
PresencePlus Vision Sensors

Ethernet/IP and Modbus/TCP Users Guide

Date: Sep. 15, 2008

Banner Engineering Corp.
9714 10th Av. N.
Minneapolis, MN 55441

- 1. Overview 2
 - 1.1 Terms and Definitions 2
- 2. PPVS Input and Output Data 3
 - 2.1 Input Data Values 3
 - 2.2 Output Data Values 3
 - 2.2.1 ACK Flags 4
 - 2.3 Sensor Operation 4
 - 2.3.1 Trigger Inspections 4
 - 2.3.2 Remote Teach 4
 - 2.3.3 Product Change 5
- 3. EIP on ControlLogix PLCs 6
 - 3.1 PPVS Assembly Objects 6
 - 3.1.1 Input Assembly Object 6
 - 3.1.2 Output Assembly Object 6
 - 3.1.3 Extended Output Assembly Objects 7
 - 3.1.4 Configuration Assembly Object - 0x80 (128 decimal) 8
- 4. EIP to PLC5 and SLC 5 9
- 5. Modbus/TCP 10
- 6. Data Format 14
 - 6.1 16 Bit Integer 14
 - 6.2 32 Bit Integer 14
 - 6.3 Floating Point 14
 - 6.4 Char String 15
- 7. Appendix - RSLogix5000 Configuration 16
 - 7.1 Tag Configurations 20
- 8. Appendix – String Input Regions 21
 - 8.1 Overview – String Input Regions 21
 - 8.1.1 Region Definitions 21
 - 8.1.2 Specifying Updated Regions 23
 - 8.1.3 Applying Region Updates 24
 - 8.2 Changing the String Input Regions 24
 - 8.3 String Regions on (EIP) ControlLogix PLCs 26
 - 8.4 String Regions on (EIP) PLC5 and SLC 5 28
 - 8.5 String Regions on Modbus/TCP 28
- 9. Appendix - Modbus/TCP Register Summary 29

1. Overview

This is the Users Guide for the Ethernet/IP (EIP) and Modbus/TCP features of the Banner PresencePlus Vision Sensor (PPVS).

Compatible devices supported are:

- Ethernet/IP connection (using the CIP protocol) to the Allen-Bradley ControlLogix family of PLCs. Both implicit and explicit messaging is supported.
- Ethernet/IP connection (using the PCCC protocol) to the Allen-Bradley SLC and PLC5 families of PLCs.
- Modbus/TCP connection to any compatible PLCs or device.

1.1 *Terms and Definitions*

PPVS	Banner PresencePlus Vision Sensor
Inspection	Stored inspection reference image plus tool instructions, stored in the PPVS flash.
Iteration	A cycle that starts with a trigger and ends with the sensor going back to the ready state. Consists of image acquisition, tool processing and output generation.
EIP	Ethernet/IP

2. PPVS Input and Output Data

The PPVS is controlled over Ethernet/IP and Modbus/TCP through the input and output data it makes available as a slave device for those protocols. In this document, the terms input and output are from the point of view of the PPVS. This differs from most protocol documentation, where these terms will be used from the PLC’s point of view.

Using input and output values, the following Sensor operation can be performed:

- Trigger Input
- Remote Teach
- Product Change
- Output indicators: pass/fail/ready/error
- Counters: pass, fail, system error, missed trigger, frame count, iteration count

On PPVS Barcode, the Test Tool’s Barcode compare string and mask can be changed.

In addition, inspection tool results can be outputted from the PPVS. In the PresencePlus PC software, CommTools configure the PPVS to output specific inspection tool results from any of the Analysis Tools.

2.1 Input Data Values

PPVS input data are values used as inputs by the PPVS, such as trigger, product select, etc. They are accessed using the EIP input assembly (PPVS_INPUT1) or Modbus/TCP and PCCC input registers. The following input values can be set from the PLC to control the PPVS:

<u>INPUT VALUE</u>	<u>USE</u>
Trigger	Transition from 0 to 1 causes system to trigger an inspection. Trigger divide and width requirements set in the PresencePlus software are ignored.
Remote Teach	Transition from 0 to 1 causes Remote Teach to occur on the next trigger.
Product Change	Transition from 0 to 1 causes a Product Change to occur.
Product Select	The inspection number to change to when a Product Change occurs.

2.2 Output Data Values

PPVS output data are values output by the PPVS to the PLC, such as pass/fail flags, ready, tool results, etc. They can be accessed using an EIP output assembly (PPVS_OUTPUT1-5) or Modbus/TCP and PCCC output registers. The following values can be retrieved from the PPVS:

<u>OUTPUT VALUE</u>	<u>USE</u>
Pass	If set, the last inspection iteration passed.
Fail	If set, the last inspection iteration failed.
Error	If set, a system error has occurred. This must be cleared using the PresencePlus software.
Ready	If set, PPVS system is ready for trigger.
I/O 1 to I/O 6	Displays the state of the I/O pins. Output delays and duration set in the PresencePlus software apply.
Inspection Number	Currently executing inspection number. Will be 0xFFFF if inspection is not running.
Error Count	Number of system errors that have occurred.
Frame Count	Number of images that have been captured. A unique ID used in system log entries.
Pass Count	Number of inspection iterations that have passed.
Fail Count	Number of inspection iterations that have failed.
Missed Triggers	Number of missed trigger incidents.
Iteration Count	Number of inspection iterations performed. Is the sum of pass/fail/missed trigger counters.

	When incremented, indicates that the triggered inspection iteration is complete.
Trigger ACK	Trigger acknowledge flag
Remote Teach ACK	Remote Teach acknowledge flag
Product Change ACK	Product Change acknowledge flag

2.2.1 ACK Flags

For each of the Input Flags (Trigger, Remote Teach, Product Change) there is a corresponding ACK flag in the Output Flags (Trigger, Remote Teach Ack, Product Change Ack). The PPVS echoes the input flag to the output ACK flag, providing a method for the PLC programmer to verify that the input flag transition was detected by the PPVS.

As an example, to use the Trigger ACK flag, the programming steps for triggering an inspection would be:

- Wait for Ready
- Set Trigger to 1
- Wait for Trigger ACK to go to 1
- Set Trigger to 0

Waiting for the Trigger ACK to go to 1 assures that the Trigger flag was received by the PPVS. This may be useful in certain system configurations, since there may be mismatched polling periods between the PLC program, the Ethernet/IP or Modbus/TCP communications and the PPVS.

2.3 Sensor Operation

Ethernet/IP and Modbus/TCP can be used to trigger inspections, remote teach and perform a product change. [The PLC programming instructions that follow assume that the PPVS has been configured and inspections created using the PresencePlus PC software.](#)

General notes:

- Input flags (Trigger, Remote Teach, Product Change) cause actions to occur on the low-to-high transition of that flag. De-assert the flag sometime after the corresponding Output ACK flag has been observed to be high.

2.3.1 Trigger Inspections

To trigger inspections, perform the following:

- Wait for the Ready flag to be high
- Assert the Trigger flag
- Wait for Trigger ACK flag to be asserted
- Wait for the Iteration Counter to be incremented. This signals that the inspection iteration is complete.
- If the Missed Trigger count has incremented, the inspection iteration did not occur.
- Check the Pass, Fail and Error flags for inspection results.

Optional steps are:

- Before triggering, verify the Inspection Number is as expected. If not, set using Product Change.
- The Ready flag can be used to determine if inspection iteration is complete, in place of the Iteration Count.

2.3.2 Remote Teach

Remote teach occurs when the Remote Teach flag is asserted, followed by the Trigger flag. To perform a remote teach, perform the following:

- Wait for the Ready flag to be high
- Assert the Remote Teach flag
- Wait for Remote Teach ACK flag to be asserted
- Assert the Trigger flag
- Wait for Trigger ACK flag to be asserted

- Wait for the Iteration Counter to be incremented. This signals that the inspection iteration is complete.
- If the Missed Trigger count has incremented, the remote teach did not occur.
- Check the Pass, Fail and Error flags for inspection results.

NOTE: Remote teach over EIP or Modbus/TCP is only enabled when Product Select is set to Hardware Input. In the PresencePlus PC software, see Run – Select – Product Select.

2.3.3 Product Change

Product change is used to change the active inspection number. To perform a product change, perform the following:

- Wait for the Ready flag
- Write the desired inspection number to the Product Select register
- Assert the Product Change flag
- Wait for Product Change ACK flag to be asserted
- Check the Inspection Number register. It will contain either the desired inspection number or 0xFFFF if a change to an invalid inspection number was performed.

Note:

- If the current inspection number is set to a number for which there is no stored inspection in the PPVS, a System Error will occur and the Inspection Number register will contain 0xFFFF. If this happens, change the inspection number to a valid value (and reassert the Product Change).

3. EIP on ControlLogix PLCs

The PPVS is controlled by a ControlLogix PLC using EIP through assembly objects. There is one input assembly (PPVS_INPUT1) and five output assemblies (PPVS_OUTPUT1 - PPVS_OUTPUT5).

For assistance with configuring the PPVS in RSLogix5000, see the Appendix.

3.1 PPVS Assembly Objects

Basic system operation of the PPVS using EIP by a ControlLogix PLC is done using the assembly objects PPVS_INPUT1 and PPVS_OUTPUT1.

3.1.1 Input Assembly Object

PPVS INPUT1 Instance 0x70 (112)

WORD #	WORD NAME
0	Input Flags
1	Product Select
2	<i>reserved</i>
3	<i>reserved</i>

Input Flags

Word Name	Bit Position															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input Flags	<i>reserved</i>	<i>reserved</i>	Product Change	Remote Teach	Trigger

NOTE: PPVS device that include a Barcode Tool use an expanded input instance (see Appendix).

3.1.2 Output Assembly Object

PPVS OUTPUT1 Instance 0x64 (100)

WORD #	WORD NAME
0	Output Flags
1	ACK Flags
2	Inspection Number
3	System Error Count
4 - 5	Frame Count
6 - 7	Pass Count
8 - 9	Fail Count
10 - 11	Missed Triggers
12 - 13	Iteration Count
14	<i>reserved</i>
15	<i>reserved</i>

Output Flags

Tag Name	Bit Position															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Flags	<i>reserved</i>	<i>reserved</i>	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Ready	<i>reserved</i>	Error	Fail	Pass
ACK Flags	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Product Change Ack	Remote Teach Ack	Trigger Ack

3.1.3 Extended Output Assembly Objects

The Extended Output Assembly Objects (PPVS_OUTPUT2 - PPVS_OUTPUT5) allow inspection tool results to be outputted over EIP. The outputting of this data is enabled using a Comm Tool in the PresencePlus software. Note that the first 16 words of PPVS_OUTPUT2 are the same as PPVS_OUTPUT1.

PPVS_OUTPUT2 Instance 0x65 (101)

WORD #	WORD NAME
0	Output Flags
1	ACK Flags
2	Inspection Number
3	System Error Count
4 - 5	Frame Count
6 - 7	Pass Count
8 - 9	Fail Count
10 - 11	Missed Triggers
12 - 13	Iteration Count
14	reserved
15	reserved
...	reserved
32	Location 1
33	
34	Location 2
35	
36	Location 3
37	
...	
236	Location 103
237	
238	Location 104
239	

$$WordNumber = 32 + ((Location - 1) * 2)$$

PPVS_OUTPUT3 Instance 0x66 (102)

WORD #	WORD NAME
0	Location 105
1	
2	Location 106
3	
4	Location 107
5	
...	
236	Location 223
237	
238	Location 224
239	

$$WordNumber = ((Location - 105) * 2)$$

PPVS_OUTPUT4 Instance 0x67 (103)

WORD #	WORD NAME
0	Location 225
1	
2	Location 226
3	
4	Location 227
5	
...	
236	Location 343
237	
238	Location 344
239	

$$WordNumber = ((Location - 225) * 2)$$

PPVS_OUTPUT5 Instance 0x68 (104)

WORD #	WORD NAME
0	Location 345
1	
2	Location 346
3	
4	Location 347
5	
...	
236	Location 463
237	
238	Location 464
239	

$$WordNumber = ((Location - 345) * 2)$$

3.1.4 Configuration Assembly Object - 0x80 (128 decimal)

The PPVS EIP implementation does not support an assembly object configuration instance. However, one is required for creation of implicit Class 1 connections on a ControlLogix family PLC. Therefore, a configuration instance is defined as instance number 0x80 (128 decimal). It's size is zero.

4. EIP to PLC5 and SLC 5

Allen-Bradley’s PLC5 and SLC 5 family of devices use PCCC communications over EIP. PPVS will support these PLCs using input and output register arrays. The Output Flags, ACK Flags and Input Flags bit definitions are the same as defined in the EIP Assembly Objects section. The terms “Input” and “Output” are from the point of view of the PPVS.

NOTE: The PPVS Barcode device uses an expanded set of N14 RW registers. See Appendix.

N7 RO REGS

N14 RW REGS

REG #	WORD NAME
0	Output Flags
1	ACK Flags
2	Inspection Number
3	System Error Count
4 - 5	Frame Count
6 - 7	Pass Count
8 - 9	Fail Count
10 - 11	Missed Triggers
12 - 13	Iteration Count
14	<i>reserved</i>
15	<i>reserved</i>
...	<i>reserved</i>
32	Location 1
33	
34	Location 2
35	
36	Location 3
37	
...	
956	Location 463
957	
958	Location 464
959	

REG #	WORD NAME
0	Input Flags
1	Product Select
2	<i>reserved</i>
3	<i>reserved</i>

$$registerNumber = 32 + ((Location - 1) * 2)$$

5. Modbus/TCP

The Modbus/TCP protocol provides device control using register and coil banks defined by the slave device. This section defines the PPVS Modbus/TCP register and coil banks. The Output Flags, ACK Flags and Input Flags bit definitions are the same as defined in the EIP Assembly Objects section. The terms “Input” and “Output” are from the point of view of the PPVS. By specification, Modbus/TCP uses TCP port 502.

Note that the output coils correspond to the Output and ACK Flags, and the input coils correspond to the Input Flags.

Note also that the PPVS Barcode device uses an expanded set of Modbus Input Registers. See Appendix for details.

A single page register summary is available in the Appendix.

Modbus Function Codes Supported

- 01: Read Coil Status
- 02: Read Input Status
- 03: Read Holding Registers
- 04: Read Input Registers
- 05: Force Single Coil
- 06: Preset Single Register
- 07: Read Exception Status
- 15: Write Multiple Coils
- 16: Preset Multiple Registers

PPVS Output Registers

The Modbus PPVS Output Registers are used to push output values from the PPVS to the PLC. These values include the Output and ACK Flags, Inspection Number, etc. In addition, PPVS tool results, configured using the CommTool, are outputted using this range of registers.

Note that some devices (such as Modicon family PLCs) cannot access data using the 30000 range of register addresses. For these devices, the PPVS output values are also available using the 40000 range of addresses (at offset 41000). See the next section. To access the PPVS Modbus/TCP Output Registers use Function Code 04 (Read Input Registers).

Output Registers (30001-30960)

REG #	WORD NAME
1	Output Flags
2	ACK Flags
3	Inspection Number
4	System Error Count
5 - 6	Frame Count
7 - 8	Pass Count
9 - 10	Fail Count
11 - 12	Missed Triggers
13 - 14	Iteration Count
15	reserved
16	reserved
...	reserved
33	Location 1
34	
35	Location 2
36	
37	Location 3
38	
...	
957	Location 463
958	
959	Location 464
960	

$$registerNumber = 33 + ((Location - 1) * 2)$$

PPVS Output Coils

The Modbus PPVS Output Coils are used to push single bit outputs from the PPVS to the PLC. The 32 bits of Output Coils can also be accessed using the bits of the first two Output Registers (Output Flags and ACK Flags).

To access the PPVS Modbus/TCP Output Coils use Function Code 02 (Read Input Status).

Output Coils (10001-10032)

	Bit Position															
Output Flags	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Coil #	Coil 16	Coil 15	Coil 14	Coil 13	Coil 12	Coil 11	Coil 10	Coil 9	Coil 8	Coil 7	Coil 6	Coil 5	Coil 4	Coil 3	Coil 2	Coil 1
Name	reserved	reserved	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	reserved	reserved	reserved	Ready	reserved	Error	Fail	Pass

	Bit Position															
ACK Flags	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Coil #	Coil 32	Coil 31	Coil 30	Coil 29	Coil 28	Coil 27	Coil 26	Coil 25	Coil 24	Coil 23	Coil 22	Coil 21	Coil 20	Coil 19	Coil 18	Coil 17
Name	reserved	reserved	Product Change Ack	Remote Teach Ack	Trigger Ack

PPVS Input Registers

The Modbus PPVS Input Registers are used by the PLC to push values to the PPVS. These values, such as the Input Flags and Product Select, are accessed in the register address range of 40001 - 41000. To write the PPVS Input Registers, use Function Codes 6/16 (Preset Single/Multiple Registers).

Also available in this range of registers are the PPVS Outputs (Output Flags, ACK Flags, tool results, etc). These outputs are available using the address range of 41001 - 42000. To read these output into the PLC use Function Code 03 (Read Holding Registers).

Note that the PLC Map generated by the CommTool shows the Modbus register location assigned to a particular CommTool tool result. When accessing these outputs using the 40000 range (Function Code 03), add an offset of 1000 to the register value. For example, if a tool result is listed in the PLC Map as residing at register 33, it can be accessed using Function Code 03 at register address 41033.

Modbus Registers (40001-42000)	
REG #	WORD NAME
1	Input Flags
2	Product Select
3 - 39	<i>reserved</i>
40 - 239	String Regions
240 - 1000	<i>reserved</i>
1001	Output Flags
1002	ACK Flags
1003	Inspection Number
1004	System Error Count
1005 - 1006	Frame Count
1007 - 1008	Pass Count
1009 - 1010	Fail Count
1011 - 1012	Missed Triggers
1013 - 1014	Iteration Count
1015 - 1032	<i>reserved</i>
1033	Location 1
1034	
1035	Location 2
1036	
1037	Location 3
1038	
...	
1957	Location 463
1958	
1959	Location 464
1960	
1961 - 2000	<i>reserved</i>

Also available as Coils 1 - 16

PLC -> Device

Also available as Coils 10001 - 10032

Device -> PLC

This range also available as registers 30001 - 31000

PPVS Input Coils

The Modbus PPVS Input Coils are used to send single bit inputs from the PLC to the PPVS. The 16 bits of Input Coils can also be accessed using the 16 bits of the Input Flags register. To write the PPVS Input Coils, use Function Codes 5/15 (Force Single/Multiple Coils).

Input Coils (00001-00015)

	Bit Position												Banner Engineering Corp.			
Input Flags	15	14	13	12	11	10	9	8	7	6	5	4	Coil 3	Coil 2	Coil 1	
Input Coil #																
Name	<i>reserved</i>	<i>reserved</i>	Product Change	Remote Teach	Trigger

6. Data Format

The PPVS stores data as one of 4 data types:

- 16 bit integer
- 32 bit integer
- Floating point
- Char string

6.1 16 Bit Integer

The PPVS stores data as a 16 bit (unsigned) integer for the following fixed-location values only:

- Output Flags
- ACK Flags
- Inspection Number
- System Error Count
- Input Flags
- Product Select

6.2 32 Bit Integer

The PPVS stores data as 32 bit (unsigned) integers for the following fixed-location values:

- Frame Count
- Pass Count
- Fail Count
- Missed Triggers
- Iteration Count

In addition, some tool results, such as counts, are stored as integers.

This integer is stored in two adjacent 16 bit word/registers. The data can be formatted in two ways, depending on the PLC 32 Bit Format setting (set in the Presence Plus PC software - System tab, Communications, Communication Tool, Industrial Ethernet).

MSW LSW

The most significant word is stored first, then the least significant word

LSW MSW

The least significant word is stored first, then the most significant word. Used for Allen-Bradley ControlLogix PLCs.

6.3 Floating Point

Floating point values in the PPVS are IEEE-754 (32 bit) single precision floating point numbers. They are used for some tool results, such as locations or distances.

This floating point value is stored in two adjacent 16 bit word/registers. Again, the data can be formatted in two ways, depending on the PLC 32 Bit Format setting.

MSW LSW

The most significant word is stored first, then the least significant word

- Bits 15-0 of first register = Bits 31-16 of the number
- Bits 15-0 of second register = Bits 0-15 of the number

LSW MSW

The least significant word is stored first, then the most significant word.

- Bits 15-0 of first register = Bits 15-0 of the number

- Bits 15-0 of second register = Bits 31-16 of the number

6.4 Char String

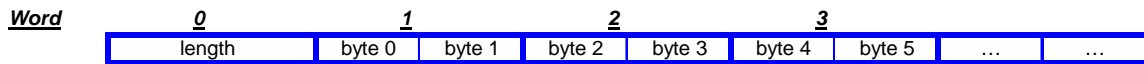
Char strings consist of a length followed by a variable number of (8 bit) character bytes. The NULL byte is not included in the length. The entire string storage area is zero filled before the string is written to it. Therefore, if there is sufficient storage allocated, the string will be NULL terminated.

NOTES:

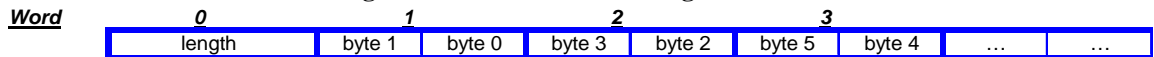
- In the Comm Tool, the user selects the number of strings to store and the maximum size of each string (in characters).
- If insufficient storage is allocated, strings are truncated.

There are two string format options in the PPVS, depending on the PLC Protocol chosen (in the Presence Plus GUI Communications tab):

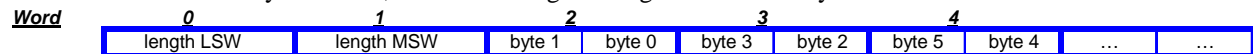
- When Modbus/TCP or PCCC is chosen, characters in the string can be packed into registers according to either the Standard String format or the ControlLogix Format¹:
 - Standard String Format is used when the Character String Order option in the Presence Plus GUI Communications Tab is set to “High Byte – Low Byte” selection. **Please note that the length of the string is stored in one 16-bit register:**



- ControlLogix packing format is used to store characters into registers when the Character String Order option in the Presence Plus GUI Communications Tab is set to “Low Byte – High Byte” selection. **Please note that in this particular case, the length of the string is still stored in one 16-bit register, and only the order of characters in each 16-bit register follows the ControlLogix format:**

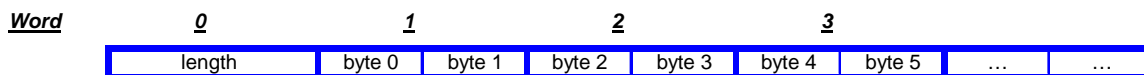


- When Allen-Bradley is chosen, the ControlLogix String Format is always used:



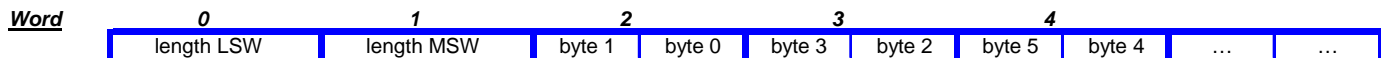
Standard String Format

In Standard String Format, the first (16 bit) word/register contains the string length. This is followed by the character string stored as 2 bytes per word/register.



ControlLogix String Format

This string format is compatible with the Allen-Bradley ControlLogix built-in string data type. This format is a 32 bit (DINT) length followed by character bytes (SINT). This results in the following string format as viewed from the PPVS:

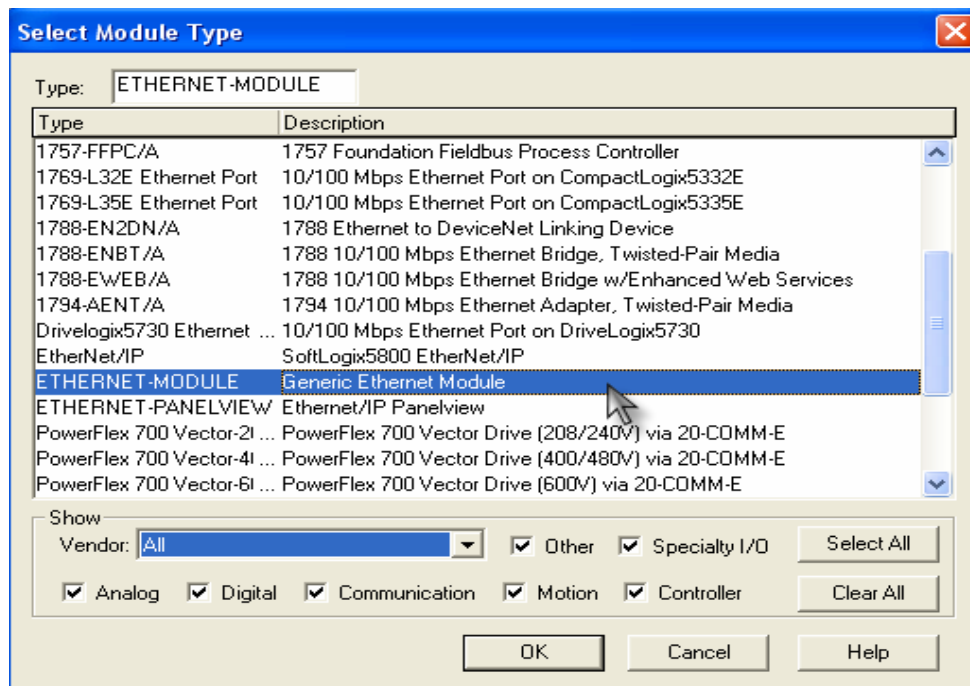
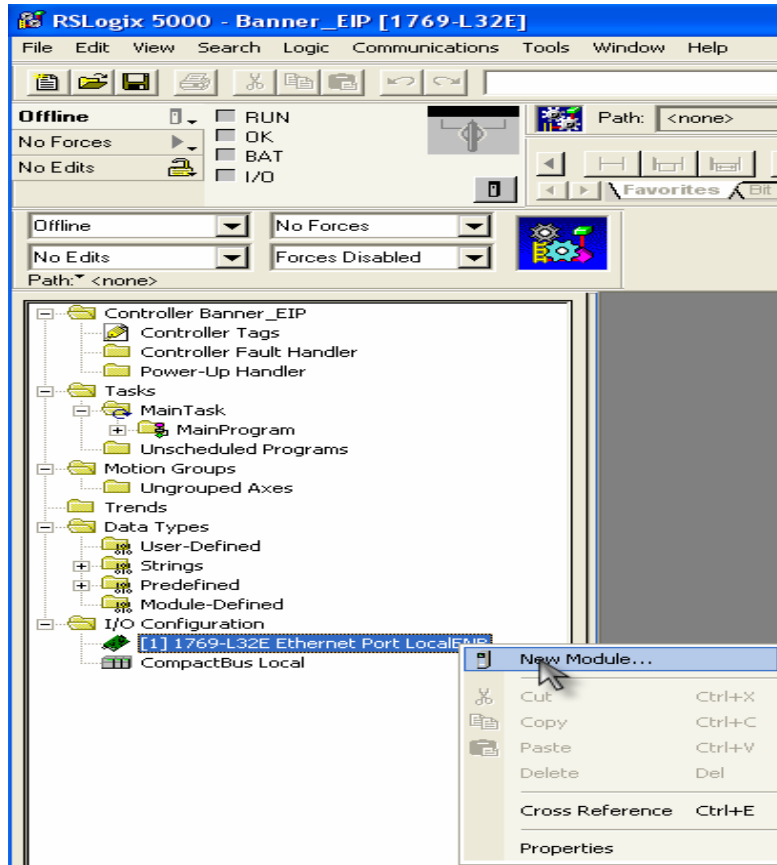


¹ The characters are packed into 16-bit Modbus/TCP registers according to ControlLogix specifications, but the length is still stored in a single 16-bit register.

7. Appendix - RSLogix5000 Configuration

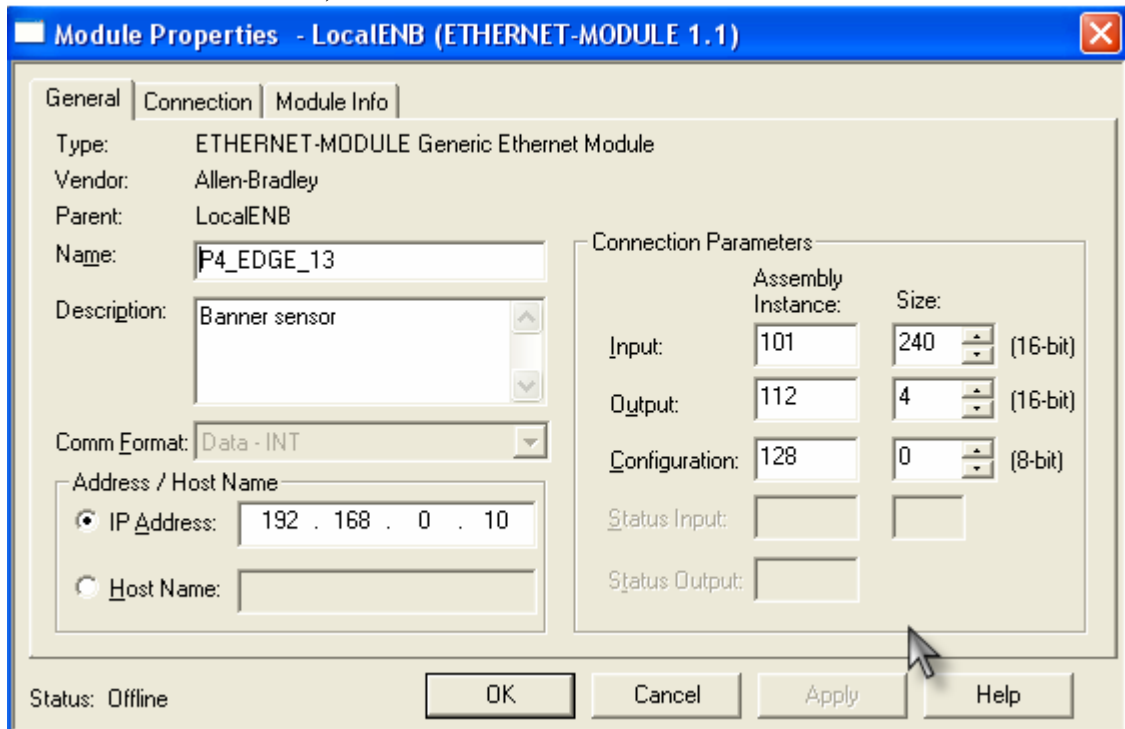
To create an implicit Class 1 configuration to the PPVS using EIP when using a ControlLogix family PLC, configure the PPVS as a “Generic Ethernet Module” under the ENET_MODULE. The following is a sample setup of Banner sensor:

1- Adding Banner sensor



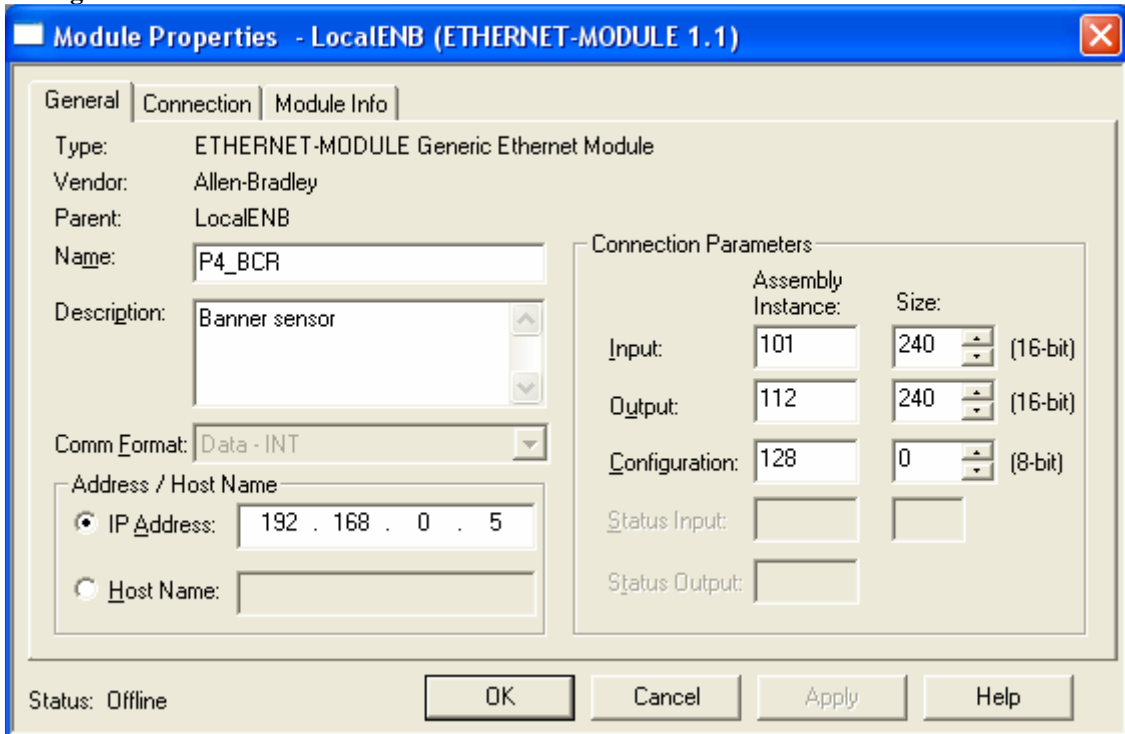
2- Configure Banner module property.

A- Configuration for all EIP sensors (except P4 BCR models, and sensors that do not have licensed Barcode or OCR/OCV tools)

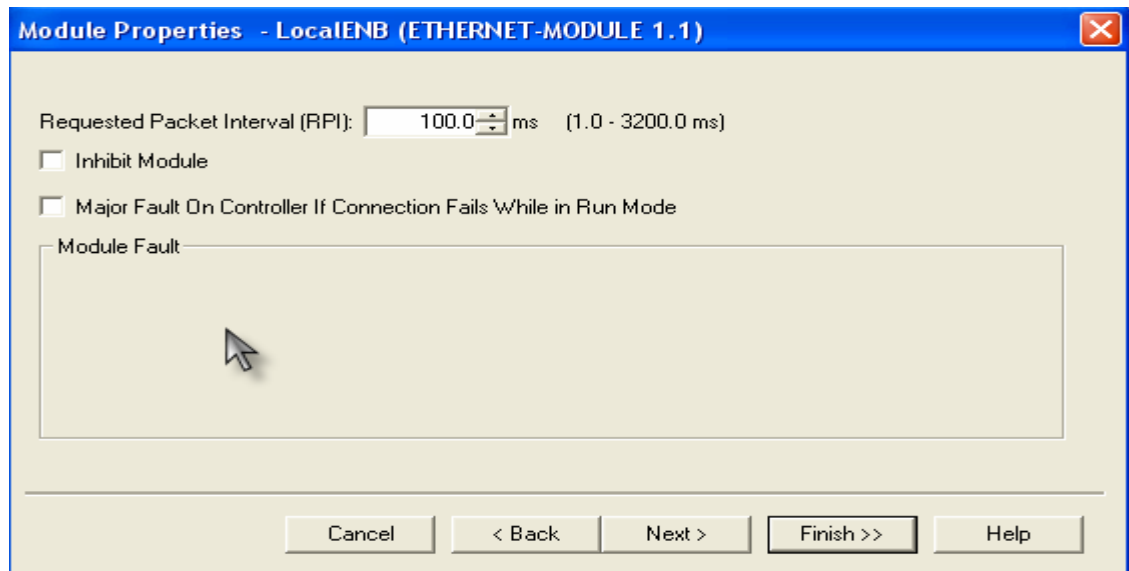


Note: The data type in the Comm Format must be changed to an INT.

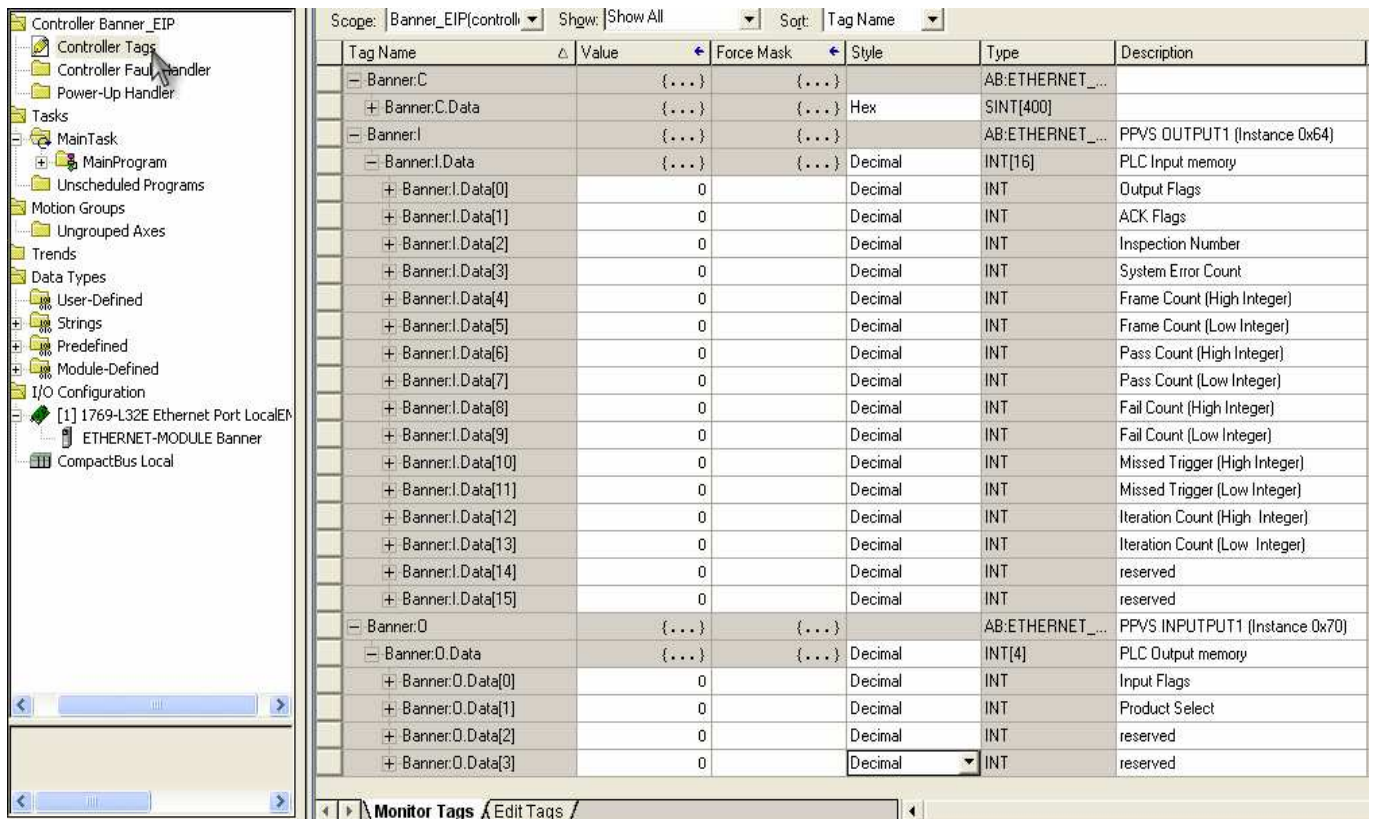
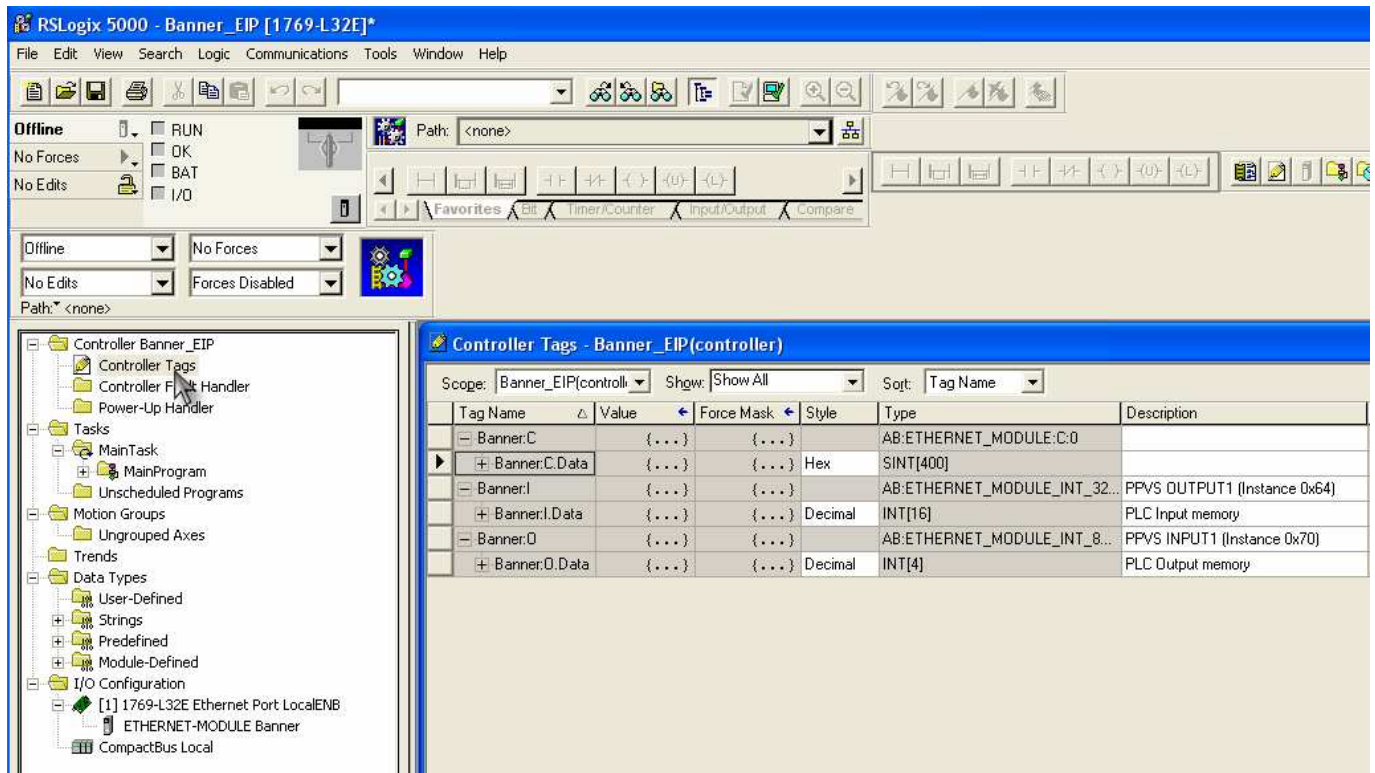
B- Configuration for P4 BCR sensors and sensors that have licensed Barcode and/or OCR/OCV tools:



Note: The data type in the Comm Format must be set to INT.



3- Locate memory map setup from Banner module to PLC memory map.

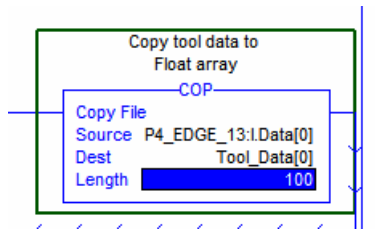


7.1 Tag Configurations

Floating Point Numbers

To read a floating point number into a Control Logix PLC, the number must be copied from the Banner INT register to an internal Real register.

Use the COP command transfer the floating point number to a real data tag.



8. Appendix – String Input Regions

PPVS devices that include a Barcode or an OCR/OCV Tool, and that, as a result, have enabled String Tool, use different input instances/registers than the other PPVS devices. This new input register configuration enables changing the String Tool's operands and OCV Expected String (when set to Industrial Ethernet) over EIP/Modbus/TCP.

8.1 Overview – String Input Regions

EIP/Modbus/TCP Updatable String Regions were developed as a result of the introduction of new string processing tools, such as String, OCR and OCV. The goal is to improve and simplify string data validation, and to allow remote string updates for multiple individual tools. String Regions allow up to 6 different tools to be updated with different strings in a single transaction.

Prior to the introduction of String Regions, remote string updates were possible using Test Tool/Barcode masked string comparison feature. This method of Barcode string comparison is still functional in the current version of PPVS software, but will be deprecated in the future releases, so its use is no longer recommended. In addition, the two string remote update methods, Test Tool/Barcode and String Regions, cannot be used together in the same inspection. If an attempt is made to do so, String Regions will be used, disabling all updates to the Test Tool/Barcode.

String Regions occupy input registers 40 through 240 in Modbus/TCP input register map, or registers 39 through 239 in the first EIP PPVS Input Instance, 0x70.

8.1.1 Region Definitions

There are a total of 6 string regions. Regions one and two are each 50 registers long, allowing strings up to 98 characters long. Regions three through six are each 25 registers long, allowing strings up to 48 characters long. Maximum length of strings stored in each of the regions depends on the tool associated with that region, and the specific operation chosen for that tool. The six regions are described below:

- Region 1
 - 50 registers long
 - Starting register 40 for Modbus/TCP, 39 for EIP
 - For OCV tool and String Tool with Compare or Find Substring Operation:
 - Holds strings up to 98 characters long
 - For String Tool with Masked String Comparison operation
 - String and its corresponding mask are each 25 registers long
 - Holds strings up to 48 characters long
 - Holds string masks up to 48 characters long
 - Starting register for the mask is 65 for Modbus/TCP, 64 for EIP
- Region 2
 - 50 registers long
 - Starting register 90 for Modbus/TCP, 89 for EIP
 - For OCV tool and String Tool with Compare or Find Substring Operation:
 - Holds strings up to 98 characters long
 - For String Tool with Masked String Comparison operation
 - String and its corresponding mask are each 25 registers long
 - Holds strings up to 48 characters long
 - Holds string masks up to 48 characters long
 - Starting register for the mask is 115 for Modbus/TCP, 114 for EIP
- Region 3
 - 25 registers long
 - Starting register 140 for Modbus/TCP, 139 for EIP
 - For OCV tool and String Tool with Compare or Find Substring Operation:
 - Holds strings up to 48 characters long
 - For String Tool with Masked String Comparison operation
 - String and its corresponding mask are each 12 registers long
 - Holds strings up to 22 characters long
 - Holds string masks up to 22 characters long

- Starting register for the mask is 152 for Modbus/TCP, 151 for EIP
- Region 4
 - 25 registers long
 - Starting register 165 for Modbus/TCP, 164 for EIP
 - For OCV tool and String Tool with Compare or Find Substring Operation:
 - Holds strings up to 48 characters long
 - For String Tool with Masked String Comparison operation
 - String and its corresponding mask are each 12 registers long
 - Holds strings up to 22 characters long
 - Holds string masks up to 22 characters long
 - Starting register for the mask is 177 for Modbus/TCP, 176 for EIP
- Region 5
 - 25 registers long
 - Starting register 190 for Modbus/TCP, 189 for EIP
 - For OCV tool and String Tool with Compare or Find Substring Operation:
 - Holds strings up to 48 characters long
 - For String Tool with Masked String Comparison operation
 - String and its corresponding mask are each 12 registers long
 - Holds strings up to 22 characters long
 - Holds string masks up to 22 characters long
 - Starting register for the mask is 202 for Modbus/TCP, 201 for EIP
- Region 6
 - 25 registers long
 - Starting register 215 for Modbus/TCP, 214 for EIP
 - For OCV tool and String Tool with Compare or Find Substring Operation:
 - Holds strings up to 48 characters long
 - For String Tool with Masked String Comparison operation
 - String and its corresponding mask are each 12 registers long
 - Holds strings up to 22 characters long
 - Holds string masks up to 22 characters long
 - Starting register for the mask is 227 for Modbus/TCP, 226 for EIP

The Masked String Comparison is a new operation implemented in the String Tool. It is identical to the Test Tool/Barcode masked string comparison feature. Because it requires both the string and the string mask, each region is divided into two sub-regions, one for the string, and the other one for the mask. The subdivision occurs on the register (word), rather than the byte boundary, therefore creating the appearance of a “lost” byte.

A table representation of string regions is given below:

EIP PPVS INPUT1 Instance 0x70 (112)

WORD #	WORD NAME
0	Input Flags
1	Product Select
2	Tool String Update Flags
3	<i>reserved</i>
...	<i>reserved</i>
39	String Region 1 (50 or 25 words)
...	...
64	(String Region 1 Mask, 25 words)
...	...
89	String Region 2 (50 or 25 words)
...	...
114	(String Region 2 Mask, 25 words)
...	...
139	String Region 3 (25 or 12 words)
...	...
151	(String Region 3 Mask, 12 words)
...	...
164	String Region 4 (25 or 12 words)
...	...
176	(String Region 4 Mask, 12 words)
...	...
189	String Region 5 (25 or 12 words)
...	...
201	(String Region 5 Mask, 12 words)
...	...
214	String Region 6 (25 or 12 words)
...	...
226	(String Region 6 Mask, 12 words)
239	

MODBUS Input Registers (40001-40240)

REG#	WORD NAME
1	Input Flags
2	Product Select
3	Tool String Update Flags
4	<i>reserved</i>
...	<i>reserved</i>
40	String Region 1 (50 or 25 words)
...	...
65	(String Region 1 Mask)
...	...
90	String Region 2 (50 or 25 words)
...	...
115	(String Region 2 Mask, 25 words)
...	...
140	String Region 3 (25 or 12 words)
...	...
152	(String Region 3 Mask, 12 words)
...	...
165	String Region 4 (25 or 12 words)
...	...
177	(String Region 4 Mask, 12 words)
...	...
190	String Region 5 (25 or 12 words)
...	...
202	(String Region 5 Mask, 12 words)
...	...
215	String Region 6 (25 or 12 words)
...	...
227	(String Region 6 Mask, 12 words)
240	...

8.1.2 Specifying Updated Regions

To allow individual region updates, a new input register, Tool String Update Flags, located at address 3 in Modbus/TCP and 2 in EIP, has been defined. The coils in this register correspond to the specific string regions or their masks. Setting the individual region or mask coil prior to applying string region updates (see next section) allows the user to update only that specific region (or its mask). Multiple mask and/or coil flags can be set in a single update. It is up to the user to clear these flags once the data copy acknowledgement from the sensor is received.

The table describing the new register is given below:

Tool String Update Flags

Tag Name	Bit Position															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
String Update Flags	Region 6 Mask	Region 5 Mask	Region 4 Mask	Region 3 Mask	Region 2 Mask	Region 1 Mask	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Region 6	Region 5	Region 4	Region 3	Region 2	Region 1

8.1.3 Applying Region Updates

To update String Regions, a new flag (coil) in the Input Flags register has been defined. This new flag, Tool String Flag, is mapped onto Coil 10 (bit offset 9) of the Input Flags register.

When this flag is set, all the modified string regions, as indicated by the Tool String Update Flags register, will be copied to the sensor's internal buffer. After the copy operation is complete, the sensor will set the Tool String Change ACK flag in the Output Flags register (see further below).

The updated string regions will be retained in the internal buffer. They will be applied to their respective tools only when the next trigger is received. Please note that by setting and clearing this flag multiple times between triggers, the data previously copied into the sensor's internal buffer may be overwritten, without ever being copied into the inspection.

It is up to the user to reset this flag after the data copy acknowledgement has been received from the sensor (please see the next paragraph).

The table describing the updated Input Flags register is given below:

Input Coils (00000-00015)

	Bit Position																
Input Reg 1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Input Coil #								Coil 9						Coil 3	Coil 2	Coil 1	
Name	reserved	reserved	reserved	reserved	reserved	reserved	reserved	Tool String Change	BCR String Change	reserved	reserved	reserved	reserved	reserved	Product Change	Remote Teach	Trigger

A corresponding Tool String Acknowledgement flag (coil) has also been added to the Output Flags register. This flag is set when the PPVS sensor finishes copying the new strings to the internal buffer. This flag is cleared after the Tool String Change flag in the Input Flags register is cleared by the user.

Please note that this flag does not indicate that tools have been updated with the new strings, it only indicates that the data was copied into the sensor's internal buffer. Actual tool updates occur only when the next trigger is received.

The table describing the updated Output Flags register is given below:

Output Coils (10000-10032)

	Bit Position															
Output Reg 1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Coil #	Coil 16	Coil 15	Coil 14	Coil 13	Coil 12	Coil 11	Coil 10	Coil 9	Coil 8	Coil 7	Coil 6	Coil 5	Coil 4	Coil 3	Coil 2	Coil 1
Name	reserved	reserved	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	reserved	reserved	reserved	Ready	reserved	Error	Fail	Pass

	Bit Position																
Output Reg 2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Output Coil #	Coil 32	Coil 31	Coil 30	Coil 29	Coil 28	Coil 27	Coil 26	Coil 25	Coil 24	Coil 23	Coil 22	Coil 21	Coil 20	Coil 19	Coil 18	Coil 17	
Name	reserved	reserved	reserved	reserved	reserved	reserved	reserved	Tool String Change ACK	BCR String Change Ack	reserved	reserved	reserved	reserved	reserved	Product Change Ack	Remote Teach Ack	Trigger Ack

8.2 Changing the String Input Regions

Operands in the String Tool and expected string in the OCV tool can be modified remotely using EIP/Modbus/TCP. Prior to attempting to do this, the desired String Tool operand and/or the OCV expected string must be associated with an Industrial Protocol string region. To create this association, select "Industrial Ethernet" as the expected string source in OCV tool drop-down dialog; or select "Industrial Ethernet" as the input tool for one of the String Tool operands. After this step is complete, a new drop-down dialog listing all the available string regions will be displayed. A suitable string region should then be selected from that drop-down dialog. After this operation is complete, the selected string region will be associated with the tool². The resulting inspection should then be stored and enabled in the sensor.

To use EIP/Modbus/TCP to change the desired string region, the following steps can be followed:

² Please note that it is possible to associate a region with more than one tool.

- Write a new string in the EIP/Modbus/TCP input instance/register space to the area specified for the region to be updated, according to the tables given in this document.
 - More than one region and more than one mask can be updated at the same time.
- Optionally, write a new mask in the instance/register space. The mask is valid only if a String Tool with Masked String Comparison operation has been created and associated with a string region.
- Set the flags in the Tool String Update Flags register that correspond to the updated regions and masks.
- Toggle the Tool String Change flag in the Input Flags register.

Toggling the Tool String Change flag causes the PPVS to copy the new strings and masks into the temporary internal buffer. To verify that the change flag was received in the PPVS, the Tool String Change ACK flag should be used.

When the next trigger is received, the strings and masks (if present) are validated, and copied into the tools that are associated with the updated regions.

To be validated, the new data stored in the EIP/Modbus/TCP input instance register space must meet the following requirements:

- Both the string and the mask (if present) must be stored in the format documented above - length followed by the string. Length is the number of characters (not including any null termination characters).
- The string length can be different than the string length of the original inspection.

The mask is optional, and is examined only if the region is associated with a String Tool with Masked String Comparison operation. If the mask length is non-zero, and the mask length equals the new compare string length, it will be used. The mask is a character string of the same length as the compare string. The following characters are valid in the mask:

- '0' (0x30) - Don't care
- '1' (0x31) - Check this character

The mask is a string of '0' and '1' characters, indicating which character positions in the compare string to test against. Any characters other than '0' and '1' in the mask will cause it to be flagged as invalid.

An example mask (using the Standard String Format) would be: [00 05 | 0x31 0x31 | 0x31 0x31 | 0x30 00]. In this example, the mask (and the compare string) are 5 characters long. The first 4 characters will be tested, the fifth character is a 'don't care'.

8.3 String Regions on (EIP) ControlLogix PLCs

The EIP Input Assembly object on the device that supports String Regions is extended to support the compare string change feature.

PPVS INPUT1 Instance 0x70 (112)

WORD #	WORD NAME
0	Input Flags
1	Product Select
2	Tool String Update Flags
3	<i>reserved</i>
...	<i>reserved</i>
39	String Region 1 (50 or 25 words)
...	...
64	(String Region 1 Mask, 25 words)
...	...
89	String Region 2 (50 or 25 words)
...	...
114	(String Region 2 Mask, 25 words)
...	...
139	String Region 3 (25 or 12 words)
...	...
151	(String Region 3 Mask, 12 words)
...	...
164	String Region 4 (25 or 12 words)
...	...
176	(String Region 4 Mask, 12 words)
...	...
189	String Region 5 (25 or 12 words)
...	...
201	(String Region 5 Mask, 12 words)
...	...
214	String Region 6 (25 or 12 words)
...	...
226	(String Region 6 Mask, 12 words)
239	

Word Name	Bit Position															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input Flags	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Tool String Change	BCR String Change	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Product Change	Remote Teach	Trigger

Tool String Update Flags

Tag Name	Bit Position															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
String Update Flags	Region 6 Mask	Region 5 Mask	Region 4 Mask	Region 3 Mask	Region 2 Mask	Region 1 Mask	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Region 6	Region 5	Region 4	Region 3	Region 2	Region 1

The EIP Output Assembly objects (Instance 0x64 and 0x65) are extended to provide an ACK bit for the BCR String Change input flag.

Output Flags

Tag Name	Bit Position															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Flags	<i>reserved</i>	<i>reserved</i>	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Ready	<i>reserved</i>	Error	Fail	Pass
ACK Flags	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Tool String Change ACK	BCR String Change Ack	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Product Change Ack	Remote Teach Ack	Trigger Ack

The Module Properties entries changes with this larger Input Assembly. The following is an example Connection Parameters for RSLogix:

	Assembly Instance	Size
Input	100	16
Output	112	240
Configuration	128	0

8.4 String Regions on EIP to PLC5 and SLC 5

The EIP N14 RW Register space, used by PLC5 and SLC5 PLCs, is extended to support string regions.

N14 RW REGS

REG #	WORD NAME
0	Input Flags
1	Product Select
2	Tool String Update Flags
3	<i>reserved</i>
...	<i>reserved</i>
39 - 238	Tool String Regions

8.5 String Regions on Modbus/TCP

The Modbus/TCP Input Registers and Coils are extended to support string regions.

Input Registers (40001-40240)

REG#	WORD NAME
1	Input Flags
2	Product Select
3	Tool String Update Flags
4	<i>reserved</i>
...	<i>reserved</i>
...	<i>reserved</i>
40 - 239	Tool String Regions

Input Coils (00000-00015)

	Bit Position															
Input Reg 1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input Coil #							Coil 10	Coil 9						Coil 3	Coil 2	Coil 1
Name	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Tool String Change	BCR String Change	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Product Change	Remote Teach	Trigger

Tool String Update Flags

Tag Name	Bit Position															
String Update Flags	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Region 6 Mask	Region 5 Mask	Region 4 Mask	Region 3 Mask	Region 2 Mask	Region 1 Mask	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Region 6	Region 5	Region 4	Region 3	Region 2	Region 1

The Modbus/TCP Output Coils are extended to provide an ACK coil for the BCR String Change input coil.

Output Coils (10000-10032)

	Bit Position															
Output Reg 1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Coil #	Coil 16	Coil 15	Coil 14	Coil 13	Coil 12	Coil 11	Coil 10	Coil 9	Coil 8	Coil 7	Coil 6	Coil 5	Coil 4	Coil 3	Coil 2	Coil 1
Name	<i>reserved</i>	<i>reserved</i>	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Ready	<i>reserved</i>	Error	Fail	Pass

	Bit Position															
Output Reg 2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Coil #	Coil 32	Coil 31	Coil 30	Coil 29	Coil 28	Coil 27	Coil 26	Coil 25	Coil 24	Coil 23	Coil 22	Coil 21	Coil 20	Coil 19	Coil 18	Coil 17
Name	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Tool String Change Ack	BCR String Change Ack	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	Product Change Ack	Remote Teach Ack	Trigger Ack

9. Appendix - Modbus/TCP Register Summary

Modbus Register Summary

Coils

Input Flag (Input coils 00001-00016)

Register	Bit Position	WORD NAME
00001	0	Trigger
00002	1	Remote Teach
00003	2	Product Change
00004	3	reserved
00005	4	reserved
00006	5	reserved
00007	6	reserved
00008	7	reserved
00009	8	BCR String Change
00010	9	Tool String Change
00011	10	reserved
00012	11	reserved
00013	12	reserved
00014	13	reserved
00015	14	reserved
00016	15	reserved

Output Flags (Output Coil 10001-10016)

Register	Bit Position	WORD NAME
10001	0	Pass
10002	1	Fail
10003	2	Error
10004	3	reserved
10005	4	Ready
10006	5	reserved
10007	6	reserved
10008	7	reserved
10009	8	I/O 1
10010	9	I/O 2
10011	10	I/O 3
10012	11	I/O 4
10013	12	I/O 5
10014	13	I/O 6
10015	14	reserved
10016	15	reserved

ACK Flag (Output coils 10017-10032)

Register	Bit Position	WORD NAME
10017	0	Trigger Ack
10018	1	Remote Teach Ack
10019	2	Product Change Ack
10020	3	reserved
10021	4	reserved
10022	5	reserved
10023	6	reserved
10024	7	reserved
10025	8	BCR Str Change ACK
10026	9	Tool Str Change ACK
10027	10	reserved
10028	11	reserved
10029	12	reserved
10030	13	reserved
10031	14	reserved
10032	15	reserved

Words

Output Registers 30001-30960

Register	WORD NAME
30001	Output Flags (coils 10001-16)
30002	ACK Flags (coils 10017-32)
30003	Inspection Number
30004	System Error Count
30005 - 6	Frame Count
30007 - 8	Pass Count
30009 - 10	Fail Count
30011 - 12	Missed Triggers
30013 - 14	Iteration Count
30015 - 32	reserved
30033	Location 1
30034	
30035	Location 2
30036	
30037	Location 3
30038	
...	
30957	Location 463
30958	
30959	Location 464
30960	

Input Registers 40001-41000

Register	WORD NAME
40001	Input Flags (Coils 00001-16)
40002	Product Select
40003	Tool String Update Flags
40003 - 39	reserved
40040 - 239	Tool String Regions
40240 - 1000	reserved

Output Registers 41001-42000 (same as 30000 Reg)

41001	Output Flags
41002	ACK Flags
41003	Inspection Number
41004	System Error Count
41005 - 1006	Frame Count
41007 - 1008	Pass Count
41009 - 1010	Fail Count
41011 - 1012	Missed Triggers
41013 - 1014	Iteration Count
41015 - 1032	reserved
41033	Location 1
41034	
41035	Location 2
41036	
41037	Location 3
41038	
...	
41957	Location 463
41958	
41959	Location 464
41960	
41961 - 2000	reserved

Output, ACK, input and Tool String Region Update flags

Output Coils (10001-10032)

	Bit Position															
Output Flags	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Coil #	Coil 16	Coil 15	Coil 14	Coil 13	Coil 12	Coil 11	Coil 10	Coil 9	Coil 8	Coil 7	Coil 6	Coil 5	Coil 4	Coil 3	Coil 2	Coil 1
Name	reserved	reserved	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	reserved	reserved	reserved	Ready	reserved	Error	Fail	Pass

	Bit Position																
ACK Flags	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Output Coil #	Coil 32	Coil 31	Coil 30	Coil 29	Coil 28	Coil 27	Coil 26	Coil 25	Coil 24	Coil 23	Coil 22	Coil 21	Coil 20	Coil 19	Coil 18	Coil 17	
Name	reserved	reserved	reserved	reserved	reserved	reserved	reserved	Tool String Change Ack	BCR String Change Ack	reserved	reserved	reserved	reserved	reserved	Product Change Ack	Remote Teach Ack	Trigger Ack

Input Coils (00000-00015)

	Bit Position																
Input Reg 1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Input Coil #								Coil 9							Coil 3	Coil 2	Coil 1
Name	reserved	reserved	reserved	reserved	reserved	reserved	reserved	Tool String Change	BCR String Change	reserved	reserved	reserved	reserved	reserved	Product Change	Remote Teach	Trigger

Tool String Update Flags

Tag Name	Bit Position															
String Update Flags	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
String Update Flags	Region 6 Mask	Region 5 Mask	Region 4 Mask	Region 3 Mask	Region 2 Mask	Region 1 Mask	reserved	reserved	reserved	reserved	Region 6	Region 5	Region 4	Region 3	Region 2	Region 1