



DeltaV/PanelView Data Connection

Introduction

The PanelView 550, 600, 900, 1000, and 1400 allow class1 (implicit) connections via Ethernet/IP from the VIM (Virtual I/O Module) connected to a DeltaV Controller. The PanelView acts as a server with the DeltaV controller receiving from and writing data to locations in the PanelView Assembly Object tables. These assembly areas may be mapped in the PanelView to data tags that are used in displays and logic. These assembly areas are continuously updated between the PanelView and the VIM. (AI tables hold values that will be sent to the VIM; the AO tables hold the values that have been sent by the VIM).

Two types of connections to DeltaV include simple and mapped. Simple connections use one data type for all data in the IO connection. Mapped connections use several DeltaV datasets with different data types and “map” portions of the data buffers of the connection to the datasets, dependent on the type of data expected in the IO buffer. The mapped connections may be used to transfer complex data structures between the two systems.

PanelView Configuration

To access data in PanelView, select the Tag Editor from the system tree entry. This opens the editor dialog; DeltaV data is mapped by selecting the “Enet-Assembly Object” Tab.

This dialog consists of a tag name field followed by fields that specify the data type, size (allowing arrays of data), and location (or assembly address) of the data for the tag. Other values are not necessary for the configuration of the communications connections, so they are not discussed here.

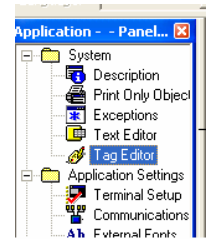


Figure 1

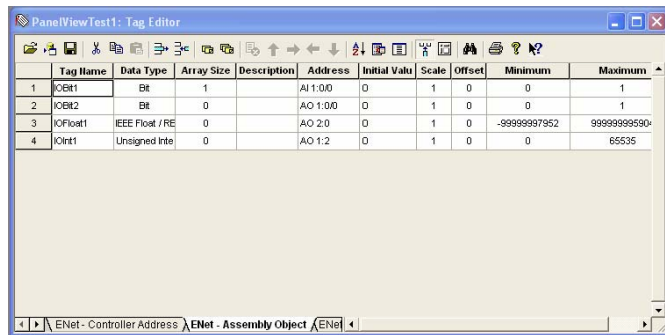


Figure 2

The PanelView assembly object addresses are referenced as AO x:y/z and AI x:y/z, where AO and AI are the Assembly Output or input object tables. X specifies the object instance number (1-8). Y is the word (16 bit) offset in the instance (0-224). Finally, /z is the bit offset (0-15) in the word for bit access (Boolean or discrete points). AO instances are written to from DeltaV while AI instances are those read by DeltaV.

The location (address) tab specifies the starting address of the data for the tag in the assembly area. The PanelView contains 8 input and 8 output tables of data in the assembly. Each table consists of 224 words of data that may be used as variables in the PanelView configuration.



The assembly area is constant in the PanelView; however, the size of the array that needs to be included in the communication connection message buffer is dependent on the highest address (and the size of the data structure assigned to this address). It is not necessary to transfer the complete data table. Using this method, the communications is initialized. The size and timing of the communications messages is determined by the VIM.

Simple Connections

For simple connections, the registers in the assembly object may be accessed in the PanelView as any of the supported data types; however, the data type should match the accessing dataset data type in DeltaV. This means if the dataset in DeltaV is a floating point value, then all PanelView tags associated with the assembly object instance populated (or read) from the DeltaV dataset should be floating point. If the DeltaV is a 16-bit unsigned, then the PanelView tags should be unsigned integers (or packed bits). In Figure 2 above, two bit tags are assigned. One is assigned to the input instance 1, and another is assigned to the output instance 1. Any bit assignments could be assigned to this, and the DeltaV dataset could be BOOL, discrete, or a 16-bit unsigned. If the set were to include 16-bit unsigned values, then only a DeltaV 16-bit unsigned data set would be appropriate. The third tag Figure 2 is a floating-point value; this is assigned to the second instance of the output table. The dataset associated with the 2nd assembly object should be defined as floating point.

See “Accessing Data in DeltaV”, “Output Assembly Table (AO) Address Mapping”, and “Input Assembly Table (AI) Address Mapping” below for detailed definitions of the mapping of datasets.

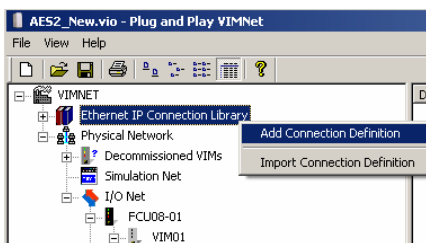
Mapped Mixed Data Type Connections

This data type is more complex to configure than the simple connection; however, communication connections may be optimized by using the mapping utility in the VimNet Explorer to transfer different data types in a single set of Ethernet/IP assembly buffers. These different types may then be mapped in DeltaV to several datasets associated with the connection.

VIM Configuration

VIM Configuration specifies specific configuration information for configuring PanelView connections to the VIM. It assumes the user is familiar with configuring the VIM using the VimNet Explore utility.

Open the VimNet Explorer. If the appropriate configuration is not already loaded, create a configuration containing the VIM and PanelView Device. Right click on the “Ethernet IP Connection Library” leaf in the navigation panel on the left of the display. Select “Add Connection Definition”.



The simplest way to configure a connection would be to create two separate connections: one for output data and a second for input data. Each of these would require only one dataset (if less than 200 bytes of data is to be transferred). If both are configured in one connection, then you will need to assign multiple datasets to the connection. The number of datasets depends on the number of registers required in each direction.

Figure 3



This opens the "VIM_EtipBufferMapping" Utility shown in Figure 4. This is the utility that allows the configuration of the connection between the VIM and the PanelView.

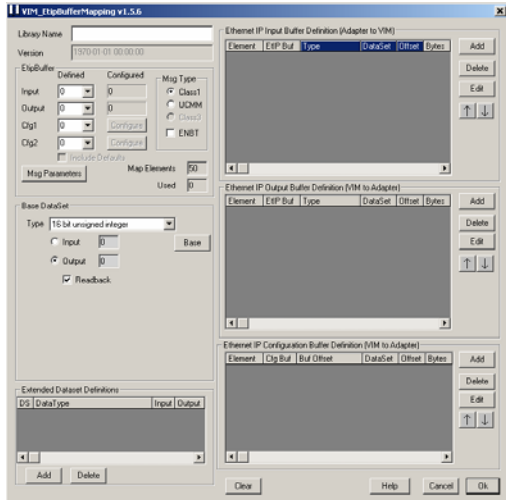


Figure 4

Enter a name in the "Library Name" field. Then configure the Etip Buffer section below. Select the message type radio button (Class1), and make sure the ENBT field is unchecked.

In the "Input" and "Output" fields, select the number of bytes that will be included in the communications message buffers. The Input specifies the size of the AI assembly data in the PanelView that will be transmitted to the VIM, while the Output specifies the bytes transmitted to the AO assembly table.

Note: It is important to note that the input buffer will include a 4-byte header as well as the AI assembly area you are capturing. This must be added to the bytes in the "EtipBuffer", "Input", "Defined" box in this dialog.

If you are using mapped IO, then you should start the Input buffer definition mapping elements with a 4 byte segment set to "unmapped". Checking the "Unmapped Bytes" checkbox in the edit dialog does this.

If you use un-mapped IO, then you must make the DeltaV input dataset 4 bytes (2 unsigned integer, 1 floating point, or 32 discrete registers, larger than the AI table in the PanelView assembly area. In this case the PanelView data is offset by the 4 bytes in DeltaV.

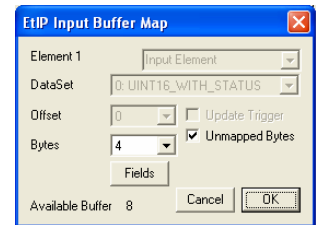


Figure 5

Next, select the "MsgParameters" button to open the "SpecialData" utility (Table 1). This utility allows you to enter the necessary parameters to configure the actual Ethernet/IP Class1 connection. First, enter the Connection Point for this connection. The connection point is related to but not the same as the PanelView Address specified in the PanelView configuration utility. Specify parameters for Originator to Target (OT) and Target to Originator (TO) connections.

Table 1). This utility allows you

Table with 3 columns: Table (AO/AI), Inputs (AI) or TO, Outputs (AO) or OT. Rows 1-8 and N/A.

Table 1

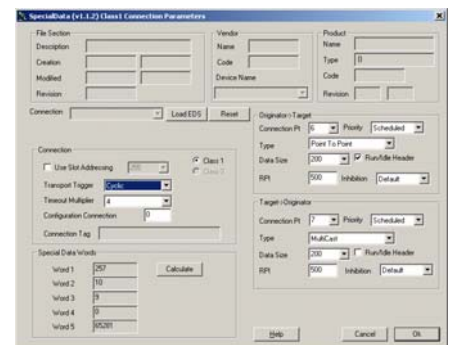


Figure 6



In Table 1, set the Originator (VIM) to Target (PanelView) at 6, in order to specify the AO1 table. The values may be from 6 to 20 (even numbers for the AO table). Next, set the Target (PanelView) to Originator (VIM) at 7, to specify AI1. The mapping for all the AO and AI tables is shown in Table 1. If the connection is one-way, then specify a connection point of 255 for the direction that will not carry data.

Both OT and TO connections should be set to "Scheduled" and Inhibition "Default". The OT connection should be "Point to Point" while TO should be "MultiCast". Make sure the OT connection has the "Run/Idle Header" checkbox checked and the TO does not. Now set the RPI (requested packet interval) to the required interval. This is the expected time for the PanelView (TO) and VIM (OT) to send data messages. The Data size for both of these was set using the "VIM_EtipBufferMapping" utility.

Finally, you need to set the overall connection parameters. These are the "Use Slot Addressing" (unchecked), Transport Trigger, "Cyclic", and Timeout Multiplier (4). Leave the Configuration Connection at 0.

The last step in the SpecialData words is to select "OK" to save these parameters into the current connection definition in the VIM_EtipBufferMapping" utility.

Using the VIM_EtipBufferMapping" utility; select the datasets to which the connections will be mapped. All connections have at least one dataset associated; this is assigned in the Base DataSet section. If there are both input and output connections, this will require two datasets (unless the data values in both assembly tables are intended to be the same). In Figure 7, the first dataset is configured as a 16-bit unsigned integer output (without readback). The second dataset is then configured as a 16-bit unsigned integer input. This allows the connection to write one set of values to the PanelView and read the second dataset back from it.

If mapping data of varied types (i.e., integer and floating point), you will need to assign datasets for both types.

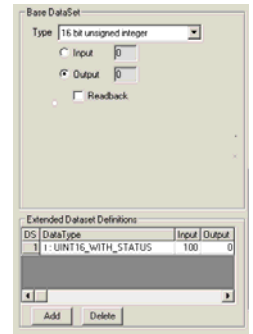


Figure 7

Next, you will set the specific registers to transfer. Select the "Add" button in the Ethernet IP Input Buffer Definition (Adapter to VIM) to add the Input Definition. This dialog allows the user to select the dataset (1:

UINT16_WITH_STATUS) for the inputs, set the offset in the dataset (0) and number of bytes (200). Both "Update Status" and "Unmapped Bytes" should be unchecked. If using mixed data types, then you will need to enter more than one element in the input (and output) buffer tables. Each element will point to a specific range of bytes in the buffer and map these to the specific dataset registers for that type.

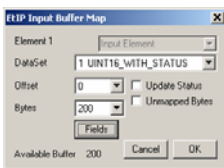


Figure 8

Selecting the "Fields" button will open a dialog (Ethernet/IP Buffer Fields) shown in Figure 9 that will allow you to enter a series of field definitions. Each field may specify a PanelView tag definition assigned to the assembly table(s) for the connection.

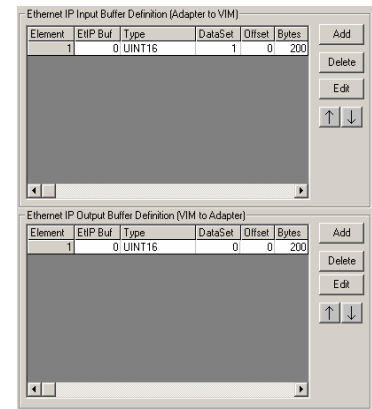
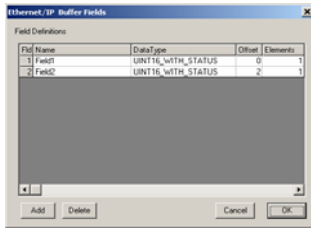


Figure 9



These have no effect on the communications between the PanelView and DeltaV; however, the field definitions allow you to document the format of the transfer buffer and will show the specific DeltaV dataset registers associated with the specified fields.

Each field is added by selecting the “Add” button. Then, fill out the parameters (including the data type of the field), the starting byte in the field, and bit (if a bit field). Also, the size (elements) may be set if the field is an array of the data type.

Figure 10

Finally, select “OK” on all of the dialog boxes to accept the configured values. You are returned to the VimNet Explorer “Ethernet IP Connection Library” with the new definition added. To add this to the VIM configuration, select the Node, the specific VIM, card, port and device for the connection, then, then right click on the device and select “Add Connection” In the dialog box, add a description, and select the specific connection definition from the “Ethernet Device Definition” combo box. The connection is added to the list and the next available dataset(s) in DeltaV are assigned. You can open this connection by selecting the plus (+) in front of it, and examine the definition of each assigned dataset here. When configuring the actual DeltaV datasets (DeltaV Explorer), you may use these definitions as a reference.



DeltaV Configuration

The actual connection between the PanelView and DeltaV is configured in VimNet Explorer. The DeltaV dataset(s) assigned in DeltaV Explorer may be on any card (57-60) / port(1-2) and device, but they must match the definition as specified in VimNet Explorer. One or more datasets may be assigned to a connection as necessary. The first dataset defines the connection; the remaining datasets for the connection (if required) must immediately follow the first.

To start, using DeltaV Explorer, select the device, right click, and select "New Dataset" to open the Dataset properties dialog.

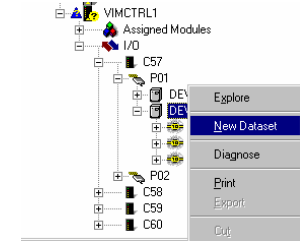


Figure 11

Select the data direction of the connection data for this dataset. If you are assigning an output assembly instance (AO) to the data, this would be "Output." If the assembly instance was input (AI), then use "Input". Do not use Output read back, as the input and output assemblies are different data tables, and the read back data would not reflect the data written. Refer to the VimNet Explorer definition of the dataset for actual requirements.

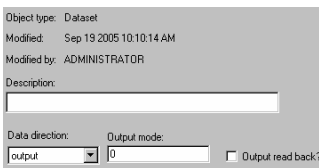


Figure 12

The DeltaV tab of the dataset property box is used to select the data type of the data that will be presented. This should be appropriate for the data that will be accessed in the PanelView. If the data in the assembly table is floating point, only a floating point dataset in DeltaV is appropriate as noted above. Mapping of datasets in the connection will allow this to be assigned separately from integer and bit registers.



Figure 13

On the PLC tab, the "Device data type" may be specified, but is not required. If specified, the first dataset associated with the connection is specified with a device data type of 39 (mapped EthernetIP Class1 client) on the PLC tab. The remaining datasets are assigned type 36 (EthernetIP extension data). In addition, the number of values on this tab will determine the size of the connection data read/written from this dataset. The total data in both directions is determined by the number of registers in the associated datasets, by the data direction, type, and size in each dataset.

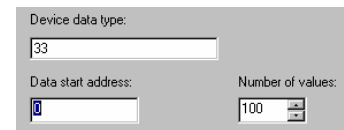


Figure 14

All words on the special data tab should be 0 (any values here are ignored).



Figure 15



Accessing Data in DeltaV

While PanelView assembly tables are mapped using zero based word addressing, the DeltaV data is mapped into datasets of up to 200 bytes each. This conversion in data addresses in DeltaV is different from that in the PanelView, and varies based on the data type associated with the dataset. Also, because the PanelView AI (data read from the PanelView) starts with 32 bits of status information, the input and output mapping differ. The mapping for each type of data is described in the sections below. It is possible to map up to 224 words of output and 226 words of input data. If more data is mapped, the connection will fail the forward open with a status of 0x01 and an extended status of 0x0109 (capture of the actual status requires examination of the data packets).



Output Assembly Table (AO) Address Mapping

All Assembly tables in the PanelView are mapped using zero based word addressing. AO x:n/b where 'x' is the table (1-8), 'n' is the word (0-223), and 'b' is the bit (for discrete points, 0-15).

PanelView	DeltaV							
Assembly Word	DS	Registers						
		Float / 32 Bit	16 Bit	Byte	Discrete / Boolean			
0	1	1	1	1	1-8			
1				2	9-16			
2			2	3	3	17-32		
3					4	33-48		
4				4	5			
5					6			
...				
99		1	50	99	197			
100					198			
101				2	1	100	199	
102							200	
103		2	3			1		
104	2							
...				
198	2	50	99	197				
199				198				
200			3	1	100	199		
201						200		
...				
222	3	12	23	1				
223				2				
...			24	23	3			
...					4			



Input Assembly Table (AI) Address Mapping

All Assembly tables in the PanelView are mapped using zero based word addressing. AI x:n/b where 'x' is the table (1-8), 'n' is the word (0-223), and 'b' is the bit (for discrete points, 0-15). For input words (read from the PanelView) the first two words in the dataset contain status information. The actual table data starts in word 3. The column "Un-mapped Status bytes" is configured using mapping elements with the first input mapping element specifying 4 bytes of un-mapped data in Input buffer (these are not transferred to the DeltaV dataset(s)).

PanelView		DeltaV				
Assembly Word		DS	Registers			
Including Status Bytes	Un-mapping Status Bytes		Float / 32 Bit	16 Bit	Byte	Discrete / Boolean
Status data	0	1	1	1	1	1-8
				2	2	9-16
					3	17-32
					4	33-48
0	1		2	3	5	
					6	
1	2			4	7	
					8	
2	3		3	5	9	
					10	
3	4			6	11	
					12	
					16	
...	
97	98	1	50	99	197	
					198	
98	99			100	199	
					200	
99	100	2	1	1	1	
					2	
100	101			2	3	
					4	
101	102		2	3	5	
					6	
102	103			4	7	
					8	
...	
197	198	2	50	99	197	
					198	
198	199			100	199	
					200	
199		3	1	1	1	
					2	
200				2	3	
					4	
...	
220	222	3	11	23	47	
221	223		12	24	48	
222	N/A		13	25	49	
					50	
223				26	51	
					52	



M Y N A H

TechNote VIM 4.0.1-1
Module: Generic Ethernet/IP VIM
Version: 4.0.1
Date: December 27, 2006

Please contact us for any questions about these TechNotes at:

MYNAH Technologies

504 Trade Center Boulevard
Chesterfield, MO 63005 USA
1 888 506 9624 (North America)
1 636 681 1555 (International)
1 636 681 1660 (fax)
email: support@mynah.com

©MYNAH Technologies 2006. All rights reserved.

Designs are marks of MYNAH Technologies, Emerson Process Management, DeltaV, and the DeltaV design are marks of one of the Emerson Process Management of companies. All other marks are property of their respective owners. The contents of this publication are presented for informational purposes only, and while every effort has been made to ensure their accuracy, they are not to be construed as warranties or guarantees, expressed or implied, regarding the products or services described herein or their use or applicability. All sales are governed by our terms and conditions, which are available on request. We reserve the right to modify or improve the design or specification of such products at any time without notice.

While this information is presented in good faith and believed to be accurate, Mynah Technologies does not guarantee satisfactory results from reliance upon such information. Nothing contained herein is to be construed as a warranty or guarantee, express or implied, regarding the performance, merchantability, fitness or any other matter with respect to the products, nor as a recommendation to use any product or process in conflict with any patent. Mynah Technologies reserves the right, without notice, to alter or improve the designs or specifications of the products described herein.