PCI 16D

PCI Bus High-speed 16-bit I/O Interface
Addendum for

NASA Cyber IOBC

008-01176-01

EDT

The information in this document is subject to change without notice and does not represent a commitment on the part of Engineering Design Team, Inc. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the agreement.

Engineering Design Team, Inc. (“EDT”), makes no warranties, express or implied, including without limitation the implied warranties of merchantibility and fitness for a particular purpose, regarding the software described in this document (“the software”). EDT does not warrant, guarantee, or make any representations regarding the use or the results of the use of the software in terms of its correctness, accuracy, reliability, currentness, or otherwise. The entire risk as to the results and performance of the software is assumed by you. The exclusion of implied warranties is not permitted by some jurisdictions. The above exclusion may not apply to you.

In no event will EDT, its directors, officers, employees, or agents be liable to you for any consequential, incidental, or indirect damages (including damages for loss of business profits, business interruption, loss of business information, and the like) arising out of the use or inability to use the software even if EDT has been advised of the possibility of such damages. Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitations may not apply to you. EDT’s liability to you for actual damages for any cause whatsoever, and regardless of the form of the action (whether in contract, tort [including negligence], product liability or otherwise), will be limited to $50.

The software described in this manual is based in part on the work of the independent JPEG Group.

Contents

Overview

The PCI 16D is a single-slot, 16-bit parallel input / output interface for PCI Bus-based computer systems. It is designed for continuous input or output between a user device and PCI Bus host memory. The PCI 16D features FIFO storage inboth directions and can support continuous data rates of up to 10 MB per second.

The Cyber IOBC is a NASA test system.

This manual describes the operation of a modified PCI 16D used to control the IOBC. This PCI 16D / IOBC is modified by configuring the Xilinx field programmable gate array on the PCI 16D to implement the IOBC protocol instead of the standard PCI 16D protocol. This manual describes only the changes for the PCI 16D / IOBC. Installation, software and the unchanged hardware are described in the *PCI16D User's Guide*, EDT part number 008-00969.

Upgrading the Firmware

After upgrading to a new device driver, it may sometimes also be necessary to upgrade the PCI interface PROM. If so, the *readme* file will say so.

To do so, follow the instructions for upgrading the firmware in the *PCI16D User's Guide*, substituting:

```
iobc.bit
```

for the firmware file name.

NOTE *The board reloads the firmware from flash ROM only during power-up. Therefore, after running pciload, the new bit file is not loaded in the Xilinx until the system has been power-cycled. Simply rebooting is not adequate.*

Testing

The PCI 16D / IOBC is tested at the factory using another PCI 16D board programmed with another Xilinx configuration to act as an IOBC. This test requires a diagnostic cable, the firmware file *iobctest.bit*, and the test program. These can be used in the field. The firmware file and test program are available from the EDT *ftp* site, and the cable is available for purchase from EDT. Ask for part number 016-1193.

Signals

This section describes how the PCI 16D/IOBC signals are connected.

Connector Pinout

The PCI 16D/IOBC uses a high-density 80-pin I/O connector. The IOBC uses a different connector. The following table describes the PCI 16D to IOBC cable adaptor: Consult the note on page 3 for wiring details.

PCI 16D	IOBC	Signal	PCI 16D	IOBC	Signal
1	P11-G11	OB15N	41	P10-G11	IB15N
2	P11-A13	OB00N	42	P10-A13	IB00N
3	P11-J13	OB14N	43	P10-J13	IB14N
4	P11-B13	OB01N	44	P10-B13	IB01N
5	P11-H13	OB13N	45	P10-H13	IB13N
6	P11-A11	OB02N	46	P10-A11	IB02N
7	P11-B5	OB12N	47	P10-B5	IB12N
8	P11-B11	OB03N	48	P10-B11	IB03N
9	P11-A5	OB11N	49	P10-A5	IB11N
10	P11-C11	OB04N	50	P10-C11	IB04N
11	P11-C7	OB10N	51	P10-C7	IB10N
12	P11-A9	OB05N	52	P10-A9	IB05N
13	P11-B7	OB09N	53	P10-B7	IB09N
14	P11-B9	OB06N	54	P10-B9	IB06N
15	P11-A7	OB08N	55	P10-A7	IB08N
16	P11-C9	OB07N	56	P10-C9	IB07N
17	note 2	GROUND	57		NC
18	P11-J10	GROUND	58		GROUND
19	P11-J9	LDIBFN	59		GROUND
20	P10-H8	GROUND	60	P11-J12	GROUND
21	P10-H7	BUSYN	61	P11-J11	INBMTN
22	P10-D18	GROUND	62		GROUND
23	P10-D17	ISLM03	63		extra output 4
24		GROUND	64		GROUND
25		extra input 1	65		extra input 2
26	P10-D16	GROUND	66	P10-D12	GROUND
27	P10-D15	ISLM12	67	P10-D11	CB14N
28	P10-F12	GROUND	68	P10-E18	GROUND
29	P10-F11	CB15N	69	P10-E17	ISLM05
30	P10-F18	GROUND	70	P10-C14	GROUND
31	P11-G7	EXTSIN	71		same as 72
32		NC	72		extra output 3
33	P10-F17	ISLM11	73	P10-C13	ISLM15
34	P11-G8	GROUND	74	P10-F16	GROUND
35		extra output 1	75	P10-F15	ISLM14
36		GROUND	76	P10-E16	GROUND
37		extra output 2	77	P10-E15	ISLM13
38	P11-G10	GROUND	78	P11-H10	GROUND
39	P11-G9	ODACCN	79	P11-H9	ODSTBN
40	note 1	GROUND	80		GROUND

NOTE IB00N to IB15N signal pairs ground wires are connected together and grounded through PCI16D P3-40. These signals are: P10-A14, P10-B14, P10-A12, P10-B12, P10-C12, P10-A10, P10-B10, P10-C10, P10-A8, *P10-B8, P10-C8, P10-A6, P10-B6, P10-H14, P10-J14, and P10-G12.*

OB00N to OB15N signal pairs ground wires are connected together and grounded through PCI16D P3-17. These signals are: P11-A14, P11-B14, P11-A12, P11-B12, P11-C12, P11-A10, P11-B10, P11-C10, P11-A8, P11-B8, P11-C8, P11-A6, P11-B6, P11-H14, P11-J14, and P11-G12.

Tables 1–3 describe each signal by name, I/O type, and polarity.

Legend

I The signal is an input to the PCI 16D/IOBC.

O The signal is an output to the PCI 16D/IOBC.

H The signal performs the function described in the table at a logic high (or +3 volts).

L The signal performs the function described in the table at a logic low.

Handshake Signals

These six signals perform the PCI 16D/IOBC data transfer cycles.

Name	I/O	Assert	Description
CB15N	O	L	The data direction signal. The PCI16D/IOBC sets CB15N true (0 volts) before beginning an input operation from the IOBC. CB15N is reset (3 volts) before an output to the IOBC.
BUSYN	O	L	The controller busy signal. The PCI16D/IOBC sets BUSYN true (0 volts) during data transfer. During multiple word transfers, BUSYN is true for the entire transfer.
ODSTBN	O	L	The output data strobe signal. The PCI16D/IOBC sets ODSTBN true (0 volts) when output data is valid on the OB[0-15]N signals. This occurs when CB15N is false and BUSYN is true.
ODACCN	I	L	The output data accepted signal. ODACCN is asserted (0 volts) by the IOBC when the output data on OB[0-15]N has been accepted. The PCI16D/IOBC resets ODSTBN when ODACCN is asserted and waits for ODACCN to be deasserted before the next ODSTBN.
LDIBFN	I	L	The input data strobe signal. LDIBFN is asserted (0 volts) by the IOBC after the PCI16D/IOBC asserts BUSYN and CB15N and INBMTN and when the IOBC has valid data on the IB[0-15]N signals. LDIBFN is reset after the PCI16D/IOBC resets INBMTN.
INBMTN	O	L	The input buffer empty signal. The PCI16D/IOBC sets the INBMTN signal (0 volts) during input operations when it is ready to accept data on the IB[0-15]N signals. The IOBC sets LDIBFN signal when the data is available.

Table 1. Handshake Signals

Data Signals

These 32 signals transfer the data.

Name	I/O	Assert	Description
IB[0-15]N	I	L	Input data—set up for 150ns before IOBC asserts LDIBFN.
OB[0-15]N	O	L	Output data—set up for 150ns before PCI16D/IOBC asserts ODSTBN

Table 2. Data Signals

IOBC Status Signals

These eight signals communicate IOBC status.

Name	I/O	Assert	Description
EXTSIN	I	L	External service input signal. The IOBC asserts the EXTSIN signal when the status inputs are valid. The PCI16D/IOBC can be programmed to generate a PCI interrupt when the EXTSIN occurs, or to begin or end DMA operations automatically on certain status conditions .
ISLM03	I	L	The power/ready signal. ISLM03 is asserted (0 volts) when power is applied to the IOBC and it is ready.
ISLM05	I	L	Not used. Always deasserted.
ISLM11	I	L	The busy signal. ISLM11 is asserted (0 volts) when the IOBC is busy.
ISLM12	I	L	The scan signal. ISLM12 is asserted (0 volts) when the IOBC is scanning.
ISLM15-13	I	L	Encoded status signals. The IOBC asserts these signals to represent 8 conditions of IOBC data transfers. All signals asserted (0 volts) represents 7, all deasserted represents 0, etc. **Value Status** 0 No Status 1 CRC Error 2 Illegal Command 3 Invalid Format 4 Data Overflow 5 Remote CRC Error 6 Unused 7 Data Available at IOBC

Table 3. IOBC Status Signals

Data Input Procedure from IOBC

Read data from the IOBC with a PCI 16D DMA read operation, either a simple `edt_read` library call for command response, or EDT ring buffer reads for continuous data input. Programmed read operations are not allowed, as the response time from the IOBC for a read would exceed the maximum PCI bus latency.

If you require a simple read of a command response , enable the PCI EXTSIN interrupt for the data available status. Issue the command as described below in the section entitled "Data Output Procedure from IOBC" on page 6. When the IOBC asserts the EXTSIN input, the driver calls the application program's PCI 16D user interrupt function (previously registered with the driver). This interrupt routine reads the status register; if the IOBC indicates data is available, program the IOBC Read Count Register with the number of words expected, and issue an `edt_read` for the number of bytes expected. When the `edt_read` returns, the data will be in the read buffer.

If continuous data input is desired when the IOBC is collecting data, disable the ENDAV interrupt for data available status.

NOTE EXTSIN with any other status, such as data overflow, can still be programmed to interrupt the user application.

In continuous data mode, configure the EDT ring buffer (using a call to the EDT DMA library). When the EXTSIN occurs, indicating IOBC data available, the PCI 16D / IOBC transfers the number of words programmed in the Read Count Register. The user application can keep track of read data progress by polling buffer completions using standard EDT ring buffer techniques.

NOTE The number of words per IOBC buffer need not equal the number of words in a ring buffer.

If the data is arriving fast, it is more efficient to program the ring buffer size for several IOBC buffers, in order to minimize the driver overhead for each DMA setup and maximize the amount of data your application gets with each ring buffer completion. If the IOBC is programmed to send data infrequently, you may wish to set up the ring buffer to be the same size as the IOBC buffer, so each IOBC buffer can be handled as it arrives.

Data Output Procedure from IOBC

Output data to the IOBC either as a DMA operation (`edt_write` call) or as a programmed write to the Data Register. Either way, poll the status register to make sure the last write is complete before issuing another write. If multiple words are to be written with one assertion of the BUSYN signal, that must be done with the DMA operation.

NOTE EDT understands that continuous writes are not required. Depending on the IOBC interaction, they may work; however, they have not been tested.

Registers

Most PCI 16D / IOBC registers are the same as the PCI 16D. This section details only those registers that are different. These registers are involved in the control and status of the IOBC interface. EDT provides an include file *iobc.h* to symbolically define these addresses and bit definitions.

The addresses listed in Figure 1 are offsets from the PCI base addresses. This base address is initialized by the PCI Local Bus host operating system at boot time. The application program reads and writes these registers with the `edt_reg_read` and `edt_reg_write` library calls, or with *ioctl* routines.

NOTE The addresses 0x80 and 0x84 are used by the pciload utility to update the gate array. User applications must not modify use these registers. Results of running pciload do not take effect until after the board has been power-cycled.

Address **Bits**	31	16	15	0
0xcc	read count		IOBC interrupt enable	
0xc8	data		status	
0xc4	configuration		command	
0xc0	0	0	0	input status
Byte	3	2	1	0
Word	1		0	

Figure 1. PCI Local Bus Addresses

IOBC Interrupt Enable Register

Size	16-bit
I/O	read-write
Address	0x0CC

Bit	IOBC_	Description
D15-11		not used
D11	ENDAV	Enable ISLM[15-13] encoded data available condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D10	ENRSVD	Enable ISLM[15-13] encoded code 6(unused) condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D9	ENRMCRC	Enable ISLM[15-13] encoded remote CRC error condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D8	ENDOVFL	Enable ISLM[15-13] encoded data overflow condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.

Bit	IOBC_	Description
D7	ENINVFMT	Enable ISLM[15-13] encoded invalid format condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D6	ENILLCMD	Enable ISLM[15-13] encoded illegal command condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D5	ENWRCRC	Enable ISLM[15-13] encoded write CRC error condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D4	ENNOSTAT	Enable ISLM[15-13] encoded no status condition to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D3	ENISLM12	Enable ISLM12 (IOBC scan) to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D2	ENISLM11	Enable ISLM11 (IOBC busy) to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D1	ENISLM05	Enable ISLM05 (not used) to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.
D0	ENISLM03	Enable ISLM03 (IOBC power/ready) to cause an EXTSIN interrupt. Write a 0 to disable and clear the interrupt.

Command Register

Size	16-bit
I/O	read-write
Address	0xC4

Bit	IOBC_	Description
D15	EN_INT	Enable PCI interrupt. Write 1 to set the interrupt on a board-wide basis. Write 0 to disable the interrupt. This bit should be set or cleared only by the EDT driver software.
D14		not used
D13	ENEXTSI	Enable EXTSIN interrupt. Write 0 to clear the interrupt. This bit is managed by the EDT driver software, though the actions taken must be implemented in the application program in a subroutine called by the driver.
D12-10		not used
D9	ODDSTART	Set to 1 when the DMA starts on an odd word boundary.
D8	BCLR	Board Clear. Set BCLR to reset the IOBC interface state machines and clear the FIFOs. This value is not stored and need not be reset.
D7-0		not used

Configuration Register

Size	16-bit
I/O	read-write
Address	0xC6

Bit	IOBC_	Description
D15–9		not used
D10	SSWAP	Short Swap. Set to reverse the positions of 16-bit DMA words in the computer’s 32-bit memory word. Some computers require this because the natural word order in that machine is different from the PCI Bus-defined word order. The default is reset, which applies for a PCI bus in an X86 achitecture host. Some hosts swap the data in their memory controllers so that this bit may be reset even on a machine with the opposite word order of the PCI.
D9	INV	Set to invert the data.
D8	SWAP	SWAP determines which byte of a PCI Bus half-word ends up on which half of the PCI 16D/IOBC 16-bit bus. When SWAP is zero, byte swapping is disabled; this is the default—the upper byte of an IOBC half word appears on the upper half of the PCI 16D/IOBC 16-bit bus.
D7–1		not used
D0	ENAUTO	ENAUTO causes data available EXTSIN status interrupts from the IOBC to be ignored by the interrupt circuit and automatically start IOBC data input operations for a previously pending DMA read. The number of IOBC words read is set by the read count register.

Status Register

Size	16-bit
I/O	read-write
Address	0xC8

Bit	IOBC_	Description
D15	INT	A value of 1 indicates that the PCI interrupt is asserted.
D14	WRBSY	A value if 1 indicates either a DMA or data register write operation is in progress from the PCI16D/IOBC and the IOBC. When this bit is 1, no new register write or DMA write operation may be started.
D13	EXTSINT	Shows when an IOBC interrupt is pending. This bit is the logical OR of bits 0-11 logically anded with the ENEXTSI in the command register.
D12	DMA_INT	A value of 1 indicates the end of DMA interrupt is asserted.
D11	DAVINT	Same as NOSTATINT except set on a data available condition.
D10	RSVDINT	Same as NOSTATINT except set on status 6 (reserved) condition.
D9	RMCRCINT	Same as NOSTATINT except set on a remote CRC error condition.
D8	DOVFLINT	Same as NOSTATINT except set on a data overflow condition.

Bit	IOBC_	Description
D7	INVFMTINT	Same as NOSTATINT except set on and invalid format condition.
D6	ILLCMDINT	Same as NOSTATINT except set on an illegal command condition.
D5	WRCRCINT	Same as NOSTATINT except set on write CRC error condition.
D4	NOSTATINT	Set if ISLM[15-13] encoded no status condition is asserted when EXTSIN is asserted, and ENNOSTAT bit of the command register is set. Cleared when ENNOSTAT bit of the command register is reset. If the ENEXTSI bit of the command register is set, NOSTATINT causes a PCI interrupt.
D3	ISLM12INT	Set if ISLM12 (IOBC scan) is asserted when EXTSIN is asserted and ENISLM12 bit of the command register is set. Cleared when ENISLM12 bit of the command register is reset. If the ENEXTSI bit of the command register is set, ISLM12INT causes a PCI interrupt .
D2	ISLM11INT	Set if ISLM11 (IOBC busy) is asserted when EXTSIN is asserted and ENISLM11 bit of the command register is set. Cleared when ENISLM11 bit of the command register is reset. If the ENEXTSI bit of the command register is set, ISLM11NT causes a PCI interrupt.
D1	ISLM05INT	Set if ISLM05 (not used) is asserted when EXTSIN is asserted and ENISLM05 bit of the command register is set. Cleared when ENISLM05 bit of the command register is reset. If the ENEXTSI bit of the command register is set, ISLM05INT causes a PCI interrupt.
D0	ISLM03INT	Set if ISLM03 (IOBC power/ready) is asserted when EXTSIN is asserted and ENISLM03 bit of the command register is set. Cleared when ENISLM03 bit of the command register is reset. If the ENEXTSI bit of the command register is set, ISLM03INT causes a PCI interrupt.

Input Status Register

Size	8-bit
I/O	read-only
Address	0xC0

Bit	IOBC_	Description
D7	EXTSI	Reflects the state of the EXTSIN IOBC output. Set to 1 when EXTSIN is asserted (0 volts).
D6	ISLM15	Reflects the state of the ISLM15 IOBC output. Set to 1 when ISLM15 is asserted (0 volts).
D5	ISLM14	Reflects the state of the ISLM14 IOBC output. Set to 1 when ISLM14 is asserted (0 volts).
D4	ISLM13	Reflects the state of the ISLM13 IOBC output. Set to 1 when ISLM13 is asserted (0 volts).
D3	ISLM12	Reflects the state of the ISLM12 IOBC output. Set to 1 when ISLM12 is asserted (0 volts).
D2	ISLM11	Reflects the state of the ISLM11 IOBC output. Set to 1 when ISLM11 is asserted (0 volts).

Bit	IOBC_	Description
D1	ISLM05	Reflects the state of the ISLM05 IOBC output. Set to 1 when ISLM05 is asserted (0 volts).
D0	ISLM03	Reflects the state of the ISLM03 IOBC output. Set to 1 when ISLM03 is asserted (0 volts).

Read Count Register

Size	16-bit
I/O	read-write
Address	0xCE

Bit	Description
D15–1	Contents of this register are loaded into a down counter and decremented with each IOBC read operation. After the counter reaches 0, the PCI16D/IOBC will deassert BUSYN and wait for another DMA read or IOBC data available condition (if ENAUTO is on) to start another set of IOBC reads. The counter is reloaded automatically from this register as required. A count of 0 transfers 1 word, up to count of 0xfffe transfering 32768 words.
D0	0

Data Register

Size	16-bit
I/O	read-write
Address	0xCA

Bit	Description
D15–0	Data written to this register is output as a single-word write to the IOBC if the IOBC write busy bit in the Status register is 0. If a DMA write is in progress or the write busy bit is set, the result of a data write is undefined. A read of this register returns the state of the IOBC output data bus (IB[15-0]N). An IOBC read is not performed. IOBC reads must be performed with DMA operations.