main heading is 'IVEK Protocol Converter'. Below it is the section 'EthernetIP Network Hardware Status' with the following data:

Module Ok:	1
Serial Number:	169759
Module ID:	42780
Hardware Variant:	1
Firmware Version:	35
Version:	2
Communication Change of State:	135
Communication State:	2
Communication Error:	3221225797
Watchdog Timeout:	1000
Host Watchdog:	1
Error Count:	1
Error Log Indicator:	0

At the bottom of the status table are two buttons: 'Config' and 'Home'.

11.4 SOFTWARE INTERFACE

The Protocol Converter provides a register set of a specific industrial Ethernet protocol to allow communication between a Terminal (PLC, Touchscreen, PC, etc.) and a Controller (Multispense 900/2000). The Protocol Converter encapsulates most of the Multispense RS232 protocol within the supported Industrial Ethernet Protocol. The Protocol Converter works in a Transmit/Response method: the Terminal Software transmits a Message Packet to the Protocol Converter, the Protocol Converter translates the Message Packet into an RS232 command (or multiple commands) and sends the RS232 command(s) to the Multispense Controller, waits for a response from the Multispense Controller, parses the response(s) from the Multispense Controller, assembles the response(s) into a single Message Packet, and responds to the Terminal with a Message Packet.

It is expected that the customer will most often implement software using a typical PLC language such as Structured Text or Ladder Logic. The customer PLC must contain a Scanner (Master) device to establish communication with the Protocol Converter and to communicate the proper registers.

11.4.1 Message Packet Structure

A Message Packet structure, when mapped to the registers of a specific industrial Ethernet protocol, provides the mechanism of data exchange between a Terminal (PLC, Touchscreen, PC, etc.) and a Controller (Multispense 900/2000) via the Protocol Converter. The Message Packet is encapsulated by whichever industrial Ethernet protocol is employed. The industrial Ethernet protocol handles all data transmission, error detection/correction, etc.; the Message Packet is strictly an application layer protocol.

The Message Packet structure consists of 9 fields that are each 2 bytes (16-bits) in length, plus 5 field arrays that are each 64 bytes (32 x 16-bits) in length. The following code snippet demonstrates the structure in structured text:

```
message_packet_t :   STRUCT
  enable : UINT := 0;
  message_id : UINT := 0;
  command : UINT := 0;
  address : UINT := 0;
  value_qty : UINT := 0;
  value1 : UINT := 0;
  value2 : UINT := 0;
  value3 : UINT := 0;
  warning_num : UINT := 0;
  value_qty_ch : ARRAY[32]OF UINT := [32(0)];
  value1_ch : ARRAY[32]OF UINT := [32(0)];
  value2_ch : ARRAY[32]OF UINT := [32(0)];
  value3_ch : ARRAY[32]OF UINT := [32(0)];
  warning_num_ch : ARRAY[32]OF UINT := [32(0)];
END_STRUCT;
```

In most applications, only the first 9 fields are necessary for proper communication. The 5 arrays fields are available for more advanced communication scenarios when the quickest possible response times are required when commanding different values to multiple channels, or for advanced error handling.

The following table provides a brief description and boundary conditions for each field of the Message Packet structure:

Message Packet Field	Length (Bytes)	Valid Values (To Protocol Converter)	Valid Values (From Protocol Converter)
Fields for All Communication			
Enable	2	0 = Ignore contents of Message Packet 1 = Use contents of Message Packet 2-65535 = (reserved)	0 = No Message Packet processed, or Message Packet is being processed 1 = Message Packet processed
Message Id	2	0 = reset Message Packet 1-65535 = Id of the Message Packet to Process, must change this value each new Message Packet	0 = Message Packet reset 1-65535 = Id of last Message Packet processed
Fields for Basic Communication			
Command	2	ASCII 'a'-'z' per the command set of the Multispense RS232 protocol to send to the Multispense.	ASCII 'a'-'z' per the command set of the Multispense RS232 protocol received from the Multispense.
Address	2	0 = send Command to all installed Multispense channels 1..32 = send Command to a single Multispense channel 99 = send Command to the Multispense master	0 = response to Command received from all installed Multispense channels 1..32 = response to Command received from single Multispense channel
Value Quantity	2	0 = use the channel array fields. 1 = send the Command using the Address and no values. 2 = send the Command using the Address & Value 1. 3 = send the Command using the Address, Value 1 & Value 2. 4 = send the Command using the Address, Value 1, Value 2 & Value 3. 5 – 65535 (reserved)	0 = invalid response or channel array fields used 1 = Command received at Address, no values 2 = Command received at Address, Value 1 contains data 3 = Command received at Address, Value 1 & Value 2 contain data 4 = Command received at Address, Value 1, Value 2, & Value 3 contain data

Value 1	2	0 - 65535, valid values dependent on specific Multispense Command. Value to send to the Multispense.	0 - 65535, valid values dependent on specific Multispense Command. Value contained in the Multispense response.
Value 2	2	0 - 65535, valid values dependent on specific Multispense Command. Value to send to the Multispense.	0 - 65535, valid values dependent on specific Multispense Command. Value contained in the Multispense response.
Value 3	2	0 - 65535, valid values dependent on specific Multispense Command. Value to send to the Multispense.	0 - 65535, valid values dependent on specific Multispense Command. Value contained in the Multispense response.
Warning Number	2	Ignored	0 = No errors, warnings, or faults 1-999 = Multispense response contained this warning 1000 = At least one Multispense channel response contained an error 1001-1999 = Multispense response contained this error 9000-9999 = Error occurred in communication from Protocol Converter to Multispense
Fields for Advanced Communication			
Value Quantity Channel [1..32]	2 x 32	0 = Do not send Command to this channel 1 = send Command to this channel using no values. 2 = send Command to this channel using Value 1. 3 = send Command to this channel using Value 1 & Value 2. 4 = send Command to this channel using Value 1 & Value 2 & Value 3 5 - 65535 (reserved)	0 = There was no Command response for this channel last processed Message Packet 1 = a Command for this channel was received with no values. 2 = a Command for this channel was received with values, Value 1. 3 = A Command for this channel was received with values, Value 1 & Value 2. 4 = A Command for this channel was received with values, Value 1, Value 2, & Value 3.
Value 1 Channel [1..32]	2 x 32	0 - 65535, valid values dependent on specific Multispense Command. Value to send to the Multispense.	0 - 65535, valid values dependent on specific Multispense Command. Value contained in the Multispense response.
Value 2 Channel [1..32]	2 x 32	0 - 65535, valid values dependent on specific Multispense Command. Value to send to the Multispense.	0 - 65535, valid values dependent on specific Multispense Command. Value contained in the Multispense response.
Value 3 Channel [1..32]	2 x 32	0 - 65535, valid values dependent on specific Multispense Command. Value to send to the Multispense.	0 - 65535, valid values dependent on specific Multispense Command. Value contained in the Multispense response.
Warning Number Channel [1..32]	2 x 32	Ignored	0 = No errors, warnings, or faults 1-999 = Multispense response contained this warning 1000 = At least one Multispense channel response contained an error 1001-1999 = Multispense response contained this error 9000-9999 = Error occurred in communication from Protocol Converter to Multispense

11.4.2 Message Packet Processing Time

The Protocol Converter may take anywhere from 10ms to 5 seconds per channel to process a received Message Packet. This means that the Terminal software should be able to handle responses that occur as quick as a single communication cycle, as well as have timeouts to indicate a communication problem if a response is not received within the worst case window (32 channels x 5 seconds = 160 seconds). If the channel array fields are not used, the worst case response window is 5 seconds.

11.4.3 Message Packet Synchronization

Two of the Message Packet structure fields provide the necessary synchronization between the Terminal (PLC, PC, etc.) and the Protocol Converter: Enable, and Message Id. Maintaining synchronization is critical to ensure that data is exchanged atomically between the Terminal and the Protocol Converter. In order to allow for quick response times (e.g., responding to a Message Packet in a single communication cycle, e.g., 10ms), the synchronization mechanism requires special attention.

These two fields can be thought of as the input to a function block. The Enable field “enables” the execution of the Protocol Converter “function block” to process the Message Packet. The Message Id provides the “command” to the Protocol Converter “function block” to process.

11.4.4 Message Packet: Enable

The Enable field indicates to the Protocol Converter that the Message Packet fields are valid, stable, and should be processed. When the Protocol Converter receives a 1 in the Enable field, it begins to process the rest of the Message Packet fields. The Protocol Converter also has an Enable field which it sets to 0 while processing the Message Packet and sets to 1 upon completion of processing the Message Packet.

Because a Message Packet may be processed in a single communication cycle, it is possible that the Terminal will not see the Enable field from the Protocol Converter transition. However, anytime the Enable field from the Protocol Converter is 1, the rest of the Message Packet is valid.

11.4.4.1 Terminal to Protocol Converter

The Enable field should be 0 during power-up and initialization. The Enable field should be set to 1 after all other fields of the Message Packet are ready for transmission to the Protocol Converter.

The Protocol Converter does not require a transition of the Enable field (i.e., once it is set to 1 the first time it may remain at 1).

11.4.4.2 Protocol Converter to Terminal

The Enable field is set to 0 during power-up and initialization and while a Message Packet is being processed. Once a Message Packet has been processed, the Enable field is set to 1.

Because a Message Packet may be processed in a single communication cycle, it is possible that this field may not transition, so do not wait upon a transition of this field in any software.

11.4.4.3 Error Checking Recommendations

The Enable field is 0 while a Message Packet is being processed by the Protocol Converter. It is possible that the Enable field could be 0 for up to 5 seconds per channel (32 channels x 5 seconds = 160 seconds) when using the channel arrays, or up to 5 seconds when not using the channel arrays. A timeout timer should be implemented to check for an Enable field that stays 0 for longer than the maximum expected time. Note, initial power-up/initialization may take much longer than 5 seconds.

11.4.5 Message Packet: Message Id

The Message Id indicates to the Protocol Converter that a new Message Packet is ready for processing. The Protocol Converter only acts upon Messages when the Message Id value transitions (and only if Enable = 1). This means a new Message Id needs to be generated for each new Message Packet.

11.4.5.1 Terminal to Protocol Converter

The Message Id should be 0 during power-up/initialization. To cause the Protocol Converter to process the first Message Packet after power-up/initialization, set the rest of the fields in the Message Packet to the desired values, set the Message Id to a new value, and then set Enable to 1. For subsequent Message Packets, set the rest of the fields to the desired values, and then change the Message Id to a new value (Enable may either be left at 1 during this time, or transitioned to 0 and then back to 1).

It is recommended to increment the Message Id by 1 for each new Message Packet, rolling over from 65535 to 1. However, other implementations are possible, including: toggling a single bit between Message Packets (e.g., toggling Message Id between 2 & 3), or toggling between the initialization Message Packet and a new Message Packet (i.e., toggling Message Id between 0 and 1). The only requirement is that the Message Id is changed each time it is desired for the Protocol Converter to process a Message.

A Message Id of 0, with Enable = 1, causes the Protocol Converter to reset its Message Packet fields to 0. This is useful for initialization of the Message Packet from the Protocol Converter to a known state.

11.4.5.2 Protocol Converter to Terminal

The Message Id indicates the last processed Message Packet. If Enable = 0, the Message Packet with that Id is being processed. If Enable = 1, the Message Packet with that Id has been processed and the rest of the fields in the Message Packet contain the response.

During normal operation, it is recommended that the Terminal change this value upon each new Message Packet, and then wait for the Protocol Converter to Message Id to be changed to the same Message Id, with Enable = 1.

A Message Id of 0 is reserved for power-up/initialization. The Protocol Gateway initializes its outgoing Message Id to 0 during power-up, and keeps it at this value until a valid Message Packet is received from the Terminal (with Enable = 1).

11.4.5.3 Error Checking Recommendations

There are several scenarios in which the Protocol Gateway and Terminal could lose synchronization (power-cycle, reset, second Terminal on the link, etc.). Because of this possibility, it is recommended that the Terminal software read the outgoing Message Id of the Protocol Converter before each new Message Packet Id is generated. It is necessary that the Terminal software transmits a Message Id that is different than the Message Id indicated by the Protocol Converter. Also, the indicated Message Id of the Protocol Converter should be continuously monitored for changes to a state other than expected (e.g., a sudden change to 0 caused by a reset/power-cycle of the Protocol Gateway).

11.4.6 Message Packet: Command

The Command field communicates the Multispense RS232 commands (see Multispense 900 or Multispense 2000 manual). The Multispense RS232 commands are decimal equivalents of ASCII characters (e.g., 97 for 'a', 122 for 'z'). The commands available to the Terminal software depend on the type and firmware version of the Multispense Controller.

A command (and its values) is always initiated by the Terminal. The Protocol Converter translates the command and values into the proper ASCII string(s) to send to the Controller. The Protocol Converter handles all RS232 communication with the Controller and, after completion, translates the Controller's command response into a Message Packet for use by the Terminal.

11.4.6.1 Terminal to Protocol Converter

Set the Command field to the numeric value of the ASCII character per the Multispense Command set.

11.4.6.2 Terminal to Protocol Gateway

The Command field is set to the numeric value of the ASCII command that was received from the Multispense Controller.

11.4.6.3 Error Checking Recommendations

It is possible that an error could occur during the RS232 communication sequence between the Protocol Converter and the Multispense Controller. Therefore, it is important to verify for each Message Packet processed that the Command received is the same as the Command sent.

11.4.7 Message Packet: Address

The Message Packet structure works with the Value Quantity and Value Quantity Channel fields to allow for broadcast, individual channel, and multiple channel communication. Most applications will use only broadcast communication (treating all Multispense channels the same). Individual Channel and Multiple Channel communication is more complex and only required when it is desired to configure individual channels differently than other channels.

Communication Type	Address	Value Quantity	Value Quantity Channel[1..32]	Notes
Broadcast	0	1..4	All 0	Command and values sent to all installed channels
Individual Channel	1..32	1..4	All 0	Command and values sent to single channel
Individual Channel (alternate)	0	0	1..4 (of desired channel only)	Command and channel values sent to single channel
Multiple Channel	0	0	1..4 (of desired channels only)	Command and channel values sent to desired channels
Mixed	0	1..4	1..4 (of desired channels only)	Command and values sent to all channels, then channel values sent to desired channels.

For advanced usage, both methods may be used in the same Message Packet (Mixed Communication Type). For an example, it may be desired to set the volume of half of the channels to one value and the other half to a different value. In this scenario, it is possible to broadcast a value to all of the channels using the broadcast fields as well as a different value to half the channels using the individual channels fields. Note, that the broadcast values are always sent to the Multispense first, followed by the individual channel fields.

When communicating with individual channels using the channel arrays, the array index of the corresponding channel field correlates to the address of the desired channel (e.g., to communicate with channel 1, use registers Value Quantity Channel [1], Value 1 Channel [1], Value 2 Channel [1], Value 3 Channel [1], and Warning Number Channel [1]). Multiple channels may be used in a single Message Packet.

11.4.8 Message Packet: Value Quantity

The Value Quantity field indicates whether the command in the Command field is to be sent to the addresses to all channels, and how many Value fields are relevant to that Command.

Each Multispense command has a different number of allowed value fields (see Multispense Manual).

11.4.8.1 Terminal to Protocol Converter

0 = use Value Quantity Channel fields to send Commands to the corresponding Multispense channels

- 1 = use the Address field to specify Multispense channel(s) and send no values fields.
- 2 = use the Address field to specify Multispense channel(s) and send Value 1.
- 3 = use the Address field to specify Multispense channel(s) and send Value 1 & Value 2.
- 4 = use the Address field to specify Multispense channel(s) and send Value 1 & Value 2 & Value 3

11.4.8.2 Protocol Converter to Terminal

- 0 = depends on the Communication Type: broadcast, the response of all channels not identical; individual using Address field, the channel did not respond; other methods, always 0.
- 1 = broadcast, all installed channels responded with no values; individual using Address field, the channel responded with no values; all other types, not used.
- 2 = broadcast, all installed channels responded with Value 1; individual using Address field, the channel responded with Value 1; all other types, not used.
- 3 = broadcast, all installed channels responded with Value 1 & Value 2; individual using Address field, the channel responded with Value 1 & Value 2; all other types, not used.
- 4 = broadcast, all installed channels responded with Value 1, Value 2, & Value 3; individual using Address field, the channel responded with Value 1, Value 2 & Value 3; all other types, not used.

11.4.8.3 Error Checking Recommendations

Many Multispense RS232 commands respond with more values than received. It is important that the Terminal software verifies that the number of values received matches the number expected for each specific command. The Multispense RS232 protocol truncates the third value field in cases where a warning or error exists. Therefore, it is possible a command that usually responds with 3 values will only respond with 2 when a warning or error is present. In this case, Value 1 & Value 2 will contain the 2 values, and Warning Num will contain the warning number. The Value Quantity field from the Protocol Converter will be set to 3 in this case. If all channels do not respond to a broadcasted command with the same number of values, the Value Quantity, Value 1, Value 2, & Value 3 fields are set to 0 and a warning is generated in the Warning Number field.

11.4.9 Message Packet: Value 1, Value 2, Value 3

The Multispense RS232 protocol allows up to three values as parameters within a command or command response. The quantity of value fields and range of each value field varies based on the specific Multispense command. If a value is used with a command that is out of range for that command, a warning will be returned by the Controller and passed on by the Protocol Gateway to the Terminal in the appropriate Warning Number field.

11.4.9.1 Terminal to Protocol Converter

The value in each Value field depends on the command being sent to the Multispense controller. The overall range of values is 0-65535, but is further limited by each command. The number of values sent is determined by the Value Quantity field. Value fields that are not being used, based on the Value Quantity field, are ignored.

11.4.9.2 Protocol Converter to Terminal

When a command is issued by the Terminal, the Protocol Converter generates a command to the Multispense controller. A Multispense Controller always responds to the Protocol Converter with a response from each installed channel if the Address is 0, or the individual channel if the Address is >0.

When broadcasting (Address =0, Value Quantity>0), the Protocol Converter generates a broadcast response for the Terminal by processing these individual channel responses. If all Controller channels respond with the same values in each value field, the corresponding Value field will contain the response value. Otherwise, the Value 1, Value 2, & Value 3 fields will contain 0 and the response values will be in the Value # Channel arrays.

11.4.9.3 Error Checking Recommendations

If all Multispense channels do not respond to the broadcasted command with the same value in each value field, the Value Quantity, Value 1, Value 2, & Value 3 fields are set to 0 and a warning is generated in the Warning Number

field. If it is desired to determine which channel(s) is causing the unexpected value, the Value # Channel array fields may be compared to the expected value to determine the errant channel.

11.4.10 Message Packet: Warning Number

The Warning Number field provides any warnings or faults received in the Multispense Controller's command response. Also, if a communication error occurs between the Protocol Converter and the Multispense Controller, the Warning Number will contain an error number.

11.4.10.1 Terminal to Protocol Converter

This field is ignored by the Protocol Converter. It is recommended to set this field to 0 for future compatibility.

11.4.10.2 Protocol Converter to Terminal

Warning/Fault/Error Number	Source	Description
0	Protocol Converter	No warnings, faults, or errors in response to Message Packet
1 to 999	Multispense Controller	Warning number (see Multispense manual)
1000 to 1999	Multispense Controller	Fault code (see Multispense manual)
9001	Protocol Converter	Either no response or an errant response was received from the Multispense
9002	Protocol Converter	An unaddressed channel generated a response to Message Packet.
9003	Protocol Converter	The internal RS232 hardware of the Protocol Converter has an error
9004	Protocol Converter	A channel responded with a different Command than that in the Message Packet.

11.4.11 Message Packet: Value Quantity Channel [1..32]

The Value Quantity Channel field indicates whether the command in the Command field is to be sent to the specific channel, and how many Value Channel fields are relevant to that Command. It is permissible to send the same command with different Value Channel field values in the same Message Packet (e.g., reading some channel values while writing to others).

Each Multispense command has a different number of allowed value fields (see Multispense Manual). If an installed channel generates a response during processing of the Message Packet, the field array will contain the response values – no matter which communication type was employed. This allows advanced error checking, even when sending a command with Value Quantity > 0.

11.4.11.1 Terminal to Protocol Converter

- 0 = Do not send the command to this channel individually.
- 1 = Send the Command to the individual Multispense channel using no values fields.
- 2 = Send the Command to the individual Multispense channels using Value 1.
- 3 = Send the Command to the individual Multispense channels using Value 1 & Value 2.
- 4 = Send the Command to the individual Multispense channels using Value 1 & Value 2 & Value 3

11.4.11.2 Protocol Converter to Terminal

This field indicates whether a response was received from the Multispense in response to the Message Packet. Note, this field is updated even when a Broadcast Command is sent (only if the channel is installed).

0 = no Command response was received from the Multispense for this channel.

- 1 = a Command response was received from the Multispense for this channel, no values received
- 2 = a Command response was received from the Multispense for this channel, Value 1 field contains response value.
- 3 = a Command response was received from the Multispense for this channel, Value 1 & Value 2 fields contain response values.
- 4 = a Command response was received from the Multispense for this channel, Value 1, Value 2, & Value 3 fields contain response values.

11.4.11.3 Error Checking Recommendations

Many Multispense RS232 commands respond with more values than received in the command to them. It is important that the Terminal software verifies that the number of values received matches the number expected for each specific command.

The Multispense RS232 protocol truncates the third value field in cases where a warning or error exists. Therefore, it is possible a command that usually responds with 3 values will only respond with 2 when a warning or error is present. In this case, Value 1 & Value 2 will contain the 2 values, and Warning Num will contain the warning number. The Value Quantity field from the Protocol Converter will be set to 3 in this case.

As mentioned above, when responses to a broadcast command (Address=0, ValueQuantity>0) are not the same the Value Quantity and Value 1, Value 2, Value 3 fields are set to 0. When this occurs, the Terminal software may check the individual channel Value Quantity Channel and Value # Channel array fields to determine which channel(s) caused the errant response.

11.4.12 Message Packet: Value 1, Value 2, Value 3 Channel [1..32]

Same function as Value 1, Value 2, and Value 3, except each channel has individual values.

11.4.12.1 Terminal to Protocol Converter

The value in each Value field depends on the command being sent to the Multispense channel. The overall range of values is 0-65,535, but is further limited by each command. The number of values sent is determined by the Value Quantity Channel field for the desired channel. Value fields that are not being used based on the Value Quantity Channel field are ignored.

11.4.12.2 Protocol Converter to Terminal

Values in all response(s) from the Multispense Controller are placed in the corresponding Value # Channel fields.

11.4.12.3 Error Checking Recommendations

The values in the Value fields should be checked against the expected values based on the specific command.

11.4.13 Message Packet: Warning Number Channel [1..32]

Same function as Warning Number except each channel has an individual value. Also, if a communication error occurs between the Protocol Converter and the Multispense Controller, the Warning Number will contain an error number.

11.4.13.1 Terminal to Protocol Converter

This field is ignored by the Protocol Converter. It is recommended to set this field to 0 for future compatibility.

11.4.13.2 Protocol Converter to Terminal

Warning/Fault/Error Number	Source	Description
0	Protocol Converter	No warnings, faults, or errors in response to Message Packet
1 to 999	Multispense Controller	Warning number (see Multispense manual)
1000 to 1999	Multispense Controller	Fault code (see Multispense manual)
9001	Protocol Converter	Either no response or an errant response was received from the Multispense
9002	Protocol Converter	An unaddressed channel generated a response to Message Packet.
9003	Protocol Converter	The internal RS232 hardware of the Protocol Converter has an error
9004	Protocol Converter	A channel responded with a different Command than that in the Message Packet.

11.4.14 Industrial Ethernet Protocol: EthernetIP

The Message Packet registers are mapped to cyclical data registers and may be communicated at rates up to 10ms. The EthernetIP adapter file, BuR-X20IF10D3-1.eds contains a generic EthernetIP EDS file for configuration with a PLC. The EthernetIP adapter configuration is as follows:

Parameter	Value
Module Name	X20IF10D3_1
Keying Method	No keying
Originator to Target, RT transfer format:	32-bit run/idle header
Target to originator, RT transfer format:	Connection is pure data and is modeless
Connection Name	Connect1
Assembly instance, IN:	Connect1, ID = 1, Data length = 338
Assembly instance, OUT:	Connect1, ID = 2, Data length = 338
Watchdog Timeout:	1000ms

The registers that map the Message Packet fields are follows (inbound separate from outbound):

Message Packet Field	EthernetIP Type
Enable	UINT
Message_Id	UINT
Command	UINT
Address	UINT
Value_Quantity	UINT
Value1	UINT
Value2	UINT
Value3	UINT
Warning_Num	UINT
Value_Quantity_Channel[1..32]	UINT[32]
Value_1_Channel[1..32]	UINT[32]
Value_2_Channel[1..32]	UINT[32]
Value_3_Channel[1..32]	UINT[32]
Warning_Num_Channel[1..32]	UINT[32]

11.4.15 Industrial Ethernet Protocol: Profinet

The protocol converter employs a B&R X20IF10E3-1, which is a Profinet RT device (slave). Two RJ45 ports with an integrated switch provide easy daisy-chain connections between devices.

Parameter	Value
Name of Station	X20if10e3-1
Process image storage format	Big Endian (MSB first)
Watchdog Timeout:	1000ms
GSDML file:	GSDML-V2.31-BuR-X20IF10E3_1-20141127.xml

The following table lists the configuration module:

Configuration Module	Slot	Subslot
X20IF10E3-1_NETX V1.3.x.x	0	1
PN-IO	0	32768
Port 1	0	32769
Port 2	0	32770

The following is the table of the inbound modules.

Message Packet Field	Type	Slot	Subslot	Address
Enable	UINT	1	1	0x0000
Message_Id	UINT			0x0002
Command	UINT			0x0004
Address	UINT			0x0006
Value_Quantity	UINT			0x0008
Value1	UINT			0x000A
Value2	UINT			0x000C
Value3	UINT			0x000E
Warning_Num	UINT			0x0010
Unused	UINT			0x0012
Value_Quantity_Channel[1..32]	UINT[32]			2
Value_1_Channel[1..32]	UINT[32]	3	1	0x0054
Value_2_Channel[1..32]	UINT[32]	4	1	0x0094
Value_3_Channel[1..32]	UINT[32]	5	1	0x00D4
Warning_Num_Channel[1..32]	UINT[32]	6	1	0x0114

The following is the table of the outbound modules.

Message Packet Field	Type	Slot	Subslot	Address
Enable	UINT	7	1	0x0000
Message_Id	UINT			0x0002
Command	UINT			0x0004
Address	UINT			0x0006
Value_Quantity	UINT			0x0008
Value1	UINT			0x000A
Value2	UINT			0x000C
Value3	UINT			0x000E
Warning_Num	UINT			0x0010
Unused	UINT			0x0012
Value_Quantity_Channel[1..32]	UINT[32]			8
Value_1_Channel[1..32]	UINT[32]	9	1	0x0054
Value_2_Channel[1..32]	UINT[32]	10	1	0x0094
Value_3_Channel[1..32]	UINT[32]	11	1	0x00D4
Warning_Num_Channel[1..32]	UINT[32]	12	1	0x0114

11.4.16 Industrial Ethernet Protocol: Modbus TCP

The following tables show the register locations of the Message Packet using the Modbus protocol. When using channel arrays, the Message Packet is larger than the maximum Modbus register transmission limit (125). In this case, the Message needs to be sent/received in pieces (up to the discretion of the designer). It is important to access the data in a way that maintains the coherence of the parts in regards to a single Message Packet.

Message Packet Input	Holding Register	Length
Enable	24576	1
Message Id	24577	1
Command	24578	1
Address	24579	1
Value Quantity	24580	1
Value 1	24581	1
Value 2	24582	1
Value 3	24583	1
Warning Num	24584	1
Value Quantity Channel [1..32]	24585	32
Value 1 Channel [1..32]	24617	32
Value 2 Channel [1..32]	24649	32
Value 3 Channel [1..32]	24681	32
Warning Num Channel [1..32]	24713	32

Message Packet Output	Output Register	Length
Enable	8192	1
Message Id	8193	1
Command	8194	1
Address	8195	1
Value Quantity	8196	1
Value 1	8197	1
Value 2	8198	1
Value 3	8199	1
Warning Number	8200	1
Value Quantity Channel [1..32]	8201	32
Value 1 Channel [1..32]	8233	32
Value 2 Channel [1..32]	8264	32
Value 3 Channel [1..32]	8297	32
Warning Number Broadcast [1..32]	8329	32

11.4.17 Message Packet Examples

The following shows an example sequence of changing the volume using the registers. The following examples assume that the Protocol Converter is connected to a 2-channel Multispense 2000. For the sake of brevity, all registers that are irrelevant for the specific example are omitted (should all be at 0 in each Message Packet). In each example, the Outbound Message Packet table is what is sent, the Inbound Message Packet is the normal expected response.

In these examples the Message Id is incremented to demonstrate how it is expected to increment for each successive packet. Any non-zero value may be used in this field as long as it changes from packet to packet.

11.4.17.1 Initial Packet (after power-up)

This packet is useful to reset the Protocol Converter message id count and to check for when the Protocol Converter power-on sequence has completed (when Inbound Enable transitions to 1).

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
Enable	1	Enable	1
Message_Id	0	Message_Id	0

11.4.17.2 Firmware Revision ('z' command)

The following is used to check the firmware version of the channel cards after power-up. Since the channel cards need to be referenced, they will respond with a warning number rather than a third value. This warning number takes place of the third value, making it impossible to read the third value until all warnings/faults/errors are cleared (this also applies to the drawback 'w' command which also requires 3 values). The following response is for firmware version JHD26214. In the example tables below, the second row of the table shows the RS232 command and response. Refer to the section in the Multispense 2000 manual describing the 'z' command to convert the numerical value to the version as text.

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0z)		(1z19016,17422*4;2z19016,17422*4)	
Enable	1	Enable	1
Message_Id	1	Message_Id	1
Command	122	Command	122
Address	0	Address	0
Value_Quantity	1	Value_Quantity	3
		Value1	19016
		Value2	17422
		Value3	0
		Warning_Num	4
		Value_Quantity_Channel[1]	3
		Value_Quantity_Channel[2]	3
		Value_1_Channel[1]	19016
		Value_1_Channel[2]	19016
		Value_2_Channel[1]	17422
		Value_2_Channel[2]	17422
		Value_3_Channel[1]	0
		Value_3_Channel[2]	0
		Warning_Num_Channel[1]	4
		Warning_Num_Channel[2]	4

Here is what it would look like if the channels were all referenced. Notice in this example that Value 3 now contains the correct firmware version value and the warning number is now 0. Also, notice the Value Quantity field is now 4 which indicates all 3 value fields contain data.

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
Retain		(1z19016,17422,262;2z19016,17422,262)	
Enable	1	Enable	1
Message_Id	1	Message_Id	1
Command	122	Command	122
Address	0	Address	0
Value_Quantity	1	Value_Quantity	4
		Value1	19016

	Value2	17422
	Value3	262
	Warning_Num	0
	Value_Quantity_Channel[1]	4
	Value_Quantity_Channel[2]	4
	Value_1_Channel[1]	19016
	Value_1_Channel[2]	19016
	Value_2_Channel[1]	17422
	Value_2_Channel[2]	17422
	Value_3_Channel[1]	262
	Value_3_Channel[2]	262
	Warning_Num_Channel[1]	0
	Warning_Num_Channel[2]	0

11.4.17.3 Status Check ('q' command)

The 'q' command checks the ready/busy status of the channels to determine whether the channels are ready to receive motion commands. Notice the '4' in the Warning_Num field indicates that the channels need to be referenced (always the case for a MS2000 channel card upon power-up). The Value 1 field indicates the Ready/Busy status (0 = Ready, >0 = busy). This example assumes the channel is ready (though in need of a reference).

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0q)		(1q0*4;2q0*4)	
Enable	1	Enable	1
Message_Id	2	Message_Id	2
Command	113 (g)	Command	113
Address	0	Address	0
Value_Quantity	1	Value_Quantity	2
		Value1	0
		Warning_Num	4
		Value_Quantity_Channel[1]	2
		Value_Quantity_Channel[2]	2
		Value_1_Channel[1]	0
		Value_1_Channel[2]	0
		Warning_Num_Channel[1]	4
		Warning_Num_Channel[2]	4

11.4.17.4 Reference ('f' command) and Status ('q' command)

The 'f' command initiates a reference command to the channels:

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0f)		(1f*4;2f*4)	
Enable	1	Enable	1
Message_Id	3	Message_Id	3
Command	102 (f)	Command	102
Address	0	Address	0
Value_Quantity	1	Value_Quantity	1
		Warning_Num	4
		Value_Quantity_Channel[1]	1
		Value_Quantity_Channel[2]	1
		Warning_Num_Channel[1]	4
		Warning_Num_Channel[2]	4

The following 'q' should be sent repeatedly to monitor the ready/busy status of the channels. The inbound packet is the expected response when both channels are busy referencing. Notice that the response is contained both in the basic registers and the field array registers.

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0q)		(1q1*4;2q1*4)	
Enable	1	Enable	1
Message_Id	4	Message_Id	4
Command	113	Command	113
Address	0	Address	0
Value_Quantity	1	Value_Quantity	2
		Value1	1
		Warning_Num	4
		Value_Quantity_Channel[1]	1
		Value_Quantity_Channel[2]	1
		Value_1_Channel[1]	1
		Value_1_Channel[2]	1
		Warning_Num_Channel[1]	4
		Warning_Num_Channel[2]	4

The following 'q' packet demonstrates the response when one of the channels (in this example, channel 1) completes referencing before the other (this is common). Notice that the basic registers are now 0 because the channel values are different.

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0q)		(1q0;2q1*4)	
Enable	1	Enable	1
Message_Id	5	Message_Id	5
Command	113	Command	113
Address	0	Address	0
Value_Quantity	1	Value_Quantity	0
		Value1	0
		Warning_Num	0
		Value_Quantity_Channel[1]	1
		Value_Quantity_Channel[2]	1
		Value_1_Channel[1]	0
		Value_1_Channel[2]	1
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	4

The following 'q' packet demonstrates the response both channels have completed their reference sequence:

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0q)		(1q0;2q0)	
Enable	1	Enable	1
Message_Id	6	Message_Id	6
Command	113	Command	113
Address	0	Address	0
Value_Quantity	1	Value_Quantity	2
		Value1	0
		Warning_Num	0
		Value_Quantity_Channel[1]	1
		Value_Quantity_Channel[2]	1

	Value_1_Channel[1]	0
	Value_1_Channel[2]	0
	Warning_Num_Channel[1]	0
	Warning_Num_Channel[2]	0

There is an alternate 'q' command (99q) that may be used to check the busy/ready status of all the channels (equivalent to the System Ready signal available on the System Logic I/O connector). This is the recommended method to check busy/ready status if not using the System Logic I/O. Before using this, review the channel "h" command, which defines the ready/busy state for different operations.

'99q' (recommended alternate to '0q') when at least one channel is busy (notice Value 1 is '1'):

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(99q)		(99q1)	
Enable	1	Enable	1
Message_Id	5	Message_Id	5
Command	113	Command	113
Address	99	Address	99
Value_Quantity	1	Value_Quantity	2
		Value1	1
		Warning_Num	0
		Value_Quantity_Channel[1]	0
		Value_Quantity_Channel[2]	0
		Value_1_Channel[1]	0
		Value_1_Channel[2]	0
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

'99q' (recommended alternate to '0q') when all channels are ready (notice Value 1 is '0'):

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(99q)		(99q0)	
Enable	1	Enable	1
Message_Id	6	Message_Id	6
Command	113	Command	113
Address	99	Address	99
Value_Quantity	1	Value_Quantity	2
		Value1	0
		Warning_Num	0
		Value_Quantity_Channel[1]	0
		Value_Quantity_Channel[2]	0
		Value_1_Channel[1]	0
		Value_1_Channel[2]	0
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

11.4.17.5 Volume ('v' command)

It is not necessary to read the volume before writing it, but the example is shown for reading the volume. This example assumes the volumes are identical in all of the channels:

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0v)		(1v400;2v400)	

Enable	1	Enable	1
Message_Id	7	Message_Id	7
Command	118	Command	118
Address	0	Address	0
Value_Quantity	1	Value_Quantity	2
		Value1	400
		Warning_Num	0
		Value_Quantity_Channel[1]	2
		Value_Quantity_Channel[2]	2
		Value_1_Channel[1]	400
		Value_1_Channel[2]	400
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

The following example shows the result of reading volume when the volumes are different. Notice that the main Value Quantity field goes to 0 and the values are available in the channel array fields only:

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0v)		(1v400;2v1000)	
Enable	1	Enable	1
Message_Id	7	Message_Id	7
Command	118	Command	118
Address	0	Address	0
Value_Quantity	1	Value_Quantity	0
		Value1	0
		Warning_Num	0
		Value_Quantity_Channel[1]	2
		Value_Quantity_Channel[2]	2
		Value_1_Channel[1]	400
		Value_1_Channel[2]	1000
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

The following example shows writing the volume of all channels to an identical value. Notice that both the main value registers and the channel array field registers contain the results:

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0v)		(1v500;2v500)	
Enable	1	Enable	1
Message_Id	8	Message_Id	8
Command	118	Command	118
Address	0	Address	0
Value_Quantity	2	Value_Quantity	2
Value1	500	Value1	500
		Warning_Num	0
		Value_Quantity_Channel[1]	2
		Value_Quantity_Channel[2]	2
		Value_1_Channel[1]	500
		Value_1_Channel[2]	500
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

The following example shows writing the volume of one channel to a different value using the address field. Notice that since only 1 channel is addressed, only 1 channel responds (i.e., channel 2 response is not contained in the message packet), and that the channel array fields are not used.

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(1v1500)		(1v1500)	
Enable	1	Enable	1
Message_Id	9	Message_Id	9
Command	118	Command	118
Address	1	Address	1
Value_Quantity	2	Value_Quantity	2
Value1	1500	Value1	1500
		Warning_Num	0
		Value_Quantity_Channel[1]	0
		Value_Quantity_Channel[2]	0
		Value_1_Channel[1]	0
		Value_1_Channel[2]	0
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

The following is an alternate method of writing the volume to a single channel using the channel field arrays. Notice that the basic value registers are not used:

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(1v1500)		(1v1500)	
Enable	1	Enable	1
Message_Id	9	Message_Id	9
Command	118	Command	118
Address	0	Address	0
Value_Quantity_Channel[1]	2	Value_Quantity_Channel[1]	2
Value_Quantity_Channel[2]	0	Value_Quantity_Channel[2]	0
Value_1_Channel[1]	1500	Value_1_Channel[1]	1500
Value_1_Channel[2]	0	Value_1_Channel[2]	0
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

The following example shows writing the volume of each channel to a different value using a single message packet. Notice again that the basic value registers are not used.

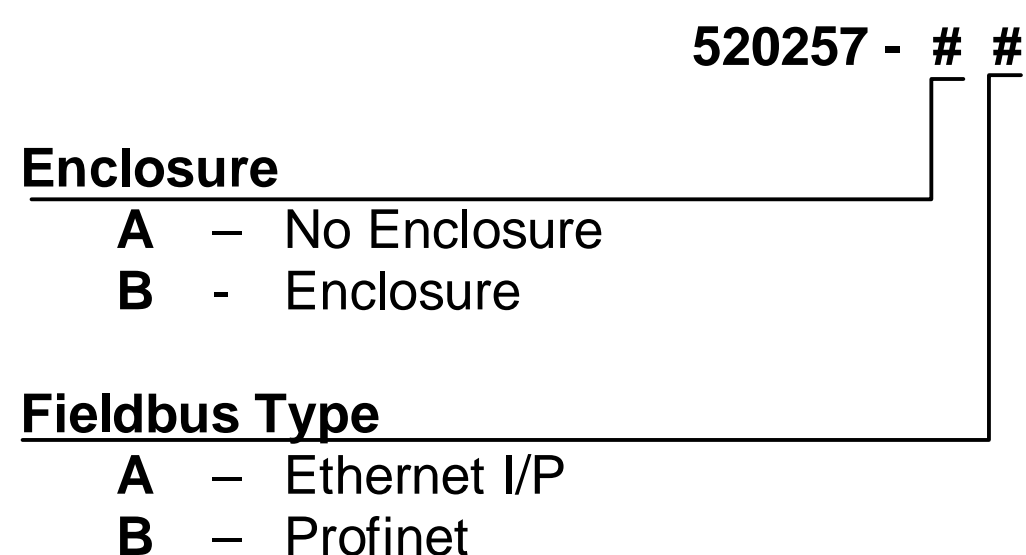
Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(1v1200) then (2v300)		(1v1200) then (2v300)	
Enable	1	Enable	1
Message_Id	10	Message_Id	10
Command	118	Command	118
Address	0	Address	0
Value_Quantity_Channel[1]	2	Value_Quantity_Channel[1]	2
Value_Quantity_Channel[2]	2	Value_Quantity_Channel[2]	2
Value_1_Channel[1]	1200	Value_1_Channel[1]	1200
Value_1_Channel[2]	300	Value_1_Channel[2]	300
		Warning_Num_Channel[1]	0
		Warning_Num_Channel[2]	0

The following example shows a more advanced feature when most channels have the same volume, with a few channels that differ. The example here uses an 8-channel MS2000, setting all the channels to one volume, and then setting channels 7 & 8 to different volumes.

Outbound Message Packet Field	Value	Inbound Message Packet Field	Value
(0v20) then (7v800) then (8v900)		(1v20;2v20;3v20;4v20;5v20;6v20;7v20;8v20) then (7v800) then (8v900)	
Enable	1	Enable	1
Message_Id	11	Message_Id	11
Command	118	Command	118
Address	0	Address	0
Value_Quantity	2	Value_Quantity	2
Value1	20	Value1	20
Warning_Num	0	Warning_Num	0
Value_Quantity_Channel[1]	0	Value_Quantity_Channel[1]	2
Value_Quantity_Channel[2]	0	Value_Quantity_Channel[2]	2
Value_Quantity_Channel[3]	0	Value_Quantity_Channel[3]	2
Value_Quantity_Channel[4]	0	Value_Quantity_Channel[4]	2
Value_Quantity_Channel[5]	0	Value_Quantity_Channel[5]	2
Value_Quantity_Channel[6]	0	Value_Quantity_Channel[6]	2
Value_Quantity_Channel[7]	2	Value_Quantity_Channel[7]	2
Value_Quantity_Channel[8]	2	Value_Quantity_Channel[8]	2
Value_1_Channel[1]	0	Value_1_Channel[1]	20
Value_1_Channel[2]	0	Value_1_Channel[2]	20
Value_1_Channel[3]	0	Value_1_Channel[3]	20
Value_1_Channel[4]	0	Value_1_Channel[4]	20
Value_1_Channel[5]	0	Value_1_Channel[5]	20
Value_1_Channel[6]	0	Value_1_Channel[6]	20
Value_1_Channel[7]	800	Value_1_Channel[7]	800
Value_1_Channel[8]	900	Value_1_Channel[8]	900
Warning_Num_Channel[1]	0	Warning_Num_Channel[1]	0
Warning_Num_Channel[2]	0	Warning_Num_Channel[2]	0
Warning_Num_Channel[3]	0	Warning_Num_Channel[3]	0
Warning_Num_Channel[4]	0	Warning_Num_Channel[4]	0
Warning_Num_Channel[5]	0	Warning_Num_Channel[5]	0
Warning_Num_Channel[6]	0	Warning_Num_Channel[6]	0
Warning_Num_Channel[7]	0	Warning_Num_Channel[7]	0
Warning_Num_Channel[8]	0	Warning_Num_Channel[8]	0

11.5 MODEL NUMBER

The model number provides important information about the specifics of your Pump Module. Refer to this number when calling IVEK Technical support. The model number for your Pump Module is located in the Title Page section of this manual.



Appendix

The following table provides the decimal value equivalent to each valid Multispense ASCII command. The command set, description, and required values are contained in the Multispense manual.

ASCII Command	Decimal Value	ASCII Command	Decimal Value
a	97	p	112
b	98	q	113
c	99	r	114
d	100	s	115
e	101	t	116
f	102	u	117
g	103	v	118
h	104	w	119
k	107	y	121
l	108	z	122
m	109		