ICP2432-to-PCI
Host Protocol Specification

DC 900-1509A

Simpact, Inc.
9210 Sky Park Court
San Diego, CA 92123
April 1997

SIMPACT

Simpact, Inc.
9210 Sky Park Court
San Diego, CA 92123
(619) 565-1865

ICP2432-to-PCI Host Protocol Specification

Contents

List of Figures

List of Tables

Preface

Introduction

Simpact's PCI bus Intelligent Communications Processor is the ICP2432. The Simpact ICP2432 hardware includes the Applied Micro Circuits Corporation (AMCC) 5933 PCI bus controller. The ICP2432 system software includes a driver in the Host I/O task (HIO), which runs in the OS/Impact environment. A host computer that wishes to communicate with the ICP2432 must have a device driver that meets this specification. These two drivers communicate with each other providing host applications with the capability of communicating with the ICP2432 (all further references to ICP mean the ICP2432).

For the ICP to become operational for normal host application reads and writes, it must be reset, downloaded with processing tasks which normally include the Host I/O task and any application tasks, and have the downloaded programs initialized.

Purpose of Document

The purpose of this document is to specify the operations of a host driver in order to be operational with Simpact's ICP2432. The HIO task on the ICP is intended to be interfaced to many different computers running under a variety of operating systems. Therefore, any host driver that follows this specification can provide a seamless interface over the PCI bus to HIO on Simpact's ICP.

Organization of Document

Chapter 1 describes the message format.

Chapter 2 describes the ICP reset procedure.

Chapter 3 describes the ICP download procedure.

Chapter 4 describes the ICP initialization procedure.

Chapter 5 describes the application read/write interface.

Document Conventions

The term "ICP" refers to the ICP2432.

Document Revision History

The revision history of the *ICP2432-to-PCI Host Protocol Specification*, Simpact document DC 900-1509A, is recorded below:

Document Revision	Release Date	Description
DC 900-1509A	April 1997	Initial release

Chapter

1

Message Format

There is a region in I/O space where both the host and ICP can communicate called the PCI Operation Registers by AMCC. To be consistent with previous driver implementations for various Simpact ICPs, this exchange region is also called the Protocol Exchange Region (PXR). The registers shown in Table 1–1 define the PXR and are used to exchange data across the PCI bus.

Table 1–1: Protocol Exchange Region (PXR) Registers

Hex Address Offset	Abbreviation	Register Name
00	OMB1	Outgoing Mailbox Register 1
04	OMB2	Outgoing Mailbox Register 2
08	OMB3	Outgoing Mailbox Register 3
0C	OMB4	Outgoing Mailbox Register 4
10	IMB1	Incoming Mailbox Register 1
14	IMB2	Incoming Mailbox Register 2
18	IMB3	Incoming Mailbox Register 3
1C	IMB4	Incoming Mailbox Register 4
20	FIFO	FIFO Register Port
24	MWAR	Master Write Address Register
28	MWTC	Master Write Transfer Count Register
2C	MRAR	Master Read Address Register
30	MRTC	Master read Transfer Count Register
34	MBEF	Mailbox Empty/Full Status Register
38	INTCSR	Interrupt Control/Status Register
3C	MCSR	Bus Master Control/Status Register

References to the PXR are made using the above abbreviations. The offsets are relative to the base address of the PXR. Refer to the *AMCC S5933 PCI Controller Data Book* for specific details on the PCI Operation Registers.

Communications between the host and the ICP take place through the Protocol Exchange Region (PXR). All data transfers are requested and queued to the appropriate host driver or ICP HIO driver. Data can be transferred in either direction over the PCI bus from memory on one processor to the other. The actual data transfer is accomplished by the HIO driver. That is, the HIO driver does the actual move of the data whether it is being stored into a host buffer or an ICP buffer.

OMB1 of the PXR defines Host-to-ICP commands and responses, and IMB1 defines ICP-to-Host commands and responses. Both the host driver and the ICP driver can read any element of the PXR, but only the host driver can write to the outgoing mailboxes (OMB1 – OMB4), and only the ICP driver can write to the incoming mailboxes (IMB1 – IMB4). These mailboxes are always read and stored as 32 bit words. Processing bytes are extracted from the 32-bit words by bit shifting. Using this algorithm to process OMB1 and IMB1 bytes, both the host driver and the ICP driver do not need to be concerned with whether the host or the ICP is a Big-Endian or Little-Endian machine. OMB1 and IMB1 are defined as shown in Table 1–2.

The IMB1 word is read by the host driver to determine what operations the ICP wants the host to perform (`icp_cmd`), or what previous host command is receiving a response (`icp_resp`). The OMB1 word is written by the host driver to command the ICP to perform some operation (`hst_cmd`), or to send a response to a previous ICP command (`hst_resp`). Both the OMB1 and IMB1 words contain the host and ICP node numbers.

Note that it is possible for either the host driver or the ICP driver to send both an acknowledgment to a previous command (which does not require either a host or ICP node number) and send a new command (which does require node numbers) together in the same PXR mailbox word (OMB1 or IMB1).

Table 1–2: OMB1 and IMB1 Definitions

	Bits	Field Name	Description
OMB1			
	12–15	hst_inode	Host's ICP node number
	8–11	hst_hnode	Host's host node number
	4–7	hst_resp	Host's ICP response
	0–3	hst_cmd	Host's ICP command
IMB1			
	12–15	icp_inode	ICP's ICP node number
	8–11	icp_hnode	ICP's host node number
	4–7	icp_resp	ICP's host response
	0–3	icp_cmd	ICP's host command

The host and ICP use node numbers to identify the senders and receivers of messages during normal application reads and writes. These node numbers are not used during the ICP download process.

The PCI bus interface causes an interrupt on the host whenever the ICP writes a word to IMB1, and causes an interrupt on the ICP whenever the host writes a word to OMB1. It is also possible for the PCI bus interface to cause an interrupt on the host whenever the ICP reads the OMB1 word, and to cause an interrupt on the ICP whenever the host reads the IMB1 word. There is one case where interrupts are enabled on the read of OMB1 or IMB1. See Section 5.1 on page 26 for a description of normal reads and writes.

The commands and responses shown in Table 1–3 are written by the host in `hst_cmd` or `hst_resp` of OMB1.

The commands and responses shown in Table 1–4 are written by the ICP in `icp_cmd` or `icp_resp` of IMB1.

Table 1–3: Host Commands and Responses

Symbolic	Hex Value	Description
HOSTx_NOP	00	Host NOP command
HOSTx_ACK	04	Host acknowledge of ICP command received – in hst_resp
HOSTx_WR_BLK	04	Host command to write a block of data during download
HOSTx_IPROC	08	Host command for the ICP to initialize
HOSTx_NAK	10	Host negative acknowledgment of ICP command
HOSTx_REJECT	10	Host rejection of ICP command
HOSTx_DLRDY	10	Host request of ICP to be ready to receive a download block
HOSTx_WR_PEND	20	Host request for the ICP to move data from the host
HOSTx_RD_PEND	21	Host request for the ICP to move data to the host

Table 1–4: ICP Commands and Responses

Symbolic	Hex Value	Description
ICPx_NOP	00	ICP NOP response – in icp_resp
ICPx_NORSP	00	ICP No Response – in icp_resp
ICPx_RDY	03	ICP command denoting that the ICP is initialized
ICPx_ACK	04	ICP acknowledge of host command received – in icp_resp
ICPx_NAK	10	ICP negative acknowledgment of host command – in icp_resp
ICPx_WR_CMPL	20	ICP command denoting that data has been moved to the ICP as requested by the host's HOSTx_WR_PEND command
ICPx_RD_CMPL	20	ICP command denoting that data has been moved to the host as requested by the host's HOSTx_RD_PEND command
ICPx_DLREQ	80	ICP command denoting that it is ready to receive a download block host command

Chapter

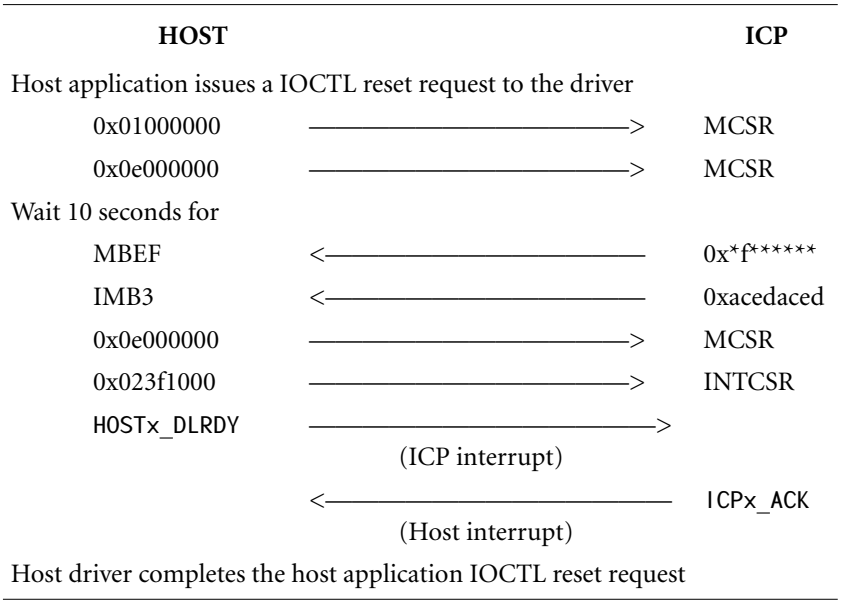
2 ICP Reset

Before anything can be accomplished on an ICP, it must be reset. A reset is initiated by a host application issuing an IOCTL (or UNIX equivalent) RESET request to the host driver. The host driver proceeds as follows:

- Reset the physical ICP devices and processor by setting the Add-On Reset pin of the MCSR (store 0x01000000 in the MCSR).
- Clear the Add-On Reset, and set the Mailbox Flags Reset, the Add-On to PCI FIFO Status Reset, and the PCI to Add-On FIFO Reset (store 0x0e000000 in the MCSR).
- For ten, one-second intervals, wait for the ICP to initialize by checking the Incoming Mailbox 3 flags to be set (check bits 24 to 27 to all be set), and for IMB3 to be initialized (IMB3 to contain 0xacedaced).
- Again, clear the Add-On Reset, and set the Mailbox Flags Reset, the Add-On to PCI FIFO Status Reset, and the PCI to Add-On FIFO Reset (store 0x0e000000 in the MCSR).
- Set 32-Bit Endian Conversion, Target Abort, Master Abort, Read Transfer Complete, Write Transfer Complete, Incoming Mailbox Interrupt, Outgoing Mailbox Interrupt, and Enable Interrupts for Incoming Mailbox Becomes Full (store 0x023f1000 in INTCSR for a Little-Endian host processor and store 0x003f1000 for a Big-Endian host processor).
- The host driver issues a `HOSTx_DLRDY` command to the ICP and waits for an acknowledgment from the ICP.

- On interrupt, the ICP sends an `ICPx_ACK` response to acknowledge receipt of the `HOSTx_DLRDY` command.
- On interrupt, the host driver completes the IOCTL request to the host application.

Figure 2–1 illustrates the process of resetting an ICP.



* Don't care hex

Figure 2–1: Resetting an ICP

Note

There is nothing to preclude a host driver implementation from having the `HOSTx_DLRDY` – `ICPx_ACK` sequence being invoked with a separate IOCTL call to the host driver. This option is left up to the driver implementer since it would be transparent to the ICP.

Chapter

3 ICP Download

The download of application tasks, including the HIO driver, is accomplished by storing blocks of executable images (data) on the ICP. The host driver initiates this process with the previous HOSTx_DLRDY – ICPx_ACK sequence. Then, on request from the ICP to receive a data block and on request from a host download application to send a data block, blocks of data are downloaded to the ICP. For each block of data sent to the ICP, the host driver sends the ICP the host address of a block of data, the address where the block of data is to be stored on the ICP, and the length of the block of data. This process works as follows:

- The ICP sends an ICPx_DLREQ command to the host driver, to request the download of a block of data.
- A host task issues a special application write request. This special write request includes the standard buffer (data image) address and buffer length, but also includes a field which specifies where the data is to be stored on the ICP. If the ICP has not yet sent the host driver an ICPx_DLREQ command, the host driver waits for the command to be received from the ICP.
- If an ICPx_DLREQ has already been received from the ICP, or on receipt of such a request, the host driver stores the length of the block of data in OMB2, and the address of the block of data in OMB3, and the address of where the block of data is to be stored in OMB4. The host driver then sends a HOSTx_WR_BLK command to the ICP.
- On interrupt, the ICP sends an ICPx_ACK response to acknowledge receipt of the HOSTx_WR_BLK.

- On interrupt, the host driver completes the special write request of the download application task.
- On completion of moving the block of data from the host to the ICP, the ICP sends an `ICPx_DLREQ` command to the host driver to request another download block.
- The above sequence continues until the host download application has downloaded all required task images to the ICP.

Figure 3–1 illustrates the process of downloading data images from the host to the ICP.

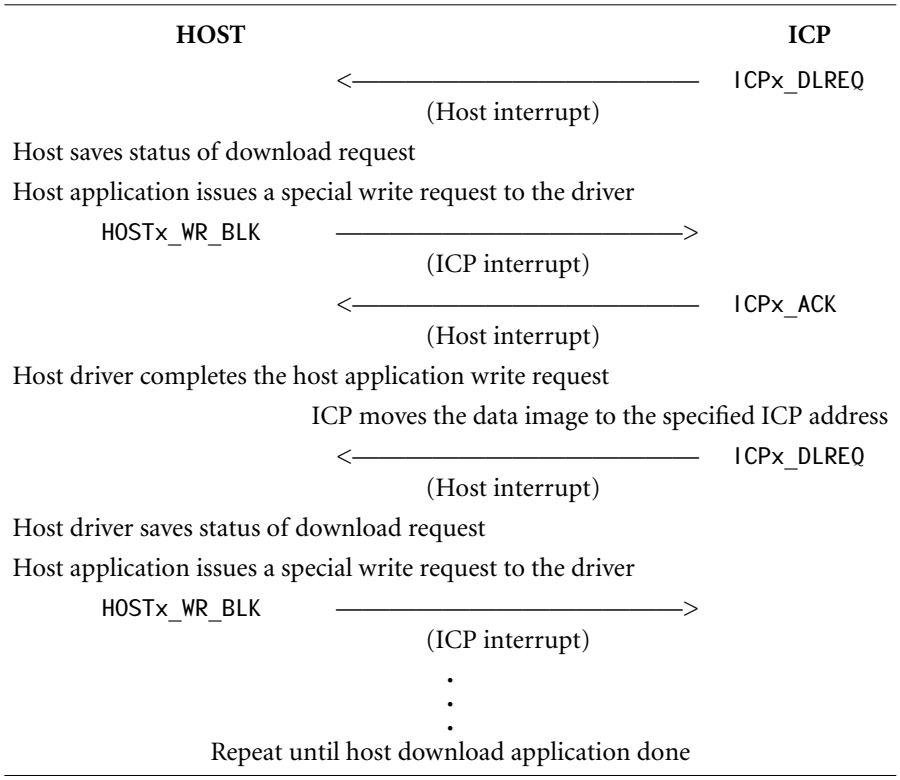


Figure 3–1: Downloading an ICP

Chapter

4 ICP Initialization

After the host download application has completed downloading all task images, the ICP tasks are ready to begin processing. A host application initiates these processes by making an IOCTL INIT request to the host driver specifying the beginning execution address. The host driver proceeds as follows:

- The driver insures that it has received the last `ICPx_DLREQ` from the ICP.
- The host driver stores the beginning execution address in OMB4. The host driver then sends a `HOSTx_IPROC` command to the ICP. The ICP does not know that all images have been download until it receives a `HOSTx_IPROC`.
- On interrupt, the ICP starts processing at the beginning execution address.
- At some point in the initial ICP application task, it calls the HIO system initialization routine. This eventually causes the HIO driver to send an `ICPx_RDY` response to acknowledge receipt of the `HOSTx_IPROC` and to signify that the ICP is ready for normal data transfer operations.

Figure 4–1 illustrates the process of initializing the ICP.

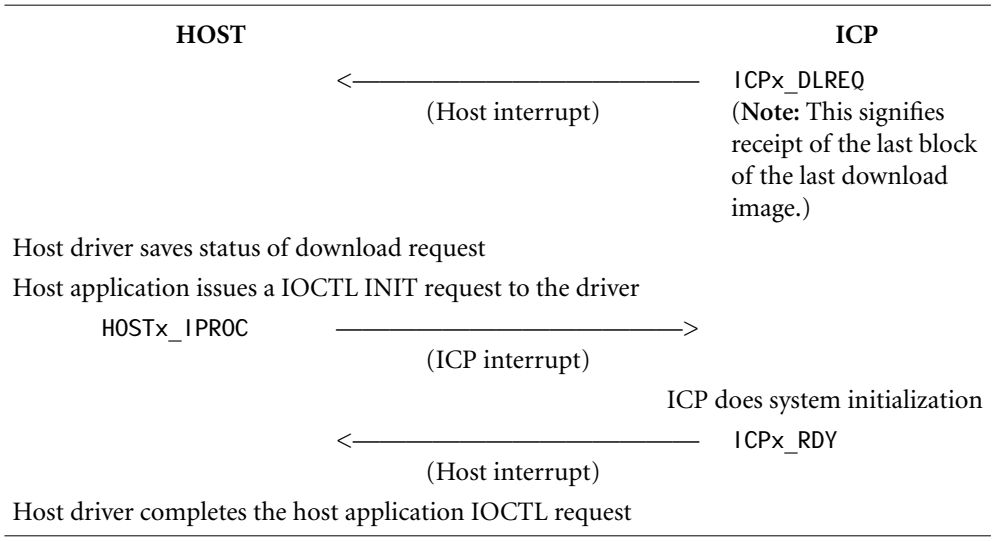


Figure 4–1: Initializing the ICP

Chapter

5 Application Read/Write Interface

ICP applications (i.e. protocol tasks) send write requests to the HIO driver on the ICP for data to be transferred to the host, and send read requests to the HIO driver when they are ready to receive data from the host. Likewise, host applications send write requests to the host driver, and send read requests to the host driver when they are ready to receive data from the ICP.

The host and ICP use node numbers to identify the senders and recipients of messages. When an ICP task or a host task issues a write request, it must specify both an ICP and a host node number. For a read request, however, only the receiver node number is specified. That is, an ICP task specifies an ICP node number on a read request, while a host task specifies a host node number on a read request.

When the host driver receives a write request from the ICP, it attempts to match the host node number with the host node number specified in a pending host read request. When the host driver receives a read request from the ICP, it attempts to match the ICP node number with the ICP node number specified in a pending host write request. Figure 5–1 illustrates this process.

The host driver can have only one unacknowledged command outstanding at a time, but it can have any number of uncompleted reads pending (no `ICPx_RD_CMPL` received) or writes pending (no `ICPx_WR_CMPL`) per ICP node number. When the host driver receives an ICP completion of a pending request, the host driver matches the ICP completion to a matching host application request.

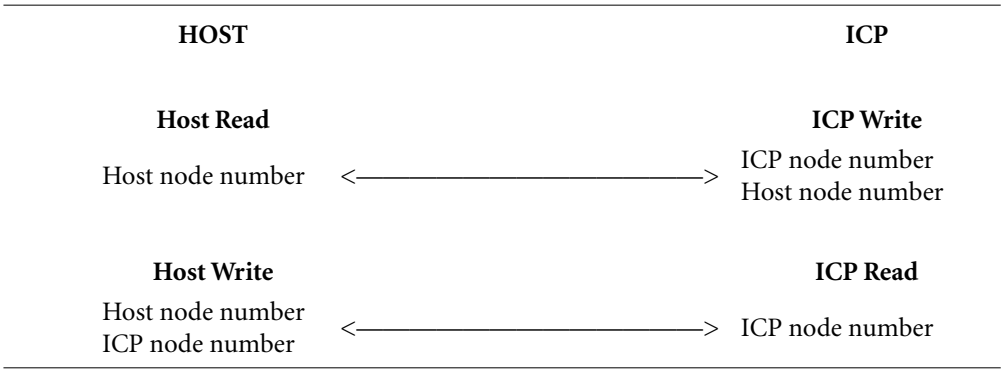


Figure 5–1: Read/Write Process

On either a HOSTx_WR_PEND or HOSTx_RD_PEND host command, the host driver needs to store the size of the data buffer in OMB2 and the address of the data buffer in OMB3. The data buffer address needs to be properly mapped to allow the PCI bus to address the buffer.

The transfer of data from the host to the ICP proceeds as follows:

- An ICP task issues a read request to the HIO driver and supplies a buffer address.
- The HIO driver queues the read request if a previous ICP command is pending to the host.
- A host application task issues a write request to the host driver.
- The host driver queues the write request if a previous command is pending to the ICP.
- The host driver issues a HOSTx_WR_PEND command to the ICP.
- On interrupt, the HIO driver on the ICP sends an ICPx_ACK response to acknowledge receipt of the HOSTx_WR_PEND command and then moves the data in the host application write buffer to the queued HIO read request buffer.

- The HIO driver completes the read request of the ICP task.
- On completion of the data move, the HIO driver sends an ICPx_WR_CMPL command to the host.
- On interrupt, the host driver completes the write request of the host application task.
- The host driver sends a HOSTx_ACK response to acknowledge receipt of the ICPx_WR_CMPL command.

Figure 5–2 illustrates the process of writing data from the host to the ICP.

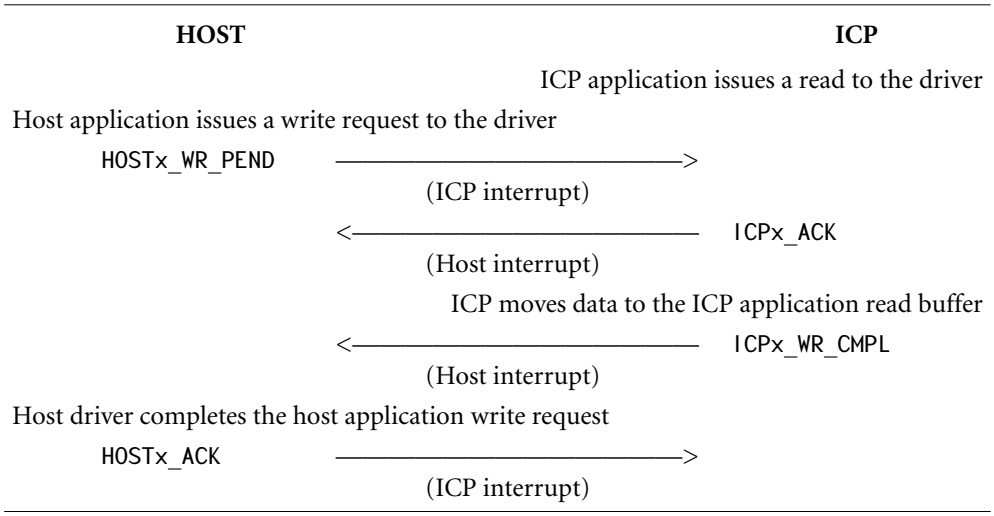


Figure 5–2: Host Writing Data to ICP

The transfer of data from the ICP to the host proceeds as follows:

- A host application task issues a read request to the host driver and supplies a buffer address.
- The host driver queues the read request if a previous host command is pending to the ICP.
- An ICP task issues a write request to the HIO driver.
- The HIO driver queues the write request if an ICP command is pending to the host.
- The host driver issues a `HOSTx_RD_PEND` command to the ICP.
- On interrupt, the HIO driver sends an `ICPx_ACK` response to acknowledge receipt of the `HOSTx_RD_PEND` command and saves the address of the host application read request buffer.
- The HIO driver moves the data from the ICP to the host application read buffer.
- On completion of the data move, the HIO driver sends an `ICPx_RD_CMPL` command to the host.
- On interrupt, the host driver sends a `HOSTx_ACK` response to acknowledge receipt of the `ICPx_RD_CMPL` command and completes the read request of the host application task.

Figure 5–3 illustrates the process of the host reading data from the ICP

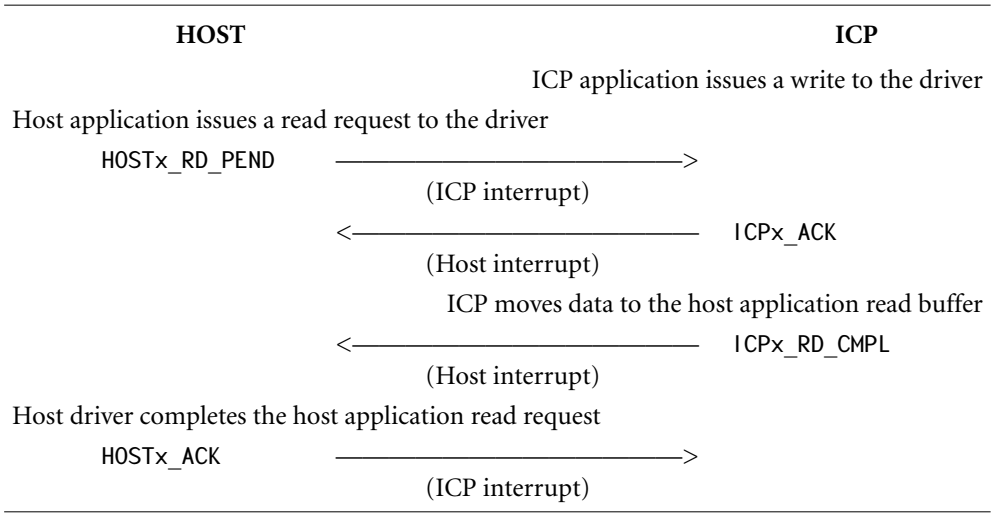


Figure 5–3: Host Reading Data from ICP

In order to speed up the interchange of data and to eliminate potential grid locks between the host and the ICP, both the host driver and the HIO driver need to adopt the following algorithms for initiation of a read or write, and interrupt processing. To simplify this algorithm, terms (variables) are defined as:

- The variable `ack_pending` implies that the host driver has received a command (`ICPx_RD_CMPL` or `ICPx_WR_CMPL`), but the host driver has not sent the ICP an acknowledgment.

- The variable `command_sent` implies that the host driver has sent a command (`HOSTx_RD_PEND` or `HOSTx_WR_PEND`), but the host driver has not received the ICP acknowledgment of the command.

- The variable `out_mbx_interrupt` implies that the host driver has enabled interrupts on Outgoing Mailbox Goes Empty. This enabled interrupt is cleared as soon as the HIO driver either stores a word into IMB1 or reads OMB1.

5.1 Initiation of a Read or Write

On receipt of an application read or write request, if the host driver is currently waiting for an acknowledgment to a previously sent read or a write command (`command_sent` is TRUE), queue the incoming read or write request to a queue for the requested ICP node, and exit while waiting for the acknowledgment.

Otherwise (`command_sent` is FALSE), enable host interrupts associated with the HIO driver reading the host outgoing mailbox (OMB1) (see first **Note** below).

If the HIO driver has not read OMB1, tested by looking at the low order 4 bits of MBEF, then exit. When the ICP reads OMB1, an interrupt is generated on the host which causes a re-invocation of the initiation of a read or write attempt.

If the HIO driver has read the outgoing mailbox, then disable host interrupts associated with the HIO driver reading the host outgoing mailbox (see second **Note** below), and post the read or write command. Posting the command is done by converting the read or write buffer address into an address that can be reached by the PCI bus and storing the word in OMB3, storing the buffer size in OMB2, and creating a word (where each byte is placed into the word by bit shifting) as follows:

- `hst_inode` and `hst_hnode` (the two high order bytes) contain the ICP node number.
- `hst_resp` contains a `HOSTx_NOP` if there is no acknowledgment pending (`ack_pending` is FALSE) or contains `HOSTx_ACK` if there is an acknowledgment pending. Set `ack_pending` to FALSE.
- `hst_cmd` contains either a write pending command (`HOSTx_WR_PEND`) or a read pending command (`HOSTx_RD_PEND`).

The word is then stored in OMB1, which causes an interrupt to occur on the ICP.

Note

Allowing interrupts on Outgoing Mailbox Goes Empty are enabled by storing 0x02001010 into the INTCSR register, and should be allowed by the host only if:

- The host driver has a command ready to be sent to the ICP.
- The host driver is not waiting for an acknowledgment to a previous command.

Note

To disable interrupts on Outgoing Mailbox Goes Empty, store 0x02011000 into the INTCSR register.

5.2 Interrupt Processing

The host interrupt processing routine receives an interrupt either when the ICP stores a word in the host incoming mailbox, IMB1, or when the Outgoing Mailbox Goes Empty interrupt is enabled and the ICP has read the host outgoing mailbox, OMB1.

Since multiple PCI devices can share the same interrupt vector, it may be necessary to determine which ICP has caused the interrupt for the current interrupt line. (Bits 16 or 17 of the INTCSR will be set).

If the interrupt is due to an Outgoing Mailbox Interrupt (bit 16 of the INTCSR), then clear the interrupt (store AND of INTCSR and 0xff011f00), set the ICP command to `HOSTx_NOP`, and set the ICP response to `ICPx_NORSP`.

In addition, if the interrupt is due to an Incoming Mailbox Interrupt (bit 17 of the INTCSR), then clear the interrupt (store AND of INTCSR and 0xff021f1f), and read IMB1. The interrupt routine reads IMB1 ONE TIME ONLY and creates:

- The ICP command word from the low order byte.

- The ICP response from the low order byte after shifting the word right 8 bits.
- The host node number from the low order byte after shifting another 8 bits.
- The ICP node number from the low order byte after shifting another 8 bits.

Regardless, the following process then occurs:

- If the ICP response is an `ICPx_ACK`, then the `command_sent` flag is set to FALSE.
- If the ICP response is anything else but an `ICPx_NORSP`, log an error and exit.
- If the ICP command is an `ICPx_RD_CMPL` or `ICPx_WR_CMPL`, then complete the read or write request to the host application that issued the read or write, and set `ack_pending` to TRUE.
- If the ICP command is anything else but an `ICPx_NOP`, log an error and exit.
- If the `command_sent` flag is FALSE at this point and there is at least one read or write request queued for the ICP, invoke the Initiation of a Read or Write process.

Customer Report Form

We are constantly improving our products. If you have suggestions or problems you would like to report regarding the hardware, software or documentation, please complete this form and mail it to Simpact at 9210 Sky Park Court, San Diego, CA 92123, or fax it to (619) 560-2838.

If you are reporting errors in the documentation, please enter the section and page number.

Your Name: _____

Company: _____

Address: _____

_____

_____

Phone Number: _____

Product: _____

Problem or
Suggestion: _____

_____

_____

_____

_____

_____

Simpact, Inc.
Customer Service
9210 Sky Park Court
San Diego, CA 92123