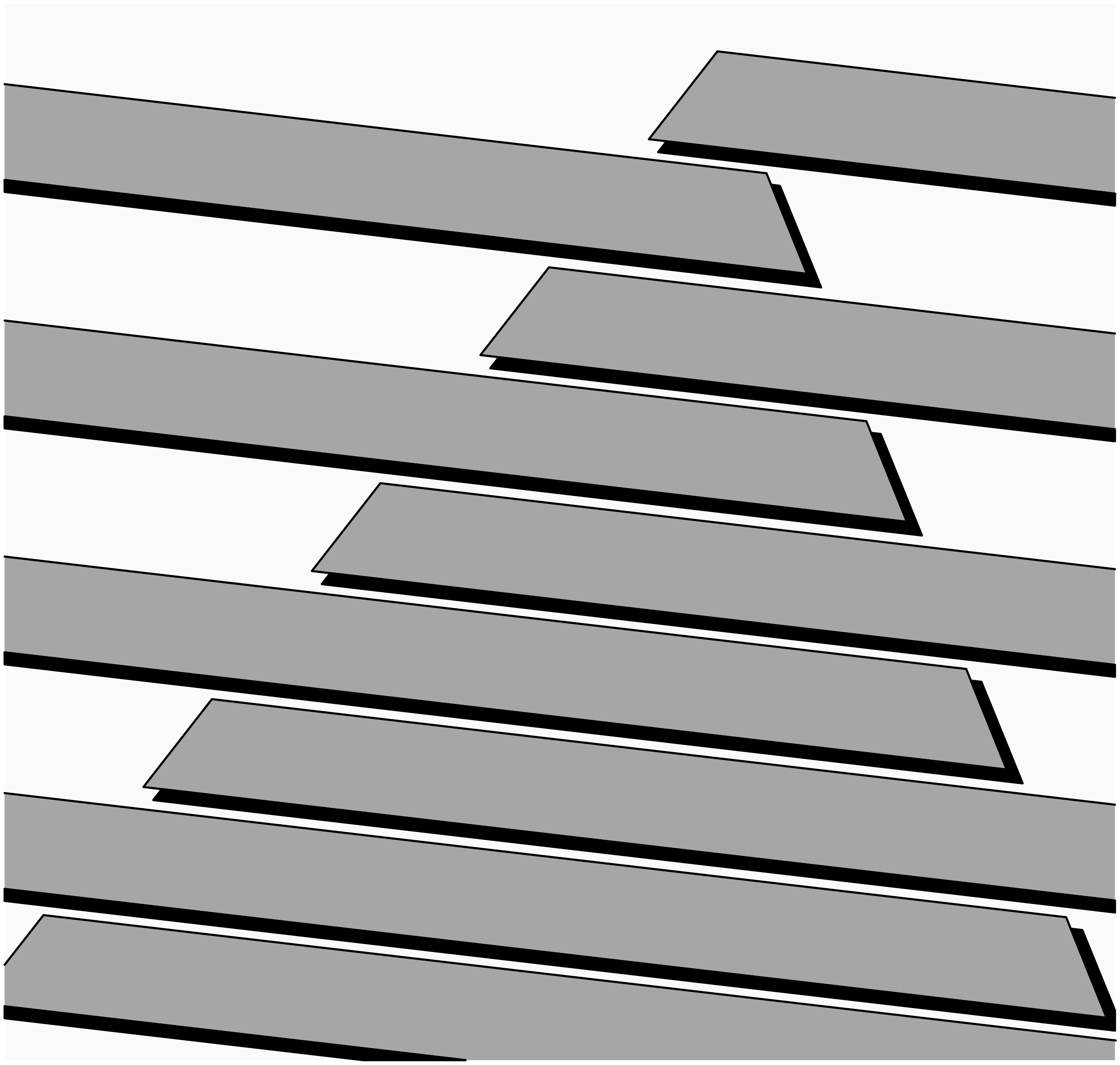
ALLEN-BRADLEY

Bulletin 5370 CVIM™ Configurable Vision Input Module

Communications Manual



Important User Information

Solid state equipment has operational characteristics differing from those of electromechanical equipment. "Safety Guidelines for the Application, Installation and Maintenance of Solid State Controls" (Publication SGI-1.1) describes some important differences between solid state equipment and hard–wired electromechanical devices. Because of this difference, and also because of the wide variety of uses for solid state equipment, all persons responsible for applying this equipment must satisfy themselves that each intended application of this equipment is acceptable.

In no event will the Allen-Bradley Company be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, the Allen-Bradley Company cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Allen-Bradley Company with respect to use of information, circuits, equipment, or software described in this manual.

Throughout this manual we use notes to make you aware of safety considerations.

ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss.

Attentions help you:

- identify a hazard
- avoid the hazard
- recognize the consequences

Important: Identifies information that is especially important for successful application and understanding of the product.

Using the Remote I/O Link (Node Adapter)

Chapter 4

Using the RS–232 Ports

Chapter 5

Tables

Figures

Chapter 1

Using this Manual

Chapter Objectives

Read this chapter to familiarize yourself with the rest of the manual. You will learn about:

- Contents of the manual.
- Intended audience.
- How to use the manual.

Software Revision

This manual describes how to communicate with a CVIM™ module (Catalog No. 5370–CVIM Series A or B) with at least firmware revision C03.

Overview of this Manual

This manual explains how to communicate with the Bulletin 5370 Configurable Vision Input Module (CVIM) module using a variety of peripheral devices.

Chapter	Title	Purpose
1	Using This Manual	Provides an overview.
2	Introduction	Describes the basic options available for communications with the CVIM module.
3	Using Local I/O	Describes how to use the discrete I/O module (Catalog No. 1771–JMB).
4	Using the Remote I/O Link (Node Adapter)	Describes how to access data through the remote I/O port with a PLC–2®, or PLC–3®, PLC–5™. Includes sample programs.
5	Using the RS–232 Ports	Describes how to access data through the RS–232 interfaces using ASCII and DF1 formatted commands. Provides sample programs.
6	Using the Pyramid Integrator™ Backplane	Describes how to access shared memory through the Pyramid Integrator™ backplane using a PLC-5/250 information processor and/or MicroVAX®. Provides sample programs.
Appendix A	Results/Configuration Data Overview	Provides an overview of the configuration and results data. Describes data formats.
Appendix B	Discrete I/O Results Bits	Provides a description of the 128 discrete input bits and 128 discrete output bits.
Appendix C	Numerical Results Data	Provides a description of inspection results data.
Appendix D	Configuration Data	Provides information on the configuration blocks.
Appendix E	ASCII Conversion Chart	Provides equivalent values for the ASCII character set.
	Glossary	
	Index	

Intended Audience

This manual was written for an experienced PLC® user or computer programmer. The user of this manual should:

- Know how to program the host device being used to communicate with the CVIM module. For example, if you are using a PLC–5 to communicate with the CVIM module, you must have a background in programming a PLC–5.
- Know terms common to the computer and programmable controller industries.
- Understand how to operate and configure the CVIM module before using this manual. You may not understand many of the terms being used unless you have read the CVIM User's Reference Manual, Catalog No. 5370–ND001.

Related Publications

Table 1.A lists related publications that you may require:

Table 1.A
Related Publications

Publication No.	Title	Purpose of Publication
Catalog No. 5370–ND003	CVIM Quick Start Manual	Describes the basics of the CVIM user interface.
Catalog No. 5370–ND001	CVIM User's Manual	Provides step–by–step procedures for the installation, configuration and operation of the CVIM module.
1772–6.8.1	PLC–2/20 Programming and Operations Manual	Provides instructions on how to program a PLC–2/20® programmable controller.
1772–6.8.3	PLC–2/30 Programming and Operations Manual	Provides instructions on how to program a PLC–2/30® programmable controller.
1772–6.8.6	Mini–PLC–2/05 Programming and Operations Manual	Provides instructions on how to program a Mini–PLC–2/05® programmable controller.
1772–6.8.2	Mini–PLC–2/15 Programming and Operations Manual	Provides instructions on how to program a Mini–PLC–2/15® programmable controller.
1775–6.7.1	PLC–3 Controller Installation and Operations Manual	Provides instructions on how to program a PLC–3 programmable controller.
1785–6.8.2	PLC–5 Family Processor Manual	Provides instructions on how to program a PLC–5 programmable controller.
5000–2.3	Allen–Bradley Pyramid Integrator Technical Overview	Provides an overview of the Pyramid Integrator.
5000–2.17	Allen–Bradley Pyramid Integrator Technical Description	Provides a technical description of the Pyramid Integrator.
5000–2.20	MicroVAX Information Processor Technical Description	Provides a technical description of the MicroVAX Information Processor.
5000–6.2.10	Allen–Bradley Pyramid Integrator Installation Manual	Provides instructions on installing Pyramid Integrator devices.
5000–6.2.10	Allen–Bradley Pyramid Integrator Start–up and Integration Manual	Provides instructions on how to use Pyramid Integrator devices.

How to Use this Manual

When using this manual, we recommend that you do the following.

1. Become familiar with the CVIM module by reading the User's Manual, Catalog No. 5370–ND001. If possible, use the CVIM module to become familiar with its operation. Only with a thorough understanding of the CVIM module will you be able to interpret the data that is stored in its memory.
2. Read Chapters 1 and 2 of this manual. After reading these introductory chapters, you will be able to determine which of the remaining chapters, some or all, you will need to read. See note below.

   Important Note: This manual is divided into chapters. It is not necessary to read all of the information contained in this manual. Chapters 1 and 2 are mandatory. You can read the remaining chapters on a "need to know basis" depending upon the information you want to read or write and the type of host device you are using.

3. Use the programming examples provided in each section as a guide to create your own programs. In some applications, you may be able to simply modify the example provided.

   These examples are included solely for illustrative purposes. Because the many variables and special requirements associated with any particular installation, Allen–Bradley Company cannot assume responsibility or liability for their applicability to your own situation.

Nomenclature

In this Chapter and in subsequent chapters we refer to the Bulletin 5370 Configurable Vision Input Module as CVIM module. In some tables we use the abbreviation "PI" to indicate the PLC–5/250 Pyramid Integrator. We have also provided a glossary in the back of this manual. Use this glossary whenever you are unsure of the meaning of a word.

Trademarks

In this manual, we use the following trademarks:

CVIM™ is a trademark of Allen–Bradley
PLC®, PLC–2®, PLC–2/20®, PLC–2/30®, PLC–2/05®,
PLC–2/15®, and PLC–3® are registered trademarks of Allen–Bradley
PLC– 5™, PLC–5/250™ are trademarks of Allen–Bradley
Pyramid Integrator™ is a trademark of Allen–Bradley
Dataliner™ is a trademark of Allen–Bradley
RediPANEL™ is a trademark of Allen–Bradley
DATAMYTE® is a registered trademark of Allen–Bradley

Microsoft® is a registered trademark of Microsoft Corporation
MicroVAX® is a registered trademark of Digital Equipment Corporation
GW BASIC™ is a trademark of Microsoft Corporation

Chapter 2

Introduction

Chapter Objectives

In this chapter we provide you with an overview of the options for communicating with the CVIM module. We also describe the types of data that can be accessed or manipulated. The descriptions in this chapter will enable you to determine the type of communications most suitable for your application. You then can proceed to the chapter of this manual that describes the selected option.

How is Data Stored in the CVIM Module?

The result and command data that you can access with a host device is stored in an area of Random Access Memory (RAM) inside the CVIM module. Configuration data which controls the operating instructions for the CVIM module is located in a separate area of memory which can be also be accessed through a host device. Refer to Appendix A for an overview of configuration/results memory. Appendix B, C, and D contain tables listing the information stored in results and configuration memory locations.

How Does the Host Device Read Configuration/Results Information?

The remainder of this chapter describes the various options you have for accessing this information. Refer to Figure 2.1. In summary, your host device will be linked to the CVIM module through one of the following ports:

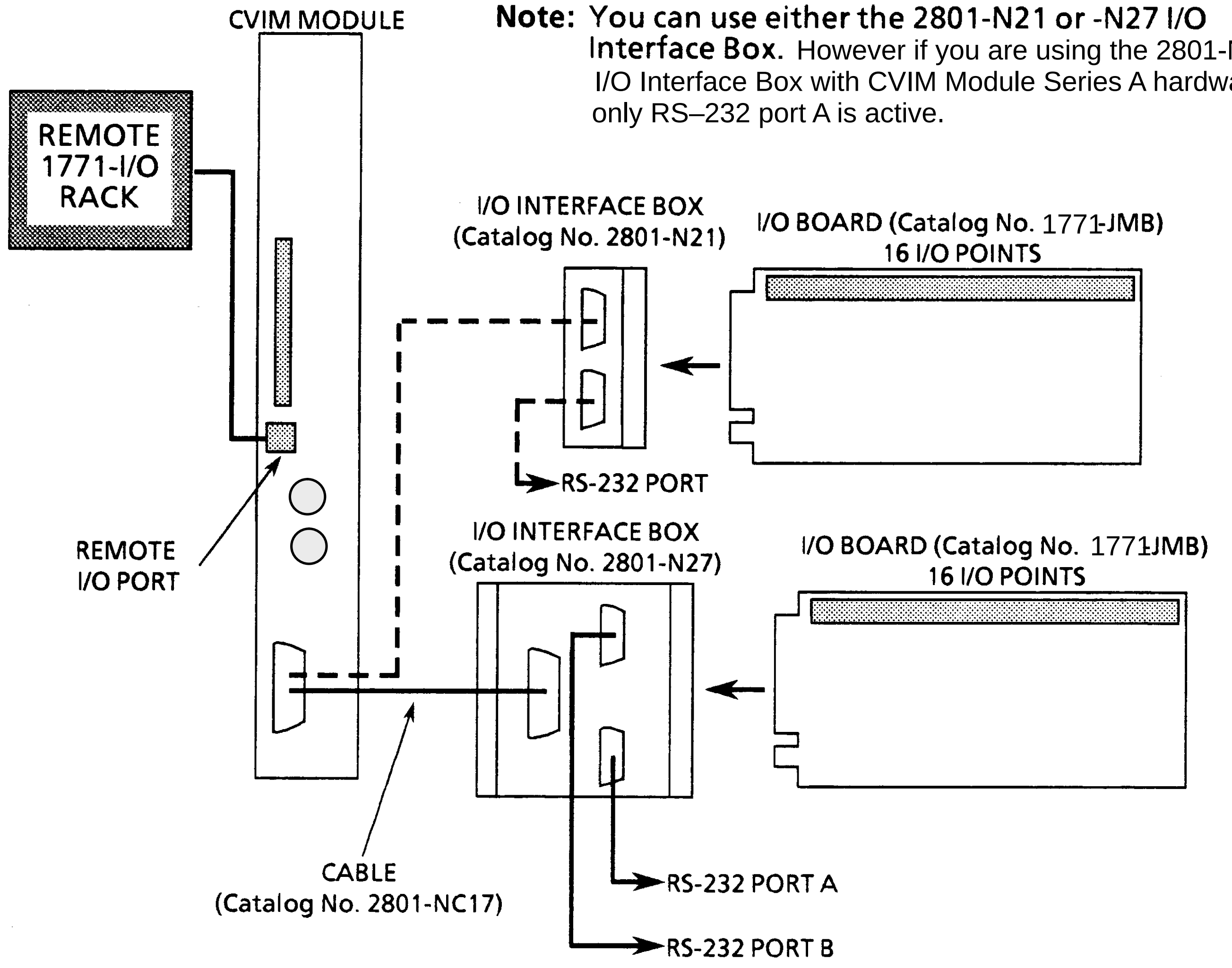
- Remote I/O (Node Adapter)
- RS–232 Interface(s)
- Pyramid Integrator Backplane
- Local I/O Board

Note: The local I/O board has sixteen discrete I/O lines. Fourteen of these lines are outputs only. One of the remaining lines is for input, and can be connected to a presence–sensing device to trigger an inspection process. The other line is not used.

How Does the Host Device Read Configuration/Results Information? (cont'd)

Figure 2.1
CVIM Module Communications Ports

Note: You can use either the 2801-N21 or -N27 I/O Interface Box. However if you are using the 2801-N27 I/O Interface Box with CVIM Module Series A hardware, only RS–232 port A is active.


CVIM MODULE
REMOTE 1771-I/O RACK
REMOTE I/O PORT
I/O INTERFACE BOX (Catalog No. 2801-N21)
I/O BOARD (Catalog No. 1771-JMB) 16 I/O POINTS
RS-232 PORT
I/O INTERFACE BOX (Catalog No. 2801-N27)
I/O BOARD (Catalog No. 1771-JMB) 16 I/O POINTS
CABLE (Catalog No. 2801-NC17)
RS-232 PORT A
RS-232 PORT B

Remote I/O (Node Adapter)

The remote I/O port (RIO) is located on the front of the CVIM module as shown in Figure 2.1. Using the remote I/O port, you can connect the following types of devices:

- Allen–Bradley Programmable Controllers (PLC–2, –3, and –5).
- Host Computers which have the Allen–Bradley IBM Bus Scanner (Catalog No. 6008–SI). The 6008–SI bus scanner is compatible with the A–B 6121/22 Industrial Computer, Industrial Terminal (Catalog Nos. 1784– T50, 1784–T35), or other IBM PC/AT compatible devices.

RS–232 Ports

As shown in Figure 2.1, the RS–232 ports are located on the I/O Interface Boxes (Catalog No. 2801–N21, –N27). The I/O Interface Box is connected to the MODULE I/O port on the front of the CVIM module. Using the RS–232 interface(s) you can connect a variety of devices which use the RS–232 standard:

- Computers
- Operator Interfaces such as Allen–Bradley Industrial Computers and Terminals with serial ports.
- I/O modules such as the Basic module (Catalog No. 1771–RB) or ASCII module (Catalog No. 1771–DA).
- DATAMYTE and Dataliner (requires USER-PAK Software (Catalog No. 5370-UPK)

Local I/O

As shown in Figure 2.1, the local I/O consists of an I/O Board (Catalog No. 1771–JMB), I/O Interface Box (Catalog No. 2801–N21, –N27), an input and up to 14 output modules as configured by the user. The Catalog No. 2801–NC17 cable connects the I/O interface box to the CVIM module.

Pyramid Integrator Backplane

Using the Pyramid Integrator backplane, you can directly communicate data between the CVIM module and other devices installed in the Pyramid Integrator chassis:

- Allen–Bradley PLC–5/250
- MicroVAX Information Processor
- Pyramid Integrator Resource Manager

What Types of Information can be Communicated?

Depending upon the type of interface in use, you can access some or all of the information listed below:

- Warning and Pass/Fail data.
- Numerical inspection results.
- Configuration data.

Discrete Bit Information

With each inspection that the CVIM module performs, individual bits are set. There are 128 bits that can be read as inputs to a host device. These bits (part of the inspection results) indicate:

- Master fault.
- Mastership.
- Configuration fault.
- Module Busy flag.
- Missed Trigger flag.
- Results Valid flag.
- Inspection Tool Pass/Fail/Warning flags.

There are 128 bits that can be set as outputs by a host device to control the operation of the CVIM module. These bits control:

- Monitor display.
- Inspection trigger.
- Toolset selection.
- Enable/disable and force discrete I/O.
- Selection of operation after reject.
- Memory storage location. RAM, EEPROM, RAM Card, or external host memory.

Note: For more information on the 128 discrete input and 128 discrete output bits refer to Appendix B.

Results Blocks

The results data for each inspection are stored in Random Access Memory (RAM) and overwrite the results of the previous inspection. The data stored in results blocks contain information regarding reference windows, inspection gages, inspection windows, etc. For a complete description of the results blocks, refer to Appendix C.

Configuration Blocks

The user developed inspection parameters of the CVIM module are stored in the CVIM module's memory as configuration blocks. This area of memory can be read or manipulated through the Remote I/O port, RS–232 ports (A & B) or Pyramid Integrator backplane. Refer to Appendix D for a complete description of the configuration blocks and their contents.

Communications Cables

If you are not using the Pyramid Integrator backplane for communications, you will have to physically link the CVIM module to the host device. If you need to create a communications cable, refer to the chapter that describes the communications port you are using.

Memory Addressing

Depending upon how you access the CVIM module results and configuration memory, you will have to address the data differently. If you refer to Appendix A, B, and C you will notice that separate columns are provided for Backplane, RS–232, and Remote I/O communications:

Note: The RS–232 protocols (ASCII and DF1) do not access data using word and bit addresses. Data is read/written in blocks. We have grouped the RS–232 and Remote I/O ports together in Appendix B, C, and D (where appropriate) for your convenience. You can ignore word and bit addresses if you are using the RS–232 ports (A & B).

Memory Addressing (cont'd)

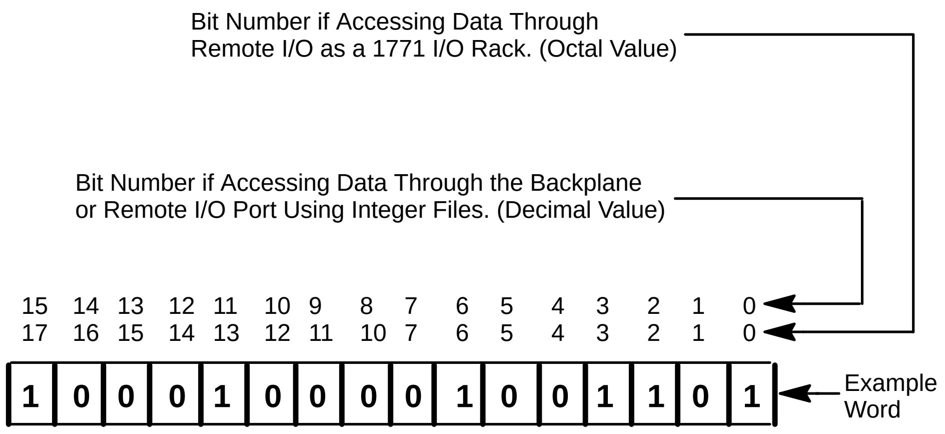
When you communicate through the Pyramid Integrator backplane all of the data words are numbered consecutively and grouped in blocks. When you use the Remote I/O port, you select a specific block and the first word in each block is word #0.

Example of Addressing Results Block 1

Word Number		
Pyramid Integrator Backplane		RS–232 and Remote I/O
Toolset 1	**Toolset 2**	**0–63**
24-87	**288-351**	

In addition, PLC I/O bit numbers are entered in octal format when referencing 1771 I/O, while PLC files and backplane communications specify a decimal bit number. Figure 2.2 illustrates how bits are numbered.

Figure 2.2
Bit Numbering


Bit Number if Accessing Data Through Remote I/O as a 1771 I/O Rack. (Octal Value)
Bit Number if Accessing Data Through the Backplane or Remote I/O Port Using Integer Files. (Decimal Value)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
17 16 15 14 13 12 11 10 7 6 5 4 3 2 1 0
1 0 0 0 1 0 0 0 0 1 0 0 1 1 0 1
Example Word

Host Designation

There are four communications ports which you can use simultaneously to access CVIM module data (Remote I/O, RS–232 Ports A & B, and Backplane). Only the host can issue commands to control the operation of the CVIM module. You can read discrete bits and numerical results information through any of the four communications ports, even through non–host devices.

Multiple Hosts

The CVIM module can operate with multiple hosts. You can select one host to perform CVIM module/host configuration transfers, and another host to perform all other CVIM/host operations. These two hosts are referred to as the configuration host (CFG) and the system host (SYS). An example of using multiple hosts is to select RS–232 A as the CFG host, and Remote I/O as the SYS host.

Note: Any CVIM communications port can be used for reading results block data regardless of whether or not the device connected to the port is selected as a host.

Note: You can select the same host (Stand Alone, Pyramid, Remote I/O, RS–232 A or B) as both the configuration host and the system host.

Chapter 3

Using Local I/O

Chapter Objectives

The objectives of this chapter are to help you plan:

- The *number* of discrete output lines (up to 14) that your application will require.
- The *function* that each output line will perform in your application.
- The *assignment* of analysis tool "results" to output lines.
- The *assignment* of status signals to output lines.
- The *electrical and mechanical connections* of the trigger (input) and output lines to your production equipment.

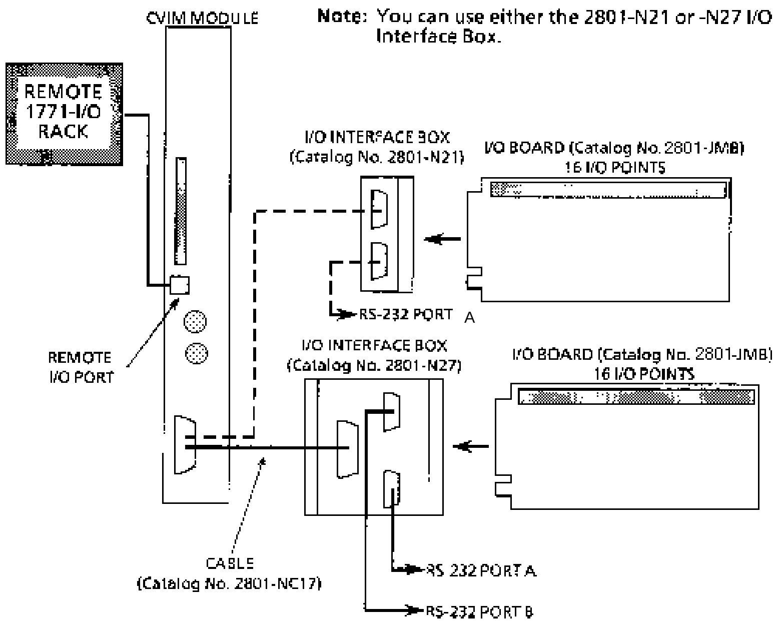
Equipment Connections

The local I/O consists of:

- I/O Interface Box (Catalog No. 2801–N21, –N27)
- I/O Board (Catalog No. 1771–JMB)
- User specified I/O modules (plug into I/O board)
- Communications Cable (Catalog No. 2801–NC17)

As shown in Figure 3.1, the communications cable (Catalog No. 2801–NC17) is connected to the MODULE I/O port on the front of the CVIM module and the connector on the I/O Interface box. The I/O board connector slides into the connector slot on the I/O Interface Box.

Figure 3.1
Local I/O Equipment Connections


CVIM MODULE
Note: You can use either the 2801-N21 or -N27 I/O Interface Box.
REMOTE 1771-I/O RACK
I/O INTERFACE BOX (Catalog No. 2801-N21)
I/O BOARD (Catalog No. 2801-JMB) 16 I/O POINTS
RS-232 PORT A
REMOTE I/O PORT
I/O INTERFACE BOX (Catalog No. 2801-N27)
I/O BOARD (Catalog No. 2801-JMB) 16 I/O POINTS
CABLE (Catalog No. 2801-NC17)
RS 232 PORT A
RS-232 PORT B

Planning Output Line Assignments

This section provides a planning sheet that you can use to lay out the *function* and *tool* assignments for output lines.

The term “function assignment” refers to the type of signal information that you want an output line to carry to your production equipment.

The term “tool assignment” refers to the tool(s) that you assign to an output line.

Note: Tools can be assigned *only* to output lines that you have assigned a “results” *function.* These output lines will carry the “pass/fail” *results* signals from the tools during each inspection.

The next section, *Planning Output Line Connections*, provides electrical and timing diagrams and data. You will need to use these diagrams to correctly identify and connect the output lines to your production equipment.

Using the Output Line Planning Sheet

The Output Line Planning Sheet is a form on which you can lay out your plans for each output line. On this form you can account for:

- The 14 output lines.
- The six output line functions.
- The 64 gages and their warning and fault outputs.
- The 48 windows and their warning and fault outputs.
- The 6 reference tools and their “pass/fail” outputs.
- The light probe with its separate red, green, and blue warning and fault outputs.

Here is an example of how an Output Line Planning Sheet could be filled out:

Example CVIM Module Output Line Planning Sheet
Output Line Functions and Tool Assignments

Line No.	Output Line Function	Tool Set No.	Gage				Window				Reference Tool		Light Probe	
			No.	Rng.	No.	Rng.	No.	Rng.	No.	Rng.	Line	Win.	Cam.	Rng.
1	Results	1	1	W	2	W	1	W	2	W				
″	″	″	3	W	4	W								
2	Results	1	1	F	2	F	1	F	2	F				
″	″	″	3	F	4	F								
3	Results	1									1	1		
4	Results	1											A	W
5	Results	1											A	F
6	Results	2	1	W	1	F								
″	″	″	2	W	2	F								
7	Strobe	1												
8	Trig. NAK	1												
9	Master Fault	1												
10	Data Valid	1												
11	Module Busy	NA												
12	Not Used													
13	Not Used													
14	Not Used													

The entries for the output lines have the following meanings:

- **Output Line 1:** The Results function is assigned to line 1. The Warning Range results (W) for gages 1–4 and windows 1 and 2 of toolset #1 are assigned to output line 1.
- **Output Line 2:** The Results function is assigned to line 2. The Fault Range results (F) for gages 1–4 and windows 1 and 2 of toolset #1 are assigned to output line 2.
- **Output Line 3:** The Results function is assigned to line 3. The “pass/fail” results for reference line 1 of toolset #1 and reference window 1 are assigned to line 3.
- **Output Line 4:** The Results function is assigned to line 4. The Warning Range result from the camera A light probe is assigned to line 4. Camera A is assigned to toolset #1.

Using the Output Line Planning Sheet (cont'd)

- **Output Line 5:** The Results function is assigned to line 5. The Fault Range result from camera A probe is assigned to line 5.
- **Output Line 6:** The Results function is assigned to line 6. The Warning *and* Fault Range results for gages 1 and 2 of toolset #2 are assigned to line 6.
- **Output Line 7:** The Strobe function for toolset #1 is assigned to line 7.
- **Output Line 8:** The Trigger NAK function for toolset #1 is assigned to line 8.
- **Output Line 9:** The Master Fault function for toolset #1 is assigned to line 9.
- **Output Line 10:** The Data Valid function for toolset #1 is assigned to line 10.
- **Output Line 11:** The Module Busy function is assigned to line 11. (Note that this function does not relate to a toolset).
- **Output Lines 12–14:** These lines are not used.

Note: Output lines 1–6 are assigned the Results function. These lines will carry "pass/fail" results from the analysis tools to your production equipment. Lines 7–11 are assigned other functions. Lines 12–14 are not used.

Here is a brief explanation of the signal functions that you can assign to the output lines:

- **Module Busy:** This signal goes *high* when the CVIM system enters the configuration mode and during a configuration download operation. Module Busy goes *low* when the system enters the run mode (whether or not triggers are present).

  You can assign the Module Busy function to only *one* output line.

Note: When configurations are being downloaded to the CVIM module, the module busy signal at the JMB board is not active.

Note: All of the remaining signal functions (except Strobe, Module Busy, and Trigger NAKs) can be configured to produce a *pulse* whose duration depends on the number of milliseconds that you assign to the Duration/1 or Duration/2 parameter. (The ‘‘1’’ and ‘‘2’’ designate toolset #1 and toolset #2).

- 1/Results: This signal occurs when the results of a tool inspection exceed the warning and/or fault limits. (The tool must be assigned to an output line that has already been assigned the Results function.)

You can assign the Results signal function to *any unassigned* output line.

As noted above, the 1/Results signal function must be assigned to an output line *before* any tool can be assigned to that line. Thus, if you wanted inspection results from Ref. Line # 2 to be assigned to output line #10, you would *first* have to assign the Results signal function to output line #10.

Note: You can assign the inspection results from *any* tool in toolset #1 to an output line to which you have already assigned the 1/Results signal function.

- 1/Data Valid: This signal occurs when the CVIM system has *completed* an inspection using toolset #1. 1/Data Valid *signals* (the “data”) are stable on all output lines assigned to the 1/Results signal function. 1/Data Valid goes low during the next inspection.

  Note: 1/Data Valid does *not* indicate whether an inspection has passed or failed. That is the task of the output lines assigned to the 1/Results signal function.

  You can assign the 1/Data Valid function to only *one* output line.

- 1/Trigger NAK: This signal occurs when the CVIM system receives a trigger input signal for toolset #1, but cannot process that trigger. The signal goes low upon the next “accepted trigger”.

  You can assign the 1/Trigger NAK function to only *one* output line.

- 1/Master Fault: This signal occurs when *any (one or more)* analysis tools in the CVIM system detects a Fail condition.

  You can assign the 1/Master Fault function to only *one* output line.

- 1/Strobe: This signal is used to trigger the strobe flash unit (if used). The signal occurs within 1 ms after the CVIM system receives a trigger input signal.

You can assign the 1/Strobe function to only *one* output.

- 1/Duration (n)ms: From 1msec to 2000msec. This value determines the pulse duration, in milliseconds of *all* pulse–type signals. A setting of zero means the signal will remain in its present state until updated by a subsequent inspection.

Note: The output duration may vary if subsequent inspections occur before the specified output duration has elapsed.

In *your* application, the function and tool assignment(s) for each output line will of course depend on the specific requirements of your production equipment.

You will find a full–page, blank copy of the planning sheet on the last page of this chapter. We suggest that you do not mark that page, but use it instead as a copy master, and use the copies to prepare your output line plans.

Using the Output Line Planning Sheet (cont'd)

Keep in mind that a *completed* planning sheet can serve also as a *record* of your output line usage. You may find it desirable to store your filled–out planning sheets in a file folder or loose leaf binder.

Using Output Signal Timing Data

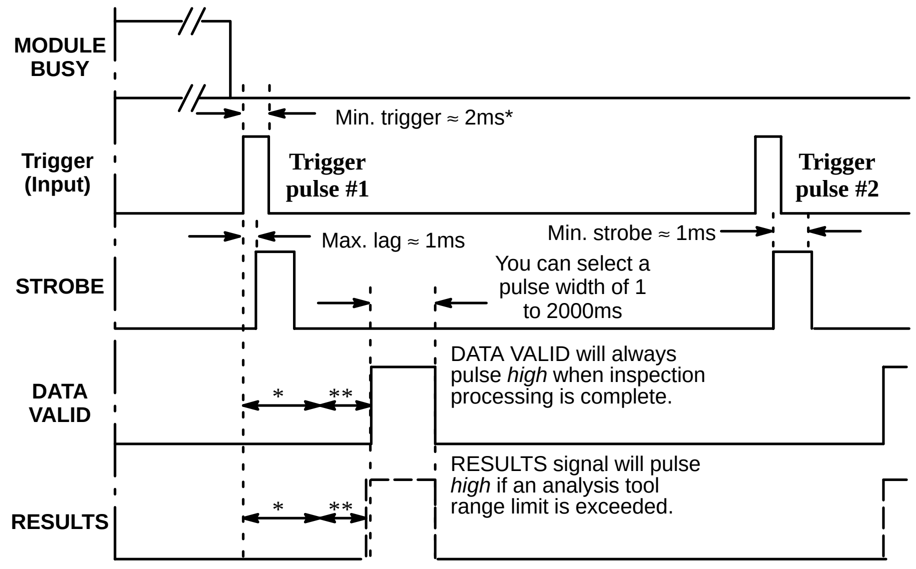
To make proper use of the signal data available to the output lines, you must first understand the timing relationships that exist between the trigger *input* signal (which *starts* each inspection cycle) and the *output* signals.

Knowing these signal timing relationships enables you to accurately *synchronize* the inspection cycles with your production equipment.

Timing charts (Figures 3.2, 3.3, and 3.4) show the timing relationships in various circumstances.

Figure 3.2 shows the relationship between the trigger leading edge and the Strobe, Data Valid, and Results signals, where the last three appear as *pulses* whose duration *you* determine during configuration.

Figure 3.2
Timing Diagram — Pulsed I/O

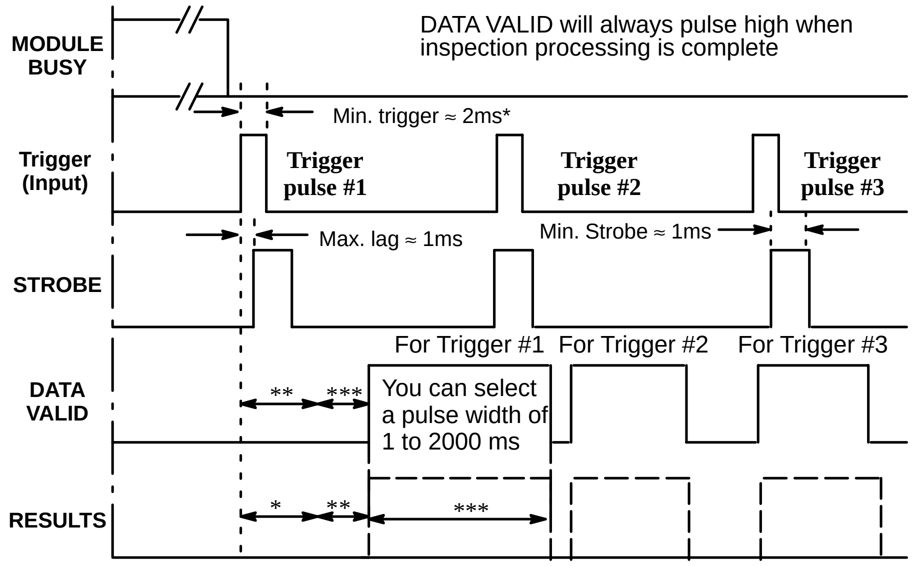

MODULE BUSY
Min. trigger ≈ 2ms*
Trigger (Input)
Trigger pulse #1
Trigger pulse #2
Max. lag ≈ 1ms
Min. strobe ≈ 1ms
STROBE
You can select a pulse width of 1 to 2000ms
DATA VALID
*
**
DATA VALID will always pulse *high* when inspection processing is complete.
RESULTS
RESULTS signal will pulse *high* if an analysis tool range limit is exceeded.

** Minimum acquisition time: 17ms for 256x256 and 512x256 Res; 34 ms for 512x512 res.
*** Analysis time (variable).

Using Output Signal Timing Data (cont'd)

In Figure 3.3, trigger pulse #2 occurs before the CVIM module has finished processing the inspection cycle started by trigger pulse #1.

Figure 3.3
Timing Diagram — Trigger #2 During Data Valid, Pulsed I/O



** Minimum acquisition time: 17ms for 256x256 and 512x256 Res; 34 ms for 512x512 res.
*** Analysis time (variable).
****RESULTS will pulse high if an analysis tool range is exceeded.

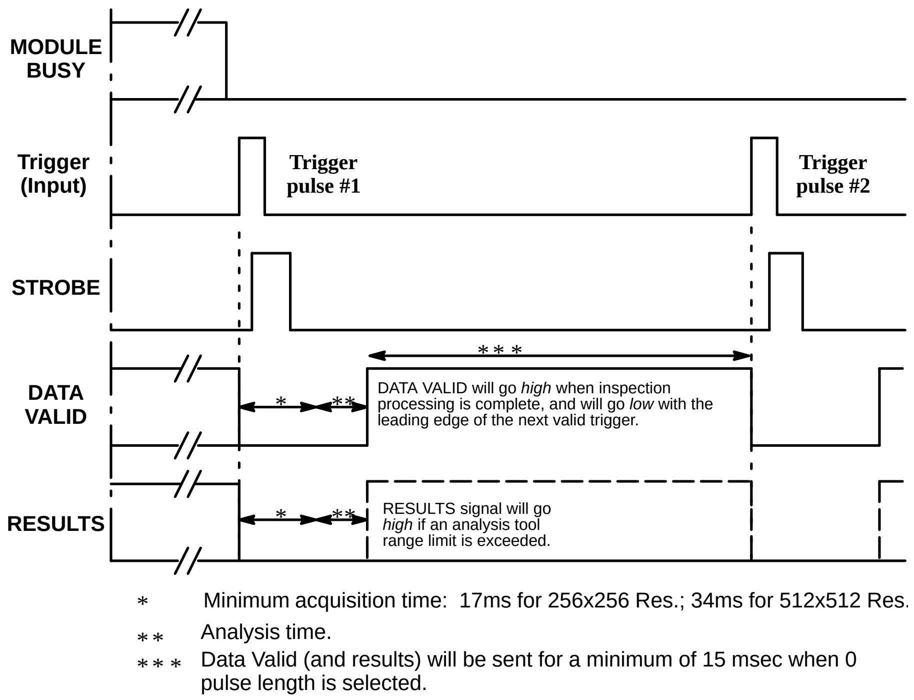
Using Output Signal Timing Data (cont'd)

Whenever these signals go *high*, they will go *low* again at the *end* of the specified pulse duration (1 to 2000ms).

Note: The Local I/O Module Busy is *high* only during system configuration.

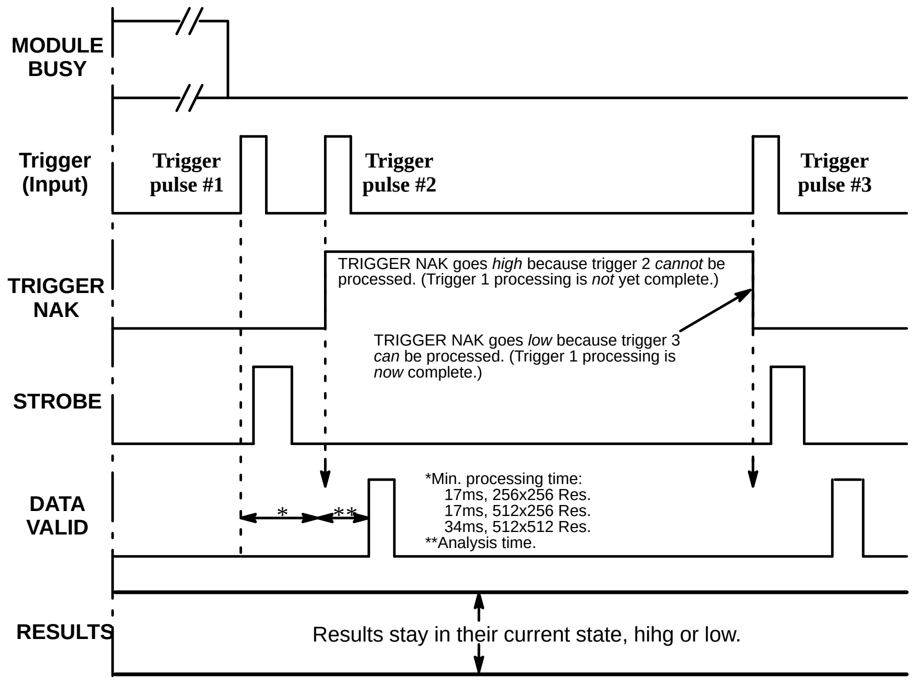
In Figure 3.4, the Data Valid, and Results signals appear as *changes in signal levels.* This will occur if, during system configuration, you select a pulse "duration" of 0 (zero) milliseconds. Data Valid will *stay* high until the leading edge of the next valid trigger signal (Trigger Pulse #2). Results stay in their current state until the leading edge of the next Trigger pulse, then change depending upon the results.

Figure 3.4
Timing Diagram — Non–Pulsed I/O


MODULE BUSY
Trigger (Input)
Trigger pulse #1
Trigger pulse #2
STROBE
DATA VALID
RESULTS
* * *
*
* *
DATA VALID will go high when inspection processing is complete, and will go low with the leading edge of the next valid trigger.
RESULTS signal will go high if an analysis tool range limit is exceeded.
* Minimum acquisition time: 17ms for 256x256 Res.; 34ms for 512x512 Res.
* * Analysis time.
* * * Data Valid (and results) will be sent for a minimum of 15 msec when 0 pulse length is selected.

In Figure 3.5, trigger pulse #2 occurs *before* the CVIM system is finished processing the inspection cycle started by trigger pulse #1. This causes the Trigger NAK signal to go *high.* Trigger NAK will *stay* high until leading edge of the next *valid* trigger pulse (trigger pulse #3).

Figure 3.5
Timing Diagram– Missed Trigger


MODULE BUSY
Trigger (Input)
Trigger pulse #1
Trigger pulse #2
Trigger pulse #3
TRIGGER NAK
TRIGGER NAK goes *high* because trigger 2 *cannot* be processed. (Trigger 1 processing is *not* yet complete.)
TRIGGER NAK goes *low* because trigger 3 *can* be processed. (Trigger 1 processing is *now* complete.)
STROBE
DATA VALID
*
**
*Min. processing time:
17ms, 256x256 Res.
17ms, 512x256 Res.
34ms, 512x512 Res.
**Analysis time.
RESULTS
Results stay in their current state, hihg or low.

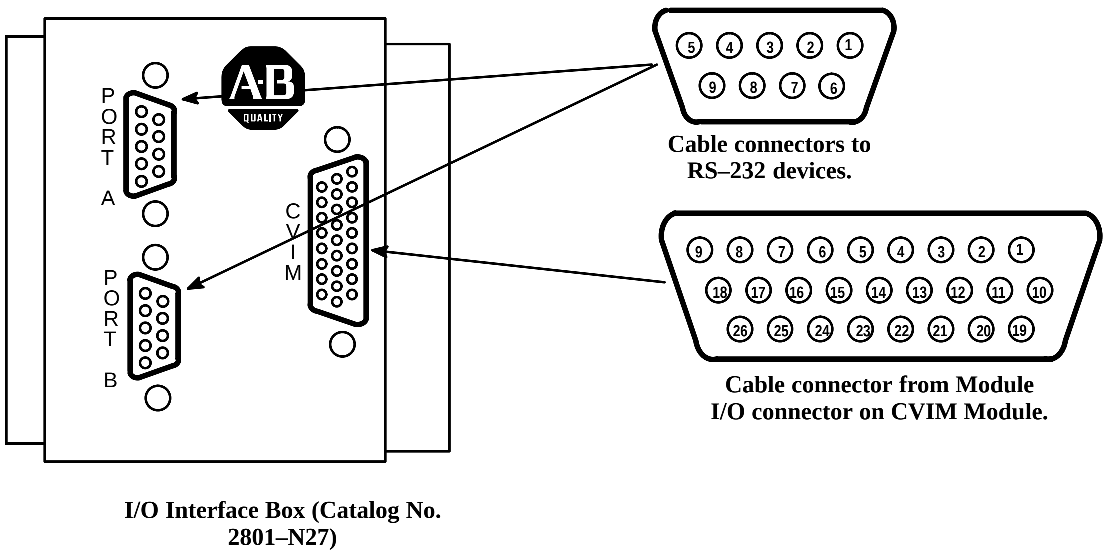
Planning Output Line Connections

This section provides diagrams of electrical connections for correctly connecting your production equipment to the CVIM module's discrete output and RS–232 lines.

Connections to RS–232 Ports (2801–N27 Interface Box)

Figure 3.6 shows the cable connectors and their pin numbers on the I/O Interface Box (Catalog No. 2801–N27).

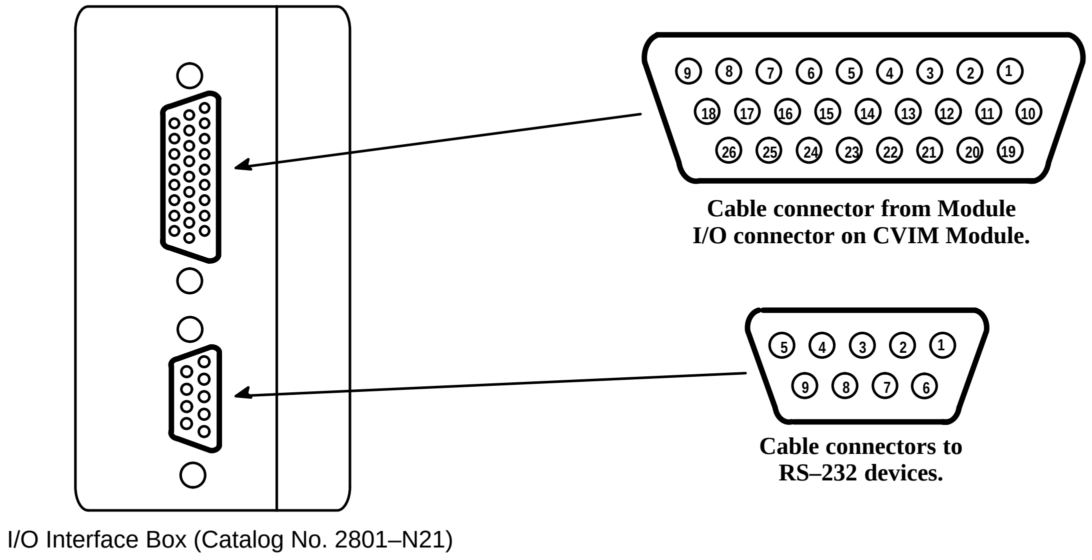
Figure 3.6
Pinouts– I/O Interface Box (Catalog No. 2801–N27)


PORT A
PORT B
CVIM
5 4 3 2 1
9 8 7 6
Cable connectors to RS–232 devices.
9 8 7 6 5 4 3 2 1
18 17 16 15 14 13 12 11 10
26 25 24 23 22 21 20 19
Cable connector from Module I/O connector on CVIM Module.
I/O Interface Box (Catalog No. 2801–N27)

Connections to RS–232 Port (2801–N21 Interface Box)

Figure 3.7 shows the cable connectors and their pin numbers on the I/O Interface Box (Catalog No. 2801–N21).

Figure 3.7
Pinouts– I/O Interface Box (Catalog No. 2801–N21)


9 8 7 6 5 4 3 2 1
18 17 16 15 14 13 12 11 10
26 25 24 23 22 21 20 19
Cable connector from Module I/O connector on CVIM Module.
5 4 3 2 1
9 8 7 6
Cable connectors to RS–232 devices.
I/O Interface Box (Catalog No. 2801–N21)

CVIM Module I/O Interface Box Connections

Tables 3.A through 3.H show the connector pin assignments with the various combinations of Series A and Series B CVIM modules connected to I/O Interface Boxes (Catalog Nos. 2801–N21, –N27) .

Table 3.A
CVIM Module I/0 Connector: Series A CVIM Module

Pin Number	Function	Pin Number	Function
1	Trigger Input Line #1	14	Output Line #12
2	Trigger Input Line #2	15	Output Line #13
3	Output Line #1	16	Output Line #14
4	Output Line #2	17	Reserved
5	Output Line #3	18	Reserved
6	Output Line #4	19	Ground (Power)
7	Output Line #5	20	Ground (Power)
8	Output Line #6	21	Ground (Chassis)
9	Output Line #7	22	Ground (Signal)
10	Output Line #8	23	TXD (Transmit Data: RS–232 A)
11	Output Line #9	24	RTS (Request to Send: RS–232 A)
12	Output Line #10	25	RXD (Receive Data: RS–232 A)
13	Output Line #11	26	CTS (Clear to Send: RS–232 A)

Table 3.B
CVIM Module I/0 Connector: Series B CVIM Module

Pin Number	Function	Pin Number	Function
1	Trigger Input Line #1	14	Output Line #12
2	Trigger Input Line #2	15	Output Line #13
3	Output Line #1	16	Output Line #14
4	Output Line #2	17	Reserved
5	Output Line #3	18	Reserved
6	Output Line #4	19	Ground (Power)
7	Output Line #5	20	Ground (Power)
8	Output Line #6	21	Ground (Chassis)
9	Output Line #7	22	Ground (Signal)
10	Output Line #8	23	TXD (Transmit Data: RS–232 A)
11	Output Line #9	24	TXD (Transmit Data: RS–232 B)
12	Output Line #10	25	RXD (Receive Data: RS–232 A)
13	Output Line #11	26	RXD (Receive Data: RS–232 B)

Table 3.C
I/O Interface Box (Catalog No. 2801–N21):
RS–232 Connector with Series A CVIM Module

Pin Number	Function	Pin Number	Function
1	No Connection	6	No Connection
2	RXD (Receive Data: RS–232 A)	7	RTS (Request to Send: RS–232 A)
3	TXD (Transmit Data: RS–232 A)	8	CTS (Clear to Send: RS–232 A)
4	Ground (Chassis)	9	No Connection
5	Ground (Signal)		

Table 3.D
I/O Interface Box (Catalog No. 2801–N21):
RS–232 Connector with Series B CVIM Module

Pin Number	Function	Pin Number	Function
1	No Connection	6	No Connection
2	RXD (Receive Data: RS–232 A)	7	TXD (Transmit Data: RS–232 B)
3	TXD (Transmit Data: RS–232 A)	8	RXD (Receive Data: RS–232 B)
4	Ground (Chassis)	9	No Connection
5	Ground (Signal)		

CVIM Module I/O Interface Box Connections (cont'd)

Table 3.E
I/O Interface Box (Catalog No. 2801–N27)
RS–232 Port A Connector Series A CVIM Module

Pin Number	Function	Pin Number	Function
1	No Connection	6	No Connection
2	RXD (Receive Data: RS–232 A)	7	+ 5V DC
3	TXD (Transmit Data: RS–232 A)	8	No Connection
4	+ 5V DC	9	No Connection
5	Ground (Signal)		

Table 3.F
I/O Interface Box (Catalog No. 2801–N27):
RS–232 Port B Connector Series A CVIM Module

Pin Number	Function	Pin Number	Function
1	No Connection	6	No Connection
2	CTS (Clear to Send: RS–232 A)	7	+ 10V DC
3	RTS Request to Send: RS–232 A)	8	No Connection
4	+ 10V DC	9	No Connection
5	Ground (Signal)		

Table 3.G
I/O Interface Box (Catalog No. 2801–N27):
RS–232 Port A Connector Series B CVIM Module

Pin Number	Function	Pin Number	Function
1	No Connection	6	No Connection
2	RXD (Receive Data: RS–232 A)	7	+ 5V DC
3	TXD (Transmit Data: RS–232 A)	8	No Connection
4	+ 5V DC	9	No Connection
5	Ground (Signal)		

Table 3.H
I/O Interface Box (Catalog No. 2801–N27):
RS–232 Port B Connector Series B CVIM Module

Pin Number	Function	Pin Number	Function
1	No Connection	6	No Connection
2	RXD (Receive Data: RS–232 B)	7	+ 10V DC
3	TXD (Transmit Data: RS–232 B)	8	No Connection
4	+ 10V DC	9	No Connection
5	Ground (Signal)		

Connections to 1771–JMB Interface

The 1771–JMB interface board is designed for direct edge connection to the I/O Interface Box, Catalog Nos. 2801–N21, –N27.

If you intend to use the 1771–JMB board and the I/O Interface Box, you will need to know the relationship between the discrete I/O line numbers and the LED numbers, the optic–isolator type, and the terminal block screws numbers on the 1771–JMB board. These are shown in the figure and table that follows.

To power the JMB logic components, you must connect an external +5VDC power supply to the (+) and (–) terminals screws shown in the board layout Figure 3.5.

Connections to 1771–JMB Interface (cont'd)

Figure 3.8 shows the layout of the 1771–JMB interface board and the adhesive–backed overlay.

Figure 3.8
Local I/O Board (Catalog No. 1771–JMB).

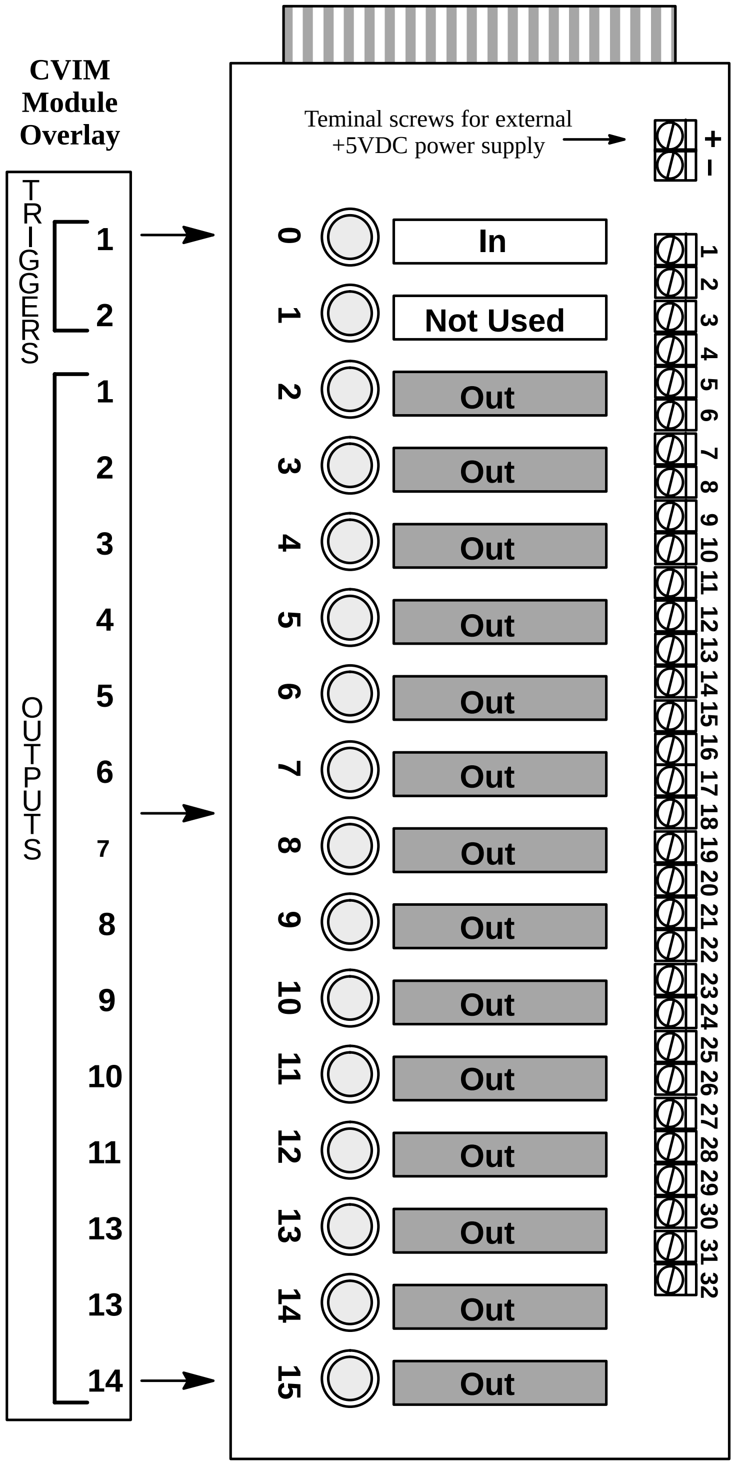


Table 3.I shows the relationship between the I/O line and optic–isolator numbers shown in Figure 3.8.

Table 3.I
Output Numbering

Discrete I/O Line Number		LED and I/O Module Number	Terminal Screw and Polarity	
Input	Output		+	–
1		0	1	2
2		1	3	4
	1	2	5	6
	2	3	7	8
	3	4	9	10
	4	5	11	12
	5	6	13	14
	6	7	15	16
	7	8	17	18
	8	9	19	20
	9	10	21	22
	10	11	23	24
	11	12	25	26
	12	13	27	28
	13	14	29	30
	14	15	31	32

Note: A self–adhesive decal (Part Number 40062-149-01) is provided with the 1771–JMB Local I/O board. This decal identifies the I/O lines. Use the chart on the next page if the decal is not in place.

OUTPUT LINE PLANNING SHEET
Output Line Functions and Assignments

Line No.	Output Line Function	Gage				Window				Reference Tool		Light Probe		
		No.	Rng.	No.	Rng.	No.	Rng.	No.	Rng.	Line	Win.	Red	Green	Blue

Chapter 4

Using the Remote I/O Link

Chapter Objectives

In this chapter we provide:

- Basic description of Remote I/O communications.
- Connection diagrams.
- Description of CVIM module setup requirements.
- Three example PLC programs for accessing CVIM module data.
- An example 6008–SI program.

Remote I/O Communications

As stated earlier, the Remote I/O port is located on the front of the CVIM module and is labeled RIO. This port allows the CVIM module to become a link in an Allen–Bradley Remote I/O network which can be up to 10,000 feet long. Data on the network can be transmitted at baud rates as high as 230K.

Maximum Link Length (Feet)	Baud Rate
10,000	57.6K
5,000	115.2K
2,500	230.4K*

* Only applies to communications between PLC-5/250 controllers in other racks.

Use twin–axial cable (Catalog No. 1770–CD) to connect the CVIM module to other devices. This cable connects to the Remote I/O port (labeled RIO) and the next device on the network. Refer to Figures 4.1 through 4.8 for connection diagrams.

Remote I/O Communications (cont'd)

Figure 4.1
PLC–5 to CVIM Module– Remote I/O Link

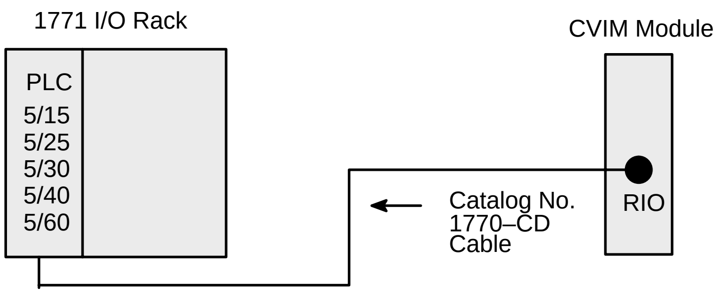


Figure 4.2
6008 SI IBM PC/AT Scanner to CVIM Module– Remote I/O Link

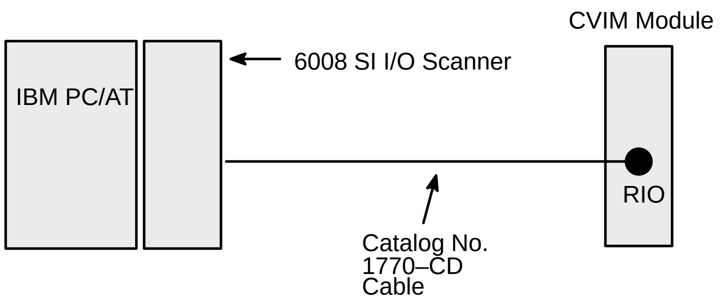
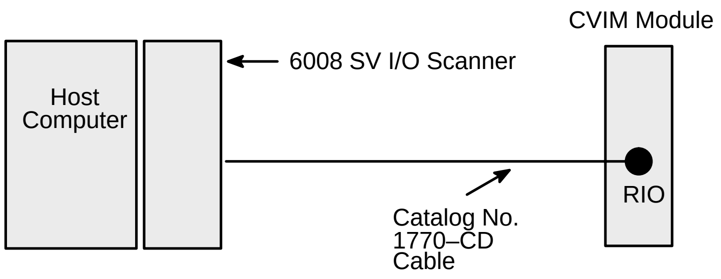


Figure 4.3
6008 SV VME Scanner to CVIM Module– Remote I/O Link



Remote I/O Communications (cont'd)

Figure 4.4
6008 SQH1/2 Q–BUS Scanner to CVIM Module–Remote I/O Link

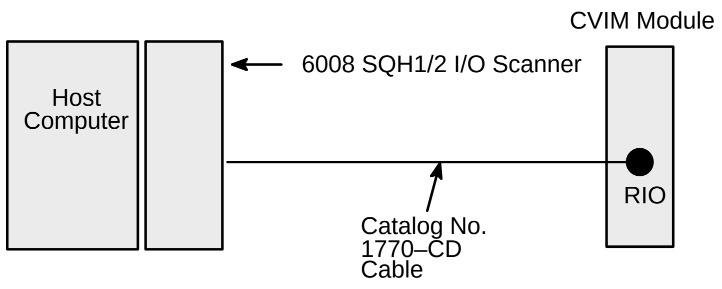


Figure 4.5
Mini PLC–2 to CVIM Module– Remote I/O Link

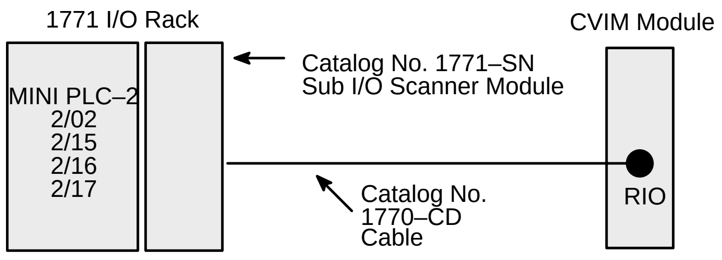
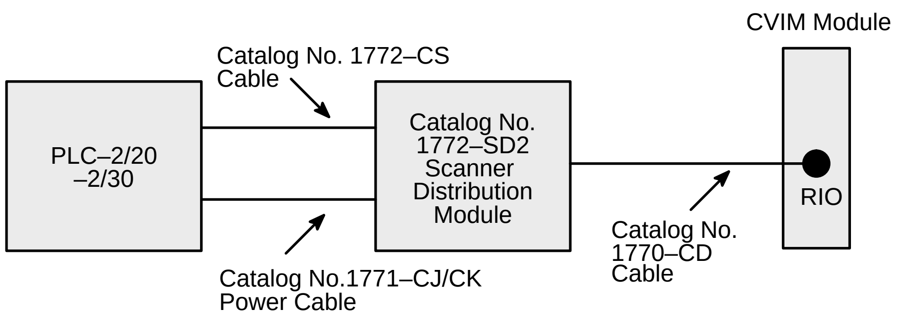

1771 I/O Rack
CVIM Module
MINI PLC–2 2/02 2/15 2/16 2/17
Catalog No. 1771–SN Sub I/O Scanner Module
Catalog No. 1770–CD Cable
RIO

Figure 4.6
PLC–2 to CVIM Module– Remote I/O Link


CVIM Module
Catalog No. 1772–CS Cable
PLC–2/20 –2/30
Catalog No. 1772–SD2 Scanner Distribution Module
RIO
Catalog No. 1770–CD Cable
Catalog No.1771–CJ/CK Power Cable

Remote I/O Communications (cont'd)

Figure 4.7
PLC–3 to CVIM Module–Remote I/O Link

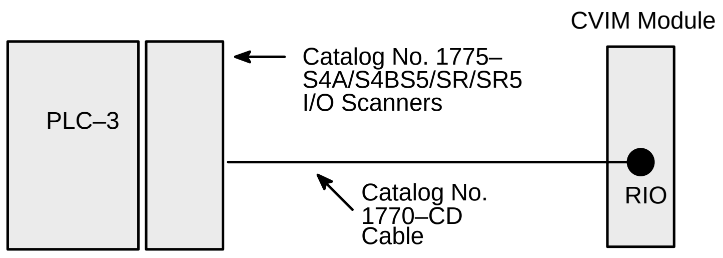
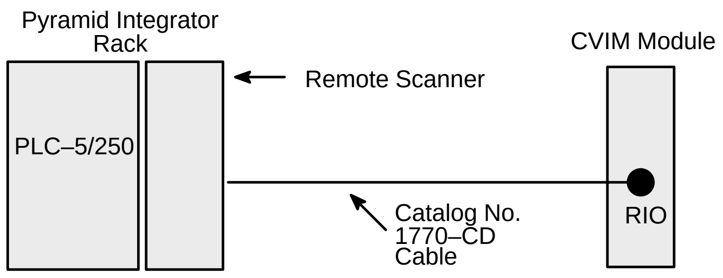
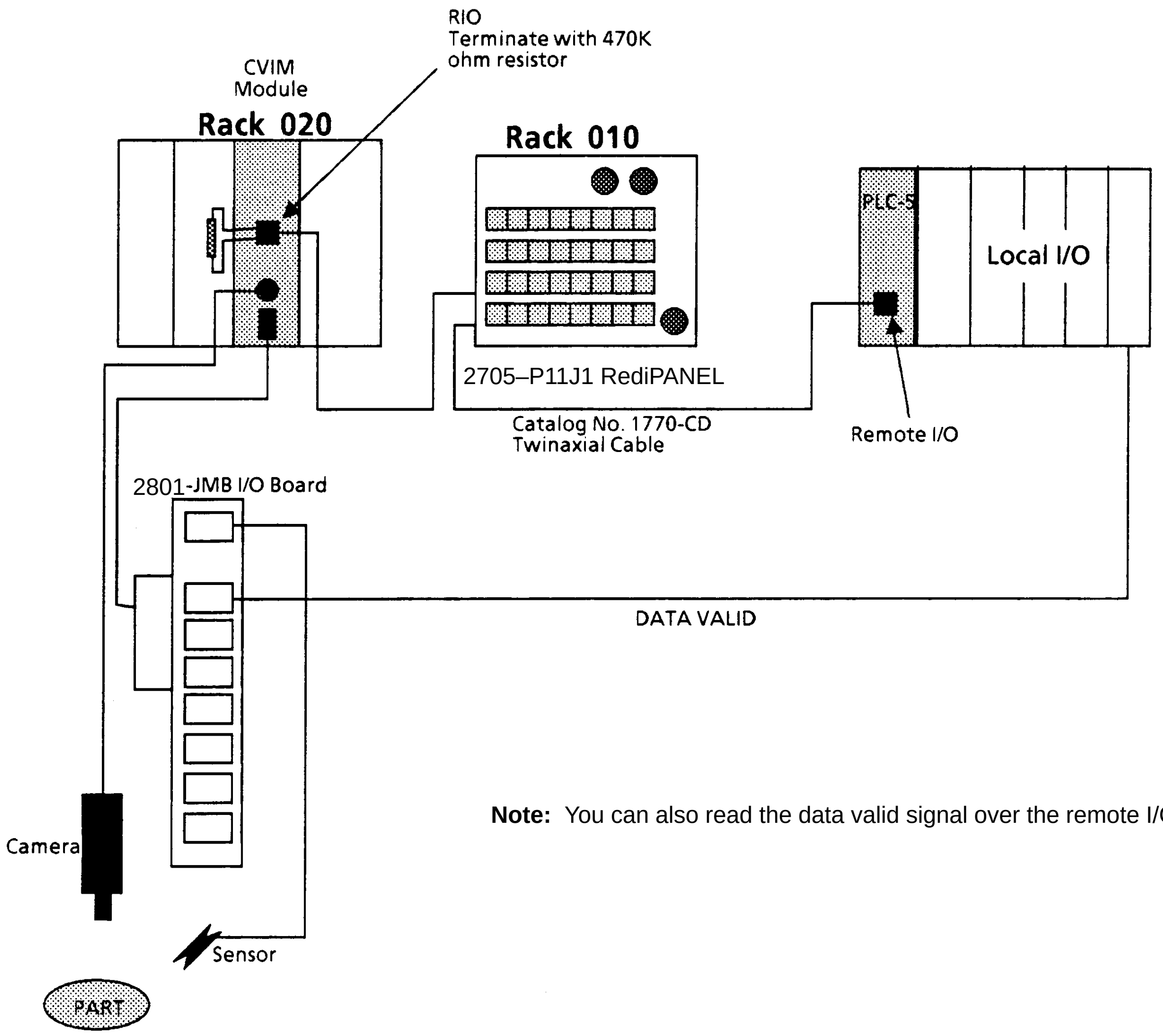

CVIM Module
PLC–3
Catalog No. 1775–S4A/S4BS5/SR/SR5 I/O Scanners
Catalog No. 1770–CD Cable
RIO

Figure 4.8
PLC–5/250 to CVIM Module– Remote I/O Link



Remote I/O Communications (cont'd)

Figure 4.9
Typical Hardware Layout for Remote I/O



Note: You can also read the data valid signal over the remote I/O link.

Remote I/O Communications (cont'd)

When installed on a Remote I/O network, the CVIM module acts as a slave device. Another device such as a PLC or computer will act as a host device. This means that the CVIM module will not initiate the sending of any data until a request is made by the host. To a host device, the CVIM module will appear simultaneously as both a full I/O rack on the network (128 input bits and 128 output bits) and as an intelligent module with block transfer capability in group 0, slot 0 in the same rack. Refer to Appendix B for a description of discrete bit data.

Note: If the CVIM module is the last node on a network, you must terminate the communication line (refer to Figure 4.9 for an example).

What Functions can be Performed over the Remote I/O Network?

A hist link can request or manipulate the following data over the Remote I/O link:

- Obtain CVIM module inspection result information. Refer to Appendix B & C.
- Upload or download CVIM module configurations for inspections. Refer to Appendix D.
- Issue Configuration Read/Write commands between the following CVIM module memory locations:
  CVIM module Random Access Memory (RAM) and CVIM module Electrically Erasable Programmable Only Memory (EEPROM). RAM is volatile and EEPROM is non-volatile.
  CVIM module RAM and RAM card. The RAM card slides into a slot on front of the CVIM module.
  CVIM module RAM and host memory.
- Change run-time display menus.
- Enable/Disable local I/O board.
- Force local I/O On or Off.

Obtaining Inspection Result Information

You can obtain inspection result information for each of the inspection tools over the Remote I/O link. There are two levels of access to this information:

- Discrete Bits. These bits indicate pass/fail/warning data.
- Result Data Words. These words contain actual inspection result data such as measured lengths, number of black pixels, etc.

Note: Refer to Appendix B for a description of the discrete bit results and Appendix C for a description of numerical results data blocks.

CVIM Module Configuration Instructions

If you are using the Remote I/O link to communicate with a PLC–2, –3, or –5 (or PLC–5/250 in another rack), you must configure the CVIM module as follows:

Select the Remote I/O port for communications:

Note: This step is not required if you are only reading results.

4. Select the setup menu <Setup>.
5. Select the environment menu <Envirn>.
6. Select the system menu <System>.
7. Select a Host menu <CFG Host> or <SYS Host>.
8. Select remote I/O option <Remote I/O>.

Note: Unless a separate configuration host is being used, set both the CFG Host & SYS Host for Remote I/O.

Configure CVIM module I/O parameters:

9. Select the I/O menu <I/O>.
10. Select <1771 Remote I/O> option.
11. Enable the Remote I/O port by selecting <Enabled>.
12. Select the rack address (octal) using the keypad.
13. Select the baud rate <57.6Kbaud> or other options.

Select the CVIM module trigger source:

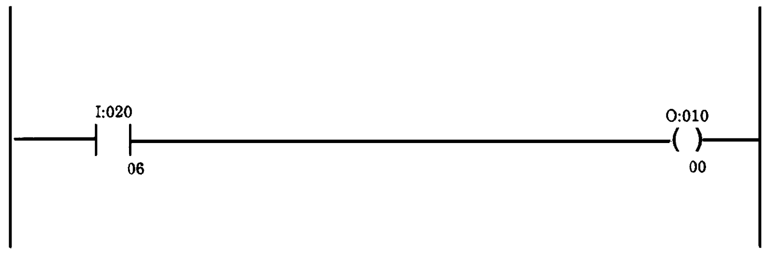
14. Select the trigger source menu <Toolset>.
15. Select the trigger source menu for the appropriate toolset <Trigger Source>.
16. Select either <I/O>, <Hosted>, or <Internally Triggered> trigger sources.

Note: The example connection diagram shown on Figure 4.9 shows a trigger using the local I/O board.

Accessing Discrete Bit Information

A PLC can directly access discrete bit information using a simple ladder program. For example:

You can use the following rung to examine the data valid bit and energize an output if the data is valid. Refer to Chapter 3 for a description of the local I/O. This example assumes that the CVIM module is in Rack 02 and the output device is in Rack 01.



Although the same basic information is provided in Appendix B, Tables 4.A and 4.B illustrate the word and bit locations of the discrete bits that can be read or manipulated using simple ladder programs. We have organized the data so that it is formatted similar to a PLC setup screen. Table 4.A shows the CVIM module Remote **Inputs** (CVIM module to PLC) if the CVIM module is rack 02. Table 4.B shows the CVIM module Remote **Outputs** (PLC to CVIM module) if the CVIM module is rack 02.

Important Note: To read results data, you must set one of the following bits (assuming CVIM module is rack 02):

- O:22/00 (Post First Part of Results to Remote I/O)
- O:22/01 (Post Second Part of Results to Remote I/O)

Note to PLC–2 Users:

When you use any PLC–2 family processor with the CVIM module, you should understand the operation of the PLC Block Transfer Done bits for Read and Write instructions. PLC–2 family processors use the input image table for these bits, all other PLCs can specify integer files for this function. This means that a PLC–2 user must use proper programming techniques to avoid confusion between the following bits:

- CVIM module discrete I/O input word 0, bit 6 (data valid toolset#1) and bit 7 (data valid toolset#2).
- PLC–2 family input image table word 0, bit 6 (BTW done bit) and bit 7 (BTR done bit).

Accessing Discrete Bit Information (cont'd)

Table 4.A
CVIM Module Remote I/O Inputs (CVIM Module to PLC) if CVIM Module is Rack 02

◄ BIT

07 / 17	06 / 16	05 / 15	04 / 14	03 / 13	02 / 12	01 / 11	00 / 10	WORD
(Not used)	1=Data Valid	0=First Bits Results	1 = Trigger Missed	1 = Module Busy	1 = PLC is Master	1 = Config. Error	(Not Used)	20
1 = Master Fault	1 = Light Probe Failed	1 = Reference Window 3 Failed	1 = Reference Window 2 Failed	1 = Reference Window 1 Failed	1 = Reference Line 3 Failed	1 = Reference Line 2 Failed	1 = Reference Line 1 Failed	
1 = Window 4 Fault	1 = Window 4 Warning	1 = Window 3 Fault	1 = Window 3 Warning	1 = Window 2 Fault	1 = Window 2 Warning	1 = Window 1 Fault	1 = Window 1 Warning	21
1 = Window 8 Fault	1 = Window 8 Warning	1 = Window 7 Fault	1 = Window 7 Warning	1 = Window 6 Fault	1 = Window 6 Warning	1 = Window 5 Fault	1 = Window 5 Warning	
1 = Window 12 Fault	1 = Window 12 Warning	1 = Window 11 Fault	1 = Window 11 Warning	1 = Window 10 Fault	1 = Window 10 Warning	1 = Window 9 Fault	1 = Window 9 Warning	22
1 = Window 16 Fault	1 = Window 16 Warning	1 = Window 15 Fault	1 = Window 15 Warning	1 = Warning 14 Fault	1 = Window 14 Warning	1 = Window 13 Fault	1 = Window 13 Warning	
1 = Window 20 Fault	1 = Window 20 Warning	1 = Window 19 Fault	1 = Window 19 Warning	1 = Window 18 Fault	1 = Window 18 Warning	1 = Window 17 Fault	1 = Window 17 Warning	23
1 = Window 24 Fault	1 = Window 24 Warning	1 = Window 23 Fault	1 = Window 23 Warning	1 = Window 22 Fault	1 = Window 22 Warning	1 = Window 21 Fault	1 = Window 21 Warning	
1 = Gage 4 Fault	1 = Gage 4 Warning	1 = Gage 3 Fault	1 = Gage 3 Warning	1 = Gage 2 Fault	1 = Gage 2 Warning	1 = Gage 1 Fault	1 = Gage 1 Warning	24
1 = Gage 8 Fault	1 = Gage 8 Warning	1 = Gage 7 Fault	1 = Gage 7 Warning	1 = Gage 6 Fault	1 = Gage 6 Warning	1 = Gage 5 Fault	1 = Gage 5 Warning	
1 = Gage 12 Fault	1 = Gage 12 Warning	1 = Gage 11 Fault	1 = Gage 11 Warning	1 = Gage 10 Fault	1 = Gage 10 Warning	1 = Gage 9 Fault	1 = Gage 9 Warning	25
1 = Gage 16 Fault	1 = Gage 16 Warning	1 = Gage 15 Fault	1 = Gage 15 Warning	1 = Gage 14 Fault	1 = Gage 14 Warning	1 = Gage 13 Fault	1 = Gage 13 Warning	
1 = Gage 20 Fault	1 = Gage 20 Warning	1 = Gage 19 Fault	1 = Gage 19 Warning	1 = Gage 18 Fault	1 = Gage 18 Warning	1 = Gage 17 Fault	1 = Gage 17 Warning	26
1 = Gage 24 Fault	1 = Gage 24 Warning	1 = Gage 23 Fault	1 = Gage 23 Warning	1 = Gage 22 Fault	1 = Gage 22 Warning	1 = Gage 21 Fault	1 = Gage 21 Warning	
1 = Gage 28 Fault	1 = Gage 28 Warning	1 = Gage 27 Fault	1 = Gage 27 Warning	1 = Gage 26 Fault	1 = Gage 26 Warning	1 = Gage 25 Fault	1 = Gage 25 Warning	27
1 = Gage 32 Fault	1 = Gage 32 Warning	1 = Gage 31 Fault	1 = Gage 31 Warning	1 = Gage 30 Fault	1 = Gage 30 Warning	1 = Gage 29 Fault	1 = Gage 29 Warning	

Accessing Discrete Bit Information (cont'd)

Table 4.B
CVIM Module Remote I/O Outputs (PLC to CVIM Module)if CVIM Module is Rack 02

BIT

07 17	06 16	05 15	04 14	03 13	02 12	01 11	00 10	WORD
(Reserved)***	(Reserved)***	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	20
(Not Used)	1 = Config Transfer	1 = I/O Request	1 = Light pen Request	1 = Trigger Toolset 2	1 = Trigger Toolset 1	1 = Unlock Setup	1 = Lock Setup	
(Not Used)	1 = Display Stat 2 Page	1 = Display Stat 1 Page	1 = Display Results Page	1 = Display I/O Page	1 = Display All Tools	1 = Display Failed Tools	1 = Display Image Only	21
(Not Used)	(Not Used)	(Not Used)	Halt on Reject	1 = Freeze Next Image	1 = Freeze All Rejects	1 = Freeze First Reject	Go on reject	
1 = Enable JMB Forces	1 = Disable JMB Forces	1 = Enable JMB Outputs	1 = Disable JMB Outputs	(Not Used)	(Not Used)	1 = Post TS2 to Remote I/O	1 = Post TS1 to Remote I/O	22
1 = Credit Card Config. (8's bit)****	1 = Credit Card Config. (4's bit)****	1 = Credit Card Config. (2's bit)****	1 = Credit Card Config. (1's bit)****	1 = RAM to Credit Card	1 = Credit Card to RAM	1 = RAM to EEPROM	1 = EEPROM to RAM	
(Not Used)	(Not Used)	1 =Toolset 2 Request Results Block	1 = Toolset 1 Request Results Block	1 = Last Block (write Only)	Block Transfer Type *	Block Transfer Type *	Block Transfer Type *	23
1 = Block Trnsfer Block No. (128's bit)	1 = Block Trnsfer Block No. (64's bit)	1 = Block Trnsfer Block No. (32's bit)	1 = Block Trnsfer Block No. (16's bit)	1 = Block Trnsfer Block No. (8's bit)	1 = Block Trnsfer Block No. (4's bit)	1 = Block Trnsfer Block No. (2's bit)	1 = Block Trnsfer Block No. (1's bit)	
1 = Force JMB Output 8 ON**	1 = Force JMB Output 7 ON**	1 = Force JMB Output 6 ON**	1 = Force JMB Output 5 ON**	1 = Force JMB Output 4 ON**	1 = Force JMB Output 3 ON**	1=Force JMB Output 2 ON**	1=Force JMB Output 1 ON**	24
(Not Used)	(Not Used)	1 = Force JMB Output 14 ON**	1 = Force JMB Output 13 ON**	1 = Force JMB Output 12 ON**	1 = Force JMB Output 11 ON**	1 = Force JMB Output 10 ON**	1 = Force JMB Output 9 ON**	
1 = Force JMB Output 8 OFF**	1 = Force JMB Output 7 OFF**	1 = Force JMB Output 6 OFF**	1 = Force JMB Output 5 OFF**	1 = Force JMB Output 4 OFF**	1 = Force JMB Output 3 OFF**	1 = Force JMB Output 2 OFF**	1 = Force JMB Output 1 OFF**	25
(Not Used)	(Not Used)	1 = Force JMB Output 14 OFF**	1 = Force JMB Output 13 OFF**	1 = Force JMB Output 12 OFF**	1 = Force JMB Output 11 OFF**	1 = Force JMB Output 10 OFF**	1 = Force JMB Output 9 OFF**	
1 = Reset Counters	1 = Reset Stats	1 = Page Down	1 = Page Up	1 = Resume Control	(Not Used)	1 = Display Toolset 2	1 = Display Toolset 1	26
(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	
(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	27
(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	

* Set these three bits to specify the type of block as follows: 001 = Results, 010 = Configuration, 100 = Template, 101 = Statistics, 111 = Programmable Results Block Write
** If both ON & OFF bits are set, the output is forced OFF.
*** Do not write to these bits.
**** The first configuration on the card is 0000.

Example Program for Accessing/Setting Discrete Bit Data

The following ladder logic program provides examples of:

- Triggering an inspection from a PLC.
- Enabling/Disabling the user access to the setup mode using the lightpen.
- Checking for valid results.
- Reading and displaying pass/fail/warning tool results (Window 1, Toolset 1).
- Controlling screen display from a PLC.

The program assumes that the CVIM module is located in rack 02 (processor address is 074 octal) and the PLC is in rack 00.

```
                                                  31 December 1989    Page 1
Ladder Listing              Processor File: CVIM.ACH                Rung 2:0

Rung 2:0
Specify Toolset 1 for remote I/O data – either this bit or 0:22/01 must be set for
the PLC to receive results
|                                                                    Post TS1   |
|                                                                    results to |
|                                                                     REM I/O   |
|                                                                      O:022    |
+-----------------------------------------------------------------------( )-----+
|                                                                        00     |

Rung 2:1
This rung acquires an image, the CVIM one shots the input (F to T transition)
| Trigger                                                                       |
| Trigger TS1                                                        Trigger    |
|                                                                     Cam 1     |
|   I:010                                                              O:020    |
+----] [----------------------------------------------------------------( )-----+
|       02                                                               12     |

Rung 2:2
The next two rungs control enable or disable the lightpen from entering the setup
mode on the black and white monitor. A keyswitch can be used here.
|                                                                               |
| Lock                                                               Disable    |
| Setup                                                               Setup     |
|     I:010                                                             O:020   |
+----] [----------------------------------------------------------------( )-----+
|       00                                                               10     |

Rung 2:3
|                                                                               |
| Unlock                                                              Enable    |
| Setup                                                               Setup     |
|     I:010                                                             O:020   |
+----] [----------------------------------------------------------------( )-----+
|       01                                                               11     |
```

Example Program For Accessing/Setting Discrete Bit Data (cont'd)

```
                                                 31 December 1989          Page 1
Ladder Listing           Processor File: CVIM.ACH                        Rung 2:4

Rung 2:4
When Data Valid bit is high, read the discrete results for window 1, then
light the correct status light.
| Data Valid                                                                    |
| Toolset 1                                    Window 1  |Window 1  |Pass       |
|                                              warning   |Fault     |light      |
|   I:020                                      I:021      I:021      O:010      |
+----] [------------------------------------+---]/[--------]/[--------( )-----+-+
|      06                                   |     00         01         10    | |
|                                           |Window 1              Fail       | |
|                                           |Fault                 light      | |
|                                           |  I:021                 O:010    | |
|                                           +---] [-------------------( )-----+ |
|                                           |     01                    11    | |
|                                           |Window 1              Warning    | |
|                                           |warning               light      | |
|                                           |  I:021                 O:000    | |
|                                           +---] [-------------------( )-----+ |
|                                                 00                    12      |

Rung 2:5
The next four rungs control the monitor display screen through the PLC link.
| Display                                                                       |
| All Tools                                                       All Tools     |
|     I:011                                                            O:021    |
+----] [----------------------------------------------------------+---(L)-----+-+
|       00                                                        |     02    | |
|                                                                 |I/O Page   | |
|                                                                 |  O:021    | |
|                                                                 +---(U)-----+ |
|                                                                 |     03    | |
|                                                                 |All Tools  | |
|                                                                 |  O:021    | |
|                                                                 +---(U)-----+ |
|                                                                       04      |

Rung 2:6
| Display                                                                       |
| I/O                                                             All Tools     |
|     I:011                                                         0:021       |
+----] [----------------------------------------------------------+---(U)-----+-+
|       01                                                        |     02    | |
|                                                                 |I/O        | |
|                                                                 |  O:021    | |
|                                                                 +---(L)-----+ |
|                                                                 |     03    | |
|                                                                 |All Tools  | |
|                                                                 |  O:021    | |
|                                                                 +---(U)-----+ |
|                                                                       04      |
```

Example Program For Accessing/Setting Discrete Bit Data (Cont'd)

```
                                                    31 December 1989      Page 1
Ladder Listing                 Processor File: CVIM.ACH                  Rung 2:7
Rung 2:7
| Display                                                           All Tools   |
| Results                                                                       |
|   I:011                                                             O:021     |
+----] [----------------------------------------------------------+---(U)----+-+
|       02                                                        |      02   | |
|                                                                 |I/O        | |
|                                                                 |  O:021    | |
|                                                                 +---(U)----+ |
|                                                                 |      03   | |
|                                                                 |All Tools | |
|                                                                 |  O:021    | |
|                                                                 +---(L)----+ |
|                                                                        04     |

Rung 2:8
This rung is equivalent to pressing the lightpen on the monitor screen.
|  Display                                                           Lightpen   |
|  All Tools                                                         Request    |
|      I:011                                                           O:020    |
+-+---] [----+----------------------------------------------------------( )-----+
| |       00  |                                                            14    |
| |Display    |                                                                 |
| |I/O        |                                                                 |
| |  I:011    |                                                                 |
| +---] [----+                                                                  |
| |       01  |                                                                 |
| |Display    |                                                                 |
| |Results    |                                                                 |
| |   I:011   |                                                                 |
| +---] [----+                                                                  |
|         02                                                                    |

Rung 2:9
|                                                                               |
+---------------------------------[END OF FILE]----------------------------------+
|                                                                               |

NO MORE FILES
```

Accessing Results and Configuration Information

A host also has access to actual results block information such as measured lengths, number of black pixels, etc. Transfer of result and configuration data is accomplished using block transfers. There are three types of blocks that can be transferred:

- Results Blocks
- Configuration Blocks
- Template Blocks

Depending upon the source and destination of the data blocks, the following transfers can be made:

Reading Results (CVIM module to SYS Host)

- Results Blocks. There are four inspection results blocks (refer to Appendix A). Three of these blocks have a preconfigured structure. You can configure the fourth block so that only the information you require is transferred. The fourth (configurable) block can be only accessed through the remote I/O port.

Transferring Configurations (CVIM module to CFG HOST and/or CFG HOST to CVIM module)

- Configuration Blocks. There are 135 configuration blocks which contain the CVIM module setup information, tool parameters, operating environment instructions, camera setups, I/O operation, and operating modes. Each block transfer is limited to 64 words maximum. You can request blocks one at a time or in groups. Refer to Appendix A (Overview) and D (Configuration Data) for a description of the configuration block data.

- Template Blocks (blocks 136 to n). These blocks (part of the configuration memory) are previously learned image templates not on-line configuration block data,

When transferring blocks of data with the CFG or SYS Hosts, note the following requirements:

- You should assign a length of 0 to all block transfer commands. This allows the CVIM module to specify the length of the block in words.

- All block transfers address the lowest Group and Module Locations (0). You must set the bits in output word 3 to designate function of Results, Configuration, or Template transfer.

- The SYS or CFG Host must initiate all block transfers.

Transferring Results Blocks

Results blocks are transferred using block transfer reads. These blocks contain inspection result information such as: tool results, fault data, etc. Of the four results blocks, three are pre–configured and one block is user configurable (refer to next section). This means that you can program the contents of the block to contain only the specific data you require. Before transferring a results block you must inform the CVIM module of the Block Transfer Type and Block Number by setting discrete bit information using simple ladder programming (refer to Table 4.B):

- Set bit 0 of output word 3 to indicate RESULTS block transfer.
- Use bit 4 of output word 3 to indicate Toolset 1.
  or
  Set bit 5 output word 3 to indicate Toolset 2.
- Use bits 10, 11, and 12 of output word 3 to indicate which of the four blocks to read. Refer to Appendix A and D.

ATTENTION: To ensure that your results data is current and valid, you should use programming logic which synchronizes the transfer of data when inspections occur.

- Use the Data Valid bits of input word 0 (bit 6 – toolset 1 or bit 7 – toolset 2) to detect when new inspection results are available. These bits are described in Chapter 3 (Local I/O). Or as an alternative, you can use the ”total number of triggers” data contained in the results block.

Note: Later in this chapter we provide an example PLC program for retrieving results data.

Configuring Results Block 4 and Statistics Block Formats

Both the programmable results block and statistic blocks are configured to contain user specified results. To configure the data in results block #4:

- Specify the information you want returned by setting the appropriate bits in the 10 word “programmable results / statistics block” in the PLC as shown in Table C.5 (Page C–15). For example, reference window 1 line gage, window 2, window 3, etc.
- If configuring a results block– Set bits 0, 1, and 2 of output word 3. This will set the CVIM module to receive the “program” for results block 4.
- If configuring a statistics block– Set bits 0 and 2 of output word 3. This will set the CVIM module to receive the “program” for the statistics block.
- Perform a Block Transfer Write to transfer the 10 word “program“ from the host to the CVIM module.
- If reading results block– Read results block 4 and check word 1 for error bits and words 2 through 63 for valid data.

Note: Refer to Appendix C and verify that your results will not require more than 62 words, this will ensure that the results will fit in the allocated block. The results are returned in the ascending order of their appearance in the programmable block (reference windows before windows, window 1 before window 2, etc.) It is the responsibility of the programmer to track the order and location of the data.

ATTENTION: The format information for the programmable results block and statistics blocks are stored in CVIM module RAM. The data does not get saved into the EEPROM with other configuration information. This means that the data will be lost when the power is turned off.

Converting Results Data

Some of the results data described in Appendix C is stored in a “16 point 16” format while other data is stored as a 32 bit integer. Refer to the following chart:

WINDOW	**FORMAT**
Luminance	**16.16***
Object	**32 bit**
Pixels	**32bit**
GAGE	**FORMAT**
Linear Measure	**16.16**
Object	**32 bit**
Pixels	**32 bit**
Edge	**32 bit**
Angular Measure	**16.16**
Light Probe	**16.16**
Reference Line	**16 bit**
Reference Window	**16 bit**
Reference Window Theta	**16.16**

*16.16 means that the first 16 bits indicate the integer and the second 16 bits the fraction (refer to Appendix A for more information). If you are transferring results data to a PLC, you may need to convert the “16 point 16” format to a PLC floating point number. You can convert results data using the following equation:

$$\text{PLC Floating Point Number} = \text{Integer} + \frac{\text{Fraction}}{65536.0}$$

The following example assumes that you are converting a “16 point 16” value of 2.75. The value 2.75 is stored as follows:

Bit#		15**	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	←PLC
Integer → N7:1 = 2	=	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
Fraction → N7:2 = .75	=	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

** This bit is the sign bit in PLC integer files (1 = Negative 0 = Positive)

To help you, we have provided the following sample program. The program begins on the next page.

Converting Results Data (cont'd)

```
                                                                Page 1
Ladder Listing            Processor Address: 000 octal        Rung 2:0

   Rung 2:0
   This program converts CVIM 16.16 to PLC Floating Point.


|  B3                                          +MOV----------------+
+--] [----------------------+------------------+MOVE               +------+-
|     1                     |                  |Source       N7:41|        |
|                           |                  |            -16384|        |
|                           |                  |Dest          F8:0|        |
|                           |                  |      49152.000000|        |
|                           |                  +-------------------+        |
|                           | +CMP---------------+ +CPT----------------+    |
|                           +-+COMPARE           +-+COMPUTE           +------+
|                           | |Expression        | |Dest          F8:0|     |
|                           | |F8:0 < 0.000000   | |      49152.000000|     |
|                           | +------------------+ |Expression        |     |
|                           |                      |F8:0 +            |     |
|                           |                      |65536.000000      |     |
|                           |                      +-------------------+    |
|                           |                      +DIV---------------------+ |
|                           +----------------------+DIVIDE                 +-+
|                           |                      |Source A          F8:0| |
|                           |                      |          49152.000000| |
|                           |                      |Source B  65536.000000| |
|                           |                      |                      | |
|                           |                      |Dest              F8:1| |
|                           |                      |              0.750000| |
|                           |                      +----------------------+ |
|                           |                      +MOV----------------+    |
|                           +----------------------+MOVE               +------+
|                           |                      |Source       N7:40|     |
|                           |                      |                 2|     |
|                           |                      |Dest          F8:2|     |
|                           |                      |          2.000000|     |
|                           |                      +-------------------+    |
                           +++
```

Converting Results Data (cont'd)

```
                           +++
¦                          ¦ +CMP---------------+ +CPT---------------+      ¦
¦                          +-+COMPARE           +-+COMPUTE           +------+
¦                          ¦ ¦Expression        ¦ ¦Dest          F8:1¦      ¦
¦                          ¦ ¦F8:2 < 0.000000   ¦ ¦          0.750000¦      ¦
¦                          ¦ +------------------+ ¦Expression        ¦      ¦
¦                          ¦                      ¦- F8:1            ¦      ¦
¦                          ¦                      +------------------+      ¦
¦                          ¦                      +ADD---------------+      ¦
¦                          +----------------------+ADD               +------+
¦                                                 ¦Source A      F8:2¦
¦                                                 ¦          2.000000¦
¦                                                 ¦Source B      F8:1¦
¦                                                 ¦          0.750000¦
¦                                                 ¦Dest          F8:3¦
¦                                                 ¦          2.750000¦
¦                                                 +------------------+

Rung 2:1
¦
+--------------------------------[END OF FILE]--------------------------------
¦
```

Transferring Configuration Blocks

You can transfer configuration block data between the CVIM module and CFG Host using block transfer reads and writes. These blocks contain the operating instructions for the CVIM module (refer to Appendix D). When transferring configuration blocks, note the following:

When the CVIM module is receiving configuration blocks from a CFG Host, the CVIM module will leave the active run mode, set the module busy bit, turn off local I/O, turn off the data valid bit, and ignore any input triggers (setup menu option is also disabled). After receiving one or more new configuration blocks (and the last block bit), the CVIM module will validate the entire configuration since many of the operating parameters are interrelated.

If the CVIM module detects an invalid configuration, the new configuration will be ignored and the CVIM module will set the Configuration Fault bit and operate using the old configuration. Note that this is true for TEMPLATE blocks. These blocks are described in the next section.

Transferring Configuration Blocks (cont'd)

You must use the discrete I/O bits in conjunction with block transfers to inform the CVIM module of the Block Transfer Type, Toolset Number, Block Number and, Last Block by setting discrete bit information using simple ladder programming (refer to Table 4.B):

- Set bit 1 of output word 3 to indicate a CONFIGURATION block transfer.
- Use bits 10 through 17 of output word 3 to indicate which block to transfer. Refer to Appendix D for block numbers.
- Set bit 3 in output word 3 to tell the CVIM to send the last block. If you forget to set this bit, the CVIM module will wait for an indefinite period of time for more data.

Note: Later in this chapter we provide an example PLC program for accessing configuration data.

Transferring Template Blocks

Part of the configuration memory is reserved for blocks of data which contain previously stored image information when using reference windows. These blocks are referred to as template blocks. Template blocks can be accessed like configuration blocks with some differences:

- A template may require different amounts of memory depending upon the size of the template and the complexity of the feature.
- Total memory storage may require up to 100 (64 word) blocks of memory.
- You **may not** alter template data, you should only upload and download the data between the CVIM and a host.
- You must keep the complete template memory intact. You may not transfer a single template by itself.
- When template data is being tansferred (to or from) the CVIM module, the CVIM module will exit the active mode and ignore incoming triggers. The CVIM will also assert the module busy bit.

Transferring Template Blocks (cont'd)

Word 1, bits 8 – 15 of the first template block indicate the total number of template blocks of the configuration. You must always upload or download *all* of the template blocks as a unit. You cannot archive only a part of the template blocks. When uploading templates from the CVIM module, the program should read the first template block and check word 1, bits 8– 15 to determine the number of template blocks to follow. The number of blocks remaining is 1 less than the total number of template blocks. When downloading templates to the CVIM module, the program must send all template blocks. Bit 8 – 15 of word 1 determine the number of blocks to send:

Note: An error in downloading templates will cause the loss of all templates presently stored in CVIM module RAM.

Example Program for Accessing Results Data

The following program provides an example of using continuous block transfer to detect and acquire new data (Reference line#1 X–position) after an inspection is triggered. The program then counts the total number of times new results were obtained by the PLC. The program assumes that the CVIM module is rack 02 and the push buttons are rack 01.

Note: This is not the most efficient method to accomplish this function. A faster method is to connect the data valid bit output on the 1771-JMB board to a PLC input (refer to Figure 4.9). You can then use the valid bit output to trigger a single read. Refer to Chapter 3 for a description of the data valid bit.

The program has the following structure:

1. Waits for push button trigger.
2. Reads present number of total triggers before sending trigger request to the CVIM module.
3. Triggers the CVIM module.
4. Continues to read total number of triggers to detect new data.
5. Retrieves new data.
6. Program waits for next push button trigger.

The program begins on the next page.

Example Program for Accessing Results Data, Cont’d

```
                                                 31 December 1989         Page 1
Ladder Listing              Processor File: CVIMBLK.ACH                 Rung 2:0

Rung 2:0
Block Transfer Results, Toolset 1, Results Block 1.
|                                                                  Block       |
|                                                                  Transfer    |
|                                                                  Results     |
|                                                                    O:023     |
+-----------------------------------------------------------------+---(L)----+-+
|                                                                 |     00   | |
|                                                                 |Specify   | |
|                                                                 |Toolset   | |
|                                                                 |  O:023   | |
|                                                                 +---(L)----+ |
|                                                                 |     04   | |
|                                                                 |Block 1 of| |
|                                                                 |4 Types   | |
|                                                                 |  O:023   | |
|                                                                 +---(L)----+ |
|                                                                       10     |

Rung 2:1
Push Button Input to Trigger a CVIM Inspection.
| I:010  B3                                                            B3      |
+----] [--]ONS[--------------------------------------------------------( )-----+
|       02      4                                                       3      |

Rung 2:2
Flags for First Block Read, and Continuous Block Read Until New Results.
|       B3                                                             B3      |
+----] [----------------------------------------------------------+----(L)---+-+
|       3                                                         |       2  | |
|                                                                 |     B3   | |
|                                                                 +---(L)----+ |
|                                                                        1     |
|
Rung 2:3
Read Results Block #1 From CVIM.
|     B3                                        +BTR--------------------+      |
+----]ONS[--------------------------------------+BLOCK TRNSFR READ      +-(EN)-+
|         2                                     |Rack                  2|      |
|                                               |Group                 0+-(DN) |
|                                               |Module                0|      |
|                                               |Control Block    N7:100+-(ER) |
|                                               |Data file          N7:0|      |
|                                               |Length                0|      |
|                                               |Continuous            Y|      |
|                                               +-----------------------+      |

Rung 2:4
Clear Block Transfer Read Error, If It Occurs.
| BTR Error                                                         BTR Enable |
|   N7:100                                                           N7:100    |
+----] [---------------------------------------------------------------(U)-----+
|       12                                                              15    |
```

Example Program for Accessing Results Data, Cont’d

```
                                                 31 December 1989         Page 2
Ladder Listing              Processor File: CVIMBLK.ACH                 Rung 2:5

Rung 2:5
Set n7:70 to the ”Total Triggers” Just Before Initiating this Inspection.
| BTR Done   |First Read                                                       |
|   N7:100     B3                                         +FLL---------------+ |
+----] [-------] [---------------------------------------|-+FILL FILE        +++
|      13        1                                      | |Source      N7:63|||
|                                                       | |Dest       #N7:70|||
|                                                       | |Length          1|||
|                                                       | +-----------------+||
|                                                       |  B3               ||
|                                                       +--(U)---------------+|
|                                                       |    1               ||
|                                                       | CVIM Trig          ||
|                                                       |   0:020            ||
|                                                       +--(L)---------------+|
|                                                             12               |

|Rung 2:6
After the Requested Inspection is Done, This Rung will Detect the ”Total Trigger”
Value Incrementing, the New Results will be Available.
| +NEG---------------+                                                B3       |
+-+NOT EQUAL         +----------------------------------------------+---(L)---++
| |Source A     N7:63|                                              |     15  ||
| |              6824|                                              |         ||
| |Source B     N7:70|                                              |         ||
| |                 0|                                              | 0.020   ||
| +------------------+                                              +---(U)---+|
|                                                                       10     |

Rung 2:7
This Rung Counts Total Number of Times New Results were Received by PLC Since Reset
Button was Pushed.
|  B3                                                    +CTU---------------+  |
+--] [--------------------------------------------------++Count Up      +-(CU)++
|      15                                               ||Counter   C5:1|     ||
|                                                       ||Preset       0+-(DN)||
|                                                       ||Accum       30|     ||
|                                                       |+--------------+     ||
|                                                       |  B3                 ||
|                                                       +--(U)----------------+|
|                                                       |      2              ||
|                                                       |  B3                 ||
|                                                       +--(U)----------------+|
|                                                             15               |
```

Example Program for Accessing Results Data, Cont'd

```
                                                 31 December 1989         Page 3
Ladder Listing              Processor File: CVIMBLK.ACH                 Rung 2:8

Rung 2:8
This Rung Resets All Flags and the Counter.
|   I:011                                                    C5:1              +
+----] [-------------------------------------------------+-(RES)-------------+-+
|      17                                                |                   | |
|                                                        |                   | |
|                                                        |                   | |
|                                                        | B3                | |
|                                                        +-(U)---------------+ |
|                                                        |    1              | |
|                                                        |  B3               | |
|                                                        +--(U)--------------+ |
|                                                        |    2              | |
|                                                        |  B3               | |
|                                                        +--(U)--------------+ |
|                                                        |    15             | |
|                                                        | 0:020             | |
|                                                        +--(U)--------------+ |
|                                                        |    12             | |
|                                                        | N7:100            | |
|                                                        +--(U)--------------+ |
|                                                        |    13             | |
|                                                        |+FLL-------------+ | |
|                                                        ++FILL FILE       +-+ |
|                                                         +Source     N7:63+   |
|                                                         +Dest      #N7:70+   |
|                                                         +Length         1+   |
|                                                         +----------------+   |

Rung 2:9
|                                                                              |
+---------------------------------[End of File]--------------------------------+
|                                                                              |
```

Example Program For Accessing Configuration Data

The following program provides an example of using bi–directional block transfers to:

- Transfer CVIM module configuration data to a PLC.
- Modify the data. In this program we move the location of Window 1, Toolset 1 up or down.
- Transfer the reconfigured data back to the CVIM module from the PLC.

```
                                                     31 December 1989        Page 1
Ladder Listing               Processor File: CVIMCNFG.ACH                   Rung 2:0
Rung 2:0

Initialize CVIM for Configuring Block Transfers.  TS1, Configures Block 42 of 135.
                                                                 Block Xfer    |
                                                                 Config        |
|                                                                    O:023     |
+------------------------------------------------------------------+---( )----+-+
|                                                                  |     01   | |
|                                                                  |Toolset 1 | |
|                                                                  |  O:023   | |
|                                                                  +---( )----+ |
|                                                                  |     04   | |
|                                                                  |32’s BIT  | |
|                                                                  |  O:023   | |
|                                                                  +---( )----+ |
|                                                                  |     15   | |
|                                                                  |8’s BIT   | |
|                                                                  |  O:023   | |
|                                                                  +---( )----+ |
|                                                                  |     13   | |
|                                                                  |2’s BIT   | |
|                                                                  |  O:023   | |
|                                                                  +---( )----+ |
|                                                                  |     11   | |

Rung 2:1
| PB Request for Moving Window 1 Location Up on Screen (SUB).                  |
| UP                                                                  UP       |
|   I:010      B3                                                         B3   |
+----] [-----[ONS]-----------------------------------------------------(L)-----+
|      14       0                                                          1   |

Rung 2:2
| PB Request to Move Window 1 Location Down on Screen (ADD).                   |
| DOWN                                                                DOWN     |
|   I:010      B3                                                         B3   |
+----] [-----[ONS]-----------------------------------------------------(L)-----+
|      15       10                                                        11   |
```

Example Program For Accessing Configuration Data Cont'd.

```
                                                 31 December 1989        Page 2
Ladder Listing              Processor File: CVIMCNFG.ACH                 Rung 2:3

Rung 2:3
Read Present Configuration Data for Window 1 (42 words).
|  UP        |BTR EN     |BTW EN                  Window 1 DATA.                |
|      B3       N7:100     N7:110     N7:110    +BTR--------------------+       |
+-+---] [----+---]/[--------]/[--------] [------+BLOCK TRNSFR READ      +-(EN)-+
| |      1   |     15         15         13     |Rack                 02|       |
| |          |                                  |Group                 0+-(DN) |
| |DOWN      |                                  |Module                0|       |
| |    B3    |                                  |Control Block    N7:100+-(ER) |
| +---] [----+                                  |Data file          N7:0|       |
|        11                                     |Length                0|       |
|                                               |Continuous            N|       |
|                                               +-----------------------+       |

Rung 2:4
Move Window 1 Location UP (SUB) or DOWN (ADD) 20 Pixels on Screen.
| BTR DN    |UP                                                                 |
|   N7:100       B3                                       +SUB----------------+ |
+----] [--------] [---------------------------------------+SUBTRACT           +-+
|      13          1                                      |Source A     N7:12| |
|                                                         |                 60| |
|                                                         |Source B         20| |
|                                                         |                   | |
|                                                         |Dest         N7:12| |
|                                                         |                 60| |
|                                                         +-------------------+ |

Rung 2:5
| BTR DN    |DOWN                                                               |
|   N7:100       B3                                       +ADD----------------+ |
+----] [--------] [---------------------------------------+ADD                +-+
|      13          11                                     |Source A     N7:12| |
|                                                         |                 60| |
|                                                         |Source B         20| |
|                                                         |                   | |
|                                                         |Dest         N7:12| |
|                                                         |                 60| |
|                                                         +-------------------+ |
                                           31 December 1989        Page 3
Ladder Listing            Processor File: CVIMCNFG.ACH            Rung 2:6

Rung 2:6
Write New Configuration Data to CVIM, Reset for Next Request to Move Window 1
Location ”Last Block” Bit Must be Sent to Inform CVIM to Revalidate and Run!!!!!!!
| BTR DN    |BTR EN    |BTR EN                                                |
|   N7:100     N7:100    N7:110      I:020     +BTW---------------------+      |
+----] [--------]/[--------]/[--------] [-----++BLOCK TRNSFR READ      +-(EN)+-+
|      13         15         15         03    ||Rack                  2|     | |
|                                             ||Group                 0+-(DN)| |
|                                             ||Module                0|     | |
|                                             ||Control Block    N7:110+-(ER)| |
|                                             ||Data file          N7:0|     | |
|                                             ||Length                0|     | |
|                                             ||Continuous            N|     | |
|                                             |+-----------------------+     | |
|                                             |  UP                          | |
|                                             |      B3                      | |
|                                             +-------(U)--------------------+ |
|                                             |           1                  | |
|                                             |  DOWN                        | |
|                                             |      B3                      | |
|                                             +-------(U)--------------------+ |
|                                             |           11                 | |
|                                             |  BTN DN                      | |
|                                             |   N7:100                     | |
|                                             +-------(U)--------------------+ |
|                                             |           13                 | |
|                                             |  Last Block                  | |
|                                             |   O:023                      | |
|                                             +-------(L)--------------------+ |
|                                                        03                    |
```

Example Program For Accessing Configuration Data Cont'd.

```
                                                 31 December 1989          Page 4
Ladder Listing              Processor File: CVIMCNFG.ACH                 Rung 2:7
Rung 2:7
Reset PB Requests, BTR BN Bit, Window 1 Data, Last Block Specifier.
| Reset PB                                          UP                           |
|   I:011                                            B3                          |
+----] [------------------------------------- +-------(U)---------------------+ |
|                                             |         1                      | |
|                                             |  DOWN                          | |
|                                             |      B3                        | |
|                                             +-------(U)---------------------+ |
|                                             |         11                     | |
|                                             |  BTR DN                        | |
|                                             |   N7:100                       | |
|                                             +-------(U)---------------------+ |
|                                             |         15                     | |
|                                             |  Last Block                    | |
|                                             |   O:023                        | |
|                                             +-------(U)---------------------+ |
|                                             |         03                     | |
|                                             |         Window 1               | |
|                                             |         Y Location             | |
|                                             | +FLL---------------------+    | |
|                                             +-+FILL FILE               +----+-+
|                                             | |Source                 0|    | |
|                                             | |Dest               #N7:0|    | |
|                                             | |Length                42|    | |
|                                             | +------------------------+    | |
|                                             |     N7:110                     | |
|                                             +-------(L)---------------------+ |
|                                             |         13                     | |

Rung 2:8
|                                                                               |
+--------------------------------[END OF FILE]----------------------------------+
|                                                                               |
NO MORE FILES
```

Example 6008–SI Program

The following program was written using Microsoft® C Version 5.10 with an Allen–Bradley 6008–SI Series B card. The program will:

- Prompt the user for the 6008–SI card address. This address is determined by the DIP switch settings on the card.
- Prompt the user for the 6008–SI card interrupt control line. This is determined by the jumper setting on the board itself.
- Initialize the 6006SI card and prompts the user for the CVIM module rack address (0–7). The CVIM module address was configured on the CVIM module monitor using the light pen.
- Display a five item menu which allows the user to perform the following functions:
  1. Trigger Toolset 1. This initiates an inspection cycle.
  2. Read Results Toolset 1. Reads the 128 discrete input bits.
  3. Read Configuration. Uploads the entire CVIM module configuration including template data.
  4. Write Configuration. Downloads the entire CVIM module configuration including template data.
  5. Quit.

Example 6008–SI Program (cont'd)

```
/* CVIM to 6008–SI sample communications program     */
/* Copyright Allen–Bradley      1–12–90         jrm,    */

/* This program was compiled using Microsoft®C Version 5.1 */
#include <stdio.h>
#include <stdlib.h>

                         /* Include the 6008–SI definitions */
#include <h_6008si.h>

#define TRIGGER_1_BIT 0x0400

                         /* define storage for configuration data */
unsigned config[135][64], configlen[135], template[256][64], templen[256];

void main()
    {

    QMR mr_pkt;
    unsigned segment;     /* segment of 6008–SI card
    unsigned status, err, CVIM_rack, block_num, numblocks, block1;
    unsigned block2, last_blk, x, t;
    int op_num, block_1;

            /* Prompt – enter address 6008–SI card */
    printf (”\n\n\nCVIM to 6008–SI communications sample program\n\n”);
    printf (”Enter hex RAM address for 6008–SI card (e.g. 0x000): ”);
    scanf (”%x”, &segment)

            /* initialize the 6008–SI */
    status = setup_6008(baud, 1, l, segment, &mr_pkt);
    if (status != OK)
        {
        printf (”Setup failed: command=%s, status=%s\n”,
            xlat_cmd(status), xlat_conf(mr_pkt.qmr_stat));
        if (status != C_AUTOCONF && status != C_SETUP)
            printf (”Scanner fatal error %d\n”, fatal_6008());
        abort();
        }

            /* Place scanner in RUN mode */
    mr_pkt.qmr_data[0] = CM_RUN;
    status = mr_wait (C_SETMODE, &mr_pkt);
    if (status != OK)
        {
        printf (”Setup failed: command=%s, status=%s\n”,
        xlat_cmd(status), xlat_conf(mr_pkt.qmr_stat));
        if (status != C_AUTOCONF && status != C_SETUP)
            printf (”Scanner fatal error %d\n”, fatal_6008());
        abort();
        }

    /* Disable host watchdog.  For sample program ONLY ––
       not recommended for any application programs. */
    host_active(–1);
```

```
        /* Get CVIM rack address from the user */
printf (”Enter CVIM remote–I/O rack number (0–7):  ”);
scanf (”%d”, &CVIM_rack);

g_oit[8*CVIM_rack + 2] |= 0x0001; /* post tool results */

        /* Start of main loop */
do
{
    printf (”\n\nOperations: \n\n”);
    printf (”1. Trigger Tool Set 1\n”);
    printf (”2. Read Results, Toolset 1\n”);
    printf (”3. Read Configuration\n”);
    printf (”4. Write Configuration\n”);
    printf (”\nEnter operation number (1–4) or –1 to quit: ”);

    scanf(”%d”, &op_num );   /* Convert user string input to a number */
    err = 0;
    switch (op_num)
        {

        case 1:         /* trigger tool set 1 */
            {
            g_oit[8*CVIM_rack] |= TRIGGER_1_BIT; /* set trigger bit to 1/*
            for (t=0; t<5000; t++);
            g_oit[8*CVIM_rack] &= ~TRIGGER_1_BIT; /* set trigger bit to 0 */
            err = g_op_stat & SO_FAULT;
            } break;

        case 2:         /* read discrete results toolset 1 */
            {
                        /* display all 8 input words in hex */
            for (x=0; x<8; x++)
                printf (”%04X ”, g_ipt[8*CVIM_rack + x]);
            printf (”\n”);
            err = g_op_stat & SO_FAULT;
            } break;

        case 3:         /* read configuration */
            {
                        /* read all config. blocks */
            for (block_num = 0; (block_num < 135) && !err; block_num++)
                err = get_CVIM_block (CVIM_rack, 2, block_num+1,
                          config[block_num], &configlen[block_num]);
             if (!err)
             {
                        /*read first template block */
              err = get_CVIM_block (CVIM)_rack, 4, 1, template[0],
                  &templen[0];
                        /* determine total no. of template blocks */
              numblocks = template[0][1] >> 8;

                        /* read remaining template blocks */
               for blocks_num = 1; (block_num < numblocks) && !err;
```

```
                     block_num++)
                    err = get_CVIM_block (CVIM_rack, 4 block_num+1,
                          template[block_num], &templen{block_num])
               }
               } break;

               case 4:         /* write configuration */
                   {
                               /* write all config. blocks */
                   for (block_num = 0; (block_num < 135) && !err; block_num++)
                       err = send_CVIM_block (CVIM_rack, 2, block_num+1,
                                 config[block_num], &configlen[block_num]);
                   }
                  } break;

                               /*send all template blocks */
                   numblocks = template[0][1] >> 8;
                   for (block_num= 0; (block_num < numblocks) && !err
                      block num++)
                   {
                     err = send_CVIM_block (CVIM_rack,
                         4 ! (block_num == numblocks–1 ? 8: 0),block_num);
                   }
                  }

                   /* wait until CVIM busy bit is low  */
                   for (t=65535; t>0 && (g_ipt[8*CVIM_rack] & 8); t––)
                       for (x=1; x<100; x++);
                   if (t==0)
                       {
                       printf (”Time–out error: CVIM busy\n”);
                       err = –1;
                       }
                   if (g_ipt[8*CVIM_rack] & 2)
                       printf (”Configuration ERROR.\n”);
                   else
                       printf (”Configuration validation OK.\n”);
                   } break;
               }              /* end switch (op_num) statement */

           if (err)
               printf (”Error code: %4x\n”,err);
           } while (op_num >= 0);

       stop_6008();          /* shut down 6008 before quitting */
       }


   int get_CVIM_block (CVIM_rack, block_type, block_num, data, length)
   unsigned CVIM_rack, block_type, block_num, *data, *length;

                  /* do a BTR (read) from the CVIM */
       {
       static QBT block_pkt;
       unsigned err, status,x;
```

```
            /* display msg for program monitoring */
    printf (”get_CVIM_block %d(%d)\n”,block_type, block_num);

            /* Tell CVIM block number and type */
    g_oit[8*CVIM_rack + 3] = block_type + block_num * 256;

            /* Initiate the block transfer read */
    block_pkt.qbt_len = 0;      /* request 0 words */
    status = bt_read(16*CVIM_rack,&block_pkt);
    err = (status != OK);
    if (!err)
    {
             /* wait for completion of BTR */
        while (!bt_done(&block_pkt));
        err = (block_pkt.qbt_stat != SC_OK);
        if (!err)
        {
             /* store the block data and length */
        *length = block_pkt.qbt_len;
        memcpy (data, block_pkt.qbt_data, *length * 2);
            }
        }
    return (err);
    }

int send_CVIM_block (CVIM_rack, block_type, block_num, data, length)
unsigned CVIM_rack, block_type, block_num, *data, *length;

            /* performs a BTW (write) to the CVIM */
{
    static QBT block_pkt;
    unsigned err, status,x;

            /* display msg for program monitoring */
    printf (”send_CVIM_block %d(%d)\n”,block_type, block_num);

            /* Tell CVIM block number and type */
    g_oit[8*CVIM_rack + 3] = block_type + block_num * 256;

            /* Initiate the block transfer write */
    block_pkt.qbt_len = *length;
    memcpy (block_pkt.qbt_data, data, *length * 2);
    status = bt_write(16*CVIM_rack,&block_pkt);
    err = (status != OK);

    if (!err)
        {
            /* wait for completion of BTW */
        while (!bt_done(&block_pkt));
        err = (block_pkt.qbt_stat != SC_OK);
        }
    return (err);
    }
```

Chapter 5

Using the RS-232 Ports

Chapter Objectives

In this chapter we describe how to:

- Connect RS–232 device(s) to the CVIM module.
- Obtain results data using ASCII or DF1 protocols.
- Upload and download configurations.

In addition, this chapter provides example programs.

RS–232 Communications

Using the RS–232 interface you can link a variety of devices to the CVIM module:

- Computers
- Operator Interfaces such as Allen–Bradley Industrial Computers and Terminals with serial ports.
- I/O modules such as the Basic Module (Catalog No. 1771–DB) or ASCII module (Catalog No. 1771–DA).
- Allen-Bradley DATAMYTE and Dataliner devices (requires USER-PAK Software, Catalog No. 5370-UPK).

All commands are simple ASCII and/or Hexadecimal strings. Refer to Appendix E for an ASCII conversion chart. These commands can be generated using a variety of programming languages (C, Fortran, BASIC). This chapter provides a sample ASCII program (written in BASIC) and a sample DF1 program (written in C).

ASCII and DF1 Protocols

There are two protocol options when you select an RS–232 communications port (A or B):

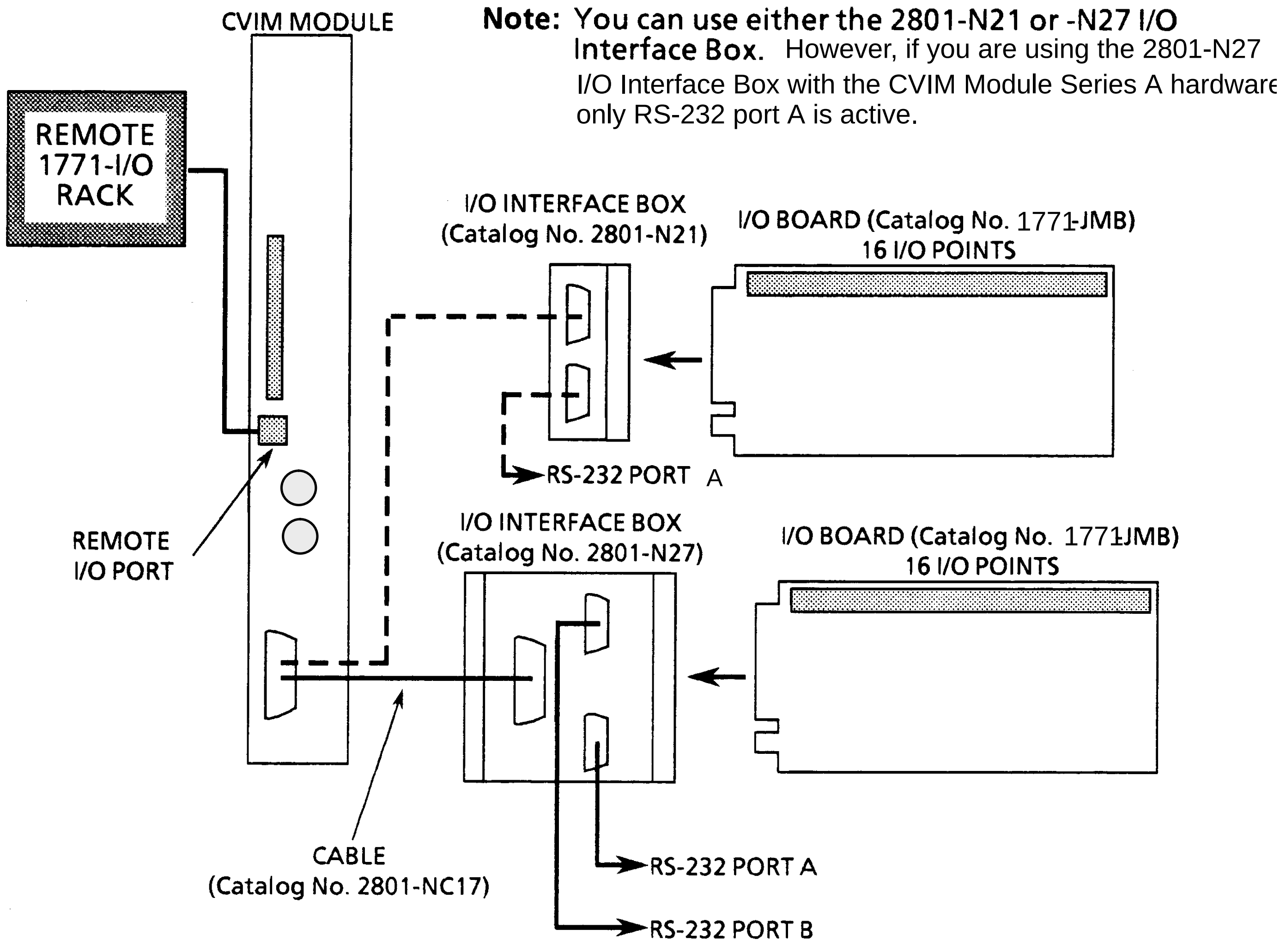
- ASCII
- DF1

This chapter describes both of these options. First we describe the ASCII protocol (page 5–5) and then the DF1 protocol (page 5–29).

Equipment Connections

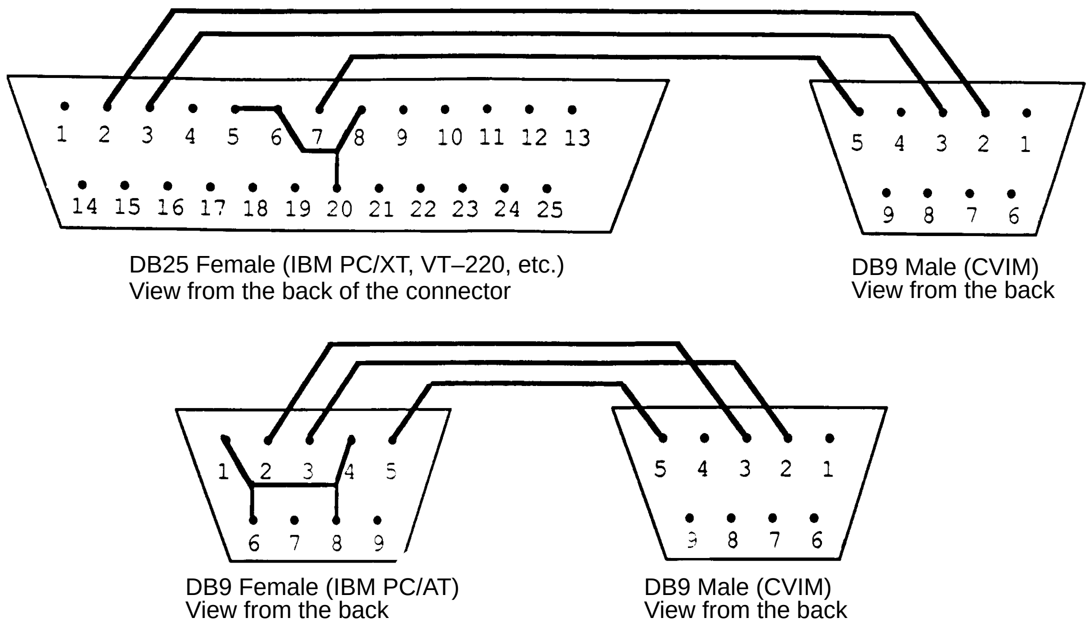
As shown in Figure 5.1, the RS–232 ports (A & B) are located on the I/O Interface Boxes (Catalog No. 2801–N21, –N27). The I/O Interface Box is connected to the MODULE I/O port on the front of the CVIM module. You will need a communications cable to link your host device to the CVIM module. Refer to Figure 5.2 for diagrams of host to I/O Interface Box cabling.

Figure 5.1
RS–232 Equipment Connections.


Note: You can use either the 2801-N21 or -N27 I/O Interface Box. However, if you are using the 2801-N27 I/O Interface Box with the CVIM Module Series A hardware only RS-232 port A is active.

CVIM MODULE
REMOTE 1771-I/O RACK
REMOTE I/O PORT
I/O INTERFACE BOX (Catalog No. 2801-N21)
I/O BOARD (Catalog No. 1771-JMB) 16 I/O POINTS
RS-232 PORT A
I/O INTERFACE BOX (Catalog No. 2801-N27)
I/O BOARD (Catalog No. 1771-JMB) 16 I/O POINTS
CABLE (Catalog No. 2801-NC17)
RS-232 PORT A
RS-232 PORT B

Figure 5.2
RS–232 Cabling.



Note: Connections for Catalog No. 2801–N27 I/O Interface Box RS–232 Port A with CVIM Series B Module is shown in this illustration. Refer to Chapter 3 for other RS–232 Connections.

What Functions can be performed over the RS–232 Interfaces?

A host device (SYS or CFG) can request or manipulate the following data through the RS-232 ports (A&B):

Obtain CVIM module results information. Refer to Appendix A, B and C (CFG or SYS host).

Upload or download CVIM module configurations for inspections. Refer to Appendix D (CFG host).

Issue Read/Write commands between the following CVIM module memory locations (CFG host):

CVIM module Random Access Memory (RAM) and CVIM module Electrically Erasable Programmable Read Only Memory (EEPROM).

CVIM module RAM and RAM card. The RAM card slides into a slot on front of the CVIM module.

Change run-time display (SYS host).

Enable/Disable local I/O board (SYS host).

Force local I/O On or Off (SYS host).

CVIM Module Configuration Instructions

If you are using the RS–232 ports (A or B), you must configure the CVIM module as follows:

Set the Baud Rate(s)

1 Select the setup menu <Setup>.
1 Select the environment menu <Environ>.
1 Select the I/O menu <I/O>.
1 Select RS–232 communications <RS–232 A> or <RS–232 B).
1 Select the Baud rate which matches your host device; from 300 to 19.2K Baud.

Note: When you select RS–232 communications, the data format is fixed as follows:

- 8 Data Bits
- 1 Stop Bit
  No Parity

Select the CFG and SYS Hosts

Note: The following steps are not necessary if you are just reading results data.

1 Select the setup menu <Setup>.
1 Select the environment menu <Environ>
1 Select the system menu <System>
1 Select a host menu <CFG Host> or <SYS Host>.
1 Select RS–232 port for host communications <RS–232A> or <RS–232B>.

Select the Protocol

1 Select the I/O menu <I/O>.
1 Select RS–232 communications <RS–232 A> or <RS–232 B>.
1 Select either <ASCII> or <DF1>.

Select the CVIM module Trigger Source

1 Select the toolset menu <Toolset>

1 Select the trigger source menu for the appropriate toolset <Trigger Source>.
1 Select either <I/O>, <Hosted>, or <Auto Trigger> trigger source. *Select hosted trigger if you are using the RS–232 trigger commands. Use I/O trigger if you are using the discrete I/O inputs as a trigger.*

Note: The next section of this chapter describes ASCII protocol followed by a description of DF1 protocol.

ASCII Protocol

In describing the ASCII Protocol we use the following conventions:

Non–printable ASCII control characters are represented as follows:

[CR] = Carriage Return
[LF] = Line Feed
___ = Space

ASCII commands are provided in large bold characters:

>RR, RB,3 [CR]

Unless _ is specified, there are no spaces between characters. Some commands have fields which can contain variable data such as number of times a command is repeated, block numbers, data, etc. These fields are shown using lowercase lettering:

>W,CBn,d [CR]

In this example, the letters **n** and **d** indicate data which is variable. The other characters indicate fixed data.

Overview

After you have made the equipment connections and configured the CVIM module for RS–232 communications, all ASCII strings generated by the host will be interpreted as commands. The CVIM module will then validate the command structure. If the command has an acceptable structure the CVIM module will reply: **[CR][LF].** Refer to Appendix E for an ASCII conversion chart. If the command has an incorrect structure the CVIM module will respond: ? **[CR] [LF].** The CVIM module will process all validated commands and discard any invalid commands. Data may or may not be returned with a command depending upon the type of command that was sent.

Note: A simple way to test the RS–232 links is to send the CVIM module a [CR]. If you have the port properly connected and the CVIM module configured for RS-232, the CVIM module should send a ? [CR][LF] in response. If no response is provided, check your connections and CVIM module configuration.

Note: Some commands cause a continuous flow of returned data. To stop the flow of data you should send another command (valid or invalid). We recommend using a [CR] to stop the transmission of data.

ASCII Character Set

The CVIM module recognizes the following ASCII characters; all other characters are ignored.

- Upper and lowercase letters A through Z (case is insignificant).

ASCII Character Set (cont'd)

- Symbols:

  > (greater than)
  * (star)
  , (comma)
  – (dash)
  (space) represented by __

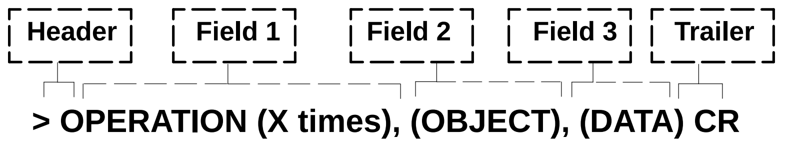
- Nonprintable control characters:

  CR (carriage return)
  LF (line feed)
  XON
  XOFF

- Numbers 0 through 9

Command Structure

Each command the host device sends to the CVIM module consists of an ASCII string of characters beginning with > and terminated with a **[CR].** Characters in between are separated into fields by commas. The following shows the structure of a typical command:



( ) Indicates Optional Information

Note: There are two modifiers that may appear in the command line:

x times modifier – This modifier is only used with certain commands to indicate the number of times the command is to be performed. The range for this value is between 0 and 255. A value of 0 indicates infinity. If you do not specify a value, a default of 1 is provided.

Toolset modifier – This modifier specifies either toolset 1 or toolset 2. **TS1** and **TS2** are the two valid entries. This modifier is only used to specify toolset dependent objects.

There are three types of fields:

Operation Field– This field contains commands directed to the CVIM module. There can only be one operation per command line. Some operations don't require any additional fields while others may require an object field, data field, or both. Note that some commands cannot be used while the CVIM module is in SETUP mode. If an operation cannot be performed because either the wrong host port has been selected or the CVIM module is in the SETUP mode, the CVIM module will respond to each command with **?[CR][LF]**.

Object Field– Object fields specify data that configures the operation of the CVIM module. There are two types of objects:

1) Toolset independent objects which do not require a toolset identifying number.

2) Toolset dependent objects which need a toolset identifying number.

The object field contains alphanumeric characters which specify one or more objects. Individual objects are specified by name. Multiple objects (of the same type) are specified with an "*" for all objects of this type or by using a "–" to indicate a range of objects.

In the description of each command we specify the objects that can be entered into a command.

- Data Field– Contains data.

XON/XOFF Flow Control

XON/XOFF characters control the flow of data between the CVIM module and the host. The XON character is transmitted by the receiving device to indicate that data can be transmitted. The XOFF character is transmitted when the receiving device cannot accept any more data (data buffers are filled). When the receiving device can accept more data, it sends another XON character. The following characters are used: XON =^Q (CTRL Q) XOFF =^S (CTRL S).

Deactivate Forces

Use the deactivate force command to return outputs on the 1771–JMB local I/O board to the CVIM module assigned functions. The deactivate forces command is:

>DF [CR]

After executing the command, the CVIM module will return: **[CR][LF].** No data is returned If you do not have the proper command structure, the CVIM module will return: **?(CR][LF].**

Echoing Data

Use the echo command to check the communications link. This command will return the same same string of characters that are sent out with the command. This command has the following structure:

>Ex,d [CR]

Where **x** specifies the number of times the CVIM module will echo the data field back to the host device. If you fail to specify an **x** value, a default value

Echoing Data (cont'd)

of 1 is assumed. **d** is the data that is to be echoed. The command is valid at any time.

For example:

>E2,HELLO [CR]

This example will cause the CVIM module to return the string:

[CR] [LF]
HELLO [CR] [LF]
HELLO [CR] [LF]

If you do not have the proper command structure the CVIM module will return:

?[CR] [LF]

Enable/Disable Outputs

Use this command to enable or disable outputs on the Local I/O Board (Catalog No. 1771–JMB). Use the following commands:

> EO [CR]	*This command enables the outputs.*
> DO [CR]	*This command disables the outputs.*

After executing the command, the CVIM module will return: **[CR] [LF]**. No data is returned. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**.

Forcing Local I/O

Use the force command to turn the local I/O outputs either on or off. This function can only be executed once per command. Use one of the following commands:

> F,On,1 [CR]	*Forces output(s) on.*
> F,On,0 [CR]	*Forces output(s) off.*

Where n is the output being forced on or off, outputs 1 through 14.

n = 1 to 14	*(individual outputs, can be non-consecutive)*
X – Y	*(range of outputs X through Y)*
*	*(all of the outputs)*

For example:

> F,O*,1 [CR]	*This example will force all outputs on.*

Another example:

> F,O3–9,0 [CR]	*This example forces outputs 3 through 9 off.*

For example:

> F,O4–6,1 [CR]	*Forces outputs 4–6 on.*
> F,O8,1 [CR]	*Forces output 8 on.*
>F,O1–4,0 [CR]	*Forces outputs 1–4 off.*

Notice that output #4 was forced on and then forced off. The force off takes precedence over the force on.

After executing a command, the CVIM module will return: **[CR][LF]**. If you do not have the proper command structure the CVIM module will return: **?[CR][LF]**. The outputs will remain in their forced states until a Deactivate Forces command is sent.

Loading Configurations

Use the load command to transfer configuration data to the CVIM module's RAM. Use one of the following commands:

> LO [CR]	*Transfers configuration from the EEPROM to the CVIM module internal RAM.*
> LO,CC,1 [CR]	*Transfers memory from the RAM Card area 1 memory to the CVIM module internal RAM.*
> LO,CC,2 [CR]	*Transfers memory from the RAM Card area 2 memory to the CVIM module internal RAM.*

This function can only be executed once per command. You cannot use this command when the CVIM module is in the SETUP mode.

After executing a command, the CVIM module will return: **[CR][LF]**. No data is returned by the command. If you do not have the proper command structure or the CVIM module is in the SETUP mode, the CVIM module will return: **?[CR][LF]**.

Lock Command

Use the lock command to disable the setup menu box so that the SETUP mode cannot be entered using the light pen. This function can only be executed once per command. There is no object associated with this command. The command has the following structure:

>L[CR]

After executing a command, the CVIM module will return: **[CR][LF].** No data is returned by the command. If you do not have the proper command

Lock Command (cont'd)

structure the CVIM module will return: **?[CR][LF].** Use the unlock command to enable the setup menu box.

Read Output Status

Use the read data command to read the status of the local I/O. This command has the following structure:

>Rx,On [CR]

Where n =	1 to 14	*(individual outputs)*
	X–Y	*(range of outputs X through Y)*
	*	*(all of the outputs)*

This function can be executed more than once per command by specifying an x times value.

For example:

> R,O14 [CR] *This example reads the status of output #14 once.*

Another example:

> R0,O*[CR] *This example continuously reads the status of all fourteen outputs.*

After executing a command, the CVIM module will return: **[CR][LF]** followed by the data. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**. The format of the requested data is an ASCII representation of the output state (**1** = ON and **0** = OFF). Each character is followed by a space. The output conditions are transmitted in numerical order (output #1 then #2, etc.). The number of characters returned depends upon the number of outputs that are read. Since there are fourteen outputs, up to 28 data characters can be returned. After the data is sent, the CVIM module will terminate the data with: **[CR][LF]**. The following is an example of returned data from the three outputs.

[CR][LF]1   0   0   [CR][LF]

Read Configuration Blocks

Use the read configuration command to read configuration data for the specified blocks (Upload Configurations). This command has the following structure:

>RC,CBn[CR]

Where n =	1 to 136	*(individual blocks)*
	X–Y	*(range of blocks X through Y)*
	*	*(all of the blocks)*

This function can only be executed once per command.

Refer to Appendix C for a description of the configuration blocks. You cannot use this command while the CVIM module is in the SETUP mode.

Examples:

>RC,CB135[CR]	*Reads configuration block 135.*
>RC,CB99,CB7,CB1[CR]	*Reads configuration blocks 1, 7, then 99.*
>RC,CB1–135[CR]	*Reads all the of configuration blocks (excluding templates).*
>RC,CB*[CR]	*Reads all the of configuration blocks (including templates).*

After executing a command, the CVIM module will return: **[CR][LF]** followed by the data. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**. The format of the requested data is an ASCII representation of the specified block(s) in bytes. Each byte is represented by two hexadecimal characters (00 through FF) followed by a space. The first two words are the signature word indicating block type and number (Refer to Appendix D). Twenty bytes of data are transmitted in a line terminated with a **[CR][LF]**. The size of the configuration block(s) determines the number of lines that are returned. The template data (CB136) is the only configuration block size that can exceed 128 bytes and therefore may require more than a single block to output the data.

Note: When you specify CB136, you are reading all of the template blocks. Word1, bits 8-15 of the first template block indicates the number of template blocks that are transmitted (all blocks except last block are 128 bytes long).

Refer to Appendix D for block description and sizes. The following is an example of how the returned data appears for command **>RC, CB-1–2 [CR]:**

Configuration Block Returned Data Format*

```
[CR] [LF]
48_01_02_00_00_00_00_00_44_65_66_61_75_6C_74_00_00_00_00_00_[CR][LF]
00_00_00_00_00_03_00_00_00_00_00_04_00_00_01_00_01_00_00_[CR][LF]
00_02_06_01_00_00_00_00_00_00_00_00_00_00_00_00_00_06_01_[CR][LF]
00_00_00_01_00_00_00_00_00_00_00_00_00_00_01_01_00_01_01–[CR][LF]
00_0D_FC_5F_9A_0A_00_19_FB_D1_[CR][LF]
                         Space Added Between Blocks for Clarity
48_02_FF_02_01_00_00_00_00_FA_00_10_00_01_01_00_00_00_00_00_[CR][LF]
00_00_00_3F_00_00_00_00_00_00_00_3F_00_00_00_00_00_00_00_64_[CR][LF]
00_32_01_2C_00_96_0A_62_00_01_00_00_00_01_00_00_00_01_00_00_[CR][LF]
00_01_00_00_00_01_00_00_00_01_00_00_00_64_00_32_01_2C_00_96_[CR][LF]
0A_62_00_01_00_00_00_01_00_00_00_00_80_00_00_00_00_00_00_00_[CR][LF]
00_00_00_00_00_00_00_01_00_00_00_00_00_00_00_00_00_00_00_01_[CR][LF]
00_00_00_00_00_00_49_CC_[CR][LF]
```

Read Inspection Results

Use this command to read the *results of the last inspection*. Refer to Appendix B for a description of the results blocks. Use the following commands:

>RRx,TSno,d [CR]

Where:	*x*	=	*Number of times command is repeated.*
	n	=	*Toolset number TS1, TS2, or S (CVIM module status)*
	o	=	*RL (specifies Reference Line)*
			RW (specifies Reference Window)
			G (specifies Gage)
			W (specifies Window)
			LP (specifies Light Probe)
	d	=	*Gage, Window, Reference Line, or Reference Window number.*

>RRx, TS1 [CR]	*Reads discrete bit results for toolset 1.* *x = Number of times command is repeated.*
>RRx,TS2 [CR]	*Reads discrete bit first results for toolset 2.* *x = Number of times command is repeated.*
>RRx,TS1RB,d [CR]	*Reads results block(s) for toolset 1* *x = Number of times command is repeated.* *d = Block number.*
>RRx,TS2RB,d [CR]	*Reads results block(s) for toolset 2.* *x = Number of times command is repeated.* *d = Block number.*
>RRx, S [CR]	*Reads CVIM module status.*

The read operation can be executed more than once per command by specifying an x times value. The data in the read results block commands indicate which results block (1, 2, 3, or 4) is being read (refer to Appendix C).

>RR0,TS1[CR]	*This command continuously reads the first discrete bit results for toolset 1. (24 bytes returned)*
>RR,TS2RB,3[CR]	*This command reads results block 3 for toolset 2. This operation is only performed once in this example. (128 bytes returned)*
>RRx,TS1RL,1[CR]	*Reads the results of toolset 1 reference line #1. (4 bytes returned)*
>RRx,TS2RW,3[CR]	*Reads the results of toolset #2 reference window #3. (28 bytes returned)*
>RRx,TS1G,21[CR]	*Reads the results of toolset #1 gage #21. (4 bytes returned)*
>RRx,TS2W,11[CR]	*Reads the results of Toolset #2 window #11. (4 bytes returned)*
>RRx,TS1LP[CR]	*Reads the results of toolset #1 light probe. (12 bytes returned)*
>RRx,S[CR]	*Reads the CVIM module status. (2 bytes returned)*

Note: Refer to Appendix B, Table B.1, RS–232 word 0 for a definition of CVIM module status.

After executing a command, the CVIM module will return: **[CR][LF]** followed by the data. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**. After reading the results, the CVIM module will return the requested data. The format of the requested data is in an ASCII representation of the specified block(s) in bytes.

If you requested results blocks, each byte is represented by two hexadecimal characters (00 through FF) followed by a space. Twenty bytes of data are transmitted in a line terminated with a **[CR][LF]**. Since the results blocks are 128 bytes in size, each block requires seven lines. Refer to Appendix C for block descriptions. The following is an example of the returned data format:

Read Inspection Results (cont'd)

Numerical Results Block Returned Data Format

```
[CR] [LF]
61_01_04_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00~00_[CR][LF]
00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00~00_[CR][LF]
00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_01_00_00_[CR][LF]
00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_32_00_00_00_00_[CR][LF]
00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_[CR][LF]
00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_[CR][LF]
00_00_00_00_00_00_00_CC_[CR][LF]
```

If you requested discrete bit information, the returned data will contain two counters and the discrete bit results. Each counter has 12 positions (10 characters, 2 spaces) reserved for a maximum value of 4,294,967,295.

Note: Counters are decimal values. All other fields are hexadecimal values.

The counter data is left justified and the remaining field is filled with spaces. The first counter contains the total number of triggers processed. The second counter contains the total number of faults. Both counters are expressed as decimal values. The results bit information (128 bits), which follows the counters, is 16 bytes long. Each byte is represented by two hexadecimal characters (00 through FF) followed by a space. The following is an example of the returned data format:

Discrete Bit Results Returned Data Format

```
[CR][LF]
1234567890__1234567890__80_00_00_00_00_00_00_00_00_00_00_00_00_00_00_00_[CR][LF]
```

Refer to Appendix B for a description of the returned bytes.

Save Configuration

Use the Save command to transfer CVIM module configuration data to the local storage area (EEPROM) or the external RAM card (credit card memory).

Note: Depending upon the card size, up to 16 configurations can be saved to the RAM card (512K card).

Use one of the following commands:

>S[CR]	*Transfers configuration data from the CVIM module RAM to the EEPROM.*
>S,CC,X [CR]	*Transfers configuration data from the CVIM module RAM to the RAM card area X (01 –16).*

For example:

>S,CC,13 [CR]	Transfers configuration data from the CVIM module RAM to the RAM card area 13.

You cannot use this command when the CVIM module is in the SETUP mode.

After executing a command, the CVIM module will return: **[CR][LF]**. No data is returned. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**.

Select Image Displayed

Use the display object commands to select the information that is displayed on the monitor:

>W,D,d [CR]

Where **d** is the data that specifies both the toolset and display to be viewed:

d = XY Where
X = 1 *(Toolset 1 displayed)* or
2 *(Toolset 2 displayed)*
Y = 1 (Image only displayed)
or 2 *(Failed tools displayed)*
or 3 *(All tools displayed)*
or 4 *(I/O page displayed)*
or 5 *(Results page displayed)*
or 6 *(Stats 1 page displayed)*
or 7 *(Stats 2 page displayed)*
or 8 *(Page up same display)*
or 9 *(Page down same display)*

>W,F,d [CR]

d = XY Where
X = 1 *(Toolset 1 displayed)* or
2 *(Toolset 2 displayed)*
Y = 1 (Go on reject)
or 2 *(Freeze on 1st reject)*
or 3 *(Freeze on all rejects)*
or 4 *(Freeze on next inspection)*
or 5 *(Halt on reject)*

>w,DC,d [CR]

d = XY Where
X = 1 *(Toolset 1 displayed)* or
2 *(Toolset 2 displayed)*
Y = 1 (Resume)
or 2 *(Reset stats)*
or 3 *(Reset counters)*
or 8 *(Page up)*
or 9 *(Page down)*

Select Image Displayed (cont'd)

Example:

>W,D,2[CR]	*This example will display toolset 1 failed tools.*

After executing the command, the CVIM module will return: **[CR][LF]**. No data is returned. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**.

Set Configurable Results

Use this command to obtain a configurable results block. The results you want are specified by a list of tools and placed in results block #4. No data is returned until you use a read inspection results command for block #4. Use the following command:

>SR,TSxd,TSxd, etc.[CR]

Where	x = 1 or 2 (specifies toolset #1 or #2)	
	d = G1, G2, G3, G1–G3, etc.	(specifies Gages)
	W1, W2, W3, W2–5, etc.	(specifies Windows)
	RL1, RL2, etc.	(specifies Reference Lines)
	RLW, RW2, etc.	(specifies Reference Windows)
	LP	(specifies Light Probe)

The returned results block will be 128 bytes including the block signature (2 bytes) and trigger counter (last 4 bytes). Refer to page C–14, the ordering of the tools and data lengths are the same as the Remote I/O configurable results block.

Example:

>SR,TS1G1,TS1W2–5[CR]	*This command places the results for gage 1 and Windows 2 through 5 in results block #4.*
>RR,TS1RB,4[CR]	*This command reads results block #4 for toolset 1.*

After executing the command, the CVIM module will return: **[CR] [LF]**. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**. Refer to Read Inspection Results command for a description of the returned data format.

Set/Read Configurable Statistics

Use the read command to read statistical data for the light probe, reference windows, gages, and windows. Use the separate set command to set the number of samples and configure the statistics block.

The set statistics command has the following structure:

>SSn,TSxd,TSxd,etc.[CR] *(Set command)*

Where n = Number of samples

Note: If n is 0, the CVIM module will continue to use the sample count configured during setup. Any other value will change the sample count.

Where	x = 1 or 2 (specifies toolset #1 or #2)	
	d = G1, G2, G3, G1–G3, etc.	(specifies Gages)
	W1, W2, W3, W2–5, etc.	(specifies Windows)
	RL1, RL2, etc.	(specifies Reference Lines)
	RLW, RW2, etc.	(specifies Reference Windows)
	LP	(specifies Light Probe)

The read statistics command has the following structure:

> RSn[CR] *(Read Statistics Command)*

Where n = Number of times statistics block is read.

Statistics are accumulated until the number of samples is reached, at which point the statistics begin to reaccumulate. The number of samples for each toolset are accumulated separately. For example, if the latest toolset specified is toolset #2, the statistics are accumulated based upon the number of triggers for toolset #2.

Examples of Set Statistics Command:

>SS50,TS1LP,TS1RW2[CR]	*This example sets the number of samples to 50,configures the block to contain light probe and reference window #2 statistics (both from toolset #1).*
>SS100,TS2G5,TS2W12[CR]	*This example sets the number of samples to 100, configures the block to contain gage #5 and window #12 statistics (both from toolset #2.*

Set/Read Configurable Statistics (cont'd)

Example of Read Statistics Command:

> RS5[CR] *This example reads the statistics block five times.*

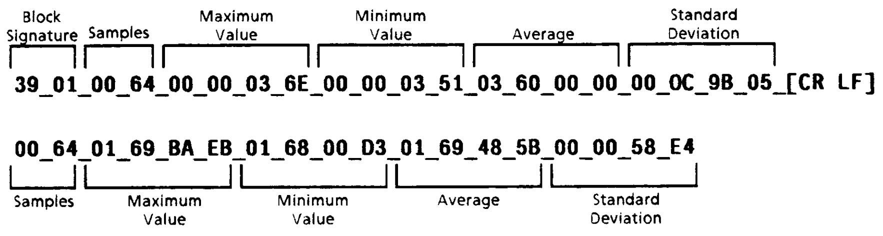
The data returned from the statistics block consists of:

- Block signature

  Number of samples, maximum, minimum, average, and standard deviation for each tool configured in the block.

The block signature is 2 bytes long. The number of samples is a 2 byte integer. The maximum and minimum values are each 4 bytes. The format of the data depends upon the operation (e.g. pixel count is an integer and linear gaging is a 16.16 fixed point value). Refer to page C–24 for data formats. Standard deviations are also 4 bytes each but are always 16.16 fixed point values. Averages are 24.8 fixed point values. Therefore, each tool statistic consists of 18 bytes with the exception of reference windows which contain 18 bytes for each feature or a total of 54 bytes. The statistics block is transmitted as two hexadecimal characters for each byte. The total number of bytes including the block signature should not exceed 128 bytes. The statistics block is read once for every number of specified samples. This means that if you read the statistics block five times with a sample number of 50, 250 triggers will have to be processed before the five reads are completed. The following shows the format of the returned data:

Statistics Block Returned Data Format



```
39_01_00_64_00_00_03_6E_00_00_03_51_03_60_00_00_00_0C_9B_05_[CR LF]

00_64_01_69_BA_EB_01_68_00_D3_01_69_48_5B_00_00_58_E4
```

Trigger Operation

Use the trigger operation command to initiate an inspection by the CVIM module. Use the following commands:

>T, TS1[CR]	*Triggers an inspection with toolset 1.*
>T, TS2[CR]	*Triggers an inspection with toolset 2.*

This function can only be executed once per command.

Note: When using this command you should make sure that the CVIM module is configured for a "hosted trigger source".

After executing a command, the CVIM module will return: **[CR][LF]**. No data is returned. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**.

Unlock Command

Use the unlock command to enable the setup menu box so that a user can access the SETUP mode using the light pen. Use the following command:

>U[CR]

This function can only be executed once per command. There is no object associated with this command. After executing a command the CVIM module will return: **[CR][LF]**. No data is returned. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**.

Write Configuration (W)
Write Configuration (WC)

Use the write command to write data to configuration memory (download configuration). Use the following commands:

>W,CBn[CR] d

or

>WC,CBn[CR] d

Where n =	1 to 136	*(individual blocks)*
	X – Y	*(range of blocks X through Y)*
	*	*(all of the blocks)*

d = the data that is being written. The format of the data is in an ASCII representation of the specified block(s) in bytes. Each byte is represented by two hexadecimal characters (00 through FF) followed by a space.

Note: The WC write command functions like the W write command but allows listing of configuration blocks.

Write Configuration (W) Write Configuration (WC) (cont'd)

This function can only be executed once per command.

Refer to Appendix D for a description of the configuration blocks. You cannot use this command when the CVIM module is in the setup mode. When the CVIM module is receiving configuration blocks from a Host, the CVIM module will leave the active run mode and ignore any input triggers (setup menu option is also disabled). After receiving one or more new configuration blocks, the CVIM module will validate the entire configuration since many of the operating parameters are interrelated.

Example:

>W,CB1 [CR] 00__F1__etc.

This example writes the data 00, F1, etc. into configuration block #1. "_" = space character.

Example:

>WC,CB1,CB30–35,CB21[CR](data)

This example writes the data into the specified blocks.

After executing the command, the CVIM module will return: **[CR][LF]**. No data is returned. If you do not have the proper command structure, the CVIM module will return: **?[CR][LF]**.

Note: We recommend that you check the discrete bit ''configuration fault'' after loading a configuration. Refer to Appendix B. You can check this bit by using the read inspection results command for toolset #1, (**>RR,S [CR]**).

Command Summary

After you have become familiar with the ASCII commands, you can use the following command summary as a quick reference guide.

Table 5.A
ASCII Command Summary

Command	Command Structure	Field Descriptions
Deactivate Forces	>DF [CR]	
Disable Outputs	>DO [CR]	
Enable Outputs	> EO [CR]	
Echo Data	>E, data [CR]	Data = ASCII string

Force Outputs	>F, On, d [CR]	n = 1 to 14
		d = 0 or 1

Load Configuration From EEPROM to RAM	>LO [CR]	
Load Configuration From RAM Card to RAM	>LO, CC, d[CR]	d = 1 to 16*
Lock	>L [CR]	
Unlock	>U [CR]	
Read Output Condition	>R, On [CR]	n = 1 to 14
Read Configurable Statistics	>RSn [CR]	n = number of times read
Read Configuration	>RC, CBn [CR]	n = 1 to 136
Read Discrete Bit Results	>RR, TSn [CR]	n = 1 or 2

Read Results Block	>RR, TSnRB, d [CR]
	
	>RR,TSno,d[CR]
	
	>RR,S**
		n = 1 or 2
		d = 1, 2, 3 or 4
		
		n = 1 or 2
		o = RL,RW,G,W,LP
		d = gage or window number
		
		S = Status

Save to EEPROM from RAM	>S [CR]	
Save to RAM Card from RAM	>S, CC, d [CR]	d = 1 to 16*

Set Configurable Results	>SR,TSxd,TSxd,etc. [CR]	x = 1 or 2
		d = G1,G2,W1,W2, RW1, RL3, LP, etc.

* The number of configurations that can be stored on a RAM card depends upon the card size (512K card can hold 16 configurations).

** Refer to Appendix B, Table B.1, RS–232 word 0 for a definition of CVIM status.

Command Summary (cont'd)

Table 5.A
ASCII Command Summary (Cont'd)

Command	Command Structure	Field Descriptions

Set Configurable Statistics	>SSn,TSxd,TSxd,etc. [CR]	n = number of samples.
		x = 1 or 2
		d = G1, G2, W1, W2, RW1, LP, etc.

Trigger Inspection	>T,TSn[CR]	n = 1 or 2
Write Display	>W, D, data [CR]	Data = XY X = 1 (TS1)2 2 (TS2) Y = 1 to 9 1 = Image only 2 = Failed Tools 3 = All Tools 4 = I/O Page 5 = Results Page 6 = Stats 1 Page 7 = Stats 2 Page 8 = Page Up 9 = Page Down
	>W, F, data [CR]	Data = XY X = 1 (TS1)2 2 (TS2) Y = 1 to 9 1 = Go On Reject 2 = Freeze On First Reject 3 = Freeze On All Rejects 4 = Freeze On Next Inspection 5 = Halt On Reject
	>W, DC, data (CRI	Data = XY X = 1 (TS1)2 2 (TS2) Y = 1 to 9 1 = Resume 2 = Reset Statistics 3 = Reset Counters 8 = Page Up 9 = Page Down

Write Configuration Block(W)	>W, CBn [CR] data	n = 1 to 136
		Data = ASCII configuration data

Write Configuration Block(WC)	>WC,CBn,CBn,etc. [CR] data	n = 1 to 136
		Data = ASCII configuration data

Explanation of ASCII Programming Example

The following sample program was written on an Allen-Bradley 1784-T50B terminal (IBM AT compatible) using GW basic. This program obtains discrete results from the CVIM module. A program user is prompted to select either toolset 1 or toolset 2. The program will then:

- Trigger an inspection.
- Detect when new data is available.
- Read all pass/fail/warning data for the selected toolset.
- Display a screen message if any of the first four windows fail.
- Prompt the user once again for a toolset number.

A basic outline of the program is as follows:

Lines 10 to 99	Initialize program variables, configure the RS–232 port for 8 bit transmissions, select no parity, select 9600 Baud, and initialize the display monitor.
Lines 100 to 130	Prompt the operator to select a trigger for toolset 1 or toolset 2.
Subroutine 2000	Reads results to find the current number of total triggers.
Subroutine 1000	Triggers the CVIM module inspection of the selected toolset.
Line 200	Causes a continuous read of CVIM module results until new results are detected. New results are detected by an incrementing of the “total trigger” data.
Subroutine 2500	Converts the CVIM module results from hexadecimal to integer.
Lines 240 to 270	Analyze the discrete fail bits for windows 1 through 4 and display a message if a failure is detected.
Line 400	Sends the program to input line 100.

The program manipulates the returned data as follows:

Explanation of ASCII Programming Example (cont'd)

Assume the ASCII string from the CVIM module is:

CR LF 2114 __ __ __ __ __ __389 __ __ __ __ __ B0__80__
A2__ 00__00__(etc.)CR LF

Note: (__ = space, LF = Line Feed, CR = Carriage Return)

The 18 element hexadecimal array after the program receives the data:

R1(0) = 2114 = Decimal representation of total triggers processed.

R1(1) = 389 = Decimal representation of total master faults (failed inspections).

R1(2) = B0 = Hexadecimal representation of discrete input word 0 low byte.

R1(3) = 80 = Hexadecimal representation of discrete input word 0 high byte.

R1(4) = A2 = Hexadecimal representation of discrete input word 1 low byte (Window 1 Fault/Warning, Window 2 Fault/Warning, etc.).

• • •

R1(17) = 00 = Hexadecimal representation of discrete input word 7 high byte (Gage 32 Fault/Warning, Gage 31 Fault/Warning, etc.).

The decimal display on the monitor will appear as follows after the program manipulates the array:

```
2114      389      176      135      162

0         0        0        0        0
0         0        0        0        0
0         0        0
```

Analysis of R1(4) for window failure:

R1(4) = 162 (decimal). The binary representation is:

1 0 1 0 0 0 1 0

The three ones in this representation indicate fail discrete input conditions in windows 1, 3, and 4 (bits 1, 5, and 7 of word 1, see Table 4.A.).

ASCII Programming Example

The following is a sample ASCII program written in BASIC:

```
1 REM     RS–232 to CVIM COMMUNICATIONS SAMPLE PROGRAM
2 REM    COPYRIGHT ALLEN–BRADLEY COMPANY, INC. 10-17-89 jrm
3 :
4 :
10 OPEN”com1:9600,n,8,1,DS”AS#1: REM Open communications channel
20 DIM R1(17): REM Allocate storage for tool set results
30 HE$=”0123456789ABCDEF”: REM Used for hex to decimal conversion
50 CLS
60 PRINT ”RS–232 TO ALLEN–BRADLEY CVIM COMMUNICATIONS PROGRAM”
70 PRINT:PRINT
99 :
100 PRINT ‘‘Press 1 or 2 to trigger tool set 1 or 2:’’;
110 K$=INKEY$: IF K$,.‘‘1’’ AND K$,.‘‘2’’ THEN 110
120 PRINT K$: TS = ASC (K$) – 48: REM Convert key ‘‘1’’ or‘‘2’’ to number 1 or 2
130 GOSUB 2000: REM Read tool set results to get # of triggers processed
140 IF R1(0)<0 THEN 100 ELSE NT = R1(0)
150 GOSUB 1000: REM Trigger an inspection
200 GOSUB 2000: IF R1(0)=NT THEN 200: REM Read until the trigger is processed
210 GOSUB 2500: REM Convert hex result string RE$ to integers
220 IF R1(0) <0 THEN 100: REM Quit on input error
230 PRINT: FOR X=0 TO 17: PRINT R1(X),: NEXT: PRINT: REM Print results
240 IF R1(4) AND 2 THEN PRINT ”Window 1 FAIL”
250 IF R1(4) AND 8 THEN PRINT ”Window 2 FAIL”
260 IF R1(4) AND 32 THEN PRINT ”Window 3 FAIL”
270 IF R1(4) AND 128 THEN PRINT ”Window 4 FAIL”
400 GOTO 100
999 :
1000 REM Subroutine to trigger an inspection on tool set TS
1050 PRINT#1,”>t,ts”; CHR$(TS+48);CHR$ (13);: REM Send the command
1080 RETURN
1999 :
2000 REM  Subroutine to read discrete results from tool set TS
2040 IF LOC(1) THEN R$=INPUT$(LOC(1),#1): REM clear out any garbage characters
2050 PRINT#1,”>rr,p1”; CHR$ (TS+48); CHR$(13);: REM Send the command
2060 CR$=INPUT$(2,#1): REM get CR/LF or ?/CR
2070 IF CR$=CHR$(13)+CHR$(10) THEN 2090
2080 PRINT”Input error”: R$=INPUT$(LOC(1),#1): R1(0)=–1: RETURN
2090 R$=INPUT$(1,#1): IF ASC(R$)<32 THEN 2090: REM ignore junk
2100 LINE INPUT#1,RE$: RE$=R$+RE$: REM get entire response
2120 R1(0) = VAL(MID$(RE$,1,9)): R1(1) = VAL(MID$(RE$,10,9))
2130 R$=INPUT$(LOC(1),#1): RETURN: REM Clear out any remaining characters
2499 :
2500 REM Subroutine to convert hex values in discrete result string RE$
2501 REM to integer values
2510 FOR RN=0 TO 15
2515 REM The following line converts each pair of hex digits to an integer
2520 D1=INSTR(HE$,MID$(RE$,25+RN*3,1))–1: D2=INSTR(HE$,MID$(RE$,26+RN*3,1))–1
2530 R1(RN+2) = 16*D1+D2: NEXT RN: RETURN
```

DF1 Protocol

The remainder of this chapter describes DF1 protocol. After you have made the equipment connections and configured the CVIM module for RS–232 communications, DF1 packets of data can be sent to the CVIM module.

What is DF1?

DF1 is an Allen–Bradley developed software convention used for RS–232 communications. DF1 provides some handshaking and data–packing formats which allow for fast communications with integrity of the data.

This chapter describes a simple application level of DF1 for communications between a CVIM module and a computer host. This application level requires that all transmitted data be preceded by a header and terminated by a trailer and a Block Check Character (BCC). In addition, ACK / NAK characters and simple time out conventions are used to ensure the integrity of the data.

A more complete implementation of DF1 can include layered software for point-to-point and multidrop links using several layers:

Data Link Layer
Transport Network Layer(s)
Application Layer

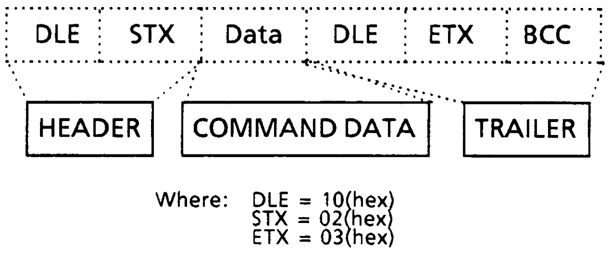
We do not provide a complete description of DF1 in this manual. We have only provided information necessary to transmit data between a host computer and the CVIM module. If you want to learn more about DF1, we suggest reading Publication 2802-800 (Line Scan Camera User’s Manual). Appendix A of this publication provides a thorough description of DF1.

DF1 Character Set

In the DF1 protocol mode, all data is transferred between the CVIM module and a host as bytes with a value between 00(hex) and FF(hex). Refer to Appendix E to convert control codes like ACK and NAK to/from hexadecimal values.

Command Structure

Each command the host device sends to the CVIM module is represented by a block of data beginning with DLE STX (Data Link Escape, Start of Transmission) and terminated with DLE ETX BCC (Data Link Escape, End Transmission, Block Check Character). The data between the header and trailer characters is the command data. The following shows the structure of a typical command:



Note: To avoid any confusion between DLE (10 hex) and data equal to 10 (hex), a value of 10(hex) is transmitted as 10(hex) 10(hex). The DLE code is transmitted simply as 10 (hex). This is referred to as “DLE Stuffing”.

The following shows the typical structure of the command data.

OPERATION	n times (H)	n times (L)	Object	Flags	Data

There are up to five fields in a command:

Operation Field — This field contains the command directed to the CVIM module. There can only be one operation per command line. Some commands don’t require any additional fields while others may require an object field,a data field, or both. Some commands cannot be used while the CVIM module is in the SETUP mode. If an operation cannot be performed because either the wrong host is selected or the CVIM module is in the SETUP mode, the CVIM module will not send a response.

n times (H) and **n** times (L) — These two fields indicate the High and Low bytes of the n times modifier. The n times modifier is used with certain commands to indicate the number of times the command is to be performed. The range for this value is 0000 to 00FF (255). A value of 0000 indicates infinity. The default value for this field is 0001.

Object Field — The Object field specifies data that configures the operation of the CVIM module.

In the description of each command we specify the objects that can entered into a command.

Flags — This optional field specifies outputs on the local I/O board or specific blocks of data.

Data Field — Contains data.

ACK/NAK, BCC Characters

After receiving a DF1 data packet, the CVIM module validates the Block Check Character.

Note: The block check character is a technique used to check the integrity of of data packet. BCC are explained in the next section.

Depending upon whether or not the BCC is validated, the following will occur:

If the BCC is not acceptable, the CVIM module will reply with a DLE NAK (Negative Acknowledgment) character and discard the data packet.

If the command has an acceptable BCC the CVIM module will reply with a DLE ACK (Positive Acknowledgment) character and try to execute the command.

After receiving a data packet and validating the BCC, one of the following will occur.

If data packet has a valid BCC but the CVIM module cannot execute the command the CVIM module will discard the data package. No message is returned. The host should be set to time out after waiting for a response.

If the command can be executed, the CVIM module will respond with any returned data packets.

After receiving the data, the host should respond with a DLE ACK to let the CVIM module know that the message was received properly. If the host returns a DLE NAK, the CVIM module will retransmit the data up to three times before discarding the data packet.

Note: Some commands request a continuous flow of data from the CVIM module. You can stop the flow of data by sending another command.

Note: A simple way to test the RS–232 links is to send the CVIM module a DLE ENQ (enquiry). If you have the port properly connected and the CVIM module is configured for RS–232, the CVIM module should send a DLE ACK or DLE NAK in response. If no response is provided, check your connections and CVIM module configuration.

Block Check Character

The block check character (BCC) is a means of checking the accuracy of each message packet transmission. It is the 2’s complement of the 8–bit sum (modulo–256 arithmetic sum) of all data bytes between the DLE STX and the DLE ETX BCC. It does not include any other message packet codes or response codes.

For example, if a message packet contained the data codes 8, 9, 6, 0, 2, 4, and 3, the message packet codes would be (in hex):

$$\underset{\text{DLE}}{10}\quad \underset{\text{STX}}{02}\quad \underbrace{08\quad 09\quad 06\quad 00\quad 02\quad 04\quad 03}_{\text{Data}}\quad \underset{\text{DLE}}{10}\quad \underset{\text{ETX}}{03}\quad \underset{\text{BCC}}{E0}$$

The sum of the data bytes in this message packet is 20 hex. The BCC is the 2’s complement of this sum, or E0 hex. This is shown in the following binary calculation:

```
0010          000020 hex
1101          11111’s complement
              +1
___________
1110          00002’s complement (E0 hex)
```

To transmit the data value 10 hex, you must use the data code DLE DLE. However, only one of these DLE data bytes is included in the BCC sum. For example, to transmit the values 8, 9, 6, 0, 10, 4, and 3 hex, you would use the following message codes:

$$\underset{\text{DLE}}{10}\quad \underset{\text{STX}}{02}\quad \underbrace{08\quad 09\quad 06\quad 00\quad \overbrace{10\quad 10}^{\text{Represents single data byte value of 10}}\quad 04\quad 03}_{\text{Data}}\quad \underset{\text{DLE}}{10}\quad \underset{\text{ETX}}{03}\quad \underset{\text{BCC}}{D2}$$

In this case, the sum of the data bytes is 2E hex because only one DLE text code is included in the BCC. So the BCC is D2 hex.

The BCC algorithm provides a medium level of data security. It cannot detect transposition of bytes during transmission of a packet. It also cannot detect the insertion or deletion of data values of zero within a packet.

Deactivate Forces

Use the deactivate force command to return outputs on the 1771–JMB local I/O board to the CVIM module assigned functions. The deactivate forces command is:

0C

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

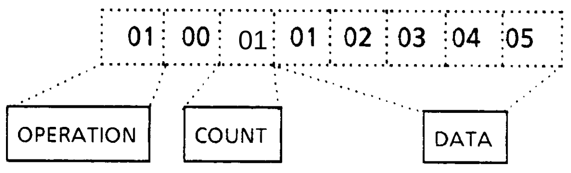
Echoing Data

Use the echo command to check the communications link. This command will return the same same string of characters that are sent out with the command. This command has the following structure:

01	00	n times	Data

Where n times specifies the number of times the CVIM module will echo the data field back to the host device. There is no object associated with this command. The command is valid at any time.

For example:



This example will cause the CVIM module to return the string:
DLE ACK DLE STX 1234512345123451234512345 DLE ETX BCC

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and echo the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

Enable/Disable Outputs

Use this command to enable or disable discrete outputs or local *Outputs* I/O. Use the following commands:

15	*This command disables the outputs.*
14	*This command enables the outputs.*

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

Forcing Local I/O

Use the force command to turn the local I/O outputs either on or off. This function can only be executed once per command. The only valid objects are the fourteen discrete outputs. This command has the following structure:

02	06	Flags	Data

Where the flags specify outputs 1 through 14:

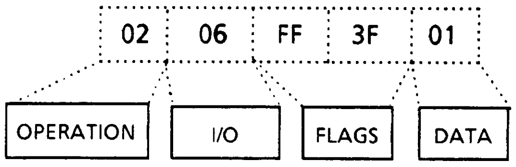
Byte 3 Bits 7 → 0								Byte 4 Bits 7 → 0							
8	7	6	5	4	3	2	1	X	X	14	13	12	11	10	9

Set individual Bits to select outputs 1 through 14

The data in this command indicates on or off (1 = ON and 0 = OFF).

Forcing Local I/O (cont'd)

For example:



This example will force all outputs on. FF sets all bits in byte 3 (outputs 1 through 8) and 3F sets bits 0 through 5 of byte 4 (outputs 9 through 14).

It is possible to have outputs forced on and off at the same time. A force off takes precedence over the force on. If multiple force commands are sent, the forced on or off outputs will be added to those already forced.

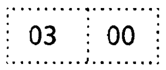
If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

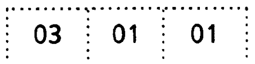
Note: The outputs will remain in their forced states until a deactivate forces command is sent.

Loading Configurations

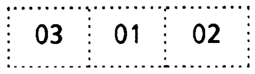
Use the load command to transfer configuration data between the CVIM module local storage area (EEPROM) and the external memory card. The RAM card slides into the slot on the front of the CVIM module. Use one of the following commands:



Transfers configuration from the EEPROM to the CVIM module internal RAM.



Transfers memory from RAM Card area 1 to the CVIM module internal RAM.



Transfers memory from RAM Card area 2 to the CVIM module internal RAM.

You cannot use these commands when the CVIM module is in the SETUP mode.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

Lock Command

Use the lock command to disable the setup menu box so that the SETUP mode cannot be entered. This function can only be executed once per command. There is no object associated with this command. The command has the following structure:

04

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

Read Output Status

Use the read command to read the status of the 14 discrete outputs on the local I/O board The command has the following structure:

05	00	n times	06	Flags

This function can be executed more than once per command by specifying an n times value.

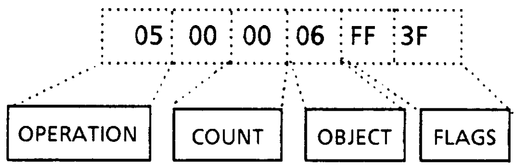
The flags specify outputs 1 through 14:

Byte 3 Bits 7 → 0								Byte 4 Bits 7 → 0							
8	7	6	5	4	3	2	1	X	X	14	13	12	11	10	9

Set individual bits to select outputs 1 through 14

Read Output Status (cont'd)

For Example:



This example will read the status of all fourteen outputs. FF sets all bits in byte 3 (outputs 1 though 8) and 3F sets bits 0 through 5 of byte 4 (outputs 9 through 14).

One byte is returned to indicate the status of the output (1 = ON and 0 = OFF). The output bytes are transmitted in numerical order (output #1 then output #2, etc.). The amount of data returned depends upon the number of outputs being read.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

Read Configuration Block Command

Use the read configuration command to read configuration data for the specified object. The command has the following structure:

06	07	Flags (17 bytes)

This function can only be executed once per command. The only valid object for this command are the configuration blocks.

The flags indicate which of the 136 configuration blocks are going to be read. Set the bits in bytes 3 through 19 of the command to specify the block(s). Use the following chart to determine which bits to set:

Use the read configuration command to read configuration data for the specified object. The command has the following structure:

Read Configuration Block Command Bytes 3–19

Byte 3 Bits 7 – 0	8	7	6	5	4	3	2	1
Byte 4 Bits 7 – 0	16	15	14	13	12	11	10	9
Byte 5 Bits 7 – 0	24	23	22	21	20	19	18	17
Byte 6 Bits 7 – 0	32	31	30	29	28	27	26	25
Byte 7 Bits 7 – 0	40	39	38	37	36	35	34	33
Byte 8 Bits 7 – 0	48	47	46	45	44	43	42	41
Byte 9 Bits 7 – 0	56	55	54	53	52	51	50	49
Byte 10 Bits 7 – 0	64	63	62	61	60	59	58	57
Byte 11 Bits 7 – 0	72	71	70	69	68	67	66	65
Byte 12 Bits 7 – 0	80	79	78	77	76	75	74	73
Byte 13 Bits 7 – 0	88	87	86	85	84	83	82	81
Byte 14 Bits 7 – 0	96	95	94	93	92	91	90	89
Byte 15 Bits 7 – 0	104	103	102	101	100	99	98	97
Byte 16 Bits 7 – 0	112	111	110	109	108	107	106	105
Byte 17 Bits 7 – 0	120	119	118	117	116	115	114	113
Byte 18 Bits 7 – 0	128	127	126	125	124	123	122	121
Byte 19 Bits 7 – 0	136	135	134	133	132	131	130	129

Set individual bits to select blocks 1 through 136.

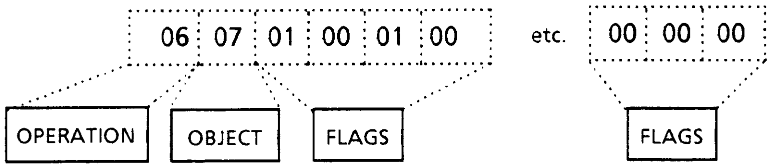
Read Configuration Block Command (cont'd)

For example: To read configuration blocks 49 and 50 you would send:

06(hex) for byte 1–	Indicates a read command.
07(hex) for byte 2–	Specifies the configuration blocks.
00(hex) for bytes 4 through 8.	
03(hex) for byte 9–	Sets the first two bits of byte 9 to indicate blocks 49 and 50.
00(hex) for bytes 10 through 19.	

Refer to Appendix C for a description of the configuration blocks. You cannot use this command while the CVIM module is in the SETUP mode.

Example:



This example reads configuration blocks 1 and 17.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

After reading the selected blocks, the CVIM module will return the requested data. Each word of a configuration block is sent as two bytes with the high byte transmitted first. A DLE (10 hex) is converted to DLE DLE (10 hex 10 hex). Refer to Appendix D for block descriptions and sizes. The following is an example of how the returned data appears (each pair of digits represents a single byte):

Configuration Block Returned Data Format*

```
48 01 02 00 00 00 00 00 44 65 66 61 75 6C 74 00 00 00 00 00
00 00 00 00 00 03 00 00 00 00 00 00 04 00 00 01 00 01 00 00
00 02 06 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 06 01
00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 01 01 00 01 01
00 0D FC 5F 9A 0A 00 19 FB D1
```

```
48 02 FF 02 01 00 00 00 00 FA 00 10 00 01 01 00 00 00 00 00
00 00 00 3F 00 00 00 00 00 00 00 3F 00 00 00 00 00 00 00 64
00 32 01 2C 00 96 0A 62 00 01 00 00 00 00 00 00 00 00 00 64
00 01 00 00 00 01 00 00 00 01 00 00 00 64 00 32 01 2C 00 96
0A 62 00 01 00 00 00 01 00 00 00 00 80 00 00 00 00 00 00 00
00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 01
00 00 00 00 00 00 49 CC
```

*We have added spaces for clarity, data is returned without spacing.

Read Results Command

Use this command to read the *results of the last inspection.* Refer to Appendix C for a description of the results blocks. Use one of the following commands:

07	00	n times	x	y	z

Where:

07 =	*Read Command*
00 =	*Always 00*
n times =	*Value for repeat read*
x =	*04 (Specifies Toolset #1)* *05 (Specifies Toolset #1)* *08 (Specifies CVIM module status)* *10 (Specifies Results Block #1)* *15 (Specifies Results Block #2)*
y =	*16 (Specifies Gage)* *17 (Specifies Window)* *18 (Specifies Reference Line)* *19 (Specifies Reference Window)* *1A (Specifies Light Probe)*
z =	*Number of window or gage being read or block result number.*

07	00	n times	04

Use this command to read the discrete bit results of toolset 1.

07	00	n times	05

Use this command to read the discrete bit results of toolset 2.

07	00	n times	10	Data

Use this command to read results blocks for toolset 1.

Where data = results block number.

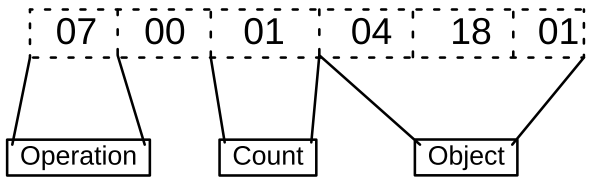
07	00	n times	15	Data

Use this command to read results blocks for toolset 2.

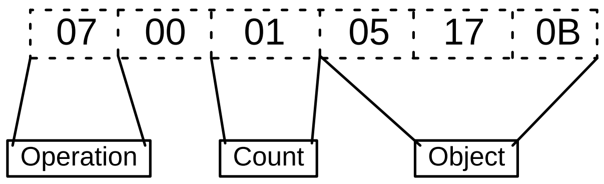
Where data = results block number.

The read results command can be executed more than once per command by specifying an n times value. This command is toolset dependent. Toolset 1 is specified by 04. Toolset 2 is specified by 05.

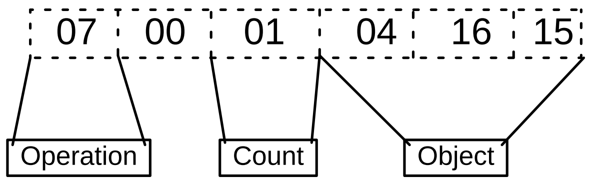
Examples:



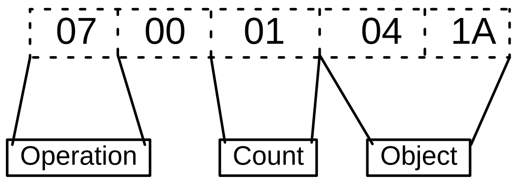
This command reads the results of toolset 1, reference line #1 (4 data bytes returned)



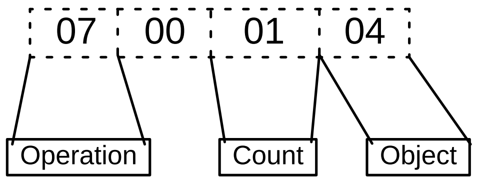
This command reads the results of toolset 2, window #11 (4 data bytes returned).



This command reads the results of toolset 1, gage #21 (4 data bytes returned).

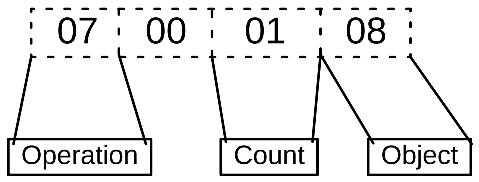


This command reads the results of toolset 1, light probe (4 data bytes returned).



This command reads the results of toolset 1, light probe (4 data bytes returned).

Read Results Command (cont'd)



This command reads the CVIM module status (2 data bytes returned).

Note: Refer to Appendix B, Table B.1, RS–232 word 0 for a definition of CVIM module status.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

The format of the requested data is an ASCII representation of the specified block(s) in bytes.

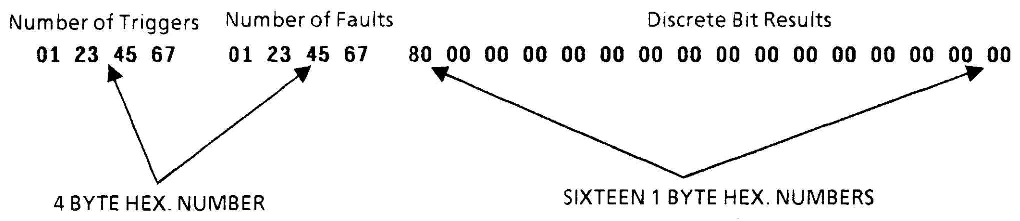
Results Block Returned Data Format*

```
61 01 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 CC
```

Refer to Appendix C for Block Descriptions

* We have added spaces for clarity, data is returned

Discrete Bit Results Returned Data Format*



Refer to Appendix B for a description of Returned Bytes

*We have added spaces for clarity; data is returned without spacing.

Save Command

Use the Save command to save CVIM module configuration data to the local storage area (EEPROM) or the external RAM card (credit card memory).

Note: Depending upon the card size, up to 16 configurations can be saved to the RAM card (512K card).

Use the following commands:

08	00

Saves configuration to EEPROM.

08	01	XX

Where XX = card storage location (01 to 16).

This function can only be executed once per command. You cannot use the commands when the CVIM module is in the SETUP mode.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

Select Image Displayed

Use the display object command to control the image that is displayed on the monitor. Use the following commands:

0B	02	Data

The data in this command specifies both the toolset and the display to be viewed.

0B (hex) = Image only displayed – Toolset 1
0C (hex) = Failed tools displayed – Toolset 1
0D (hex) = All tools displayed – Toolset 1
0E (hex) = I/O page displayed – Toolset 1
0F (hex) = Results page displayed – Toolset 1
10 (hex) = Stats 1 page displayed – Toolset 1

Select Image Displayed (cont'd)

11 (hex) = Stats 2 page displayed – Toolset 1
12 (hex) = Page up same display – Toolset 1
13 (hex) = Page down same display – Toolset 1

15 (hex) = Image only displayed – Toolset 2
16 (hex) = Failed tools displayed – Toolset 2
17 (hex) = All tools displayed – Toolset 2
18 (hex) = I/O page displayed – Toolset 2
19 (hex) = Results page displayed – Toolset 2
1A (hex) = Stats 1 page displayed – Toolset 2
1B (hex) = Stats 2 page displayed – Toolset 2
1C (hex) = Page up same display – Toolset 2
1D (hex) = Page down same display – Toolset 2

0B	13	Data

The data in this command specifies both the toolset and the display to be viewed.

0B (hex) = Go on reject – Toolset 1
0C (hex) = Freeze on first reject – Toolset 1
0D (hex) = Freeze on all reject – Toolset 1
0E (hex) = Freeze on next inspection – Toolset 1
0F (hex) = Halt on reject – Toolset 1

15 (hex) = Go on reject – Toolset 2
16 (hex) = Freeze on first reject – Toolset 2
17 (hex) = Freeze on all reject – Toolset 2
18 (hex) = Freeze on next inspection – Toolset 2
19 (hex) = Halt on reject – Toolset 2

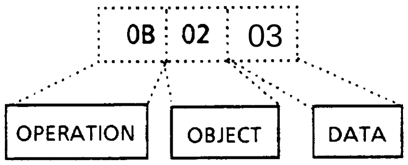
0B	14	Data

The data in this command specifies both the toolset and the display to be viewed.

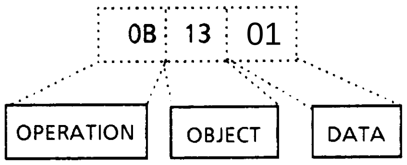
0B (hex) = Resume – Toolset 1
0C (hex) = Reset Statistics – Toolset 1
0D (hex) = Reset counters – Toolset 1
12 (hex) = Page up – Toolset 1
13 (hex) = Page down – Toolset 1

15 (hex) = Resume – Toolset 2
16 (hex) = Reset Statistics – Toolset 2
17 (hex) = Reset counters – Toolset 2
18 (hex) = Page up – Toolset 2
19 (hex) = Page down – Toolset 2

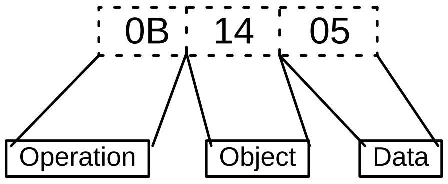
Examples:



This example displays all tools in toolset 1.



This example selects go on reject in toolset 1.



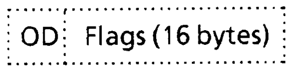
This example selects page down in toolset 2.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

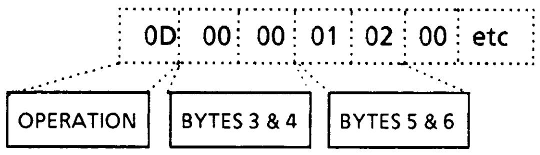
Set Configurable Results

Use this command to obtain a configurable results block. The results you want are specified by a list of tools and placed in results block #4. No data is returned until you use a read inspection results command for block #4. Use the following command:



The flags indicate which toolsets are specified. Refer to Table C.5 in Appendix C. Set the bits to 1 for the tools you want.

Example:



This command configures results block #4 to contain Toolset 1, Window #9 and Window #18 data.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

Note: The set configurable results command only sets the contents of the configurable results block (block #4). You must use a read results command to obtain the data (07 00 nn 10 04, *where nn specify the number of times the command is performed).* Refer to Read Inspection Results command for the format of the returned data.

The returned results bloc will be 128 bytes including the block signature (2 bytes) and the total number of triggers (last 4 bytes). Refer to page C–16. The ordering of the tools and data lengths are the same as the Remote I/O configurable results block.

Set/Read Configurable Statistics

Use the read command to read statistical data for the light probe, reference windows, gages, and windows. Use the separate set command to set the number of samples and configure the statistics block.

The set configurable statistics command has the following structure:

16	00	nn	x	Flags (16 Bytes)

Where nn specifies the number of samples and x specifies the toolset (04 = toolset 1, 05 = toolset2).

Note: If nn is 00, the CVIM module will continue to use the sample count configured during setup. Any other value will change the sample count.

The flags indicate which items are placed in the statistics block. Refer to Table C.5 in Appendix C. Set the bits to 1 for the statistics you need.

The read configurable statistics command has the following structure:

17	00	n times

Where nn specifies the number of times the statistics block is read.

Statistics are accumulated until the number of samples is reached, at which point the statistics begin to reaccumulate The number of samples for each toolset are accumulated separately. For example, if the toolset is toolset #2, the statistics are accumulated based upon the number of triggers for toolset #2.

Example of Set Command:

16	00	32	04	Flags (16 Bytes)

This example sets the number of samples to 50 using toolset 1.

16	00	64	05	Flags (16 Bytes)

This example sets the number of samples to 100 using toolset 2.

Example of Read Command:

17	00	05

This example reads the statistics block five times.

The data returned from the statistics blocks consists of:

Block signature

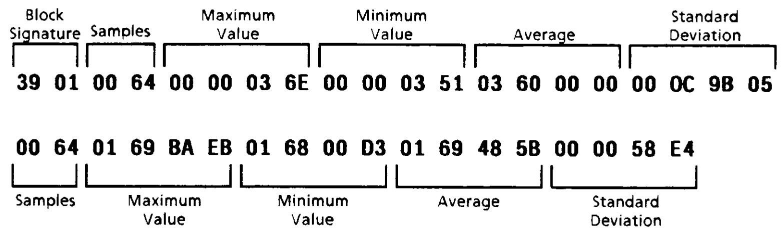
Number of samples, maximum, minimum, average and standard deviation for each toll configured in the block.

Set/Read Configurable Statistics (cont'd)

The block signature is 2 bytes long. The number of samples is a 2 byte integer. The maximum and minimum values are each 4 bytes. The format of the data depends upon the operation (e.g. pixel count is an integer and linear gaging is a 16.16 fixed point value). Refer to page C–24 for data formats. The average and standard deviation are also 4 bytes each but are always 16.16 fixed point values.

Therefore, each tool statistic consists of 18 bytes with the exception of reference windows which contain 18 bytes for each feature or a total of 54 bytes. The statistics block is transmitted as two hexadecimal characters for each byte. The total number of bytes including the block signature should not exceed 128 bytes. The statistics block is read once for every number of specified samples. This means that if you read the statistics block five times with a sample number of 50, 250 triggers will have to be processed before the five reads are completed. The following shows the format of the returned data:

Statistics Block Returned Data Format*



```
39 01 00 64 00 00 03 6E 00 00 03 51 03 60 00 00 00 0C 9B 05

00 64 01 69 BA EB 01 68 00 D3 01 69 48 5B 00 00 58 E4
```

*Data is transmitted without spaces, spacing added for clarity.

Trigger Operation Command

Use the trigger operation command to initiate an inspection by a toolset. Use the following commands:

09 04

Triggers an inspection using toolset 1.

09 05

Triggers an inspection using toolset 2.

This function can only be executed once per command.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command and return the data. If the command structure is invalid, the CVIM module will not execute the command or respond.

Note: When using this command you should make sure that the CVIM module is configured for a “hosted trigger source”.

Unlock Command

Use the unlock command to enable the user interface (monitor and keyboard) so that a user can access the SETUP. This command has the following structure:

0A

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

Write Configuration Blocks

Use the write command to write data to configuration blocks.

Note: You can also write to the monitor display to select what is displayed. Refer to Select Image Displayed.

This command has the following structure:

0B	07	Flags

Where the flags specify the configuration blocks being written to. There are 135 configuration blocks. Refer to the following diagram.

Write Configuration Blocks (cont'd)

Write Configuration Block Command Bytes 3–29

Byte 3

Bits 7 – 0

8	7	6	5	4	3	2	1

Byte 4

Bits 7 – 0

16	15	14	13	12	11	10	9

Byte 5

Bits 7 – 0

24	23	22	21	20	19	18	17

Byte 6

Bits 7 – 0

32	31	30	29	28	27	26	25

Byte 7

Bits 7 – 0

40	39	38	37	36	35	34	33

Byte 8

Bits 7 – 0

48	47	46	45	44	43	42	41

Byte 9

Bits 7 – 0

56	55	54	53	52	51	50	49

Byte 10

Bits 7 – 0

64	63	62	61	60	59	58	57

Byte 11

Bits 7 – 0

72	71	70	69	68	67	66	65

Byte 12

Bits 7 – 0

80	79	78	77	76	75	74	73

Byte 13

Bits 7 – 0

88	87	86	85	84	83	82	81

Byte 14

Bits 7 – 0

96	95	94	93	92	91	90	89

Byte 15

Bits 7 – 0

104	103	102	101	100	99	98	97

Byte 16

Bits 7 – 0

112	111	110	109	108	107	106	105

Byte 17

Bits 7 – 0

120	119	118	117	116	115	114	113

Byte 18

Bits 7 – 0

128	127	126	125	124	123	122	121

Byte 19

Bits 7 – 0

136	135	134	133	132	131	130	129

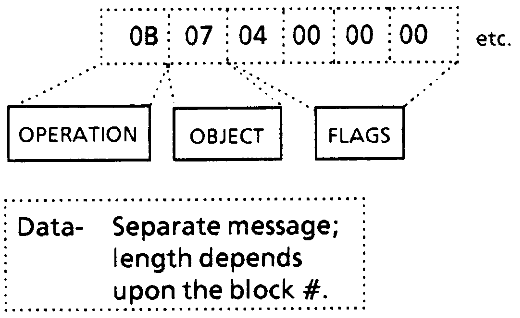
Set individual bits to select blocks 1 through 136.

This function can only be executed once per command.

Each configuration block is sent in a separate data packet (Header/Data/Trailer).

Refer to Appendix D for a description of the configuration blocks. You cannot use this command when the CVIM module is in the setup mode. After writing to the CVIM module, the CVIM module will validate all of the configuration blocks (refer to Chapter 4 for a description of memory validation).

Example:



This example writes the data into configuration block #3.

If the BCC is not valid, the CVIM module will respond with a DLE NAK and the command will not be executed.

If the BCC is valid, the CVIM module will respond with a DLE ACK. Then the CVIM module will validate the command structure. If the command is valid, the CVIM module will execute the command. If the command structure is invalid, the CVIM module will not execute the command or respond.

Note: We recommend that you check the discrete bit ‘‘Configuration Fault’’ after loading a configuration. Refer to Appendix B. Since no data is returned with this command, the load operation may fail and you could detect this by checking the fault bit. You can check this bit by using the read discrete bit results command for toolset #1.

Command Summary

After you have become familiar with the DF1 commands, you can use the following command summary as a quick reference guide.

Table 5.B
DF1 Command Summary

Command	Command Structure	Field Descriptions
Deactivate Forces	0C	

Echo Data	01 00 nn Data	nn = times repeated
		Data = Hexadecimal string

Enable Outputs Disable Outputs	14 15	
Force Outputs	02 06 Flags Flags Data	Flags indicate outputs Data = 00 (on) or 01 (off)
Load Configuration From EEPROM to RAM	03 00	

Load Configuration From Credit Card to RAM*	03 01 Data	Data = 01 (area 1)
		02 (area 2)
		● ● ●
		15(area 15)
		16(area 16)

Lock Unlock	04 0A	
Read Output Condition	05 00 nn 06 Flags Flags	nn = times repeated Flags indicate outputs
Read Configuration	06 07 Flags	Flags indicate blocks
Read Discrete Bit Results	07 00 nn 04 07 00 nn 05	First discrete bit results Second discrete bit results nn = times repeated

* The number of configurations that can be stored on a RAM card depends upon the card size (512K card can hold 16 configurations).

Table 5.B
DF1 Command Summary (Cont'd)

Command	Command Structure	Field Descriptions
Read Results Block	07 00 n x y z	n = times repeated x = 04 (Toolset 1) 05 (Toolset 5) y = 16(Gages) 17 (Window) 18 (Reference Line) 19 (Reference Window 1A (Light Probe) z = Window, Gage, Reference Line, or Reference Window No.

	07 00 n 10 Data	Toolset 1 Results
		n = times repeated
		Data = Block Number(s) 1, 2, 3 or 4

	07 00 n 15 Data	Toolset 2 Results
		n = times repeated
		Data = Block Number(s) 1, 2, 3 or 4

	07 00 n 04	Toolset 1 discrete bit results n = times repeated
	07 00 n 05	Toolset 2 discrete bit results n = times repeated
	07 00 n 08	Read Status
Save to EEPROM from RAM	08 00	

Save to RAM Card from RAM	08 01 Data	Data = 01 (area 1)
		02 (area 2)
		● ● ●
		15(area 15)
		16(area 16)

Set Configurable Results	OD Flags	Flags indicate specific tools
Set Configurable Statistics	16 00 n Flags	Flags indicate specific tools n = times repeated
Read Configurable Statistics	17 00 n	n = times repeated
Trigger Inspection	09	Trigger
Write Configuration Block	0B 07 Flags Data	Flags indicate blocks

Command Summary (cont‘d)

Table 5.B
DF1 Command Summary (Cont’d)

Command	Command Structure	Field Descriptions
Write Display	0B 02 Data	Data = 0B–Image Only Toolset 1 0C–Failed Tools Toolset 1 0D–All Tools Toolset 1 0E–I/O Page Toolset 1 0F–Results Page Toolset 1 10–Stats 1 Page Toolset 1 11–Stats 2 Page Toolset 1 12–Page Up Same Display Toolset 1 13–Page Down Same Display 15–Image Only Toolset 2 16–Failed Tools 17–All Tools Toolset 2 18–I/O Page Toolset 2 19–Results Page Toolset 2 1A–Stats 1 Page Toolset 2 1B–Stats 2 Page Toolset 2 1C–Page Up Same Display Toolset 2 1D–Page Down Same Display Toolset 2
	OB 13 Data	Data = 0B–Go On Reject Toolset 1 0C–Freeze On First Reject Toolset 1 0D–Freeze On All Reject Toolset 1 0E–Freeze On Next Inspection Toolset 1 0F–Halt On Reject Toolset 1 15–Go On Reject Toolset 2 16–Freeze On First Reject Toolset 2 17–Freeze On All Reject Toolset 2 18–Freeze On Next Inspection Toolset 2 19–Halt On Reject Toolset 2
	OB 14 Data	Data = 0B–Resume Toolset 1 0C–Reset Statistics Toolset 1 0D–Reset Counters Toolset 1 12–Page Up Toolset 1 13–Page Down Toolset 1 15–Resume Toolset 2 16–Reset Statistics Toolset 2 17–Reset Counters Toolset 2 18–Page Up Toolset 2 19–Page Down Toolset 2

DF1 Programming Example

The following is a sample DF1 program written in C. The program configures the host computer’s serial port for 9600 Baud communications. The program then displays a menu which prompts the user to select one of the following operations:

0. Echo the word “HELLO” to test the communications port.
1. Trigger the CVIM module to perform an inspection using toolset #1.
2. Read the discrete bit results for toolset 1.
3. Read the results block #1 for toolset 1.
4. Read gage #1 statistics for toolset 1.
5. Trigger the CVIM module to perform an inspection using toolset #2.
6. Read the discrete bit results for toolset 2.
7. Read the results block #1 for toolset 2.
8. Read gage #1 statistics for toolset 2.
9. Read configuration block n.
10.. Write configuration block n.
11.. Change display to results page, toolset 1.
12. Page up, toolset 1 display.
13. Page down, toolset 1 display.
14.. Change display to results page, toolset 2.
15. Page up, toolset 2 display.
16. Page down, toolset 2 display.

The program demonstrates a simple implementation of the DF1 protocol. Your actual application program may require some enhancements such as increased error checking or time–out conventions.

The sample program begins on the next page.

DF1 Programming Example (cont'd)

```
/* CVIM RS–232 Communication example program using DF1 protocol */
/* Copyright Allen–Bradley  12-5-89 jrm, aes */

This sample program was compiled using Microsoft C Version 5.1 */

#include <stdio.h>
#include <stdlib.h>
#include <bios.h>

#define STX 0x02  /* Start of Text character */
#define ETX 0x03  /* End of Text character */
#define ENQ 0x05  /* Enquire */
#define DLE 0x10  /* Data Link Escape (Control char) */
#define ACK 0x06  /* Positive acknowledgement */
#define NAK 0x15  /* Negative acknowledgement */

#define COMMFLAGS 0x8E00  /* defines bits to check for comm error */

#define MAX BUFFER         128
                           /* define storage for configuration data */
unsigned char config [135] [MAX BUFFER] configlen [135]
template [256] [MAX BUFFER] templen [256[

unsigned char last_response = 0;

void main()
    {
    int x, op_num, portnum, err, replen, reslen, numblocks;
    unsigned char reply[200], results[MAX BUFFER];

    /* Get Serial port number from user */
    printf (”Enter port number (1 for COM1 or 2 for COM2):”);
    scanf (”%d”, &portnum));

    /* make portnum either 0 or 1 */
    portnum = (portnum – 1) & 0x01;

    /* Open comm channel to at 9600 baud */
    _bios_serialcom (_COM_INIT, portnum–1, _COM_CHR8 | _COM_STOP1 |
    _COM_NOPARITY | _COM_9600);
```

```
/* Print options menu on the screen */
do  {
    printf (”\n\nOperations: \n\n”);
    printf (”0. Echo ’HELLO’\n”);
    printf (”1. Trigger Tool Set 1\n”);
    printf (”2. Read Discrete Results tool set 1\n”);
    printf (”3. Read Results Block 1, toolset 1\n”);
    printf (”4. Read gage 1 statistics, tool set 1\n”);
    printf (”5 Trigger tool set 2\n”);
    printf (”6 Read discrete results, tool set 2\n”);
    printf (”7 Read results block 1, tool set 2\n”);
    printf (”8 Read gage statistics, tool set 2\n”);
    printf (”9 Read configuration\n”);
    printf (”10 Write configuration\n”);
    printf (”11 Change to results page, tool set 1\n”);
    printf (”12 Page up, tool set 1\n”);
    printf (”13 Page down, tool set 1\n”);
    printf (”14 Change to results page, tool set 2\n”);
    printf (”15 Page up, tool set 2\n”);
    printf (”16 Page down, tool set 2\n”);
    printf (”\nEnter operation number (0–16) or –1 to quit: ”);

     /* Convert user string input to a number */
     scanf(”%d”, &op_num );

    replen = err = 0;    /* Initialize control variables */
    switch (op_num)      /* Determine what user selected */
        {

        case 0:      /* echo hello */
            {
            err = send_message (portnum,”\001\000\001HELLO”,8);
            if (!err)
                err = get_message(portnum,reply,&replen);
        break;

        case 1:      /* trigger tool set 1 */
            {
            err = send_message (portnum,”\011”/004”,2); /* no reply */
              break;

        case 2:      /* read discrete results, tool set 1 */
            err = send_message (portnum,”\007\000\001\004”,4);
            if (!err)
                err = get_message(portnum,reply,&replen);
            break;
```

DF1 Programming Example (cont'd)

```
case 3:      /* read results block 1, tool set  1 */
   err = send_message (portnum, ”\007\000\001\020\001”,5);
    if (!err)
    {
   err = get_message(portnum, results, &reslen);
    if (!err)
    {
    printf (”Results block #1:\n”);
     */ Display the results block */
        for (x=0; x<reslen; x+=2)
            printf (”%04X ”,results[x]*256 + results[x+1]);
        printf (”\n”);
        }
    }
    break;

case 4:
  err =send_message(portnum,”\026\000\062\004\000\000\000\000”
       ”\001\000\000\000\000\000\000\000\000\000\000\000”,20);

    for (x=0; x<5000 x++);  /* give CVIM time to prepare */

     if (!err)
     {
        err = send_message(portnum, ”\27\0\1”,3));
     if (!err)
     {
       err = get_message(portnum, results, & reslen);
       for (x=0; x<5000 x++);  /* give CVIM time to prepare */
        if (!err)
        {
         printf (”Gage #1 Stats \n”);
         for (x=0; x<reslen; x+=2);
           printf (”%04x”, results [x] *256 + results [x+1]);
        printf (”\n”);
         }
      }
   }
   break;

case 5:      /* trigger tool set #2 */
  err =send_message(portnum,”\011\005”, 2); /* no reply */
    break;

case 6:      /* read discrete results tool set 2 */
  err =send_message(portnum,”\007\000\001\005”,4);
   if (!err)
```

```
     err = get_message(portnum,reply, & replen);
      break;

   case 7:      /* read results block 1 tool set 2 */
     err =send_message(portnum,”\007\000\001\025\001”,5);
      if (!err)
   {
     err = get_message(portnum,results, & reslen);
      if (!err)
            printf (”Results block #1:\n”);
   /* Display the results block */
   for (x=0; x<reslen; x+=2);
           printf (”%04x”, results [x] *256 + results [x+1]);
            printf (”\n”);
            }
         }
       break;

   case 8:
     err =send_message(portnum,”\026\000\062\004\000\000\000\000”
          ”\001\000\000\000\000\000\000\000\000\000\000\000”,20);
        for (x=0; x<5000 x++);  /* give CVIM time to prepare */
        if (err)
        {
        err = send_message(portnum, ”\27\0\1”, 3);
         if (err)
         {
          err = get_message(portnum,results, & reslen);
          for (x=0; x<5000 x++);  /* give CVIM time to prepare */
           if (err)
           {
            printf (”Gage #1 Stats \n”);
            for (x=0; x<reslen; x+=2);
              printf (”%04x”, results [x] *256 + results [x+1]);
           printf (”\n”);
            }
         }
      }
      break;

case 9:           /* read configuration */
         /* read all config. blocks */
     err =send_message(portnum,”\6\7\377\377\377\377\377\377\377”
          ”\377\377\377\377\377\377\377\377\377\377”,19);
       if (err)
       {
         /* read config blocks 1-135 */
         for (x—0 (x<135) && !err; x++)
        err = get_message(portnum, config [x] & config [x]);
```

DF1 Programming Example (cont’d)

```
       if (err)
       {
         /* read first template block */
       err = get_message(portnum, template [0] & templen [0]);

         /* Determine how many template blocks follow */
         numblocks = template [0] [2];
         for (x=1 (X<numblocks) && !err; x++)
       {
       err = get_message(portnum, template [x] & templen [x]);
            }
         }
      }
      break;

case 10:          /* write configuration */
      /* write entire config. */
    err =send_message(portnum,”\13\7\377\377\377\377\377\377\377”
          ”\377\377\377\377\377\377\377\377\377\377”,19);

      for (x=0; x<5000 x++);  /* give CVIM time to prepare */
      if (err)
      {
         /* write config blocks 1-135 */
         for (x—0 (x<135) && !err; x++)
        err = send_message(portnum, config [x] & config [x]);
       if (err)
       {
         /* write all template blocks */
         numblocks = template [0] [2];
         for (x=0 (X<numblocks) && !err; x++)
       err = send_message(portnum, template [x] & templen [x]);
       err = get_message(portnum, template [0] & templen [0]);
         }
      }
      break;

case 11:
    err =send_message(portnum,”\013\002\017”, 3);
      break;

 case 12:
    err =send_message(portnum,”\013\002\022”, 3);
      break;

 case 13:
    err =send_message(portnum,”\013\002\023”, 3);
      break;
```

```
          case 14:
             err =send_message(portnum,”\013\002\031”, 3);
               break;

          case 15:
             err =send_message(portnum,”\013\002\034”, 3);
               break;

          case 16:
             err =send_message(portnum,”\013\002\035”, 3);
               break;

           default:
               break;

            }    /* End switch (op_num) statement */

                 if (err)
                  printf (”Error code: %04xn”,err);

                 if (replen)
                 {
                  printf (”Response ”);
                   for (x=0; x<replen; x++)
                       printf (”%02X”, reply [x]);
                   printf (”n”);
                 }
              } while (op num>=0);     */ End do loop */
           }

/* Transmits the message pointed to by msg, consisting of len characters.
   If the message is not acknowledged, re–transmits it up to 2 more times.
    Returns zero if successful or nonzero if an error occurs. */

    int send_message (portnum, msg, len)
    int portnum                */ Communications port # */
    unsigned character *msg    */ Pointer to message data */
    int len;                   */ Length of message */
    {
         int x, ch, chksum, err, retry=3;

         do
         {

        /* if an incoming char is waiting, clear it out */
        while (_bios_serialcom(_COM_STATUS, portnum, 0) & 256)
            _bios_serialcom(_COM_RECEIVE, portnum, 0);
```

DF1 Programming Example (cont’d)

```
/* send DLE STX to initiate message transfer */
_bios_serialcom(_COM_SEND,portnum,DLE);
_bios_serialcom(_COM_SEND,portnum,STX);

/* Send all bytes of the selected command & compute checksum */
for (x=chksum=0; x<len; x++)
    {
    _bios_serialcom(_COM_SEND,portnum, msg[x]);
    chksum += msg[x];

    /* If data within the message was 0x10 (a DLE), send another
    DLE so the receiver knows it is data, not a control character.
    (But don’t include the second DLE in the checksum) */

    if (msg[x] == DLE)
        _bios_serialcom(_COM_SEND,portnum,msg[x]);
    }

/* send DLE ETX and CHKSUM */
_bios_serialcom(_COM_SEND,portnum,DLE);
_bios_serialcom(_COM_SEND,portnum,ETX);
_bios_serialcom(_COM_SEND,portnum,(–chksum)&0xFF);

/* check for DLE ACK */
ch = _bios_serialcom(_COM_RECEIVE,portnum,0);
err = ch & COMMFLAGS;
if (err || ((ch & 0xFF) != DLE))   /* Mask character to 8 bits */
    err |= 0x01;            /* no DLE on send */
if (!err)
    {
    ch = _bios_serialcom(_COM_RECEIVE,portnum,0);
    err = ch & COMMFLAGS;

     /* Mask character to 8 bits
    if (err || ((ch & 0xFF) != ACK)) */
        err |= 0x02;             /* no ACK on send */
    }
 } while (err && (––retry > 0)); /* if any error, retry */
return (err);
}

/*  Receives a message into buffer pointed to by msg & places
    length in *len.  If the checksum is invalid or a timeout occurs,
    sends NAK and attempts to re–receive up to 2 more times.
    Returns zero on success or nonzero if an error occurs. */

int get_message (portnum, msg, len)
int portnum;         /* Serial port number */
unsigned char *msg;    /* Pointer to message buffer */
int     *len;         /* Pointer to return length of message */
```

```
{
int         good_string = 0;
int         message_started = 0;
int         ch, err, retry=4;
int         length = 0;
unsigned char    *msg_start_ptr, df1_bcc;

msg_start_ptr = msg;

while ( !good_string && retry )
    {
    ch = _bios_serialcom(_COM_RECEIVE,portnum, 0);
    err |= ch & COMMFLAGS;
    if (( ch & 0xFF ) == DLE )
        {
        ch = _bios_serialcom(_COM_RECEIVE,portnum,0);
        err |= ch & COMMFLAGS;
        switch( ch & 0xFF )
            {
            case STX:
                message_started = 1;
                break;

            case ETX:
                message_started = 0;
                ch = _bios_serialcom(_COM_RECEIVE,portnum,0);
                err |= ch & COMMFLAGS;
                df1_bcc = –df1_bcc;
                if (( (ch & 0xFF) == df1_bcc ) && !err )
                    {
                    _bios_serialcom(_COM_SEND,portnum,DLE);
                    _bios_serialcom(_COM_SEND,portnum,ACK);
                       last responsem = 1;
                      good_string = 1;
                    }
                else
                    {
                    _bios_serialcom(_COM_SEND,portnum,DLE);
                    _bios_serialcom(_COM_SEND,portnum,NAK);
                    if ( ––retry )
                        {
                        err = 0;
                        length = 0;
                        df1_bcc = 0;
                        msg = msg_start_ptr;
                        }
                    }
                break;
```

DF1 Programming Example (cont'd)

```
          case DLE:
              if ( message_started )
                  {
                  if ( ++length > MAX_BUFFER )
                      {
                      _bios_serialcom(_COM_SEND,portnum,DLE);
                      _bios_serialcom(_COM_SEND,portnum,NAK);
                      last_response = 0;
                      message_started = 0;
                      err = 0;
                      length = 0;
                      df1_bcc = 0;
                      msg = msg_start_ptr;
                      }
                  else
                      {
                      *msg++ = ( ch & 0xFF );
                      df1_bcc += ( ch & 0xFF );
                      }
                  }
              break;

          case ENQ:
           if ( last_response )
           {
          _bios_serialcom(_COM_SEND,portnum,DLE);
          _bios_serialcom(_COM_SEND,portnum,NAK);
         }
     else
         {
          _bios_serialcom(_COM_SEND,portnum,DLE);
          _bios_serialcom(_COM_SEND,portnum,NAK);
         }
         break;
       default:
         break;
          }
     }
    else if ( message_started )
     {
     if ( ++length > MAX_BUFFER )
          {
          _bios_serialcom(_COM_SEND,portnum,DLE);
          _bios_serialcom(_COM_SEND,portnum,NAK);
          last_response = 0;
          message_started = 0;
          err = 0;
```

```
            length = 0;
            df1_bcc = 0;
            msg = msg_start_ptr;
            }
        else
            {
            *msg++ = ( ch & 0xFF );
            df1_bcc += ( ch & 0xFF );
            }
        }
    }
*len = length;
return (err);
}
```

Chapter 6

Using the Pyramid Integrator Backplane

Chapter Objectives

This chapter:

- Describes the Pyramid Integrator Backplane.
- Describes backplane communication techniques.
- Describes CVIM module setup requirements
- Contains a sample PLC–5/250 program.

Note: Refer to Publication No. 5000–2.3 (Allen–Bradley Pyramid Integrator Technical Overview) for a description of the basic hardware components and valid configurations. Use the related publications chart in Chapter 1 to reference other Pyramid Integrator manuals as required.

What Information can be Accessed?

Through the backplane, you can access an area of memory called Shared Memory. Shared Memory consists of 1024 words (approximately half of which are presently used). Shared memory contains:

- CVIM module Discrete Bit Information (refer to Appendix B). These bits include pass/fail/warning data for inspections and command bits for CVIM module operation modes.

- Results Data (refer to Appendix C).

- In addition, you can access the CVIM module setup and configuration data through the backplane. Refer to Appendix D.

Host Designation

There are four communications ports which you can *simultaneously* use to access CVIM module data (Remote I/O, RS–232 (A&B), and Backplane). Only one of the communications ports can be designated as the *host* at any given time. Only the *host* can issue commands to control the operation of the CVIM module, trigger inspections, upload/download configurations, and change displays. You can read discrete bits and numerical results information through any of the three communications ports, even through non–host devices.

Note: See Chapter 2 for a description of multiple hosts.

What Functions can be Performed Over the Backplane?

A MicroVAX information processor, PLC–5/250, or other device in the Pyramid Integrator rack can request or manipulate the following data through the backplane.

- Obtain CVIM module inspection result information. Refer to Appendix B & C (CFG or SYS Host).
- Upload or download CVIM module configurations for inspections. Refer to Appendix D (CFG Host).
- Issue Read/Write commands between the following CVIM module memory locations (CFG Host):

  CVIM module Random Access Memory (RAM) and Electrically Erasable Programmable Read Only Memory (EEPROM).

  CVIM module RAM and RAM card. The RAM card slides into a slot on front of the CVIM module.

  CVIM module RAM and host memory.
- Change run–time display menus (SYS Host).
- Enable/Disable local I/O board (SYS Host).
- Force local I/O On or Off (SYS Host).

Note: When communicating with a device through the Pyramid Integrator backplane, CVIM module results are posted in shared memory immediately after processing. When communicating with a device through the other ports, results are only available at the end of the inspection program.

CVIM Module Configuration Instructions

If you are using the Pyramid Integrator backplane for communications, you must configure the CVIM module as follows:

Select the backplane for communications:

Note: This step is not required if you are only reading results.

1. Select the setup menu <Setup>.
2. Select the environment menu <Environ>.
3. Select the system menu <System>.
4. Select a Host menu <CFG Host:> or <SYS Host:>.
5. Choose Pyramid Integrator backplane option (twice) <Pyramid>.

Select the CVIM module trigger source:

6. Select the toolset menu <Toolset>.

If you are using the Pyramid Integrator backplane for communications, you must configure the CVIM module as follows:

7. Select the trigger source menu for the appropriate toolset <Trigger Source>.
8. Select either <I/O>, <Hosted> trigger sources or <Autotrigger>.

Note: When changing the host to/from the Pyramid Integrator, you must “pick” the selection twice and then reboot the CVIM module. **Make sure that you save your configuration first!**

Obtaining Inspection Result Information Using a PLC–5/250

If you are accessing results through a PLC–5/250, the 1024 words of shared memory are numbered 0 through 1023. The PLC–5/250 treats the shared memory area like an integer file. Table 6.A provides a summary of the data. Refer to Appendix A, B, or C for more detailed information.

Table 6.A
Shared Memory Overview

Block Name	Backplane Word #
Handshake	0
Reserved	1
Discrete I/O Outputs (Host Command Block)	2 to 9
Reserved	10 to 15
Discrete I/O Inputs (TS1) (Tool Inspection Results)	16 to 23
Results Block 1 (TS1)	24 to 87
Results Block 2 (TS1)	88 to 151
Results Block 3 (TS1)	152 to 215
Results Block 4 (TS1)	216 to 279
Dicrete I/O Inputs (TS2) (Tool Inspection Results)	280 to 287
Results Block 1 (TS2)	288 to 351
Results Block 2 (TS2)	352 to 415
Results Block 3 (TS2)	416 to 479
Results Block 4 (TS2)	480 to 543
Reserved	544 to 1022
Interrupt Control Word	1023
Configuration and Templates	Not Applicable

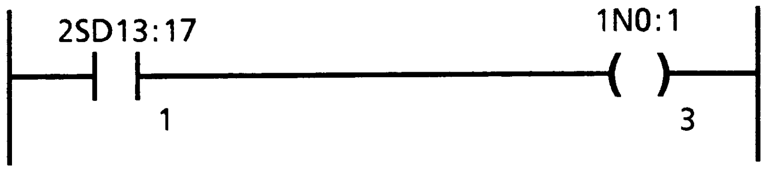
Obtaining Inspection Result Information Using a PLC–5/250 (cont'd)

The address of CVIM module shared memory is always SD13, with the addressing as follows:

CVIM Thumbwheel No.	SD13	Word/Bit No.

For example: Assume that the CVIM module has a thumbwheel setting of #2. The data that you want to read is window 1 fault flag in Toolset 1. Refer to Appendix B, the bit you want to read is bit 01 of word 17. The PLC bit address would then be:

2SD13:17/01



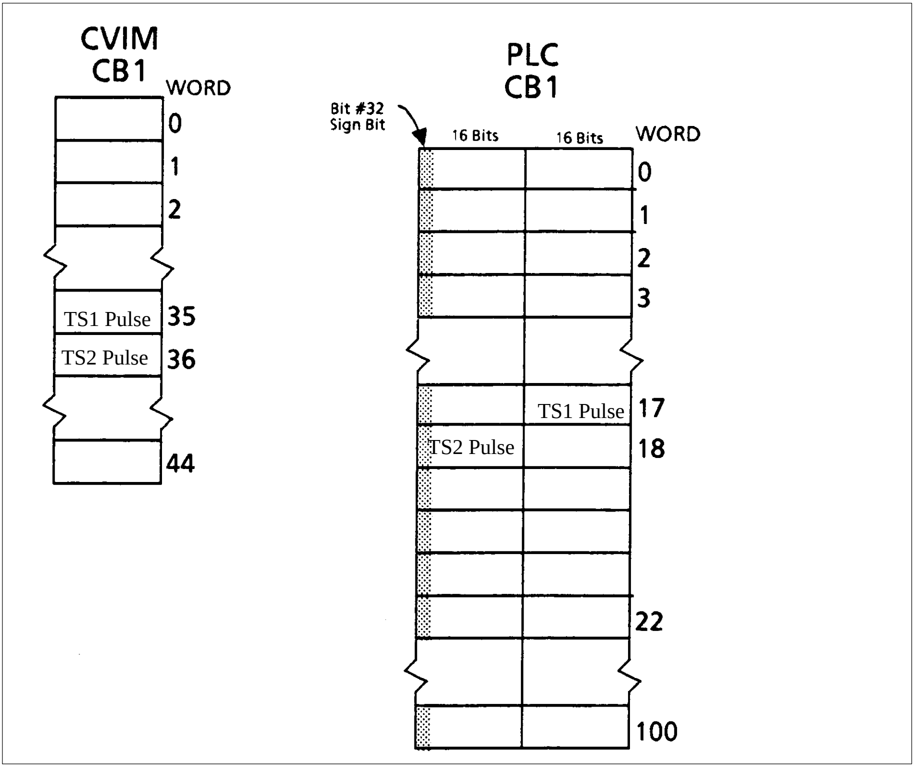
To access numerical values, you can use any PLC instruction which manipulates file values. Values can be read individually or in groups. For example, you can use the compute (CPT) function to move a single word from the shared data (SD) file (resides in resource manager) to an integer file on the local logic processor. Other PLC file instructions such as copy (FAL) or compare (FSC) can also be used to manipulate shared data results.

Note: A CVIM family processor in the same rack *must* have different thumbwheel settings.

Manipulating Configuration Data Using a PLC–5/250

When you transfer configuration data from the CVIM to a PLC–5/250 you use a message command and designate a long integer file as the destination (internal table address). Long integer files contain 32 bit elements. With this format, configuration block 1 would be arranged as shown in Figure 6.1.

Figure 6.1
32 Bit Long Integer Files



Use the message instruction (MSG) to transfer configuration data between the PLC and the CVIM. The message instruction can transfer up to 10,000 elements of data/commands. When used for Pyramid Integrator backplane instructions, the message instruction commands the resource manager module to transfer data between two module addresses. Use the RS–232 ASCII command set in the external data table address field to perform the desired function. To read a configuration of a CVIM with a thumbwheel setting of #2 into the Resource Manager long integer file #9 starting at element 0, the message instruction parameters would be entered as follows:

F1	Requested size:	0
F2	Priority:	NORMAL
F3	Local/Remote:	LOCAL
F4	Local Link Type:	DH +
F5	REM Link Type	N/A
F6	Station ID:	Node# = 0
F7	Module ID:	Class = CVIM TW# = 2 Port # = 1
F8	COMM CMND	TYPED READ
F9	Internal Data Table Address	0L9:0
F10	External Data Table Address	”> RC,CB*”

Sample PLC–5/250 Program

The following program shows how to trigger an inspection, and/or upload an entire CVIM configuration for archiving and later downloading.

Rung #1 Triggers the CVIM to perform an inspection using toolset#1 upon false transition of 1N0:0/00.

Rung #2 Reads the integer value of gage 1 and places value in 1N0:02.

Rung #3 One shots 1N0:1/0 when 1N0:02 has a false to true transition.

Rung #4 Sends message to read configuration:

Class:	CVIM TW#1
	Port:1
Internal Data Table Address:	0L9:0
External Data Table Address:	”>RC,CB*”

All other message parameters are not critical.

Rung #5 One shots 1N0:1/3 when 1N0:0/3 has from false to true transition.

Rung #6 Sends message to write configuration:

Class:	CVIM TW#1
	Port:1
Internal Data Table Address:	0L9:0
External Data Table Address:	”>W,CB*”

All other message parameters are not critical.

Note: See Remote I/O (Chapter 3) for converting 16.16 data to a PLC floating point value. Also see Chapter 3 for additional sample programs, and Chapter 5 for RS–232 commands.

The program begins on the next page.

```
                                                  16 January  1990   Page 3
Ladder Listing      Processor File: CVIM1, Addr: 003             Rung 1STEP0:0
Rung 1STEP0:0
|                                                                           |
+---------------------------------[TOP OF FILE]----------------------------------+
|                                                                           |

Rung 1STEP0:1
|   1N0:0                                                         1SD13:1023  |
+----] [--------------------------------------------------------------( )--+
|       0                                                             10    |

Rung 1STEP0:2
|   1N0:0                                                  +CPT-----------+  |
+----] [---------------------------------------------------+COMPUTE       +--|
|       1                                                  |Dest     1NO:2|  |
|                                                          |             0|  |
|                                                          |Expression    |  |
|                                                          |1SD13:65      |  |
|                                                          +--------------+  |
Rung 1STEP0:3
|   1N0:0                                       +OSR------------------+      |
+----] [----------------------------------------+ONE SHOT RISING      +-(OB)-+
|       2                                       |Output Bit    1NO:1/0|      |
|                                               |Storage Bit   1NO:1/1+-(SB) |
|                                               +---------------------+      |

Rung 1STEP0:4
|   1N0:1                                       +MSG------------------+      |
+----] [----------------------------------------+SEND/RECEIVE MESSAGE +-(EN)-+
|       0                                       |Control Block 0MSG0:0+-(DN) |
|                                               |                     +-(ER) |
|                                               +---------------------+      |

Rung 1STEP0:5
|   1N0:0                                       +OSR------------------+      |
+----] [----------------------------------------+ONE SHOT RISING      +-(OB)-+
|       3                                       |Output Bit    1NO:1/2|      |
|                                               |Storage Bit   1N0:1/3+-(SB)
|                                               +---------------------+      |

Rung 1STEP0:6
|   1N0:1                                       +MSG------------------+      |
+----] [----------------------------------------+SEND/RECEIVE MESSAGE +-(EN)-+
|       3                                       |Control Block 0MSG0:1+-(DN) |
|                                               |                     +-(ER) |
|                                               +---------------------+      |

Rung 1STEP0:7
|                                                                           |
+----]AFI[--------------------------------------------------------------(EOT)--+
|                                                                           |


Rung 1STEP0:8
|                                                                           |
+---------------------------------[END OF FILE]----------------------------------+
|                                                                           |
```

Sample PLC–5/250 Program (cont'd)

CONTROL BLOCK 0MSG0:0

F1	Requested Size(element):	0
F2	Priority:	HIGH
F3	Local/Remote:	LOCAL
F4	Local Link Type:	DH+
F5	Remote Link Type:	N/A
F6	Station ID:	Node# = 0
F7	Module ID:	Class = CVIM Tw# = 1 Port# = 1
F8	Communication Command:	TYPED READ
F9	Internal Data Table Addr:	0L9:0
F10	External Data Table Addr:	">rc,cb*"
	Parameters:	N/A

CONTROL BLOCK OMSG0:1

F1	Requested Size(element):	0
F2	Priority:	HIGH
F3	Local/Remote:	LOCAL
F4	Local Link Type:	DH+
F5	Remote Link Type:	N/A
F6	Station ID:	Node# = 0
F7	Module ID:	Class = CVIM Tw# = 1 Port# = 1
F8	Communication Command:	TYPED WRITE
F9	Internal Data Table Addr:	0L9:0
F10	External Data Table Addr:	">w,cb*"
	Parameters:	N/A

Obtaining Inspection Result Information Using a MicroVAX Information Processor

If you are accessing results through a MicroVAX information processor, you should use the standard library functions to access the data. The MicroVAX library instructions are called up using a simplified "C" type language. For example:

DTL__READ__W(Address, &Result, &Error);

Refer to the MicroVAX manuals for the available library routines.

Appendix A

Results/Configuration Data Overview

Introduction

This appendix provides an overview of the word and bit addresses of data stored in memory if you are communicating with the CVIM through the REMOTE I/O, Backplane, or RS–232 port.

Also provided is an explanation of how the CVIM stores fractional data.

Overview

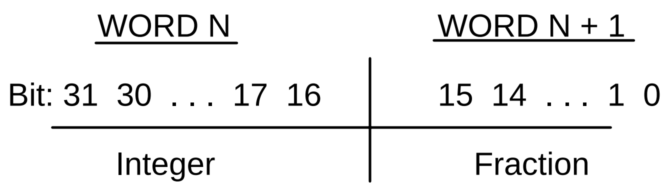
Table A.1 provides an overview of shared memory. Detailed descriptions of each word and bit functions can be found in the other appendices:

- Appendix B — Discrete Results Bits
- Appendix C — Numerical Results Data
- Appendix D — Configuration Data

Block Name	Backplane Word # XSD13i*	Backplane Word #
Handshake	0	N/A
Reserved	1	N/A
Discrete I/O Outputs (Host Command Block)	2 to 9	O:20 to O:27
Reserved	10 to 15	N/A
Discrete I/O Inputs (TS1) (Tool Inspection Results)	16 to 23	I:20 to I:27
Results Block 1 (TS1)	24 to 87	0 to 63
Results Block 2 (TS1)	88 to 151	0 to 63
Results Block 3 (TS1)	152 to 215	0 to 63
Results Block 4 (TS1)	216 to 279	0 to 63
Discrete I/O Inputs (TS2) (Tool Inspection Results)	280 to 287	I:20 to I:27
Results Block 1 (TS2)	288 to 351	0 to 63
Results Block 2 (TS2)	352 to 415	0 to 63
Results Block 3 (TS2)	416 to 479	0 to 63
Results Block 4 (TS2)	480 to 543	0 to 63
Reserved	544 to 1022	0 to 63
Interrupt Control Word	1023	N/A
Configuration and Templates	Not Applicable	Not Applicable

Fractional Notation

Inspections which produce results that are fractional are represented using two words (32 bits). The first 16 bits are the integer portion and the second 16 bits are the fractional portion:



The integer portion of the value is interpreted as a standard 16 bit signed integer where each bit is equal to:

Integer Bit Value $2^{(n)}$

Where n is the bit number.

Bits in the fractional portion of the value are interpreted as:

Fractional Bit Value $2^{(n-16)}$

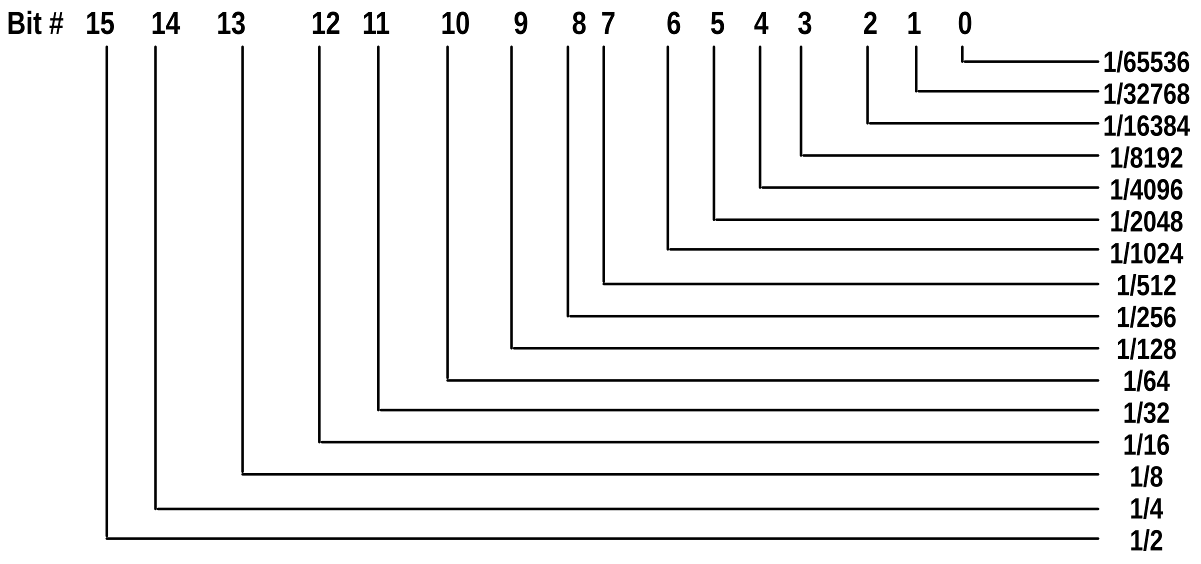
Where n is the bit number.

For example: bit 15 in the fractional portion of the value is equal to 1/2:

$$2^{(15-16)} = 2^{(-1)} = 1/2$$

We have provided the following chart to assist you:

Figure A.1
Fractional Notation

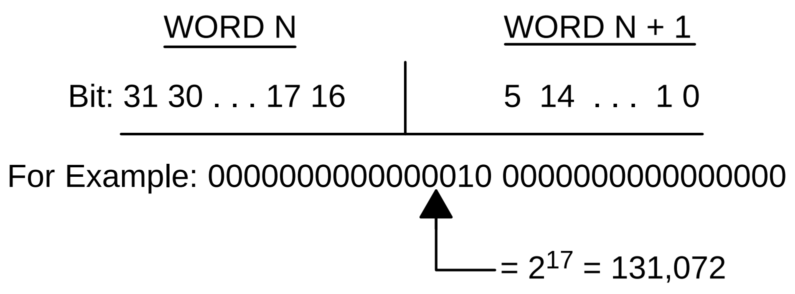


For example: 0000000000000011.0100000000000000 = 3.25

Note: For a sample PLC program which converts 16 point 16 values to floating point values, refer to Chapter 4.

32 Bit Integer Format

The CVIM stores some of the data as a 32 bit integer. The first byte contains bits 16 through 31 of the integer and the next byte contains bits 0 through 15 of the integer.



Template Blocks

The template blocks (part of configuration memory) contain previously learned image templates, not on-line configuration parameters. Word 1, bits 8-15 (third byte sent using RS-232 port) of the first template block indicate the total number of template blocks in the configuration. You must always upload or download *all* of the template blocks as a unit. You cannot archive only a part of the template blocks. When uploading templates from the CVIM, the program should read the first template block and check word 1, bits 8-15 (third byte sent using RS-232) to determine the number of template blocks to follow. The number of blocks remaining is then 1 less than the total number of template blocks. When downloading templates to the CVIM, the program must send all template blocks. Bits 8-15 of word 1 determine the number of blocks to send.

Appendix B

Discrete IIO Results Bits (Host Input & Output Bits)

Introduction

This appendix lists the function of both the discrete bit inputs and outputs. These bits can be accessed through the Remote I/O port and Pyramid Integrator backplane. You cannot manipulate these bits through the RS–232 ports (A or B) but you can perform many of their functions.

Discrete Bit Inputs

With each inspection that the CVIM module performs, individual bits are set. There are 128 bits that can be set as inputs to a host device. These bits (part of the inspection results) indicate:

- Configuration faults.
- Module Busy flag.
- Missed Trigger flags.
- Results Valid flags.
- Inspection Tool Pass/Fail/Warning flags

Note: If you are using the Remote I/O link, the bits for inspection tool pass/warning/fail apply to either toolset 1 or toolset 2. You select which toolset by setting bits 4 and 5 in word 3 of the discrete output bits.

Note: For your convenience, we have provided bit numbers for both a PLC (octal) and Pyramid Integrator (decimal).

Note to PLC–2 Users: When you use any PLC–2 family processor with the CVIM node adapter, you should understand the operation of the PLC Block Transfer Done bits for Read and Write instructions. PLC–2 family processors use the input image table for these bits all other PLCs can specify integer files for this function. This means that a PLC–2 user must use proper programming techniques to avoid confusion between the following bits:

- CVIM discrete I/O input word 0, bit 6 (data valid toolset #1) and bit 7 (data valid toolset #2).

- PLC–2 family input image table word 0, bit 6 (BTW done bit) and bit 7 (BTR done bit).

Table B.1 lists the discrete bit inputs.

Discrete Bit Inputs (cont'd)

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
16	280	0	0	0	Not Used		
16	280	0	1	1	Configuration Error	0 = No Error 1 = Error	Configuration error bit is set after any invalid configuration or template block write to the CVIM. This flag is also set after validation errors.
16	280	0	2	2	Mastership Flag	0 = Not Master 1 = Master	The device which reads this bit as 1 is the host. Not applicable for RS–232 communications.
16	280	0	3	3	Module Busy	0 = Not Busy 1 = Busy	Module Busy bit is set during the SETUP mode and during a download (sending configurations to the CVIM or reading/writing templates).
16	280	0	4	4	Trigger 1 NAK	0 = OK 1 = Trigger 1 Missed	
16	280	0	5	5	Trigger 2 NAK	0 = OK 1 = Trigger 2 Missed	
16	280	0	6	6	Toolset 1 Data Valid	0 = Not Valid 1 = Results Valid	Data Valid bit is reset when a user enters the SETUP mode. Refer to Chapter 3.
16	280	0	7	7	Toolset 2 Data Valid	0 = Not Valid 1 = Results Valid	Data Valid bit is reset when a user enters the SETUP mode. Refer to Chapter 3.
16	280	0	8	10	Reference Line 1 Flag	0 = Pass 1 = Failed	
16	280	0	9	11	Reference Line 2 Flag	0 = Pass 1 = Failed	
16	280	0	10	12	Reference Line 3 Flag	0 = Pass 1 = Failed	
16	280	0	11	13	Reference Window 1 Flag	0 = Pass 1 = Failed	
16	280	0	12	14	Reference Window 2 Flag	0 = Pass 1 = Failed	
16	280	0	13	15	Reference Window 3 Flag	0 = Pass 1 = Failed	
16	280	0	14	16	Light Probe Flag	0 = Pass 1 = Failed	
16	280	0	15	17	Master Fault	0 = No Fault 1 = Fault	Master fault bit is set if any tool fails an inspection.
17	281	1	0	0	Window 1 Warning Flag	0 = Pass 1 = Failed	

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
17	281	1	1	1	Window 1 Fault Flag	0 = Pass 1 = Fail	
17	281	1	2	2	Widow 2 Warning Flag	0 = Pass 1 = Fail	
17	281	1	3	3	Window 2 Fault Flag	0 = Pass 1 = Fail	
17	281	1	4	4	Window 3 Warning Flag	0 = Pass 1 = Fail	
17	281	1	5	5	Window 3 Fault Flag	0 = Pass 1 = Fail	
17	281	1	6	6	Window 4 Warning Flag	0 = Pass 1 = Fail	
17	281	1	7	7	Window 4 Fault Flag	0 = Pass 1 = Fail	
17	281	1	8	10	Window 5 Warning Flag	0 = Pass 1 = Fail	
17	281	1	9	11	Window 5 Fault Flag	0 = Pass 1 = Failed	
17	281	1	10	12	Window 6 Warning Flag	0 = Pass 1 = Failed	
17	281	1	11	13	Window 6 Fault Flag	0 = Pass 1 = Failed	
17	281	1	12	14	Window 7 Warning Flag	0 = Pass 1 = Failed	
17	281	1	13	15	Window 7 Fault Flag	0 = Pass 1 = Failed	
17	281	1	14	16	Window 8 Warning Flag	0 = Pass 1 = Failed	
17	281	1	15	17	Window 8 Fault Flag	0 = Pass 1 = Failed	
18	282	2	0	0	Window 9 Warning Flag	0 = Pass 1 = Failed	
18	282	2	1	1	Window 9 Fault Flag	0 = Pass 1 = Failed	
18	282	2	2	2	Window 10 Warning Flag	0 = Pass 1 = Failed	
18	282	2	3	3	Window 10 Fault Flag	0 = Pass 1 = Failed	

Discrete Bit Inputs (cont'd)

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
18	282	2	4	4	Widow 11 Warning Flag	0 = Pass 1 = Fail	
18	282	2	5	5	Window 11 Fault Flag	0 = Pass 1 = Fail	
18	282	2	6	6	Window 12 Warning Flag	0 = Pass 1 = Fail	
18	282	2	7	7	Window 12 Fault Flag	0 = Pass 1 = Fail	
18	282	2	8	10	Window 13 Warning Flag	0 = Pass 1 = Fail	
18	282	2	9	11	Window 13 Fault Flag	0 = Pass 1 = Fail	
18	282	2	10	12	Window 14 Warning Flag	0 = Pass 1 = Fail	
18	282	2	11	13	Window 14 Fault Flag	0 = Pass 1 = Failed	
18	282	2	12	14	Window 15 Warning Flag	0 = Pass 1 = Failed	
18	282	2	13	15	Window 15 Fault Flag	0 = Pass 1 = Failed	
18	282	2	14	16	Window 16 Warning Flag	0 = Pass 1 = Failed	
18	282	2	15	17	Window 16 Fault Flag	0 = Pass 1 = Failed	
19	283	3	0	0	Window 17 Warning Flag	0 = Pass 1 = Failed	
19	283	3	1	1	Window 17 Fault Flag	0 = Pass 1 = Failed	
19	283	3	2	2	Window 18 Warning Flag	0 = Pass 1 = Failed	
19	283	3	3	3	Window 18 Fault Flag	0 = Pass 1 = Failed	
19	283	3	4	4	Window 19 Warning Flag	0 = Pass 1 = Failed	
19	283	3	5	5	Window 19 Fault Flag	0 = Pass 1 = Failed	
19	283	3	6	6	Window 20 Warning Flag	0 = Pass 1 = Failed	

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
19	283	3	7	7	Widow 20 Fault Flag	0 = Pass 1 = Fail	
19	283	3	8	10	Window 21 Warning Flag	0 = Pass 1 = Fail	
19	283	3	9	11	Window 21 Fault Flag	0 = Pass 1 = Fail	
19	283	3	10	12	Window 22 Warning Flag	0 = Pass 1 = Fail	
19	283	3	11	13	Window 22 Fault Flag	0 = Pass 1 = Fail	
19	283	3	12	14	Window 23 Warning Flag	0 = Pass 1 = Fail	
19	283	3	13	15	Window 23 Fault Flag	0 = Pass 1 = Fail	
19	283	3	14	16	Window 24 Warning Flag	0 = Pass 1 = Failed	
19	283	3	15	17	Window 24 Fault Flag	0 = Pass 1 = Failed	
20	284	4	0	0	Gage 1 Warning Flag	0 = Pass 1 = Failed	
20	284	4	1	1	Gage 1 Fault Flag	0 = Pass 1 = Failed	
20	284	4	2	2	Gage 2 Warning Flag	0 = Pass 1 = Failed	
20	284	4	3	3	Gage 2 Fault Flag	0 = Pass 1 = Failed	
20	284	4	4	4	Gage 3 Warning Flag	0 = Pass 1 = Failed	
20	284	4	5	5	Gage 3 Fault Flag	0 = Pass 1 = Failed	
20	284	4	6	6	Gage 4 Warning Flag	0 = Pass 1 = Failed	
20	284	4	7	7	Gage 4 Fault Flag	0 = Pass 1 = Failed	
20	284	4	8	10	Gage 5 Warning Flag	0 = Pass 1 = Failed	
20	284	4	9	11	Gage 5 Fault Flag	0 = Pass 1 = Failed	

Discrete Bit Inputs (cont'd)

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
20	284	4	10	12	Gage 6 Warning Flag	0 = Pass 1 = Fail	
20	284	4	11	13	Gage 6 Fault Flag	0 = Pass 1 = Fail	
20	284	4	12	14	Gage 7 Warning Flag	0 = Pass 1 = Fail	
20	284	4	13	15	Gage 7 Fault Flag	0 = Pass 1 = Fail	
20	284	4	14	16	Gage 8 Warning Flag	0 = Pass 1 = Fail	
20	284	4	15	17	Gage 8 Fault Flag	0 = Pass 1 = Fail	
21	285	5	0	0	Gage 9 Warning Flag	0 = Pass 1 = Failed	
21	285	5	1	1	Gage 9 Fault Flag	0 = Pass 1 = Failed	
21	285	5	2	2	Gage 10 Warning Flag	0 = Pass 1 = Failed	
21	285	5	3	3	Gage 10 Fault Flag	0 = Pass 1 = Failed	
21	285	5	4	4	Gage 11 Warning Flag	0 = Pass 1 = Failed	
21	285	5	5	5	Gage 11 Fault Flag	0 = Pass 1 = Failed	
21	285	5	6	6	Gage 12 Warning Flag	0 = Pass 1 = Failed	
21	285	5	7	7	Gage 12 Fault Flag	0 = Pass 1 = Failed	
21	285	5	8	10	Gage 13 Warning Flag	0 = Pass 1 = Failed	
21	285	5	9	11	Gage 13 Fault Flag	0 = Pass 1 = Failed	
21	285	5	10	12	Gage 14 Warning Flag	0 = Pass 1 = Failed	
21	285	5	11	13	Gage 14 Fault Flag	0 = Pass 1 = Failed	
21	285	5	12	14	Gage 15 Warning Flag	0 = Pass 1 = Failed	

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
21	285	5	13	15	Gage 15 Fault Flag	0 = Pass 1 = Fail	
21	285	5	14	16	Gage 16 Warning Flag	0 = Pass 1 = Fail	
21	285	5	15	17	Gage 16 Fault Flag	0 = Pass 1 = Fail	
22	286	6	0	0	Gage 17 Warning Flag	0 = Pass 1 = Fail	
22	286	6	1	1	Gage 17 Fault Flag	0 = Pass 1 = Fail	
22	286	6	2	2	Gage 18 Warning Flag	0 = Pass 1 = Failed	
22	286	6	2	2	Gage 18 Fault Flag	0 = Pass 1 = Failed	
22	286	6	3	3	Gage 19 Warning Flag	0 = Pass 1 = Failed	
22	286	6	4	4	Gage 19 Fault Flag	0 = Pass 1 = Failed	
22	286	6	5	5	Gage 20 Warning Flag	0 = Pass 1 = Failed	
22	286	6	7	7	Gage 20 Fault Flag	0 = Pass 1 = Failed	
22	286	6	8	10	Gage 21 Warning Flag	0 = Pass 1 = Failed	
22	286	6	9	11	Gage 21 Fault Flag	0 = Pass 1 = Failed	
22	286	6	10	12	Gage 22 Warning Flag	0 = Pass 1 = Failed	
22	286	6	11	13	Gage 22 Fault Flag	0 = Pass 1 = Failed	
22	286	6	12	14	Gage 23 Warning Flag	0 = Pass 1 = Failed	
22	286	6	13	15	Gage 23 Fault Flag	0 = Pass 1 = Failed	
22	286	6	14	16	Gage 24 Warning Flag	0 = Pass 1 = Failed	
22	286	6	15	17	Gage 24 Fault Flag	0 = Pass 1 = Failed	

Discrete Bit Inputs (cont'd)

Table B.1
Discrete Bit Inputs

Word#			Bit#		Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O	DECIMAL	OCTAL			
Toolset 1	Toolset 2		PI	PLC			
23	287	7	0	0	Gage 25 Warning Flag	0 = Pass 1 = Fail	
23	287	7	1	1	Gage 25 Fault Flag	0 = Pass 1 = Fail	
23	287	7	2	2	Gage 26 Warning Flag	0 = Pass 1 = Fail	
23	287	7	3	3	Gage 26 Fault Flag	0 = Pass 1 = Fail	
23	287	7	4	4	Gage 27 Warning Flag	0 = Pass 1 = Failed	
23	287	7	5	5	Gage 27 Fault Flag	0 = Pass 1 = Failed	
23	287	7	6	6	Gage 28 Warning Flag	0 = Pass 1 = Failed	
23	287	7	7	7	Gage 28 Fault Flag	0 = Pass 1 = Failed	
23	287	7	8	10	Gage 29 Warning Flag	0 = Pass 1 = Failed	
23	287	7	9	11	Gage 29 Fault Flag	0 = Pass 1 = Failed	
23	287	7	10	12	Gage 30 Warning Flag	0 = Pass 1 = Failed	
23	287	7	11	13	Gage 30 Fault Flag	0 = Pass 1 = Failed	
23	287	7	12	14	Gage 31 Warning Flag	0 = Pass 1 = Failed	
23	287	7	13	15	Gage 31 Fault Flag	0 = Pass 1 = Failed	
23	287	7	14	16	Gage 32 Warning Flag	0 = Pass 1 = Failed	
23	287	7	15	17	Gage 32 Fault Flag	0 = Pass 1 = Failed	

Discrete Bit Outputs

There are 128 bits that can be set as outputs from a host device to control the operation of the CVIM. These bits control:

- Monitor display.
- Camera trigger.
- Toolset selection.
- I/O forcing.
- Selection of operation after reject.
- Memory storage location. RAM, EEPROM, RAM Card, or external host memory.

Refer to Tables B.2 and B.3. Table B.2 only applies to Backplane communications. Table B.3 applies to both Backplane and Remote I/O communications.

Table B.2
Backplane Handshake Bits

Word #		Bit # PI	Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O				
0	N/A	0	Host Data Lock – Toolset #1	0 = CVIM May Write 1 = CVIM Write Inhibited	The backplane host should write to bit 0 to prevent the CVIM from modifying toolset 1 results data.
0	N/A	1	Host Data Lock – Toolset #2	0 = CVIM May Write 1 = CVIM Write Inhibited	The backplane host should write to bit 1 to prevent the CVIM from modifying Toolset 2 results data.
0	N/A	2	CVIM Data Lock – Toolset #1	0 = CVIM Not Writing 1 = CVIM Writing	The CVIM sets bit 2 while writing to Toolset 1. The bit will be set back to 0 after writing.
0	N/A	3	CVIM Data Lock – Toolset #2	0 = CVIM Not Writing 1 = CVIM Writing	The CVIM sets bit 3 while writing while writing to Toolset 2. The bit will be set back to 0 after writing.
0	N/A	4–15	Reserved		

Discrete Bit Outputs (cont'd)

Note: When using the backplane, don't write directly to word 2. Write to word 1023. The CVIM will copy the data from word 1023 to word 2.

Note: When communicating with a device through the Pyramid Integrator backplane, CVIM module results are posted in shared memory immediately after processing. When communicating with a device through the other ports, results are only available at the end of the inspection program.

Table B.3
Discrete Bit Outputs

Word #		Bit #		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL	OCTAL			
		PI	PLC			
2	0	0 – 7	0 – 7	Not Used		
2	0	8	10	Lock Request	0 = No Change 1 = Lock	Do not set bits 8 and 9 at the same time. If you set both bits, the CVIM will be unlocked. You must reset this bit to 0, then back to 1 to repeat a lock request.
2	0	9	11	Unlock Request	0 = No Change 1 = Unlock	You must reset this bit to 0, then back to 1 to repeat an unlock request.
2	0	10	12	Host Trigger Toolset 1	0 = No Trigger 1 = Trigger	You must reset this bit to 0, then back to 1 to repeat a trigger request.
2	0	11	13	Host Trigger Toolset 2	0 = No Trigger 1 = Trigger	You must reset this bit to 0, then back to 1 to repeat a trigger request.
2	0	12	14	Light Pen Request	0 = No Request 1 = Request	Bit 12 commands are specified in: Output words 1 and 6 (Remote I/O) Output words 3 and 8 (Backplane) You must reset bit to 0, then back to 1 to repeat a light pen request.

Table B.3
Discrete Bit Outputs

Word #		Bit #		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC			
2	0	13	15	I/O Request	0 = No Request 1 = Request	Bit 13 commands are specified in: Output words 2, 4, and 5 (Remote I/O) Output words 4, 6, and 7 (Backplane) You must reset bit to 0, then back to 1 to repeat an I/O request.
2	0	14	16	Configuration Move	0 = No Request 1 = Request	Bit 14 commands are specified in: Output word 2 (Remote I/O) Output word 4 (Backplane) You must reset bit to 0, then back to 1 to repeat a configuration move request.
2	0	15	17	Not Used		

Discrete Bit Outputs (cont'd)

Table B.3
Discrete Bit Outputs

Word#		Bit#		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL	OCTAL			
		PI	PLC			
3	1	0 – 7	0 – 7	Runtime Display Control	00000000 = No Change 00000001 = Display Image Only 00000010 = Display Failed Tool 00000100 = Display All Tools 00001000 = Display I/O Page 00010000 = Display Results Page 00100000 = Display Statistics 1 Page 01000000 = Display Statistic 2 Page	

Table B.3
Discrete Bit Outputs

Word#		Bit#		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC			
3	1	8 – 15	10 – 17	Freeze on Reject Control	00000000 = No Change 00000001 = Go On Reject 00000010 = Freeze First Reject 00000100 = Freeze All Rejects 00001000 = Freeze Next Image 00010000 = Halt On Reject	
4 Not Applicable to Backplane	2	N/A	0 – 3	Toolset Flag Select	0000 = Not Valid 0001 = Toolset 1 0010 = Toolset 2	Bits 0 – 3 control which set of toolset results are assigned to discrete input words 1 through 7.
4	2	4 – 5	4 – 5	Discrete I/O Control	00 = No Change 01 = Disable Outputs 10 = Enable Outputs	Bits 4 through 7 refer to the local I/O module (Catalog Number 1771–JMB).
4	2	6 – 7	6 – 7	Forced I/O Control	00 = No Change 01 = Disable Forces 10 = Enable Forces	Bits 6 and 7 act upon the bitmap you set up in words: 4 and 5 (Remote I/O) 6 and 7 (Backplane)

Discrete Bit Outputs (cont'd)

Table B.3
Discrete Bit Outputs

Word#		Bit#		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC			
4	2	8 – 11	10 – 13	Configuration Move Control	0000 = No Request 0001 = EEPROM to RAM 0010 = RAM to EEPROM 0100 = RAM Card to RAM 1000 = RAM to RAM Card	
4	2	12 – 15	14 – 17	RAM Card Index	0000 = RAM Card Configuration 1 1111 = RAM Card Configuration 16	Bits 12 through 15 are numeric fields which specify RAM card locations 1 through 16.

Table B.3
Discrete Bit Outputs

Word #		Bit #				
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC	Definition	Usage	Notes
5 Reference only. Not used for Backplane.	3	N/A	0 – 2	Block Transfer Type	000 = Invalid Request 001 = Results Block 010 = Configuration Block 100 = Template Block 101 = Statistics Block – See Notes 111 = Programmable Block Transfer – See Notes	This word only applies to Remote I/O interface. To send programmable block transfer to the CVIM, set bits 0, 1, and 2 to equal 111 and then send a block transfer write. No other bits (last block, toolset, or block number) need to be sent. After sending a configuration block or template block to the CVIM, you should check the condition of the configuration fault bit (Word 0, Discrete Bit Inputs) To send programmable statistics , set bits 0, 1, and 2 to equal 101 and then send a block transfer write to the CVIM. No other bits (last block , toolset, or block number) need to be sent.
5 Reference only. Not used for Backplane.	3	N/A	3	Last Block Flag	0 = Not Last Block 1 = Last Block	Bit 3 only applies to block transfer writes.
5 Reference only. Not used for Backplane.	3	N/A	4 – 7	Toolset Request	0000 = No Toolset 0001 = Toolset 1 0010 = Toolset 2	Bits 4 through 7 only apply to results block.
5 Reference only. Not used for Backplane.	3	N/A	10 – 17	Block Transfer Block Number	0 = Invalid Number See Notes	Results block numbers may be from 1 to 4. Configuration block numbers may be from 1 to 136. Template block numbers may be from 1 to 255 (variable).

Discrete Bit Outputs (cont'd)

Table B.3
Discrete Bit Outputs

Word#		Bit#		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC			
6	4	0	0	Local I/O Output 1 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	1	1	Local I/O Output 2 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	2	2	Local I/O Output 3 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	3	3	Local I/O Output 4 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	4	4	Local I/O Output 5 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	5	5	Local I/O Output 6 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	6	6	Local I/O Output 7 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	7	7	Local I/O Output 8 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	8	10	Local I/O Output 9 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	9	11	Local I/O Output 10 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	10	12	Local I/O Output 11 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	11	13	Local I/O Output 12 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	12	14	Local I/O Output 13 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	13	15	Local I/O Output 14 Force ON	0 = No Force 1 = Force On	Refers to Catalog Number 1771–JMB Local I/O Board.
6	4	14 – 15	16 – 17	Not Used		

Table B.3
Discrete Bit Outputs

Word#		Bit#		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC			
7	5	0	0	Local I/O Output 1 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	1	1	Local I/O Output 2 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	2	2	Local I/O Output 3 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	3	3	Local I/O Output 4 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	4	4	Local I/O Output 5 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	5	5	Local I/O Output 6 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	6	6	Local I/O Output 7 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	7	7	Local I/O Output 8 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	8	10	Local I/O Output 9 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	9	11	Local I/O Output 10 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	10	12	Local I/O Output 11 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	11	13	Local I/O Output 12 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	12	14	Local I/O Output 13 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	13	15	Local I/O Output 14 Force OFF	0 = No Force 1 = Force Off	Refers to Catalog Number 1771–JMB Local I/O Board.
7	5	14 – 15	16 – 17	Not Used		

Discrete Bit Outputs (cont'd)

Table B.3
Discrete Bit Outputs

Word#		Bit#		Definition	Usage	Notes
PI Backplane	RS–232 and Remote I/O	DECIMAL PI	OCTAL PLC			
8	6	0 – 1	0 – 1	Toolset Display Control	00 = No Change 01 = Display Toolset 1 10 = Display Toolset 2	
8	6	2	2	Not Used		
8	6	3	3	Resume Control	0 = No Change 1 = Resume	Use in conjunction with light pen request: Word 0 bit 12 (Remote) Word 2 bit 12 (Backplane)
8	6	4 – 5	4 – 5	Page Control	00 = No Change 01 = Page Up 10 = Page Down	Use in conjunction with light pen request: Word 0 bit 12 (Remote) Word 2 bit 12 (Backplane)
8	6	6	6	Reset Statistics	0 = No Change 1 = Reset	Use in conjunction with light pen request: Word 0 bit 12 (Remote) Word 2 bit 12 (Backplane)
8	6	7	7	Reset Counters	0 = No Change 1 = Reset	Use in conjunction with light pen request: Word 0 bit 12 (Remote) Word 2 bit 12 (Backplane)
8	6	8 – 15	10 – 17	Not Used		

Numerical Results Data

Results Block Overview

There are 4 results blocks for each toolset. The following is an overview of the blocks.

Block Number 1 Contains:

- Block Transfer Signature
- Discrete Output Copy
- Brightness Probe
- X/Y Reference Lines #1, #2, and #3
- Feature Finder #1
- Windows 1–8
- Gages 1–8
- Total Number of Triggers, Missed Triggers, and Master Faults

Block #2 Contains:

- Block Transfer Signature
- Windows 9–16
- Gages 9–22
- Windows 17–24
- Total Number of Triggers

Block #3 Contains:

- Block Transfer Signature
- Gages 23–32
- Feature Finders #2 and #3
- Total Number of Triggers, Missed Triggers, and Master Faults

Block #4 (Remote I/O Port only) Contains:

- Block Transfer Signature
- Status Word
- User Programmable Block Transfer Data
- Total Number of Triggers

Block Transfer Signature

The block transfer signature is for user information only. The CVIM places the signature in each block sent to the PLC for identification and does not care if the PLC changes the signature prior to sending a block back to the CVIM.

Bits 0–7 designate the block number:

00000000 = Not Valid
00000001 = Block #1
00000010 = Block #2
. . .
11111111 = Block #255

Bits 8–10 designate the toolset:

000 = Not Valid
001 = Toolset #1
010 = Toolset #2
011 to 111 = Reserved

Bits 11 and 12 designate the block type:

00 = Results
01 = Configuration
10 = Templates
11 = Statistics

Bits 13–15 specify the module thumbwheel number:

000 to 111

Results Block #1

Table C.1 shows the function of each word in Results Block #1.

Table C.1
Numerical Results Data – Results Block 1

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
24	288	0	0 – 15	Block Transfer Signature		
25	289	1	0 – 15	Reserved		
26	290	2	0 – 15	Brightness Probe Integer		16 . 16 value. Refer to Appendix A.
27	291	3	0 – 15	Brightness Probe Fraction		16 . 16 value. Refer to Appendix A.
28	292	4	0 – 15	Reference Line #1X Position		From upper left corner of display. 16 bit integer.
29	293	5	0 – 15	Reference Line #1Y Position		From upper left corner of display. 16 bit integer.
30	294	6	0 – 15	Reference Line #2X Position		From upper left corner of display. 16 bit integer.
31	295	7	0 – 15	Reference Line #2Y Position		From upper left corner of display. 16 bit integer.
32	296	8	0 – 15	Reference Line #3X Position		From upper left corner of display. 16 bit integer.
33	297	9	0 – 15	Reference Line #3Y Position		From upper left corner of display. 16 bit integer.
34	298	10	0 – 15	Reference Window #1X1 Position		Feature #1. From upper left corner. 16 bit integer.
35	299	11	0 – 15	Reference Window #1Y1 Position		Feature #1. From upper left corner. 16 bit integer.
36	300	12	0 – 15	Reference Window #1X2 Position		Feature #2. From upper left corner. 16 bit integer.
37	301	13	0 – 15	Reference Window #1Y2 Position		Feature #2. From upper left corner. 16 bit integer.
38	302	14	0 – 15	Reference Window #1X3 Position		Feature #3. From upper left corner. 16 bit integer.
39	303	15	0 – 15	Reference Window #1Y3 Position		Feature #3. From upper left corner. 16 bit integer.
40	304	16	0 – 15	Reference Window #1X–Center		Centroid of enabled feature. 16 bit integer.
41	305	17	0 – 15	Reference Window #1Y–Center		Centroid of enabled feature. 16 bit integer.
42	306	18	0 – 15	Reference Window #1 Theta Integer		16 . 16 value. Refer to Appendix A. Only if 2 or 3 Features are enabled.
43	307	19	0 – 15	Reference Window #1 Theta Faction		Only if 2 or 3 Features are enabled.

Results Block #1 (cont’d)

Table C.1
Numerical Results Data – Results Block 1

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
44	308	20	0	Reference Window #1X1/Y1 Pass/Fail Bit	0 = Pass 1 = Fail	Feature 1.
44	308	20	1	Reference Window #1X2/Y2 Pass/Fail Bit	0 = Pass 1 = Fail	Feature 2.
44	308	20	2	Reference Window #1X3/Y3 Pass/Fail Bit	0 = Pass 1 = Fail	Feature 3.
44	308	20	3	Combination Pass/Fail Bit	0 = Pass 1 = Fail	0 if all enabled features pass.
44	308	20	4 – 15	Not Used		
45	309	21	0 – 15	Score Reference Window #1, Feature #1		
46	310	22	0 – 15	Score Reference Window #1, Feature #2		
47	311	23	0 – 15	Score Reference Window #1, Feature #3		
48 – 49	312 – 313	24 – 25	0 – 15	Window #1 Value		Luminance – 16 . 16 Object – 32 bit integer Pixel – 32 bit integer Template – 32 bit integer Gradient – 32 bit integer
50 – 51	314 – 315	26 – 27	0 – 15	Window #2 Value		Same as window #1.
52 – 53	316 – 317	28 – 29	0 – 15	Window #3 Value		Same as window #1.
54 – 55	318 – 319	30 – 31	0 – 15	Window #4 Value		Same as window #1.
56 – 57	320 – 321	32 – 33	0 – 15	Window #5 Value		Same as window #1.
58 – 59	322 – 323	34 – 35	0 – 15	Window #6 Value		Same as window #1.
60 – 61	324 – 325	36 – 37	0 – 15	Window #7 Value		Same as window #1.
62 – 63	326 – 327	38 – 39	0 – 15	Window #8 Value		Same as window #1.

Table C.1
Numerical Results Data – Results Block 1 (cont'd)

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
64 – 65	328 – 329	40 – 41	0 – 15	Gage #1 Value		Angular & Linear Measure – 16 . 16 Edge – 32 bit integer Object – 32 bit integer Pixel – 32 bit integer X Position & Y Position – 16 . 16
66 – 67	330 – 331	42 – 43	0 – 15	Gage #2 Value		Same as Gage #1.
68 – 69	332 – 333	44 – 45	0 – 15	Gage #3 Value		Same as Gage #1.
70 – 71	334 – 335	46 – 47	0 – 15	Gage #4 Value		Same as Gage #1.
72 – 73	335 – 336	48 – 49	0 – 15	Gage #5 Value		Same as Gage #1.
74 – 75	337 – 338	50 – 51	0 – 15	Gage #6 Value		Same as Gage #1.
76 – 77	339 – 340	52 – 53	0 – 15	Gage #7 Value		Same as Gage #1.
78 – 79	341 – 342	54 – 55	0 – 15	Gage #8 Value		Same as Gage #1.
80	344	56	0 – 15	Reserved		
81	345	57	0 – 15	Reserved		
82	346	58	0 – 15	Reserved		
83	347	59	0 – 15	Missed Triggers		
84	348	60	0 – 15	Task Master Faults – Most Significant Word		32 bit integer. Value is incremented if at least one tool fails.
85	349	61	0 – 15	Task Master Faults – Least Significant Word		
86	350	62	0 – 15	Task Triggers – Most Significant Word		32 bit integer.
87	351	63	0 – 15	Task Triggers – Least Significant Word		

Results Block #2

Table C.2 shows the function of each word in results block #2.

Table C.2
Numerical Results Data – Results Block 2

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
88	352	0	0 – 15	Block Transfer Signature		
89	353	1	0 – 15	Reserved		
90 – 91	354 – 355	2 – 3	0 – 15	Window #9 Value		Luminance – 16 . 16 Object – 32 bit integer Pixel – 32 bit integer Template – 32 bit integer Gradient – 32 bit integer
92 – 93	356 – 357	4 – 5	0 – 15	Window #10 Value		Same as Window #9.
94 – 95	358 – 359	6 – 7	0 – 15	Window #11 Value		Same as Window #9.
96 – 97	360 – 361	8 – 9	0 – 15	Window #12 Value		Same as Window #9.
98 – 99	362 – 363	10 – 11	0 – 15	Window #13 Value		Same as Window #9.
100 – 101	364 – 365	12 – 13	0 – 15	Window #14 Value		Same as Window #9.
102 – 103	366 – 367	14 – 15	0 – 15	Window #15 Value		Same as Window #9.
104 – 105	368 – 369	16 – 17	0 – 15	Window #16 Value		Same as Window #9.
106 – 107	370 – 371	18 – 19	0 – 15	Gage #9 Value		Angular & Linear Measure – 16 . 16 Edge – 32 bit integer Object – 32 bit integer Pixel – 32 bit integer X Position & Y Position – 16 . 16
108 – 109	372 – 373	20 – 21	0 – 15	Gage #10 Value		Same as Gage #9.
110 – 111	374 – 375	22 – 23	0 – 15	Gage #11 Value		Same as Gage #9.
112 – 113	376 – 377	24 – 25	0 – 15	Gage #12 Value		Same as Gage #9.
114 – 115	378 – 379	26 – 27	0 – 15	Gage #13 Value		Same as Gage #9.
116 – 117	380 – 381	28 – 29	0 – 15	Gage #14 Value		Same as Gage #9.
118 – 119	382 – 383	30 – 31	0 – 15	Gage #15 Value		Same as Gage #9.
120 – 121	384 – 385	32 – 33	0 – 15	Gage #16 Value		Same as Gage #9.
122 – 123	386 – 387	34 – 35	0 – 15	Window #17 Value		Luminance – 16 . 16 Object – 32 bit integer Pixel – 32 bit integer Template – 32 bit integer Gradient – 32 bit integer
124 – 125	388 – 389	36 – 37	0 – 15	Window #18 Value		Same as Window #17.
126 – 127	390 – 391	38 – 39	0 – 15	Window #19 Value		Same as Window #17.

Table C.2
Numerical Results Data – Results Block 2

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
128 – 129	392 – 393	40 – 41	0 – 15	Window #20 Value		Same as Window #17.
130 – 131	394 – 395	42 – 43	0 – 15	Window #21 Value		Same as Window #17.
132 – 133	396 – 397	44 – 45	0 – 15	Window #22 Value		Same as Window #17.
134 – 135	398 – 399	46 – 47	0 – 15	Window #23 Value		Same as Window #17.
136 – 137	400 – 401	48 – 49	0 – 15	Window #24 Value		Same as Window #17.
138 – 139	402 – 403	50 – 51	0 – 15	Gage #17 Value		Angular & Linear Measure – 16 . 16 Edge – 32 bit integer Object – 32 bit integer Pixel – 32 bit integer X Position & Y Position – 16 . 16
140 – 141	404 – 405	52 – 53	0 – 15	Gage #18 Value		Same as Gage #17.
142 – 143	406 – 407	54 – 55	0 – 15	Gage #19 Value		Same as Gage #17.
144 – 145	408 – 409	56 – 57	0 – 15	Gage #20 Value		Same as Gage #17.
146 – 147	410 – 411	58 – 59	0 – 15	Gage #21 Value		Same as Gage #17.
148 – 149	412 – 413	60 – 61	0 – 15	Gage #22 Value		Same as Gage #17.
150	414	62	0 – 15	Task Trigger – Most Significant Word		32 bit integer.
151	415	63	0 – 15	Task Trigger – Least Significant Word		

Results Block #3

Table C.3 shows the function of each word in results block #3.

Table C.3
Numerical Results Data – Results Block 3

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
152	416	0	0 – 15	Block Transfer Signature		
153	417	1	0 – 15	Reserved		
154 – 155	418 – 419	2 – 3	0 – 15	Gage #23 Value		Angular & Linear Measure – 16 . 16 Edge – 32 bit integer Object – 32 bit integer Pixel – 32 bit integer X Position & Y Position – 16 . 16
156 – 157	420 – 421	4 – 5	0 – 15	Gage #24 Value		Same as Gage #23
158 – 159	422 – 423	6 – 7	0 – 15	Gage #25 Value		Same as Gage #23
160 – 161	424 – 425	8 – 9	0 – 15	Gage #26 Value		Same as Gage #23
162 – 163	426 – 427	10 – 11	0 – 15	Gage #27 Value		Same as Gage #23
164 – 165	428 – 429	12 – 13	0 – 15	Gage #28 Value		Same as Gage #23
166 – 167	430 – 431	14 – 15	0 – 15	Gage #29 Value		Same as Gage #23
168 – 169	432 – 433	16 – 17	0 – 15	Gage #30 Value		Same as Gage #23
170 – 171	434 – 435	18 – 19	0 – 15	Gage #31 Value		Same as Gage #23
172 – 173	436 – 437	20 – 21	0 – 15	Gage #32 Value		Same as Gage #23
174	438	22	0 – 15	Reference Window #2X1 Position		Feature #1 16 bit integer
175	439	23	0 – 15	Reference Window #2Y1 Position		Feature #1 16 bit integer
176	440	24	0 – 15	Reference Window #2X2 Position		Feature #2 16 bit integer
177	441	25	0 – 15	Reference Window #2Y2 Position		Feature #2 16 bit integer
178	442	26	0 – 15	Reference Window #2X3 Position		Feature #3 16 bit integer
179	443	27	0 – 15	Reference Window #2Y3 Position		Feature #3 16 bit integer
180	444	28	0 – 15	Reference Window #2X – Center		Centroid of enabled feature. 16 bit integer.
181	445	29	0 – 15	Reference Window #2Y – Center		Centroid of enabled feature. 16 bit integer.

Table C.3
Numerical Results Data – Results Block 3

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
182	446	30	0 – 15	Reference Window #2 Theta (Integer)		16 . 16 value. Refer to Appendix A. Only if 2 or 3 features are enabled.
183	447	31	0 – 15	Reference Window #2 (Fraction)		Only if 2 or 3 features are enabled.
184	448	32	0	Reference Window #2X1/Y1 Pass/Fail Bit	0 = Pass 1 = Fail	Feature #1.
184	448	32	1	Reference Window #2X2/Y2 Pass/Fail Bit	0 = Pass 1 = Fail	Feature #2.
184	448	32	2	Reference Window #2X3/Y3 Pass/Fail Bit	0 = Pass 1 = Fail	Feature #3.
184	448	32	3	Combination Pass/Fail Bit	0 = Pass 1 = Fail	0 if all enabled features pass.
184	448	32	4 – 15	Not Used		
185	449	33	0 – 15	Score Reference Window #2, Feature #1		
186	450	34	0 – 15	Score Reference Window #2, Feature #2		
187	451	35	0 – 15	Score Reference Window #2, Feature #3		
188	452	36	0 – 15	Reference Window #3X1 Position		Feature #1 16 bit integer.
189	453	37	0 – 15	Reference Window #3Y1 Position		Feature #1 16 bit integer.
190	454	38	0 – 15	Reference Window #3X2 Position		Feature #2 16 bit integer.
191	455	39	0 – 15	Reference Window #3Y2 Position		Feature #2 16 bit integer.
192	456	40	0 – 15	Reference Window #3X3 Position		Feature #3 16 bit integer.
193	457	41	0 – 15	Reference Window #3Y3 Position		Feature #3 16 bit integer.

Results Block #3 (cont'd)

Table C.3
Numerical Results Data – Results Block 3

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
194	458	42	0 – 15	Reference Window #3X – Center		Centroid of enabled feature. 16 bit integer.
195	459	43	0 – 15	Reference Window #3Y – Center		Centroid of enabled feature. 16 bit integer.
196	460	44	0 – 15	Reference Window #3 Theta (Integer)		16 . 16 value. Refer to Appendix A. Only if 2 or 3 Features are enabled.
197	461	45	0 – 15	Reference Window #3 (Fraction)		Only if 2 or 3 Features are enabled.
198	462	46	0	Reference Window X1/Y1 Pass/Fail Bit	0 = Pass 1 = Fail	Feature #1 16 bit integer.
198	462	46	1	Reference Window X2/Y2 Pass/Fail Bit	0 = Pass 1 = Fail	Feature #2 16 bit integer.
198	462	46	2	Reference Window X3/Y3 Pass/Fail Bit	0 = Pass 1 = Fail	Feature #3 16 bit integer.
198	462	46	3	Combination Pass/Fail Bit	0 = Pass 1 = Fail	0 if all enabled tools pass.
198	462	46	4 – 15	Not Used		
199	463	47	0 – 15	Score Reference Window #3, Feature #1		
200	464	48	0 – 15	Score Reference Window #3, Feature #2		
201	465	49	0 – 15	Score Reference Window #3, Feature #3		
202 – 209	466 – 473	50 – 57	0 – 15	Reserved		
210	474	58	0 – 15	Reserved		
211	475	59	0 – 15	Task Missed Triggers		16 bit integer.
212	476	60	0 – 15	Task Masters Faults – Most Significant Word		32 bit integer. Increments if at least 1 tool fails.

Table C.3
Numerical Results Data – Results Block 3

Word #			Bit #	Definition	Usage	Notes
PI Backplane		RS–232 and Remote I/O				
Toolset 1	Toolset 2					
213	477	61	0 – 15	Task Master Faults – Least Significant Word		
214	478	62	0 – 15	Task Triggers – Most Significant Word		32 bit integer. Refer to Appendix A.
215	479	63	0 – 15	Task Triggers – Least Significant Word		

Results Block #4

The fourth results block may be formatted and read through any of the communications ports. This block has user defined contents. This block may contain up to 64 words of data; see Table C.4.

Table C.4
Numerical Results Data – Results Block #4

Word #	Bit #	Function	Notes
0	0 – 15	Block Transfer Signature	
1	0 – 15	Status	The contents of this word will be 0 if all of the requested data fits in the 64 word results block. Otherwise, the contents will be non–0.
2 – 61	0 – 15	User Defined Inspection Results Data	
62 – 63	0 – 15	Task Triggers (32 Bits)	

Each type of result requires a specific number of words. Use the following as a guideline when setting up a programmable block transfer.

	Number of Words	Format
Module Status	1 Word	
X/Y Reference Line	2 Words	16 Bit Integer
Window	2 Words	
	(Object & Pixel)	32 Bit Integer
	(Luminance)	16 Point 16
	(Template)	32 Bit Integer
	(Gradient)	32 Bit Integer
Brightness Probe	2 Words	16 Point 16
Reference Window	14 Words	16 Bit Integer and 16 Point 16
Gage	2 Words	
	(Linear Measure)	16 Point 16
	(Edge, Pixel, Object)	32 Bit Integer
	X, Y Position	16 Point 16

If you request more results than will fit into 62 words, the CVIM will truncate the data and set an error bit in word 1.

Note: Refer to discrete output word description (word 3) on Page 4 – 18 for information on how to send a programmable block format request to the CVIM.

Note: Words 2 through 5 select tools for Toolset 1. Words 6 through 9 select tools for Toolset 2.

Note: Word 0, block signature and word 1 are not used.

Use Table C.5 to set the contents of the programmable results block and statistics block.

Table C.5
Programmable Results/Statistics Block Configuration

1771 Node Adapter Results or Statistics			Function	RS–232 Results or Statistics		
Word # Toolset 1	Word # Toolset 2	Bit #		Byte # Toolset 1	Byte # Toolset 2	Bit #
2	6	0	Module Status (NA for statistics)	0	8	0
2	6	1	Light Probe	0	8	1
2	6	2	Reference Line 1 (NA for statistics)	0	8	2
2	6	3	Reference Line 2 (NA for statistics)	0	8	3
2	6	4	Reference Line 3 (NA for statistics)	0	8	4
2	6	5	Reference Window 1	0	8	5
2	6	6	Reference Window 2	0	8	6
2	6	7	Reference Window 3	0	8	7
2	6	8	Window 1	1	9	0
2	6	9	Window 2	1	9	1
2	6	10	Window 3	1	9	2
2	6	11	Window 4	1	9	3
2	6	12	Window 5	1	9	4
2	6	13	Window 6	1	9	5
2	6	14	Window 7	1	9	6
2	6	15	Window 8	1	9	7
3	7	0	Window 9	2	10	0
3	7	1	Window 10	2	10	1
3	7	2	Window 11	2	10	2
3	7	3	Window 12	2	10	3
3	7	4	Window 13	2	10	4
3	7	5	Window 14	2	10	5
3	7	6	Window 15	2	10	6
3	7	7	Window 16	2	10	7
3	7	8	Window 17	3	11	0
3	7	9	Window 18	3	11	1
3	7	10	Window 19	3	11	2
3	7	11	Window 20	3	11	3

Results Block #4 (cont'd)

Table C.5
Programmable Results/Statistics Block Configuration

1771 Node Adapter Results or Statistics			Function	RS–232 Results or Statistics		
Word # Toolset 1	**Word # Toolset 2**	**Bit #**		**Byte # Toolset 1**	**Byte # Toolset 2**	**Bit #**
3	7	12	Window 21	3	11	4
3	7	13	Window 22	3	11	5
3	7	14	Window 23	3	11	6
3	7	15	Window 24	3	11	7
4	8	0	Gage 1	4	12	0
4	8	1	Gage 2	4	12	1
4	8	2	Gage 3	4	12	2
4	8	3	Gage 4	4	12	3
4	8	4	Gage 5	4	12	4
4	8	5	Gage 6	5	13	5
4	8	6	Gage 7	5	13	6
4	8	7	Gage 8	5	13	7
4	8	8	Gage 9	5	13	0
4	8	9	Gage 10	5	13	1
4	8	10	Gage 11	5	13	2
4	8	11	Gage 12	5	13	3
4	8	12	Gage 13	5	13	4
4	8	13	Gage 14	5	13	5
4	8	14	Gage 15	5	13	6
4	8	15	Gage 16	5	13	7
5	9	0	Gage 17	6	14	0
5	9	1	Gage 18	6	14	1
5	9	2	Gage 19	6	14	2
5	9	3	Gage 20	6	14	3
5	9	4	Gage 21	6	14	4
5	9	5	Gage 22	6	14	5
5	9	6	Gage 23	6	14	6
5	9	7	Gage 24	6	14	7
5	9	8	Gage 25	7	15	0
5	9	9	Gage 26	7	15	1
5	9	10	Gage 27	7	15	2

Table C.5
Programmable Results/Statistics Block Configuration

1771 Node Adapter Results or Statistics			Function	RS–232 Results or Statistics		
Word # Toolset 1	**Word # Toolset 2**	**Bit #**		**Byte # Toolset 1**	**Byte # Toolset 2**	**Bit #**
5	9	11	Gage 28	7	15	3
5	9	12	Gage 29	7	15	4
5	9	13	Gage 30	7	15	5
5	9	14	Gage 31	7	15	6
5	9	15	Gage 32	7	15	7

Statistics Block

The Statistics Block can be formatted and read through any of the communications ports. The Statistics Block has user defined contents. This block may contain up to 64 words of data; see Table C.6.

Table C.6
Statistics Block

Word #	Bit #	Function	Notes
0	0 – 15	Block Transfer Signature	
1 – 63	0 – 15	User Defined Statistics Data	

Each type of statistic requires a specific number of words. Use the following as a guideline when setting up a programmable block transfer.

Block Signature	= 1 Word
Number of Samples	= 1 Word (2 Byte Integer)
Minimum and Maximum	= 4 Bytes
Window (Object and Pixel)	32 Bit Integer
Window (Luminance)	16 Point 16
Widow (Template)	32 Bit Integer
Brightness Probe	16 Point 16
Reference Window	16 Bit Integer & 16 Point 16
Gage (Linear Measure)	16 Point 16
Gage (Edge, Pixel, Object)	32 Bit Integer
Average	= 4 Bytes (16 Point 16 for fixed point values)
Standard Deviation	= 4 Bytes (16 Point 16)

Each tool statistic consists of 18 bytes with the exception of reference windows which contain 18 bytes for each feature or a total of 54 bytes. The statistics block is transmitted as two hexadecimal characters for each byte. The total number of bytes including the block signature should not exceed 128 bytes.

Configuration Data

Configuration Block Overview

There are 135 configuration blocks. The following is an overview of the blocks.

Block Number 1	System Environment (45 words).
Block Numbers 2 and 3	Camera A and B Definition (61 words each camera).
Block Numbers 4 through 6	Toolset 1 Reference Lines 1 through 3 (30 words).
Block Numbers 7 through 9	Toolset 1 Reference Windows 1 through 3 (36 words).
Block Number 10 through 41	Toolset 1 Gages 1 through 32 (28 words).
Block Numbers 42 through 65	Toolset 1 Windows 1 through 24 (37 words).
Block Numbers 66 through 68	Toolset 2 Reference Lines 1 through 3 (30 words).
Block Numbers 69 through 71	Toolset 2 Reference Windows 1 through 3 (36 words).
Block Numbers 72 through 103	Toolset 2 Gages 1 through 32 (28 words).
Block Numbers 104 through 127	Toolset 2 Windows 1 through 24 (37 words).
Block Numbers 128 through 135	Polygon Blocks 1 through 8 (37 words).

Note: When reading the configuration blocks, the PLC should set the block length to 0. This will allow the CVIM module to set the block size based upon the block number received. The CVIM module will then only send the amount of data required for each block type. This helps reduce the overall transfer time when writing. If the PLC sets the block length to 0 (64 words) when writing to a CVIM module, the CVIM module will ignore excess data at the end of each block. Alternatively, the PLC may set the exact length to reduce the transfer time.

Configuration Block #1

Table D.1 shows the function of each word in the system environment configuration block.

Table D.1
Configuration Block #1– System Environment

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1–3	0–15	Reserved		
4	0–7	Configuration ID (Char. 2)		
4	8–15	Configuration ID (Char. 1)		
5	0–7	Configuration ID (Char. 4)		
5	8–15	Configuration ID (Char. 3)		
6	0–7	Configuration ID (Char. 6)		
6	8–15	Configuration ID (Char. 5)		
7	0–7	Configuration ID (Char. 8)		
7	8–15	Configuration ID (Char. 7)		
8	0–7	Configuration ID (Char. 10)		
8	8–15	Configuration ID (Char. 9)		
9	0–7	Configuration ID (Char. 12)		
9	8–15	Configuration ID (Char. 11)		
10	0–7	Configuration ID (Char. 14)		
10	8–15	Configuration ID (Char. 13)		
11	0–7	Configuration ID (Char. 16)		
11	8–15	Configuration ID (Char. 15)		
12	0–15	Reserved		
13	0	Monitor Type	0 = B/W 1 = Color	
13	1	Remote I/O Enable	0 = Disabled 1 = Enabled	
13	2 & 4	Protocol–Port A	0 = ASCII 1 = DF1 2 = Not Specified 3 = Not Specified	Port A, see bits 5 & 6 for port B.
13	3	Host Standby Mode	0 = Standby Enabled 1 = Standby Disabled	
13	5 & 6	Protocol–Port B	0 = ASCII 1 = DF1 2 = Not Specified 3 = Not Specified	Port B, see bits 2 & 4 for port A.
13	7–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Table D.1
Configuration Block #1– System Environment (cont'd)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
14–15	0–15	Reserved		
16	0–7	Toolset Display Status	0 = Disabled 1 = Enabled	All tools appear simultaneously while setting a given tool.
16	8–15	Reserved		
17	0–7	Reserved		
17	8	Learn Mode	0 = Disabled 1 = Enabled	
17	9	Outputs Enable	0 = Disabled 1 = Enabled	
17	10	Freeze Enable	0 = Disabled 1 = Enabled	
17	11	Halt Enable	0 = Disabled 1 = Enabled	
17	12–15	Runtime Toolset Display	0 = Toolset 1 1 = Toolset 2	
18–31	0–15	Reserved		
32	0–15	Toolset 1 Pulse Width		In milliseconds.
33	0–15	Toolset 2 Pulse Width		In milliseconds.
34–44	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 2 & 3

Tables D.2 shows the function of each word in the camera definition configuration blocks.

Table D.2
Configuration Block #2 & 3 – Camera Definition

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0–7	Camera Low Reference	0 = Minimum Value . . . 100 = Maximum Value	These values do not correspond with the display on the help screen.
1	8–15	Camera High Reference	105 = Minimum Value . . . 255 = Maximum Value	These values do not correspond with the display on the help screen.
2	0–7	Light Probe Status	0 = Disabled 1 = Same Field 2 = Next Field	
2	8–15	Reserved		
3	0–15	Light Probe X Location	16 = Minimum Value . . . 504 = Maximum Value	
4	0–15	Light Probe Y Location	8 = Minimum Value . . . 232 = Maximum Value	
5–9	0–15	Reserved		
10	0–15	Fail Range High (Integer)		Words 10 and 11 represent a 16 (bit) . 16 (bit) fixed point decimal value.
11	0–15	Fail Range High (Fraction)		
12	0–15	Fail Range Low (Integer)		Words 12 and 13 represent a 16(bit) . 16(bit) fixed point decimal value.
13	0–15	Fail Range Low (Fraction)		
14	0–15	Warning Range High (Integer)		Words 14 and 15 represent a 16 (bit). 16 (bit) fixed point decimal value.
15	0–15	Warning Range High (Fraction)		
16	0–15	Warning Range Low (Integer)		Words 16 and 17 represent a 16 (bit) . 16 (bit) fixed point decimal value.
17	0–15	Warning Range Low (Fraction)		
18–60	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 4–6

Tables D.3 shows the function of each word in the reference line 1–3 (Toolset 1) configuration blocks.

Table D.3
Configuration Blocks #4–6 – Reference Lines 1–3 (Toolset 1)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0–7	Reserved		
1	8	Enable	0 = Disabled 1 = Enabled	
1	9–15	Reserved		
2–7	0–15	Reserved		
8	0–7	X–Line Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.
8	8–15	X–Line High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.
9	0–15	Reserved		
10	0–15	X–Line Head X Position		From upper left corner.
11	0–15	X–Line Head Y Position		From upper left corner.
12	0–15	X–Line Tail X Position		From upper left corner.
13	0–15	X–Line Tail Y Position		From upper left corner.
14	0–15	Reserved		
15	0–7	Y–Line Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.
15	8–15	Y–Line High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.
16	0–15	Reserved		
17	0–15	Y–Line Head X Position		From upper left corner.
18	0–15	Y–Line Head Y Position		From upper left corner.
19	0–15	Y–Line Tail X Position		From upper left corner.
20	0–15	Y–Line Tail Y Position		From upper left corner.
21	0–15	Reserved		
22	0–7	X/Y–Line Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.
22	8–15	X/Y–Line High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 4 – 6 (cont'd)

Table D.3
Configuration Blocks #4–6 – Reference Lines 1–3 (Toolset 1) continued

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
23	0–15	Reserved		
24	0–15	X/Y–Line Head X Position		From upper left corner.
25	0–15	X/Y–Line Head Y Position		From upper left corner.
26	0–15	X/Y–Line Tail X Position		From upper left corner.
27	0–15	X/Y–Line Tail Y Position		From upper left corner.
28–29	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 7 – 9

Table D.4 shows the function of each word in the reference window 1–3 (Toolset 1) configuration blocks.

Table D.4
Configuration Blocks #7–9 – Reference Windows 1–3 (Toolset 1)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0–7	Reserved		
1	8	Enable	0 = Disabled 1 = Enabled	
1	9–15	Reserved		
2–7	0–15	Reserved		
8	0–15	Feature 1 Search Window X Location		Relative to the upper left corner.
9	0–15	Feature 1 Search Window Y Location		Relative to the upper left corner.
10	0–15	Feature 1 Search Window Width		
11	0–15	Feature 1 Search Window Height		
12–15	0–15	Reserved		
16	0–7	Reserved		
16	8–15	Feature 1 Score	0 = Minimum Value 255 = Maximum Value	
17	0–15	Feature 2 Search Window X Location		Relative to the upper left corner.
18	0–15	Feature 2 Search Window Y Location		Relative to the upper left corner.
19	0–15	Feature 2 Search Window Width		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Table D.4
Configuration Blocks #7–9 – Reference Windows 1–3 (Toolset 1)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
20	0–15	Feature 2 Search Window Height		
21–24	0–15	Reserved		
25	0–7	Reserved		
25	8–15	Feature 2 Score	0 = Minimum Value 255 = Maximum Value	
26	0–15	Feature 3 Search Window X Location		Relative to the upper left corner.
27	0–15	Feature 3 Search Window Y Location		Relative to the upper left corner.
28	0–15	Feature 3 Search Window Width		
29	0–15	Feature 3 Search Window Height		
30–33	0–15	Reserved		
34	0–7	Reserved		
34–35	8–15	Feature 3 Score	0 = Minimum Value 255 = Maximum Value	

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 10 – 41

Table D.5 shows the function of each word in the gage 1–32 (Toolset 1) configuration blocks.

Table D.5
Configuration Blocks #10–41 – Gages 1–32 (Toolset 1)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0	Enable	0 = Disabled 1 = Enabled	
1	1–15	Reserved		
2–3	0–15	Reserved		
4	0–7	Low Threshold/Gray Scale Factor	0 – 63 0 – 197	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 10 – 41 (cont'd)

Table D.5
Configuration Blocks #10–41 – Gages 1–32 (Toolset 1)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
4	8–15	High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.
5	0–15	Reserved		
6	0–15	Gage Head X Position		
7	0–15	Gage Head Y Position		
8	0–15	Gage Tail X Position		
9	0–15	Gage Tail Y Position		
10	0–15	Gage X Center Position (Circular Gage)		
11	0–15	Gage Y Center Position (Circular Gage)		
12	0–15	Radius of Circular Gage		
13–16	0–15	Reserved		
17	0–15	Fail Range High (Integer)		Words 17 and 18 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
18	0–15	Fail Range High (Fraction)		
19	0–15	Fail Range Low (Integer)		Words 19 and 20 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
20	0–15	Fail Range Low (Fraction)		
21	0–15	Warning Range High (Integer)		Words 21 and 22 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
22	0–15	Warning Range High (Fraction)		
23	0–15	Warning Range Low (Integer)		Words 23 and 24 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
24	0–15	Warning Range Low (Fraction)		
25–27	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 42 – 65

Table D.6 shows the function of each word in the window 1–24 (Toolset 1) configuration blocks.

Table D.6
Configuration Blocks #42–65 – Windows 1–24 (Toolset 1)

Remote I/O & RS–232 Word #	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0	Enable	0 = Disabled 1 = Enabled	
1	1–15	Reserved		
2–4	0–15	Reserved		
5	0– 7	Window Low Threshold	0 = Low Limit . . . 63 = High Limit	
5	8–15	Window High Threshold	0 = Low Limit . . . 63 = High Limit	
6–10	0–15	Reserved		
11	0–15	Window X Location (Bounding Box)		
12	0–15	Window Y Location (Bounding Box)		
13	0–15	Window Width (Bounding Box)		
14	0–15	Window Height (Bounding Box)		
15	0–15	Mask X Location (Bounding Box)		
16	0–15	Mask Y Location (Bounding Box)		
17	0–15	Mask Width (Bounding Box)		
18	0–15	Mask Height (Bounding Box)		
19–27	0–15	Reserved		
28	0–15	Fail Range High (Integer)		Words 28 and 29 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
29	0–15	Fail Range High (Fraction)		
30	0–15	Fail Range Low (Integer)		Words 30 and 31 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
31	0–15	Fail Range Low (Fraction)		
32	0–15	Warning Range High (Integer)		Words 32 and 33 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
33	0–15	Warning Range High (Fraction)		
34	0–15	Warning Range Low (Integer)		Words 34 and 35 represent a 16(bit) . 16(bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
35–36	0–15	Warning Range Low (Fraction)		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 66 – 68

Table D.7 shows the function of each word in the reference line 1–3 (Toolset 2) configuration blocks.

Table D.7
Configuration Blocks #66–68 – Reference Lines 1–3 (Toolset 2)

Remote I/O & RS–232 Word #	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0–7	Reserved		
1	8	Enabled	0 = Disabled 1 = Enabled	
1	9–15	Reserved		
2–7	0–15	Reserved		
8	0–7	X–Line Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.
8	8–15	X–Line High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.
9	0–15	Reserved		
10	0–15	X–Line Head X Position		From upper left corner.
11	0–15	X–Line Head Y Position		From upper left corner.
12	0–15	X–Line Tail X Position		From upper left corner.
13	0–15	X–Line Tail Y Position		From upper left corner.
14	0–15	Reserved		
15	0–7	Y–Line Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.
15	8–15	Y–Line High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.
16	0–15	Reserved		
17	0–15	Y–Line Head X Position		From upper left corner.
18	0–15	Y–Line Head Y Position		From upper left corner.
19	0–15	Y–Line Tail X Position		From upper left corner.
20	0–15	Y–Line Tail Y Position		From upper left corner.
21	0–15	Reserved		
22	0–7	X/Y–Line Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.
22	8–15	X/Y–Line High Threshold/ Gradient Threshold	0–63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Table D.7
Configuration Blocks #37–39 – Reference Lines 1–3 (Toolset 2)

Remote I/O & RS–232 Word #	Bit #	Definition	Usage	Notes
23	0–15	Reserved		
24	0–15	X/Y–Line Head X Position		From upper left corner.
25	0–15	X/Y–Line Head Y Position		From upper left corner.
26	0–15	X/Y–Line Tail X Position		From upper left corner.
27	0–15	X/Y–Line Tail Y Position		From upper left corner.
28–29	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 69–71

Table D.8 shows the function of each word in the reference window 1–3 (Toolset 2) configuration blocks.

Table D.8
Configuration Blocks #69–71 – Reference Windows 1–3 (Toolset 2)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0–7	Reserved		
1	8	Enable	0 = Disabled 1 = Enabled	
1	9–15	Reserved		
2–7	0– 15	Reserved		
8	0– 15	Feature 1 Search Window X Location		Relative to the upper left corner.
9	0– 15	Feature 1 Search Window Y Location		Relative to the upper left corner.
10	0– 15	Feature 1 Search Window Width		
11	0– 15	Feature 1 Search Window Height		
12–15	0–15	Reserved		
16	0–7	Reserved		
16	8–15	Feature 1 Score	0 = Minimum Value 255 = Maximum Value	
17	0–15	Feature 2 Search Window X Location		Relative to the upper left corner.
18	0–15	Feature 2 Search Window Y Location		Relative to the upper left corner.

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 69–71 (cont'd)

Table D.8
Configuration Blocks #69–71 – Reference Windows 1–3 (Toolset 2)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
19	0–15	Feature 2 Search Window Width		
20	0–15	Feature 2 Search Window Height		
21–24	0–15	Reserved		
25	0–7	Reserved		
25	8–15	Feature 2 Score	0 = Minimum Value 255 = Maximum Value	
26	0–15	Feature 3 Search Window X Location		Relative to the upper left corner.
27	0–15	Feature 3 Search Window Y Location		Relative to the upper left corner.
28	0–15	Feature 3 Search Window Width		
29	0–15	Feature 3 Search Window Height		
30–33	0–15	Reserved		
34	0–7	Reserved		
34–35	8–15	Feature 3 Score	0 = Minimum Value 255 = Maximum Value	

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 72–103

Table D.9 shows the function of each word in the gage 1–32 (Toolset 2) configuration blocks.

Table D.9
Configuration Blocks #72–103 – Gages 1–32 (Toolset 2)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0	Enable	0 = Disabled 1 = Enabled	
1	1–15	Reserved		
2–3	0–15	Reserved		
4	0–7	Low Threshold/Gray Scale Factor	0 – 63 0 – 39	If binary operation, value is used as the threshold. If gray scale operation, value is scale factor.

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Table D.9
Configuration Blocks #72–103 – Gages 1–32 (Toolset 2)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
4	8–15	High Threshold/Gradient Threshold	0 – 63 0 – 197	If binary operation, value is used as the threshold high. If gray scale operation, value is gradient threshold.
5	0–15	Reserved		
6	0–15	Gage Head X Position		
7	0–15	Gage Head Y Position		
8	0–15	Gage Tail X Position		
9	0–15	Gage Tail Y Position		
10	0–15	Gage X Center Position (Circular Gage)		
11	0–15	Gage Y Center Position (Circular Gage)		
12	0–15	Radius of Circular Gage		
13–16	0–15	Reserved		
17	0–15	Fail Range High (Integer)		Words 17 and 18 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
18	0–15	Fail Range High (Fraction)		
19	0–15	Fail Range Low (Integer)		Words 19 and 20 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
20	0–15	Fail Range Low (Fraction)		
21	0–15	Warning Range High (Integer)		Words 21 and 22 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
22	0–15	Warning Range High (Fraction)		
23	0–15	Warning Range Low (Integer)		Words 23 and 24 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
24	0–15	Warning Range Low (Fraction)		
25–27	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 104–127

Table D.10 shows the function of each word in the window 1–24 (Toolset 2) configuration blocks.

Table D.10
Configuration Blocks #104–127 – Windows 1–24 (Toolset 2)

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
0	0–15	Block Transfer Signature		
1	0	Enable	0 = Disabled 1 = Enabled	
1	1–15	Reserved		
2–4	0–15	Reserved		
5	0–7	Window Low Threshold	0 = Low Limit . . . 63 = High Limit	
5	8–15	Window High Threshold	0 = Low Limit . . . 63 = High Limit	
6–10	0–15	Reserved		
11	0–15	Window X Location (Bounding Box)		
12	0–15	Window Y Location (Bounding Box)		
13	0–15	Window Width (Bounding Box)		
14	0–15	Window Height (Bounding Box)		
15	0–15	Mask X Location (Bounding Box)		
16	0–15	Mask Y Location (Bounding Box)		
17	0–15	Mask Width (Bounding Box)		
18	0–15	Mask Height (Bounding Box)		
19–27	0–15	Reserved		
28	0–15	Fail Range High (Integer)		Words 28 and 29 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
29	0–15	Fail Range High (Fraction)		
30	0–15	Fail Range Low (Integer)		Words 30 and 31 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
31	0–15	Fail Range Low (Fraction)		
32	0–15	Warning Range High (Integer)		Words 32 and 33 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
33	0–15	Warning Range High (Fraction)		
34	0–15	Warning Range Low (Integer)		Words 34 and 35 represent a 16 (bit) . 16 (bit) fixed point decimal value or 32 bit integer. Refer to Appendix A.
35–36	0–15	Warning Range Low (Fraction)		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Configuration Blocks 128–135

Table D.11 shows the function of each word in the polygon configuration blocks.

Table D.11
Polygon Configuration Blocks #128–135

Remote I/O & RS–232 Word #*	Bit #	Definition	Usage	Notes
1	0–15	Block Transfer Signature		
2–36	0–15	Reserved		

* Refer to Chapter 6 for Pyramid Integrator long word descriptions.

Template Blocks 136–

The template blocks begin at block #136. The number of template blocks stored in memory is variable. Word 1, bits 8–15 (third byte sent using RS–232) of the first template block indicate the total number of template blocks in the configuration. You must always upload or download *all* of the template blocks as a unit. You cannot archive only a part of the template blocks. When uploading templates from the CVIM module, the program should read the first template block and check word 1, bits 8–15 (third byte sent using RS–232) to determine the number of template blocks to follow. The number of blocks remaining is then 1 less than the total number of template blocks. When downloading templates to the CVIM module, the program must send all template blocks. Bits 8–15 of word 1 determine the number of blocks to send.

ASCII Conversion Table

ASCII or Control Char.	Decimal Value	Hex Value	ASCII or Control Char.	Decimal Value	Hex Value	ASCII or Control Char.	Decimal Value	Hex Value	ASCII or Control Char.	Decimal Value	Hex Value
NUL	0	0	[Space]	32	20	@	64	40	’	96	60
SOH	1	1	!	33	21	A	65	41	a	97	61
STX	2	2	”	34	22	B	66	42	b	98	62
ETX	3	3	#	35	23	C	67	43	c	99	63
EOT	4	4	$	36	24	D	68	44	d	100	64
ENQ	5	5	%	37	25	E	69	45	e	101	65
ACK	6	6	&	38	26	F	70	46	f	102	66
BEL	7	7	’	39	27	G	71	47	g	103	67
BS	8	8	(	40	28	H	72	48	h	104	68
HT	9	9	)	41	29	I	73	49	i	105	69
LF	10	A	*	42	2A	J	74	4A	j	106	6A
VT	11	B	+	43	2B	K	75	4B	k	107	6B
FF	12	C	,	44	2C	L	76	4C	l	108	6C
CR	13	D	–	45	2D	M	77	4D	m	109	6D
SO	14	E	.	46	2E	N	78	4E	n	110	6E
SI	15	F	/	47	2F	O	79	4F	o	111	6F
DLE	16	10	0	48	30	P	80	50	p	112	70
DC1	17	11	1	49	31	Q	81	51	q	113	71
DC2	18	12	2	50	32	R	82	52	r	114	72
DC3	19	13	3	51	33	S	83	53	s	115	73
DC4	20	14	4	52	34	T	84	54	t	116	74
NAK	21	15	5	53	35	U	85	55	u	117	75
SYN	22	16	6	54	36	v	86	56	v	118	76
ETB	23	17	7	55	37	W	87	57	w	119	77
CAN	24	18	8	56	38	X	88	58	x	120	78
EM	25	19	9	57	39	Y	89	59	y	121	79
SUB	26	1A	:	58	3A	Z	90	5A	z	122	7A
ESC	27	1B	;	59	3B	[	91	5B	{	123	7B
FS	28	1C	<	60	3C	\	92	SC	|	124	7C
GS	29	1D	=	61	3D	]	93	5D	}	125	7D
RS	30	1E	>	62	3E	^	94	5E	~	126	7E
US	31	1F	?	63	3F	_	95	5F			

A

ACK
An abbreviated term for Positive Acknowledgment. A control code that indicates that the previous transmission block was received.

address
A character or group of characters that identifies a register, a particular part of storage, or some other data source or destination. To refer to a device or an item of data by its address.

ASCII
The character set and code described in American National Standard Code for Information Interchange, ANSI X3.4–1977. Each ASCII character is encoded with 8–bits including parity check.

B

backplane
A printed circuit card located in the back of a rack, which has sockets into which specific boards fit for interconnection.

BASIC
Acronym for Beginner's All–Purpose Symbolic Instruction Code. A problem solving, algebra–like programming language.

block
A group of words considered as a unit.

bit
An acronym for Binary Digit. The smallest unit of information in the binary numbering system. Represented by the digits 0 and 1.

byte
A unit of data that contains 8 bits

C

centroid
Midpoint of x and y axis of an object.

CVIM
Allen-Bradley trademark for Configurable Vision Input Module. Pronounced as "See VIM".

D

data link
The communication(s) lines, related controls, and interface(s) for the transmission of data between two or more devices.

F

fixed point
A number system in which the position of the decimal point is fixed in respect to one end of a string of numbers.

flag
An indicator. A single bit of a memory location, used to detect and remember the occurrence of some event.

floating point
A system of representing numerical quantities with a variable number of places in which the location of the point does not remain fixed.

G

gray scale
In monochromatic displays, variations in brightness level used to enhance the contrast among the displayed features.

H

handshaking
Two–way communication between two devices to effect a data transfer. Handshaking operations are based on a Data–Ready/Data–Received signal scheme that assures orderly data transfer.

hex
Abbreviated form of the word hexadecimal.

hexadecimal
A base 16 numbering system.

hexadecimal numbering system
A numbering system using the equivalent of the decimal number 16 as a base. Because only a single character is allowed for each absolute value, the hexadecimal numbering system uses the 10 symbols of the decimal system for values 0 through 9, and the first six letters of the alphabet to represent values 10 through 15 (a through F). The positional significance of the hexadecimal symbols is based upon the progression of powers of 16. The highest number that can be represented in the units position is 15.

I

Image
A photographic picture, e.g., as being picked up by a TV camera. Mathematically, an image can be described by a function of 2 variables f(x,y), usually defined over a rectangular region. X and y are the region coordinates, and f(x,y) represents the gray scale value of the point (x,y) in the region.

I/O
Acronym for Input/ Output.

L

left justified
A field of numbers (decimal, binary, etc.) with no zeros or spaces to the left.

lightpen
A hand held photosensitive input device used to designate a location on a display screen.

N

NAK
An abbreviated term for Negative Acknowledgment. A control code that indicates the previous transmission block was not received correctly.

P

parity bit
A parity bit is added to a binary array to make the sum of all the bits always odd or always even; a fundamental check.

pixel
An element of a picture. In order for a computer to analyze a picture, the picture is broken up into a series of picture elements called pixels. Each pixel is assigned a brightness level which is the average of the area in the pixel. In computer vision systems, the pixels is the smallest area of resolution in a picture.

PLC
Allen-Bradley trademark for programmable logic controller.

Q

Q–bus
A set of electrical conductors that carry specific signals to several other circuits.

R

RS–232
Standard electrical interface.

S

standard deviation
A measure of the dispersion around a mean value.

serial port
A communications connector through which data is transmitted one bit at a time.

space character
A graphic character that is usually represented by a blank site in a series of graphics. The space character, though not a control character, has a function equivalent that of a formal effect or that causes the printer or display to move one position forward without producing the printing or display of any graphics.

string
A sequence of ASCII characters.

subroutine
A series of computer instructions which perform a specific task for other routines. It is distinguishable from from a main routine in that it requires as one of its parameters a location specifying where to return to the main program after its function has been accomplished.

T

TTL
A signal processing system in which data in the form of low level electrical signals is processed through circuits either discretely or through integrated circuits comprised primarily of transistors.

W

word
A unit of data which contains two bytes (16 bits).

Catalog Number 5370-ND002 Series C

40062-138-01 (C)