

# PCI-VIDEO

## User Manual

**For Revision: N/R**

**General Standards Corporation**

**8302A Whitesburg Drive**

**Huntsville, AL 35802**

**Phone: (205) 880-8787**

**Fax: (205) 880-8788**

**URL: [www.generalstandards.com](http://www.generalstandards.com)**

**E-mail: [support@generalstandards.com](mailto:support@generalstandards.com)**

# PREFACE

---

## **General Standards Corporation**

Preliminary, Revised: September, 1997

Copyright (C) 1997 **General Standards Corp.**

Additional copies of this manual or other literature may be obtained from:

### **General Standards Corporation**

8302A Whitesburg Dr.

Huntsville, Alabama 35802

Tele: (205) 880-8787

FAX: (205) 880-8788

E-mail: [support@generalstandards.com](mailto:support@generalstandards.com)

Design Engineer's E-mail: [lgavitt@traveller.com](mailto:lgavitt@traveller.com)

URL: [www.generalstandards.com](http://www.generalstandards.com)

Design Engineer's URL: [www.hsv.tis.net/~lgavitt](http://www.hsv.tis.net/~lgavitt)

The information in this document is subject to change without notice.

**General Standards Corp.** makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Although extensive editing and reviews are performed before release to ECO control, **General Standards Corp.** assumes no responsibility for any errors that may exist in this document. No commitment is made to update or keep current the information contained in this document.

**General Standards Corp.** does not assume any liability arising out of the application or use of any product or circuit described herein, nor is any license conveyed under any patent rights or any rights of others.

**General Standards Corp.** assumes no responsibility for any consequences resulting from omissions or errors in this manual, or from the use of information contained herein.

**General Standards Corp.** reserves the right to make any changes, without notice, to this product to improve reliability, performance, function, or design.

### **All rights reserved**

No part of this document may be copied or reproduced in any form or by any means without prior written consent of **General Standards Corp.**

This user's manual provides information on the specifications, theory of operation, register level programming, installation of the board, and information required for customized hardware/software development.

## RELATED PUBLICATIONS

---

The following manuals and specifications provide the necessary information for in depth understanding of the specialized parts used on this board.

EIA Standard for the RS-422A Interface (EIA order number EIA-RS-422A)

PCI Local Bus Specification Revision 2.1 June 1, 1995. Questions regarding the PCI specification be forwarded to:

PCI Special Interest Group  
P.O. Box 14070  
Portland, OR 97214  
(800) 433- 5177 (U.S.)  
(503) 797-4207 (International)  
(503) 234-6762 (FAX)

## PCI VIDEO Documentation History

---

- 1) Note: Documentation Revision N/R for Assembly N/R Completed and Released on February 24, 1997.
- 2) August 14, 1997: All Chapters, table of contents, preface, merged, to create one file.
- 3) September 15, 1997: Figure 1.0-1 updated, headings formatted in order to create a new table of contents.
- 4) September 30, 1997: Inserted PCI Spec to Related Publications

# TABLE of CONTENTS

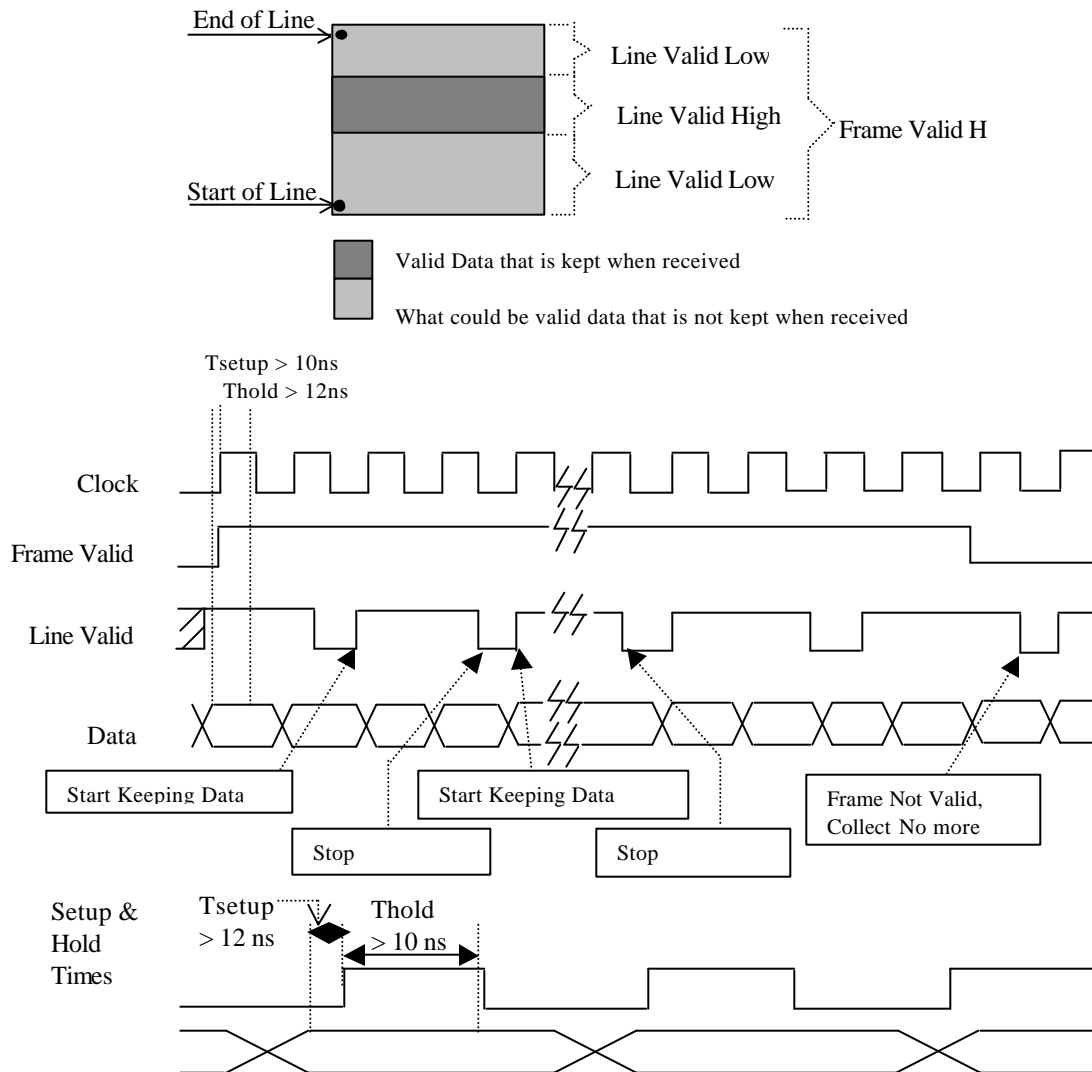
---

<b>CHAPTER 1: INTRODUCTION.....</b>	<b>5</b>
1.0 FUNCTIONAL DESCRIPTION.....	5
1.1 CABLE INTERFACE.....	6
1.2 FIFOs.....	6
1.3 REGISTERS AND THEIR USES .....	6
1.3.1 BOARD CONTROL REGISTER.....	6
1.3.2 BOARD STATUS REGISTER.....	6
1.3.3 FIFOs.....	7
1.3.4 READ N LINES COUNTER.....	7
1.3.5 KEEP N LINE FIRST REGISTER.....	7
1.3.6 KEEP N LINE SECOND REGISTER .....	7
1.3.7 DISCRETE I/O REGISTER.....	7
1.3.8 INTERRUPT CONTROL REGISTER.....	7
1.3.9 INTERRUPT STATUS REGISTER.....	7
<b>CHAPTER 2: PROGRAMMING.....</b>	<b>8</b>
2.0 PCI-VIDEO PROGRAMMING INFORMATION.....	8
2.1 PCI-VIDEO REGISTER MAP.....	8
2.2 PCI-VIDEO BIT MAP.....	9
2.2.1 BOARD CONTROL: (Offset 0x00).....	9
2.2.2 BOARD STATUS: (Offset 0x04).....	9
2.2.3 FIFO I/O: (Offset 0x08).....	9
2.2.4 READ N LINES COUNTER: (Offset 0x0C).....	9
2.2.5 KEEP N LINE FIRST: (Offset 0x10).....	9
2.2.6 KEEP N LINE SECOND: (Offset 0x14) .....	9
2.2.7 DISCRETE I/O (Offset 0x18).....	9
2.2.8 INTERRUPT CONTROL: (Offset 0x20).....	10
2.2.9 INTERRUPT STATUS: (Offset 0x24).....	10
2.3 BIT DESCRIPTIONS FOR REGISTERS .....	11
2.3.1 BOARD CONTROL: (Offset 0x00).....	11
2.3.2 BOARD STATUS: (Offset 0x04).....	11
2.3.3 FIFO I/O: (Offset 0x08).....	12
2.3.4 READ N LINES COUNTER: (Offset 0x0C).....	12
2.3.5 KEEP N LINE FIRST: (Offset 0x10).....	12
2.3.6 KEEP N LINE SECOND: (Offset 0x14).....	12
2.3.7 DISCRETE I/O: (Offset 0x18).....	12
2.3.8 INTERRUPT CONTROL: (Offset 0x20).....	12
2.3.9 INTERRUPT STATUS: (Offset 0x24).....	13
2.4 INITIALIZATION .....	15
2.5 RESETS .....	15
2.6 INTERRUPTS .....	15
2.7 KEEP N LINE FIRST AND KEEP N LINE SECOND REGISTERS.....	16
2.8 DISCRETE I/O PORT .....	16
2.9 FIFOs.....	16
<b>CHAPTER 3: HARDWARE CONFIGURATION.....</b>	<b>17</b>
3.0 THE ON-BOARD RECEIVE CLOCK.....	17
3.1 EEPROM JUMPER (J2).....	17
3.2 CABLE PINOUT .....	18

# CHAPTER 1: INTRODUCTION

## 1.0 FUNCTIONAL DESCRIPTION

The PCI-Video Board provides a high-speed parallel digital receive interface. It is capable of receiving data transfers of up to 40 Mbytes per second from the cable and 132 Mbytes per second on the PCI interface. This board has the added feature of transferring data indefinitely without host intervention. The PCI-Video is easily set up and operated by writing a few programming instructions to the board. Once the data link is established, the desired transfers can be performed and will become transparent to the user. The PCI-Video also offers one (1) bi-directional discrete I/O port, the ability to receive a window of data within a frame of data See Figure 1.0-1.



**Note:** Data is kept on the rising edge of the clock starting when Frame is Valid (high) and on the rising edge of Line Valid, Data will then be stored until the falling edge of Line Valid (even if Frame Valid has gone

Figure 1.0-1: Reception of a Frame of Data

The PCI-Video Board includes a DMA Controller, 64/128Kb of FIFO buffering, a cable input/output controller, and cable transceivers (RS-422/485). The DMA on this board is intended for reading the FIFOs. After the DMA is initialized and started, the host CPU will be free to proceed with other duties and need to respond only to interrupts. The DMA controller is capable of transferring data to host memory using D32 transfers; whereas the FIFO memory provides a means for continuous transmission of data without interrupting the DMA or requiring intervention from the host CPU. The board also provides for interrupt generation for various states of the board, including End-Of-Transfer, FIFO Almost Empty, FIFO Almost Full, and more.

## 1.1 CABLE INTERFACE

There are two (2) individually controlled ports on the cable, both of which employ differential I/O transceivers with RS485/422 compatibility (part number: 75ALS171DW).

- a. Primary Data Path: This port provides for high-speed reception of data (up to 40 Mbytes per second) with a data width of 16 bits. Refer to cable pin-out Table 3.2-1.
- b. Discrete I/O Port: The discrete I/O port is an 8-bit port used for receiving one (1) character of data at time, without any handshaking. This is a totally independent register that may be used for a variety of purposes, i.e., status of the board, or commands to or from an external peripheral. Refer to cable pin-out Table 3.2-1.

## 1.2 FIFOs

The FIFOs on the PCI-Video are used for buffering the data received. The FIFOs are loaded by the cable control logic and read by either the CPU or the DMA.

## 1.3 REGISTERS AND THEIR USES

### 1.3.1 BOARD CONTROL REGISTER

The Board Control Register is strictly under software control and provides the following functionality's:

- a. resets the board;
- b. erases the contents of the FIFOs;
- c. controls when a reception of data is started;
- d. turns the status LED on or off;
- e. controls the direction of the discrete I/O port.

### 1.3.2 BOARD STATUS REGISTER

The Board Status Register is used to return information to the software about the most current status of the board, at the time of the reading. Listed below is the information that this register will contain:

- a. if the board has been reset and the FIFOs have been initialized;
- b. if the FIFOs have been over run by the cable;
- c. if the DMA caused an under run of the FIFOs;
- d. if a new frame was started before the cycle of the previous frame was completed;
- e. the current state of the Frame\_Valid signal;
- f. the current state of the Line\_Valid signal;
- g. and status of the FIFOs.

### 1.3.3 FIFOs

The 64/128Kbyte FIFOs are used for buffering data. This gives the software a means of buffering the data before it is retrieved from the FIFOs. The FIFOs are also used by the DMA for the same purpose. This eliminates unnecessary PCI bus arbitration which provides for faster and more efficient bus cycles for transfers; hence, a faster and more efficient system.

### 1.3.4 READ N LINES COUNTER

The Read N Lines Counter will return the number of lines received in the frame that just came in.

### 1.3.5 KEEP N LINE FIRST REGISTER

The Keep N Line First Register is initialized by the software to instruct the receive logic as to what line the logic will begin keeping data.

### 1.3.6 KEEP N LINE SECOND REGISTER

The Keep N Line Second Register is initialized by the software to instruct the receive logic as to the exact line where the logic will stop keeping data.

### 1.3.7 DISCRETE I/O REGISTER

The Discrete I/O Register is an 8-bit I/O port that will receive one (1) character of data from the cable. The value placed on the cable will not change unless it is changed by the software. This port is useful for communicating various information via the cable, i.e., instructions on, or status about, the data being received through the primary data path. These commands are completely definable by the software.

### 1.3.8 INTERRUPT CONTROL REGISTER

The Interrupt Control Register provides the software with a means of selecting what conditions will be allowed to generate an interrupt.

### 1.3.9 INTERRUPT STATUS REGISTER

The Interrupt Status Register serves as a dual-purpose register. Each bit in this register operates independently of each other. If an interrupt condition is enabled, in the Interrupt Control Register, the appropriate bit, in the Interrupt Status Register, will indicate if an interrupt has occurred or not, and it will continue to indicate this until the software resets that bit. If an interrupt bit is not enabled, in the Interrupt Control Register, then the appropriate bit, in the Interrupt Status Register, will indicate whether or not the condition exists for an interrupt request.

## CHAPTER 2: PROGRAMMING

---

### 2.0 PCI-VIDEO PROGRAMMING INFORMATION

The PCI Video Card complies with the **plug-n-play** concept. That is, at the time of power-up, an attempt will be made by the CPU to set up the board to meet the configuration requirements of the system. In doing this, the CPU will map the amount of I/O space requested by the PCI-Video Card and return the configuration base address into the PCI Configuration Register (Offset 0x18) of this Board.

### 2.1 PCI-VIDEO REGISTER MAP

**Table 2.1-1 Register Map**

Offset Address	Size	Access*	Register Name	Value after Reset
0x00	D32	RW	Board Control	0xXXXX0000
0x04	D32	RO	Board Status	0xXXXXC001
0x08	D32	RO	FIFOs	EMPTY
0x0C	D32	RO	Read N Lines Counter	0xXXXX0000
0x10	D32	RW	Keep N Line First	0xXXXX0000
0x14	D32	RW	Keep N Line Second	0xXXXX0000
0x18	D32	RW	Discrete I/O	0xXXXX0000
0x20	D32	RW	Interrupt Control	0xXXXX0000
0x24	D32	RC	Interrupt Status	0xXXXX3000
0x100	D32	RW	DMA Ch 0 Mode	0x00000003
0x104	D32	RW	DMA Ch 0 PCI Address	0x00000000
0x108	D32	RW	DMA Ch 0 Local Address	0x00000000
0x10C	D32	RW	DMA Ch 0 Transfer byte Count	0x00000000
0x110	D32	RW	DMA Descriptor Pointer	0x00000000
0x114	D32	RW	DMA Ch 1 Mode	0x00000003
0x118	D32	RW	DMA Ch 1 PCI Address	0x00000000
0x11C	D32	RW	DMA Ch 1 Local Address	0x00000000
0x120	D32	RW	DMA Ch 1 Transfer Byte Count	0x00000000
0x124	D32	RW	DMA Ch1 Descriptor Pointer	0x00000000
0x128	D32	RC	DMA Command/Status Register	0x00001010
0x12C	D32	RW	DMA Arbitration Register 0	0x00000000
0x130	D32	RW	DMA Arbitration Register 1	0x00000000

\* RO - read only

RW - read/write capability

RC - read clear (a write clears the specified bits)

## 2.2 PCI-VIDEO BIT MAP

### 2.2.1 BOARD CONTROL: (Offset 0x00)

- D0** Software Board Reset
- D1** Software FIFO Reset
- D2** Reserved
- D3** Turn Receiver On
- D4..6** Reserved
- D7** Status LED
- D8** Discrete Dir is to Cable H
- D9..31** Reserved

### 2.2.2 BOARD STATUS: (Offset 0x04)

- D0** Board Ready
- D1** FIFO Over Run
- D2** FIFO Under Run
- D3** Frame Start Before Ready
- D4** Frame Valid
- D5** Line Valid
- D6..11** These bits are set to a binary '0'
- D12** FIFO Empty L
- D13** FIFO Almost Empty L
- D14** FIFO Almost Full L
- D15** FIFO Full L
- D16..31** Reserved

### 2.2.3 FIFO I/O: (Offset 0x08)

- D0..15** Data 0..15
- D16..31** Reserved

### 2.2.4 READ N LINES COUNTER: (Offset 0x0C)

- D0..15** Data 0..15
- D16..31** Reserved

### 2.2.5 KEEP N LINE FIRST: (Offset 0x10)

- D0..15** Data 0..15
- D16..31** Reserved

### 2.2.6 KEEP N LINE SECOND: (Offset 0x14)

- D0..15** Data 0..15
- D16..31** Reserved

### 2.2.7 DISCRETE I/O: (Offset 0x18)

- D0..7** Data 0..7
- D8..31** Reserved

2.2.8 INTERRUPT CONTROL: (Offset 0x20)

- D0** Interrupt on Positive Edge of Frame\_Valid
- D1** Interrupt on Negative Edge of Frame\_Valid
- D2** Interrupt on Completion of Cycle of Frame
- D3** Reserved
- D4** Interrupt on PLX request for interrupt
- D5..11** Reserved
- D12** Interrupt on FIFO empty
- D13** Interrupt on FIFO almost empty
- D14** Interrupt on FIFO almost full
- D15** Interrupt on FIFO full
- D16..31** Reserved

2.2.9 INTERRUPT STATUS: (Offset 0x24)

- D0** Positive edge of Frame\_Valid
- D1** Negative edge of Frame\_Valid
- D2** Completion of Cycle of Frame
- D3** Reserved
- D4** Plx Request Interrupt
- D5..11** Reserved
- D12** FIFO empty
- D13** FIFO almost empty
- D14** FIFO almost full
- D15** FIFO almost full
- D16..31** Reserved

## 2.3 BIT DESCRIPTIONS FOR REGISTERS

### 2.3.1 BOARD CONTROL: (Offset 0x00)

- D0** Software Board Reset  
Writing a *1* to this bit will generate a self-timed pulse that is used to reset the on-board logic and the FIFOs.  
There is no need for the software to clear this bit, the bit will clear itself.
- D1** Software FIFO Reset  
Writing a *1* to this bit will generate a self-timed pulse that will be used to reset the FIFOs.  
There is no need for the software to clear this bit, the bit will clear itself.
- D2** Reserved
- D3** Turn Receiver On  
Writing a *1* to this bit will turn the Receiver on.  
Writing a *0* to this bit will turn the Receiver off.
- D4..6** Reserved
- D7** Status LED  
Writing a *1* to this bit will turn the status LED off.  
Writing a *0* to this bit will turn the status LED on.
- D8** Discrete Direction is to Cable H  
Writing a *1* to this bit will turn on the discrete I/O port drivers to the cable.  
Writing a *0* to this bit will turn on the receivers for the discrete I/O port from the cable.
- D9..31** Reserved

### 2.3.2 BOARD STATUS: (Offset 0x04)

- D0** Board Reset L  
*1* will indicate a reset has occurred, the FIFOs have been initialized and the board is ready for operation.  
*0* will indicate the board is not ready for operation.
- D1** FIFO OverRun  
*1* will indicate that the FIFO has been loaded when it was already full.  
*0* will indicate that the FIFO has not been loaded after it was full.
- D2** FIFO UnderRun  
*1* will indicate that the FIFO has been read when it was already empty.  
*0* will indicate that the FIFO is not being read before it was empty.
- D3** Frame Start Before Ready  
*1* will indicate a frame was started before the FIFOs were emptied.  
*0* will indicate a frame has not started before the FIFOs were emptied.
- D4** Frame Valid H  
*1* will indicate that the current frame is valid.  
*0* will indicate that the current frame is not valid.
- D5** Line Valid H  
*1* will indicate that the current line is valid.  
*0* will indicate that the current line is not valid.
- D6..11** These bits are set to 0
- D12** FIFO Empty L  
*1* will indicate the FIFO is not empty.

- 0* will indicate the FIFO is empty.
  - D13** FIFO Almost Empty L
    - 1* will indicate the FIFO is not almost empty.
    - 0* will indicate the FIFO is empty.
  - D14** FIFO Almost Full L
    - 1* will indicate the FIFO is not almost full.
    - 0* will indicate the FIFO is almost full.
  - D15** FIFO Full L
    - 1* will indicate the FIFO is not full.
    - 0* will indicate the FIFO is full.
  - D16..31** Reserved
- 2.3.3 FIFO I/O: (Offset 0x08)
- D0..15** Data 0..15
  - D16..31** Reserved
- 2.3.4 READ N LINES COUNTER: (Offset 0x0C)
- D0..15** Data 0..15
  - D16..31** Reserved
- 2.3.5 KEEP N LINE FIRST: (Offset 0x10)
- D0..10** Data 0..10
  - D11..31** Reserved
- 2.3.6 KEEP N LINE SECOND: (Offset 0x14)
- D0..10** Data 0..10
  - D11..31** Reserved
- 2.3.7 DISCRETE I/O: (Offset 0x18)
- D0..7** Data 0..7
  - D8..31** Reserved
- 2.3.8 INTERRUPT CONTROL: (Offset 0x20)
- D0** Interrupt on Positive Edge of Frame\_Valid
    - 1* will allow an interrupt on the positive edge of Frame\_Valid.
    - 0* will disallow an interrupt on the positive edge of Frame\_Valid.
  - D1** Interrupt on Negative Edge of Frame\_Valid.
    - 1* will allow an interrupt on the negative edge of Frame\_Valid.
    - 0* will disallow an interrupt on the negative edge of Frame\_Valid.
  - D2** Interrupt on Completion of Cycle of Frame.
    - 1* will allow an interrupt on the Completion of a Cycle of a Frame.
    - 0* will disallow an interrupt on the Completion of a Cycle of a Frame.
  - D3** Reserved
  - D4** Interrupt on Plx Request Interrupt
    - 1* will allow an interrupt on the Plx Request.
    - 0* will disallow an interrupt on the Plx Request

**D5..11** Reserved

**D12** Interrupt on FIFO Empty

*1* will allow an interrupt on FIFO empty.

*0* will disallow an interrupt on FIFO empty.

**D13** Interrupt on FIFO Almost Empty

*1* will allow an interrupt on FIFO almost empty.

*0* will disallow an interrupt on FIFO almost empty.

**D14** Interrupt on FIFO Almost Full

*1* will allow an interrupt on FIFO almost full.

*0* will disallow an interrupt on FIFO almost full.

**D15** Interrupt on FIFO Full

*1* will allow an interrupt on FIFO full.

*0* will disallow an interrupt on FIFO full.

**D16..31** Reserved

### 2.3.9 INTERRUPT STATUS: (Offset 0x24)

**D0** Positive Edge of Frame\_Valid

If this bit is enabled as an interrupt

a *1* will indicate an interrupt on the positive edge of Frame\_Valid has occurred.

a *0* will indicate an interrupt on the positive edge of Frame\_Valid has not occurred.

If this bit is not enabled as an interrupt

a *1* will indicate that the positive edge of Frame\_Valid currently exists.

a *0* will indicate that the positive edge of Frame\_Valid does not currently exist.

**D1** Negative Edge of Frame Valid

If this bit is enabled as an interrupt

a *1* will indicate an interrupt on the negative edge of Frame\_Valid has occurred.

a *0* will indicate an interrupt on the negative edge of Frame\_Valid has not occurred.

If this bit is not enabled as an interrupt

a *1* will indicate that the negative edge of Frame\_Valid currently exists.

a *0* will indicate that the negative edge of Frame\_Valid does not currently exist.

**D2** Completion of Cycle of Frame

If this bit is enabled as an interrupt

a *1* will indicate an interrupt on the Completion of a Cycle of a Frame has occurred.

a *0* will indicate an interrupt on the Completion of a Cycle of a Frame has not occurred.

If this bit is not enabled as an interrupt

a *1* will indicate that the Completion of a Cycle of a Frame currently exists.

a *0* will indicate that the Completion of a Cycle of a Frame does not currently exist.

**D3** Reserved

**D4** Plx Request Interrupt

If this bit is enabled as an interrupt

a *1* will indicate an interrupt on the Plx Request has occurred.

a *0* will indicate that an interrupt on the Plx Request has not occurred.

If this bit is not enable as an interrupt

a *1* will indicate that the Plx is requesting an Interrupt.

a *0* will indicate that the Plx is not Requesting an Interrupt

**D5..11** Reserved

**D12** FIFO Empty

If this bit is enabled as an interrupt

a *1* will indicate that an interrupt on the FIFO empty has occurred.

a *0* will indicate that an interrupt on the FIFO empty has not occurred.

- If this bit is not enabled as an interrupt  
a *1* will indicate that the FIFO is currently empty  
a *0* will indicate that the FIFO is not currently empty.
- D13** FIFO Almost Empty  
If this bit is enabled as an interrupt  
a *1* will indicate that an interrupt on the FIFO almost empty has occurred.  
a *0* will indicate that an interrupt on the FIFO almost empty has not occurred.  
If this bit is not enabled as an interrupt  
a *1* will indicate that the FIFO is currently almost empty.  
a *0* will indicate that the FIFO is not currently almost empty.
- D14** FIFO Almost Full  
If this bit is enabled as an interrupt  
a *1* will indicate that an interrupt on the FIFO almost full has occurred.  
a *0* will indicate that an interrupt on the FIFO almost full has not occurred.  
If this bit is not enabled as an interrupt  
a *1* will indicate that the FIFO is currently almost full.  
a *0* will indicate that the FIFO is not currently almost full.
- D15** FIFO Full  
If this bit is enabled as an interrupt  
a *1* will indicate that an interrupt on the FIFO full has occurred.  
a *0* will indicate that an interrupt on the FIFO full has not occurred.  
If this bit is not enabled as an interrupt  
a *1* will indicate that the FIFO is currently full.  
a *0* will indicate that the FIFO is not currently full.
- D16..31** Reserved

## 2.4 INITIALIZATION

Initializing the PCI-Video Card will generally need to be done only once by the software, unless the mode needs to change. The software is responsible for tracking any changes; for making all changes necessary to meet the needs of the application; and for making all the adjustments when requirements change. Most of the configuration status can be determined by reading the Board Control Register. Upon power-up, and also after a board reset is performed, the board will be in a state where the following initialization will apply:

- a. all cable drivers will be in receive mode;
- b. the discrete I/O port will be in receive mode;
- c. all interrupts will be disabled;
- d. the FIFOs will be empty;
- e. the Keep First N and Keep Second N Line Registers will be in a state that will keep all of the data received;
- f. the receive logic will be disabled.

When **programming** the PCI Video Card, the following areas will be of interest:

- a. the direction of the discrete I/O port;
- b. what interrupts, if any, will be enabled;
- c. at what word during the reception of a Valid\_Line will the data be considered valid (and kept)
- d. at what word during the reception of Valid\_Frame will the data be considered invalid (and not kept).

## 2.5 RESETS

There are two (2) bits on this board that are used as resets to the local side. Both bits are located in the Board Control Register: D0 is Board Reset and D1 is FIFO Reset. These bits perform a reset when the software writes a 1 to them. After writing a 1, the software does not need to return to clear the bits, the bits operate as a self-timed pulse that will return to 0 after they have been asserted long enough to perform the reset(s). For further details on these resets refer to the PCI-Video Register Map, See Table 2.1-1.

**D0: Board Reset** will reset the local logic, clear the FIFOs, and place the appropriate registers into a known state.

**D1: FIFO Reset** will reset the FIFOs.

The FIFOs are reset by either a hardware reset of the board, a software FIFO reset (Board Control Register bit (1)), or a software board reset (Board Control Register bit (0)).

## 2.6 INTERRUPTS

In order for this board to generate interrupts to the PCI bus, Bits 8, 11, and 16 of the PLX Interrupt Control/Status Register must be set to a 1. These bits must be set to a 1 in order for the interrupts to occur.

The next step in initializing the interrupt, is to specify which interrupts are to be allowed. The board allows the software to enable some interrupts and leave others disabled. This is accomplished by writing a 1 to the appropriate bits in the Interrupt Control Register (ICR). For example, to enable the interrupt for FIFO Almost Empty, the software will need to write a 1 to bit 13 of the ICR. This bit will not need to be changed again until the need to disable this specific interrupt. This enable can be a one time process which will allow many interrupts to occur.

Multiple interrupts from the same cause are prevented via the Interrupt Status Register (ISR). Writing to the ISR is the method by which the software acknowledges to the board that it has received the previous interrupt request and signals to the board that it may now generate any other interrupts that may occur. This register will need some

attention from the software after each interrupt has occurred. Following the previous example, when this interrupt has occurred, the software will find that bit 13 of the ISR is now a 1, indicating that a FIFO Almost Empty interrupt has occurred. This bit will remain a 1 and will not allow any additional interrupts to be generated until the software performs a write to this register. To re-enable the FIFO Almost Empty interrupt, the software must write a 1 to bit 13. This will clear the occurrence of the interrupt.

The enabling, latching, and clearing of the FIFO Almost Empty status bit will not effect the other bits of the register. This means that if the software receives the interrupt for FIFO Almost Empty (the only interrupt currently enabled) the software may very well find that the FIFO is now Empty, (indicated by bit 12 being a 1). Since this bit is not enabled as an interrupt, it is acting as a status bit. If it is enabled now, it will immediately generate an interrupt.

## 2.7 KEEP N LINE FIRST AND KEEP N LINE SECOND REGISTERS

The Keep N Line First and Second Registers are individual 11-bit registers that are used to specify when the board is to begin considering the valid data to be really valid, and when to stop considering it to be valid.

## 2.8 DISCRETE I/O PORT

The discrete I/O port is an 8-bit bi-directional port that is under exclusive control of the software. It is independent of the rest of the board.

The direction of this port is determined by bit 8 of the Board Control Register. If this bit is a 1, then the direction of this port is to the cable. The cable drivers are turned on and the cable receivers are turned off. A read of this register will return the last value written to this register.

If bit 8 of the Board Control Register is a 0, then the direction of this port is from the cable. The cable drivers are turned off and the cable receivers are turned on. A read of this register will return the current value on the cable.

If the direction is set to receive from the cable, then a write to this register will not effect the data on the cable and will not be readable until the direction is changed. However, the value written will be retained in an internal register and will be available upon changing the direction of the port.

## 2.9 FIFOs

There are four (4) FIFOs on the PCI Video Board. Each FIFO is 8 bits wide x 16/32Kbytes deep each, totaling 64/128Kbytes worth of data. The FIFOs serve to provide buffering for the receive data. These FIFOs are accessible to the CPU, DMA, and to the cable.

Before the FIFOs can be used by the cable, they should be reset by the software at least once.

## CHAPTER 3: HARDWARE CONFIGURATION

---

### 3.0 THE ON-BOARD RECEIVE CLOCK

The on-board oscillator, U22, is used as the master clock for the synchronous logic. The oscillator is factory installed at 33 MHz and may be changed to a lesser value to accommodate a different speed clock. Using the 33MHz oscillator gives the DMA a transfer rate of approximately 132 Mb/sec. General Standards Corporation should be contacted for any desired changes that will increase the data rates.

Any standard dip oscillator, 8-pin or 14-pin, will fit into the socket of U22.

### 3.1 EEPROM JUMPER (J2)

The EEPROM jumper (J2) is a 1x2 header. This jumper is for manufacturer uses only. It should not be necessary for any users of the PCI-Video to perform any operations involving this jumper; this section is an only explanation of its purpose.

This jumper connects the PNP EEPROM to the PCI chip set, called the PCI9060, from PLX Technologies. When this board is first assembled, the contents of the PNP EEPROM is considered, by General Standards Corporation, to be unknown. On power up or reset, this EEPROM is used to configure the boards PCI interface so that it may operate properly with plug-n-play systems. If this EEPROM is not programmed with the correct data, then the system could easily be frozen, rendering it inoperable. This jumper is used to disconnect the EEPROM from the PCI chip set; therefore forcing the PCI chip set into believing that the EEPROM is not present or has been erased. This will force the PCI chip set into a default state which will allow the system to complete the power up or reset cycle. After the system is running, the jumper may now be installed and the EEPROM programmed with the correct values. This cycle, involving the PNP EEPROM and the jumper J2, should only need to be performed one time. This is all done during the testing of each board. Once the PNP EEPROM has been programmed with the correct values, J2 should never need to be removed. If the jumper shunt for J2 is removed, the board can be expected to operate incorrectly.

Summary of Jumper Shunt for J2:

This jumper must **not** be installed on first power up, but must be permanently installed before programming the PNP EEPROM. The removal of the jumper will prevent the EEPROM from programming the PLX upon power up or reset, thus, causing the PLX to go into a default configuration. When the PLX is in its default configuration, the performance of the board is to be considered unknown and untested by General Standards Corporation.

### 3.2 CABLE PINOUT

**Table 3.2-1 Cable Pinout**

Pin No.	Cable Signal Name	On-Board Identification	Pin No.	Cable Signal Name	On-Board Identification
1	NC	NC	35	NC	NC
2	GSDHI0	RX D0 HI	36	GSDLO0	RX D0 LO
3	GSDHI1	RX D1 HI	37	GSDLO1	RX D1 LO
4	GSDHI2	RX D2 HI	38	GSDLO2	RX D2 LO
5	GSDHI3	RX D3 HI	39	GSDLO3	RX D3 LO
6	GSDHI4	RX D4 HI	40	GSDLO4	RX D4 LO
7	GSDHI5	RX D5 HI	41	GSDLO5	RX D5 LO
8	GSDHI6	RX D6 HI	42	GSDLO6	RX D6 LO
9	GSDHI7	RX D7 HI	43	GSDLO7	RX D7 LO
10	GSDHI8	RX D8 HI	44	GSDLO8	RX D8 LO
11	GSDHI9	RX D9 HI	45	GSDLO9	RX D9 LO
12	NC	NC	46	NC	NC
13	GSDHI10	RX D10 HI	47	GSDLO10	RX D10 LO
14	GSDHI11	RX D11 HI	48	GSDLO11	RX D11 LO
15	GSDHI12	RX D12 HI	49	GSDLO12	RX D12 LO
16	GSDHI13	RX D13 HI	50	GSDLO13	RX D13 LO
17	NC	NC	51	NC	NC
18	NC	NC	52	NC	NC
19	GSDHI14	RX D14 HI	53	GSDLO14	RX D14 LO
20	GSDHI15	RX D15 HI	54	GSDLO15	RX D15 LO
21	GSUSRHI0	Discrete D0 HI	55	GSUSRLO0	Discrete D0 LO
22	NC	NC	56	NC	NC
23	NC	NC	57	NC	NC
24	GSUSRHI1	Discrete D1 HI	58	GSUSRLO1	Discrete D1 LO
25	GSVSHI	Frame Valid HI	59	GSVSLO	Frame Valid LO
26	GSDVHI	Line Valid HI	60	GSDVLO	Line Valid LO
27	GSUSRHI2	Discrete D2 HI	61	GSUSRLO2	Discrete D2 LO
28	GSUSRHI3	Discrete D3 HI	62	GSUSRLO3	Discrete D3 LO
29	GSDCLKHI	Cable Clk HI	63	GSDCLKLO	Cable Clk LO
30	GSUSRHI4	Discrete D4 HI	64	GSUSRLO4	Discrete D4 LO
31	GSUSRHI5	Discrete D5 HI	65	GSUSRLO5	Discrete D5 LO
32	GSUSRHI6	Discrete D6 HI	66	GSUSRLO6	Discrete D6 LO
33	GSUSRHI7	Discrete D7 HI	67	GSUSRLO7	Discrete D7 LO
34	Gnd	Gnd	68	Gnd	Gnd