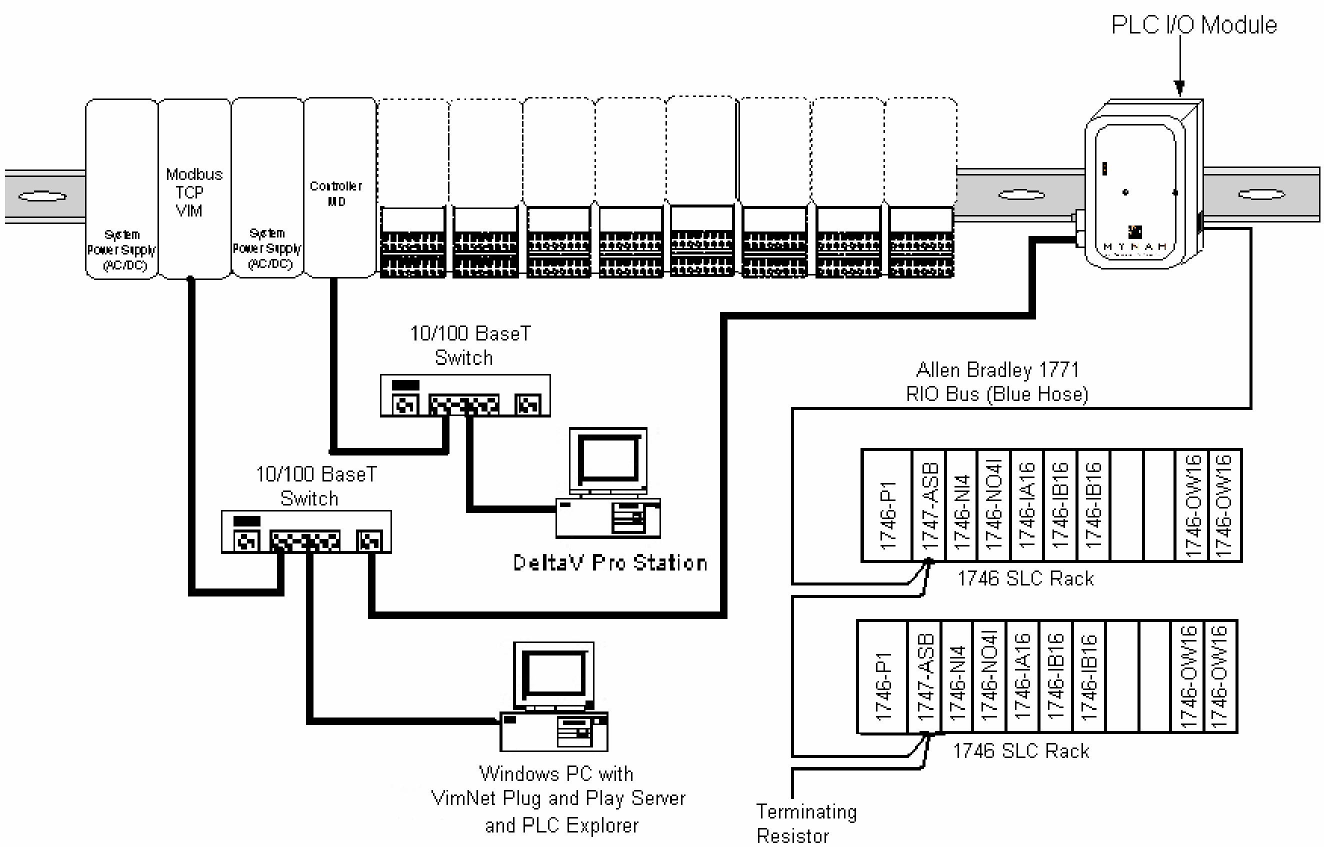
AB1771 PIO SLC (1746) Rack RIO/DeltaV Communications Configuration

Introduction

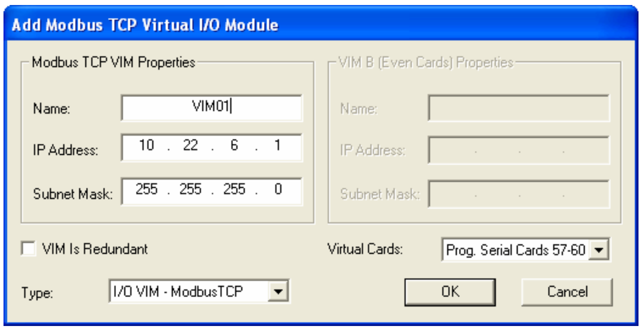
The AB1771 PIO module acts as an interface between a DeltaV controller and the Allen Bradley 1771 universal RIO bus. This module responds to ModbusTCP commands from the VIM module to read and write data tables, and asynchronously sends and receives IO data from the SLC racks via the 1771 protocol. The VIM emulates four serial cards; each card has two ports that may each hold up to 16 datasets. The datasets hold data transferred to and from the PIO module. The data is mapped according to the format detailed at the end of the document. Discrete IO, and rack status, data may be read into one dataset (up to 10 racks). This means up to 10 racks may be maintained using only 3 datasets on a port. Analog data is transferred to and from the rack using block transfers. Up to four block transfers may be included in one dataset (depending on the number of registers required for each transfer). As many datasets may be configured as needed on one or more ports. Multiple cards may be assigned if needed; datasets on any card in the VIM will communicate with the PIO.



This system is configured using two utilities, the VimNet plug and play server is used to configure the VIM device connection to the PIO, and the PLCExplorer utility configures the PIO module and creates a FHX import file to configure the DeltaV datasets and initial IO landing modules. These utilities are discussed below.

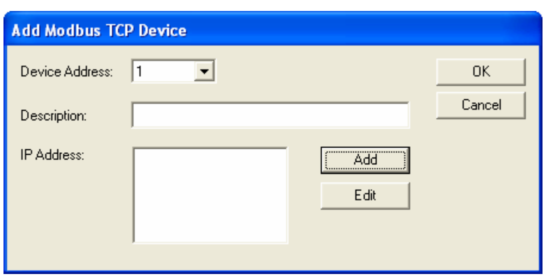
VIM Configuration

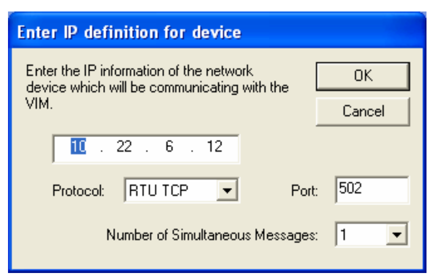
For details of this see the “Modbus TCP Master Driver for DeltaV Virtual I/O Module”. Briefly this module allows the creation of a configuration of VIM modules assigned to a DeltaV controller. The process entails adding a controller to the I/O Net branch of the IO tree, then from the controller menu (right click) select “New IO VIM”. In the configuration dialog, specify the name, IP address (and subnet). The type is “I/O VIM – ModbusTCP”, and the Virtual Cards are either “Prog. Serial Cards 57-60” or “Prog. Serial Cards 57-60”.



The “VIM is Redundant” check box should be left unchecked, as the AB1771 RIO network is a simplex network. Finally, selecting OK will enter the VIM module in the tree. Now from the VIM right click menu select “Commission” to find and assign an actual VIM to this branch.

When the VIM is added to the tree, it automatically populates the cards and ports available in DeltaV. Select a port that will be used to communicate with the PIO, right click, and select “Add Device”. This opens the dialog to assign a device address (1-64) in DeltaV to the IP address that has been assigned to the VIM. After selecting an address, selecting “Add” will open the IP definition dialog.

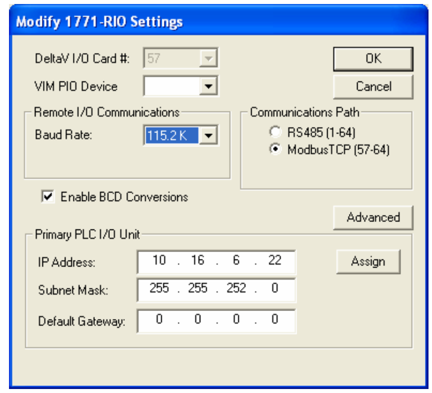




Specify the the IP address of the PIO, “RTU TCP” as the protocol, port 502, and set the number of simultaneous messages to 16. Finally, select “OK”on both dialogs to accept settings and save the VimNet configuration.

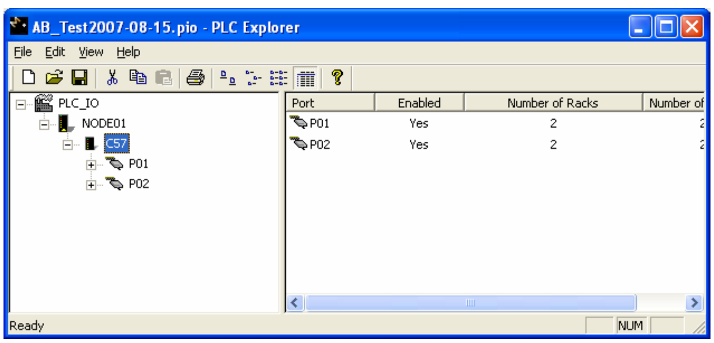
PIO Configuration

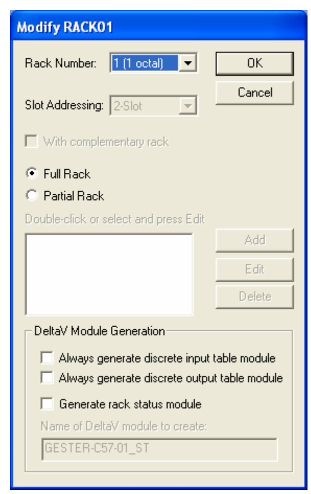
The PIO is configured using PLCExplorer, see the Allen Bradley 1771 Remote I/O Driver Users manual for details. This utility displays a tree of IO based on the PLC. Right click on the root of this tree (PLC_IO) and select "Specify IP Address" to enter the IP address of the network card on the PC that will be used to communicate with the PIO modules. Next from the same menu, select "Add Controller" and then controller type of "DeltV". Rename the Node to the same name as the DeltaV controller that has the VIM card attached.



From the controller right click and select "Add Card", and "1771 RIO". In the Card selection dialog, select the card (must be from 57-64 and match one of the cards configured in the VIM (see above). This is the location to specify the RIO network baud rate, and the communications to the PIO (check ModubsTCP). Finally enter the PIO IP address, subnet and if necessary the default gateway, these may be found automatically if the PIO is on the network by selecting the "Assign" button. Select "Ok" to enter this card in the IO tree.

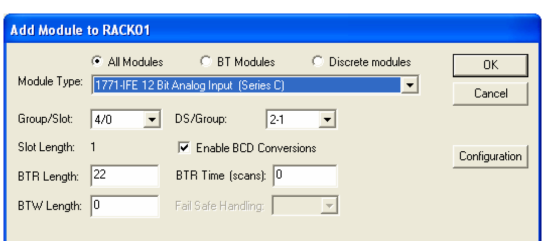
The card and two DeltaV ports are entered into the tree.



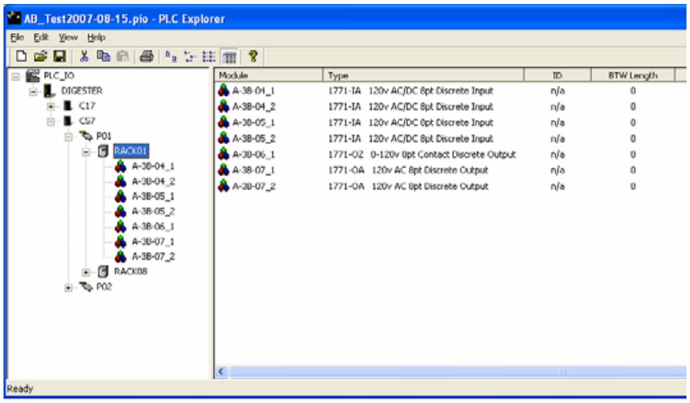


Up to 16 racks may be assigned to a port, but this number is not directly related to the number of datasets assigned to the port. The number datasets is determined by both the number of racks and type of data accessed in each rack. To add racks, select a port, right click and select "Add Rack". The rack definition dialog will be opened, select the rack to communicate with, the slot addressing type of the rack, whether the rack specifies a complementary pair, and whether full or partial (add actual partial rack configuration).

Select the rack, then for all modules in the rack, select "Add Module" from the rack context menu. Select the module from those in the list box. For analog modules if the module is not present select "user defined". Modify the values in this dialog as necessary, and set the configuration parameters by selecting the "Configuration" button. Selecting "Ok" enters the module into the tree under the rack, the generated name may be edited to local specifications.



Use the main application menu "File/Save" menu item to save the complete (or in progress) PLCExplorer definition for later use.

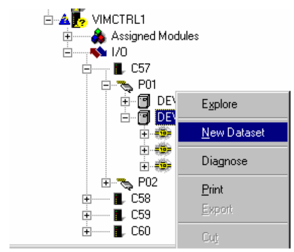


Finally when the configuration is complete, the use may select "Download Config" from the card context menu to update the configuration in the PIO module. Once the PIO is configured, this is stored internally and the PIO will recover with the configuration whenever power is restored.

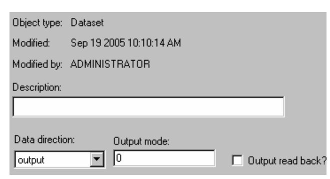
The configuration may be exported into an FHX file for import into DeltaV using the "Export FHX" from the card context menu. User will be prompted for the DeltaV area to import modules, the OPC server (used to get DeltaV version information for FHX file), then and the location to store the file.

DeltaV Configuration

The user may create the DeltaV configuration in two ways. The first is to import the FHX file that was exported from PLC Explorer (above). This will create the card definitions as well as adding IO landing modules to the configuration. Alternatively, the user may manually configure the cards, and then create IO modules as necessary.



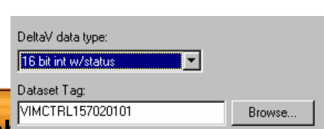
To manually configure DeltaV, use DeltaV Explorer to add the cards. All four cards emulated by the Vim (either 57-60, or 61-64) must be configured. If a port on the card is not used, leave it disabled. Add a device to a port and assign the device address configured in the VIM on the same port. This device address will specify the IP address to access for all datasets assigned to a device. Select the device, right click, and select "New Dataset" to open the Dataset properties dialog. Select the data direction of the connection data for this dataset. If you are assigning output (AO) data, this would be "Output" (and check "Output read back"), if input data (AI), then use "Input". Refer to the VimNet Explorer definition of the dataset for actual requirements. Output mode should be 0 (block mode).

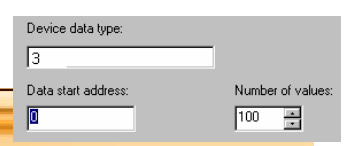


The DeltaV tab of the dataset property box is used to select the data type of the data that will be presented. This type should be always be "16-bit unsigned int w/status", as all data (including DI and DO) is accessed via analog calls.

On the PLC tab, the "Device data type" specifies the ModbusTCP communications message type:

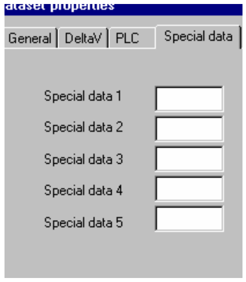
2	Input registers (AI)





3	Holding Registers (AO) as well as DI and DO (in AO format)

The Data start address is used to specify the starting location for reading data. This is one less than the first register that will be accessed, i.e., to read the AI registers 1 to 100, specify 0, for AO 101-200 specify 100. Specify the number of values to read/write, this depends on the amount of data required in one operation.



All special words but word 3 on the special data tab need be should be 0. Word 3 specifies a delay interval (in 5 ms intervals) in the messaging between the VIM and the PIO. This should be 0 for discrete modules to allow the fastest updates. For analog values this may be extended to a longer interval (set to 200 ms by PLC Explorer), as analog data normally changes more slowly than discrete. The status datasets may be set to 1000 as this is for diagnostic purposes only.

IO Data mapping

Discrete IO access via AI reads

For discrete IO, the Modbus AI read input registers command (FC = 04) may be used. The discrete values are read into unsigned 16-bit register datasets and the individual bits may be accessed with Boolean fans. This provides a faster data access than the read discrete input (DI) command (FC = 02) as a single DeltaV dataset may capture up to 9 racks of discrete IO data in a single read..

Rack to AI register mapping

All discrete input data in the PIO may be accessed via registers 300501 to 300820. Data for each rack is accessed consists of 8 words of IO data (read only) followed by two words of status data for the rack communications (see Rack discrete register group organization below). This status data is always updated in conjunction with the IO data and may be used for fail/recovery processing. The IO data is mapping to rack slots is dependent on the address formatting configured in the rack (see Slot discrete data mapping below).

AI Register	Rack Identification	
	Decimal	Octal
300501-300510	0	0
300511-300520	1	1
300521-300530	2	2
300531-300540	3	3
300541-300550	4	4
300551-300560	5	5
300561-300570	6	6
300571-300580	7	7
300581-300590	8	10
300591-300600	9	11
300601-300610	10	12
300611-300620	11	13
300621-300630	12	14
300631-300640	13	15
300641-300650	14	16
300651-300660	15	17
300661-300670	16	20
300671-300680	17	21
300681-300690	18	22
300691-300700	19	23
300701-300710	20	24
300711-300720	21	25
300721-300730	22	26
300731-300740	23	27
300741-300750	24	30
300751-300760	25	31
300761-300770	26	32
300771-300780	27	33
300781-300790	28	34
300791-300800	29	35
300801-300810	30	36
300811-300820	31	37

Holding (40xxxx) register map

The holding registers are mapped into access to status and IO data. Several address ranges are reserved for each of these types, each with a different organization to allow for optimization of access..

Global Status read/write access

The global status data may be read from the PIO via the first 360 holding registers. These registers may be written to, allowing modification of the current status (mainly resetting incrementing status values). Only some of the registers will actually be updated with a write.

Register	Rack	Register		Modbus TCP Registers	
		#	W	Low Byte	High Byte
400001	Global	80	N	SST Check Byte2 (0x04)	SST Check Byte1 (0xC3)
400002		81	N	SST Ticker	SST Check Byte3 (0xAA)
400003		82	N	RIO Baud Rate (0000=57.6, 00FF=115.2, FFFF=230.4K)	
400004		83	N	Card State (00FF=config, 0000=run, FFFF=transition)	
400005		84	N	Mode (FF=RUN, 00=PRG)	Rack communications status (COMM) (FF=OK, 00=Error)
400006		85	N	Watch dog timer (in 10 ms units, 0=OFF or 1-255)	Current Process time (in ms)
400007		86	N	Current rack (scanning)	Communications Type (0=RS485, 1=EtherentIP, 2=ModbusTCP)
400008		87	Y	RIO Scan Time (max)	RIO Scan Time (min)
400009		88	N	AB driver module revision number	
400010		89	N	AB driver status (0=OK, 1-FF are Errors)	
400011		90	Y	Sync Byte (Incremented by driver at end of each IO scan)	
400012		91	N	Auto-configure time (in millisecond units, 1-65535 ms)	
400013		92	N	Reserved (currently Debug message trigger)	
400014		93	N	BT Edit Index (1-16)	Reserved
400015		94	N	Reserved	
400016		95	N	Rack Enable 17-32	
400017		96	N	Rack Enable 1-16	
400018		97	N	Auto-Configure	
400019		98	N	Watch dog timer, 1-255 in 10 ms units	
400020		99	N	RIO Baud Rate	
400021		100	N	Module Mode bits	
				Bit	Description
				0	Mode Code 0=Program, 1=Run)
				16	Watchdog override (0=WD on, 1=WD off)
400022		N/A	N	Rack Active 17-32, controller polling	
400023		N/A	N	Rack Active 1-16, controller polling	
400024		N/A	N	Rack Active 17-32, field responding	
400025		N/A	N	Rack Active 1-16, field responding	
400026		N/A	N	Processing range high rack	Processing range low rack
400027		N/A	N	Watchdog Update Interval	
400028		N/A	N	ModbusTCP connections	Connection Type (0=Modbus, 1=485)
400029		N/A	N	ModbusTCP Messages (number of messages received)	
400030		N/A	N	ModbusTCP timing (interval between messages, mean of last 20 values)	
400031		N/A	N	ModbusTCP message interval (time since last message)	
400032		N/A	N	Reserved	
400033		N/A	N		

400034		N/A	N		
400035		N/A	N		
400036		N/A	N		
400037		N/A	N		
400038		N/A	N		
400049		N/A	N		
400040		N/A	N		
400041	0	74	N	Size	Rack Status
400042		75	Y	CRC Errors	Rack TO's
400043		76	Y	Protcl Errors	Fail
400044		77	Y	BT Protcl Errors	BT Errors
400045		78	Y	BT Control word bits (bits 0-15) match to BT1 to BT16, 0=Enabled, 1=Disabled	
400046		17	N	Block Transfers	Rack Enable
400047		18	N	BTR Status bits	
400048		19	N	BTW Status bits	
400049		20	N	BTR Error bits	
400050		21	N	BTW Error bits	
400051	1	74	N	Size	Rack Status
400052		75	Y	CRC Errors	Rack TO's
400053		76	Y	Protcl Errors	Fail
400054		77	Y	BT Protcl Errors	BT Errors
400055		79	Y	BT Control word bits (bits 0-15) match to BT1 to BT16, 0=Enabled, 1=Disabled (Register 79 not accessible via RS485)	
400056		17	N	Block Transfers	Rack Enable
400057		18	N	BTR Status bits	
400058		19	N	BTW Status bits	
400059		20	N	BTR Error bits	
400050		21	N	BTW Error bits	
400061	2	Rack status repeats as above			
400071	3				
400081	4				
400091	5				
400101	6				
400111	7				
400121	8				
400131	9				
400141	10				
400151	11				
400161	12				
400171	13				
400181	14				
400191	15				
400201	16				
400211	17				
400221	18				
400231	19				
400241	20				
400251	21				
400261	22				
400271	23				
400281	24				
400291	25				
400301	26				
400311	27				
400321	28				
400331	29				
400341	30				
400351	31	74	N	Size	Rack Status
400352		75	Y	CRC Errors	Rack TO's
400353		76	Y	Protcl Errors	Fail

400354		77	Y	BT Protcl Errors	BT Errors
400355		79	Y	BT Control word bits (bits 0-15) match to BT1 to BT16, 0=Enabled, 1=Disabled (Register 79 not accessible via RS485)	
400356		17	N	Block Transfers	Rack Enable
400357		18	N	BTR Status bits	
400358		19	N	BTW Status bits	
400359		20	N	BTR Error bits	
400360		21	N	BTW Error bits	

Discrete IO access via AO writes

For discrete IO, the Modbus AO registers 400501-400820 may be used to write new output data to the discrete output tables in the PIO. This uses the write coils (FC = 0x10) may be used. The discrete values are written from unsigned 16-bit register datasets and the individual bits may be accessed with Boolean fans. This provides a faster data access than the writing coils (DO) command (FC = 0x01) as a single DeltaV dataset may access up to 9 racks of discrete IO data with a single write..

Rack to AI register mapping

All discrete output data in the PIO may be accessed via registers 400501 to 400820. Data for each rack is accessed consists of 8 words of IO data followed by two words of status data for the rack communications (see Rack discrete register group organization below). This status data is always updated in conjunction with the IO data and may be used for fail/recovery processing. The IO data is mapping to rack slots is dependent on the address formatting configured in the rack (see Slot discrete data mapping below).

AO Register	Rack Identification	
	Decimal	Octal
400501-400510	0	0
400511-400520	1	1
400521-400530	2	2
400531-400540	3	3
400541-400550	4	4
400551-400560	5	5
400561-400570	6	6
400571-400580	7	7
400581-400590	8	10
400591-400600	9	11
400601-400610	10	12
400611-400620	11	13
400621-400630	12	14
400631-400640	13	15
400641-400650	14	16
400651-400660	15	17
400661-400670	16	20
400671-400680	17	21
400681-400690	18	22
400691-400700	19	23
400701-400710	20	24
400711-400720	21	25
400721-400730	22	26
400731-400740	23	27
400741-400750	24	30
400751-400760	25	31
400761-400770	26	32
400771-400780	27	33
400781-400790	28	34
400791-400800	29	35
400801-400810	30	36
400811-400820	31	37

Dataset 2-5 Organization

The PIO table-set 2 to 5 for each rack are addressable using the format 4xx201 to 4xx500, where xx is the rack address (01 to 32) to access. This area holds the block transfer data for each rack.

Compressed Format			
Dataset	**Modbus Addresses**	**Word**	**Description**
2	4xx101-4xx125	1-25	BT 1
	4xx126-4xx150	26-50	BT 2
	4xx151-4xx175	51-75	BT 3
	4xx176-4xx200	76-100	BT 4
3	4xx201-4xx225	1-25	BT 5
	4xx226-4xx250	26-50	BT 6
	4xx251-4xx275	51-75	BT 7
	4xx276-4xx300	76-100	BT 8
4	4xx301-4xx325	1-25	BT 9
	4xx326-4xx350	26-50	BT 10
	4xx351-4xx375	51-75	BT 11
	4xx376-4xx400	76-100	BT 12
5	4xx401-4xx425	1-25	BT 13
	4xx426-4xx450	26-50	BT 14
	4xx451-4xx475	51-75	BT 15
	4xx476-4xx500	76-100	BT 16

**** where “xx” os replaced with 01 to 32 for racks 0-31, using base 1 for rack number.**

Compressed format allow minimal spacing of DS space. Each block transfer is sequentially configured; the actual location in the rack is not significant. Location of the BT is found in the BT definition table in DS1. Only sufficient datasets to hold the defined BT’s must be assigned to the port device (rack). Minimum would be 1 (no BT modules), allowing 10 racks per port.

Rack discrete register group organization

Discrete data for each rack is accessed consists of 8 words of IO data followed by two words of status data for the rack communications. This status data is always updated in conjunction with the IO data and may be used for fail/recovery processing.

Register Offset	Data Description		
0-7	Slot data dependent on slot addressing and organization.		
8	Rack Enabled word, this is generated from RS485 dataset 1 word 17, (read/only, but accepts writes without changing data), this word is set by PLC Explorer.		
	Byte	Description	
	Low	Rack group Configured if bit is set, this comes from PLC Explorer rack configuation	
		Bit	Description
		0	Group 1 Configured
		1	Group 2 Configured
		2	Group 3 Configured
		3	Group 4 Configured
		4	Group 5 Configured
		5	Group 6 Configured
		6	Group 7 Configured
		7	Group 8 Configured
	High	Rack group enabled if bit is set, this is enabled by PLC, disabled on comm. Loss	
		Bit	Description
		0	Group 1 Enabled
		1	Group 2 Enabled
		2	Group 3 Enabled
		3	Group 4 Enabled
		4	Group 5 Enabled
		5	Group 6 Enabled
		6	Group 7 Enabled
		7	Group 8 Enabled
9	Rack status word, this is generated from RS485 dataset 1 word 74. Read/only, but accepts writes without changing data, this word is set by the PIO dependent on the rack-enabled word (8 above).		
	Low	Group status (each 16 bit word), each bit represents a group, 1=run, 0=other.	
		Bit	Description
		0	Group 1 State (1= Ok)
		1	Group 2 State (1= Ok)
		2	Group 3 State (1= Ok)
		3	Group 4 State (1= Ok)
		4	Group 5 State (1= Ok)
		5	Group 6 State (1= Ok)
		6	Group 7 State (1= Ok)
		7	Group 8 State (1= Ok)
	High	Rack status	
		Bit	Description
		0	Quarter Rack 1 Configured
		1	Quarter Rack 1 Configured
		2	Quarter Rack 1 Configured
		3	Quarter Rack 1 Configured
		4	Quarter Rack 1 Enabled
		5	Quarter Rack 1 Enabled
		6	Quarter Rack 1 Enabled
		7	Quarter Rack 1 Enabled

Slot discrete data mapping

Slot Data mapping for data registers (offset 0-7) of each rack data register group. This is dependent on the addressing configured for the rack (see setting rack dip switches in the PIO manual, or the Allen Bradley ASB module documentation).

Addressing			AO Offsets	
	Slot	Group	Register	Bit
2-Slot	0	0	0	0-7
	0	1	0	8-15
	1	0	1	0-7
	1	1	1	8-15
	2	0	2	0-7
	2	1	2	8-15
	3	0	3	0-7
	3	1	3	8-15
	4	0	4	0-7
	4	1	4	8-15
	5	0	5	0-7
	5	1	5	8-15
	6	0	6	0-7
	6	1	6	8-15
	7	0	7	0-7
	7	1	7	8-15
1-Slot	0	N/A	0	0-15
	1		1	0-15
	2		2	0-15
	3		3	0-15
	4		4	0-15
	5		5	0-15
	6		6	0-15
	7		7	0-15
½ Slot	0	N/A	0	0-15
			1	0-15
	1		2	0-15
			3	0-15
	2		4	0-15
			5	0-15
	3		6	0-15
			7	0-15

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