ISS
Interconnect
Systems
Solution

Transform your ISA adapter card into PCI today.

Low Cost General Purpose PCI Controller
PCI-ISA-001
PCI Controller with ISA-like Add-on Interface

Data Book

December 6, 1999
Version 1.01

Data Book

IMPORTANT NOTICES

Interconnect Systems Solution (ISS) reserves the right to make changes to its products or to discontinue any semiconductor devices or any of it's services without notice, and advises it's customers to obtain the most current information from ISS before placing orders.

Although an effort has been made to keep this document accurate, some information may be incorrect.

—o—

PCI-ISA-001 is not designed, intended, or authorized for use as critical components in life support devices or systems without the express written approval of the president of ISS, where—

- Life support devices or systems are devices or systems which, (i) are intended for surgical implant into the body, or (ii) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
- A critical component is any component of a life systems which are intended for surgical implant support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Inclusion of PCI-ISA-001 in such applications is understood to be entirely at the risk of the customer.

—o—

Any Trade-mark included in this Data Book is used for references only and are owned by their respective owners.

—o—

Reader may obtain further information about the PCI standard from—

PCI Local Bus Specification, Revision 2.1

Published by:

PCI Special Interest Group
2575 N.E. Kathryn #17, Hillsboro, OR 97124, USA
800-433-5177, pcisig@pcisig.com, http://www.pcisig.com

—o—

ISS can be contacted at,

Interconnect Systems Solution
22691 Lambert Street, Suite 503, Lake Forest, CA 92630-1614, USA
949-587-0628, info@iss-us.com, http://www.iss-us.com

PCI-ISA-001

Data Book

Table of Contents

PCI-ISA-001

Low Cost General Purpose PCI Controller
PCI-ISA-001
PCI Controller with ISA-like Add-on Interface

Features

- **PCI interface**
  - ✓ 3.3v PCI compliant I/O.
  - ✓ 5v tolerant PCI I/O.
  - ✓ 32–bit 33 MHz PCI 2.1 Local Bus.
  - ✓ Configuration register support for maximum of 3 add-on functions.
  - ✓ PCI master/target capable.
- **ISA like Add-on interface**
  - ✓ 8–bit data, 11–bit address.
  - ✓ Interface supports full DMA capability consistent with ISA DMA architecture.
  - ✓ Independent address mappings of each Add-on function within the 11–bit address space.
- **Built in DMA engine**
  - ✓ Engine can access both I/O and memory space on the PCI side.
  - ✓ Engine can access both the address mapped and the DMA spaces in the add-on device through ISA like interface.
- **4-pin Serial Non-volatile RAM interface**
  - ✓ Auto download of Serial E^2PROM content at reset time for personalization.
  - ✓ Field upgrade support; In-Circuit Serial E^2PROM update can be performed from the PCI side.

1.0 Description

PCI is being used in the industry for several years now replacing ISA slots. The PCI Local Bus is a high performance bus with multiplexed address and data lines. The bus is intended for use as an interconnect mechanism between highly integrated peripheral controller components, peripheral add-in boards, and processor/memory systems.

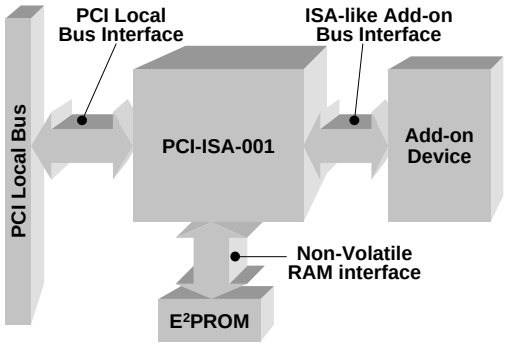
PCI Local Bus offers features and benefits in many areas to achieve multiple price-performance points and can enable functions that allow differentiation at the system and component level. Some of the benefits are,

Robust High Performance Bus – Low latency burst transfer on a Synchronous bus with Hidden central arbitration. Provides parity on both data and address, and allows implementation of robust client platforms;

Low Cost – Optimized for direct silicon with minimal glue logic and in typical ASIC processes. Inexpensive packaging and small number of pins. Small form factor add-in boards;

Platform Independent Plug-and-Play Environment – Processor independent auto configuration support of PCI Local Bus add-in boards and components. PCI devices contain registers with the

Figure 1-1: PCI-ISA-001 Interfaces



device information required for configuration. Full multi-master capability allowing any PCI master peer-to-peer access to any PCI master/target.

In order to enjoy these benefits a large number of adapter cards tend to use single component solutions with integrated PCI interface using ASIC or FPGA technologies. Some other adapter cards use off-the-shelf PCI controller chips with the add-in function. However currently these off the shelf controller chips are targeted to a broad market segment at varying performance levels. Therefore the Add-on interfaces of these controller chips are complex in nature. However, there are still a large number of off-the-self ISA components used in the adapter card market. These ISA chips operate on a very simple ISA-like interface and demand a low cost PCI controller chip with a simple ISA-like interface for building adapter cards. Similarly, prototyping, and small volume designs that may not demand high performance can employ ISA-like interface also with a low cost PCI controller chip having a simple ISA-like interface with the same ease. The low cost PCI-ISA-001 PCI controller chip provides the needed ISA-like interface to enable such a market.

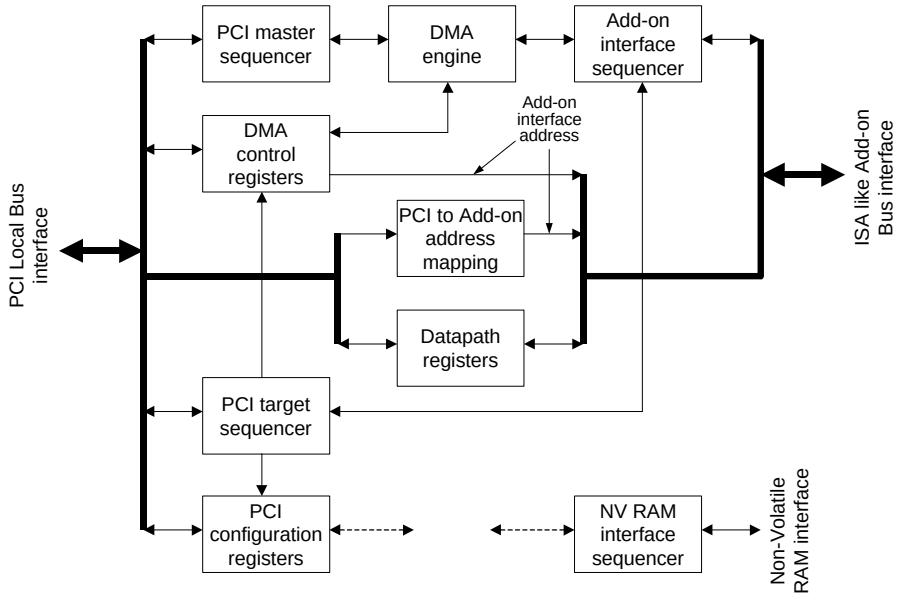
1.1 Architecture

PCI-ISA-001 is a powerful yet simple to use PCI controller supporting the need for the designs that leverage on the usefulness of ISA-like interface. At the basic level it can act as a target and translate all the accesses on the PCI side to the Add-on interface and present the benefits of the PCI plug-and-play architecture. Address mappings for these accesses from the PCI side to the add-on device is automatically handled by PCI-ISA-001 without the need for any external logic on the Add-on interface. At a more sophisticated level, designs that require more autonomous operation to free up the processor from directly participating in the data transfer process at the Add-on interface, the PCI-ISA-001 provides an elegant DMA engine with various addressing modes. If the DMA engine is programmed to perform the data transfer, PCI-ISA-001 will become the bus master on the PCI side and manage the data transfer between the system memory and the add-on device.

The block diagram in Figure 1-2 shows the major functional elements in PCI-ISA-001. The three interfaces shown in the block diagram are PCI Local Bus interface, ISA-like Add-on Bus interface, and the Non-volatile serial RAM interface. Data movement can happen between PCI and the add-on interface during normal operation. At the initial reset time there is also data movement at the Non-volatile RAM interface while the contents of certain PCI configuration registers and the PCI to add-on device address mapping logic inside the PCI-ISA-001 is loaded through the Non-volatile serial RAM interface from a E^2PROM. Once loaded, they act as read-only bits inside PCI-ISA-001. There can also be data movement between the PCI bus interface and the non-volatile RAM interface, to upload/download E^2PROM content by the device driver. This allows field configuration and upgrade of your PCI card using PCI-ISA-001.

The Following sections describe the major functional elements of the PCI-ISA-001.

Figure 1-2: PCI-ISA-001 Block Diagram



1.1.1 PCI target sequencer

The PCI target sequencer state machine performs all the necessary PCI Local bus interface signal handshakes when the PCI-ISA-001 is selected as a target by the current PCI bus master The current PCI bus master will provide an address on the PCI bus and can perform read or write accesses to the target. The PCI target sequencer state machine generates the control signals to control the data movement while it is acting as a target. During this time the accesses can be to—

- PCI configuration registers.
- DMA control registers.
- Serial E^2PROM. The target sequencer provides control for Non-volatile RAM via some register bits for such an access.
- The add-on devices or logic connected to the Add-on interface. The target sequencer provides control for ISA like add-on interface sequencer for such an access.

1.1.2 PCI configuration registers

These registers are defined in the PCI specification and are required for a PCI agent operation. They are a vital part of the PCI plug and play architecture. The target sequencer provides controls for any access to these registers from the PCI interface. The system initialization software such as boot BIOS accesses the configuration registers to obtain information and initializes these registers as part of the system initialization in a typical system.

As part of the initialization, one or more windows within the PCI addressing space will be allocated for the target to be mapped into, and those mapping are also saved in the configuration registers by the BIOS so that PCI-ISA-001 knows when it needs to respond during a PCI access.

Certain information in the PCI configuration registers such as PCI vendor ID are read-only and are loaded from the E^2PROM during hardware reset.

1.1.3 PCI to Add-on address mapping

When the current bus master wants to access registers on the Add-on device or logic connected to the Add-on interface it presents the corresponding PCI address on the PCI bus to select the PCI-ISA-001 as a target. PCI-ISA-001 will respond to the PCI address to claim the access and subsequently the PCI address needs to be mapped to the address space at the Add-on interface.

PCI-ISA-001 supports more than one function on the Add-on interface. During system initialization each of these functions will be allocated separate windows in the PCI address space. As a result these windows are also mapped independently within the Add-on interface address space. During an access PCI-ISA-001 will determine which window within PCI address space is addressed and select the appropriate mapping at the Add-on interface.

The PCI to Add-on address mapping information is loaded from the E^2PROM during hardware reset to the PCI to Add-on Address Mapping Registers and can not be directly accessed or changed during normal operation.

1.1.4 Datapath registers

The PCI data bus used for PCI-ISA-001 is 32 bits. The add-on interface is 8 bits. The datapath registers provide storage and assembly and disassembly function between these interfaces.

1.1.5 Add-on interface sequencer

The Add-on interface sequencer state machine performs all the necessary Add-on bus interface signal handshakes when the PCI-ISA-001 is accessing the add-on devices. There are two cases when such an access to any add-on device will exist—

- A PCI bus master accesses the add-on function while the PCI-ISA-001 is acting as a PCI target. In such case the PCI target sequencer will control the operation of the Add-on interface sequencer.
- DMA engine also accesses the add-on function as part of a DMA transfer. In such case the DMA engine will provide controls for the Add-on interface sequencer.

1.1.6 DMA control registers

Before a DMA transfer can start, the DMA control register are initialized by the controlling software such as a device driver. These registers hold necessary information like the PCI bus address, Add-on interface address, data transfer direction, and the transfer size for the DMA transfer to take place.

1.1.7 PCI master sequencer

As the DMA operation begins, PCI-ISA-001 will need to become the PCI bus master to perform the data transfer. The PCI master sequencer state machine performs all the necessary PCI Local bus interface signal handshakes when the PCI-ISA-001 acts as a PCI bus master.

1.1.8 DMA engine

The DMA engine performs the data transfer using the source address, the destination address, transfer size from the DMA control registers. To access data on the PCI side it controls the PCI master sequencer and to access on the add-on side it controls Add-on interface sequencer. During transfer the engine increments source address and destination address, and decrements the transfer count. The transfer stops when the transfer count becomes zero.

1.1.9 Non-volatile RAM interface sequencer

At initial hardware reset time when the system powers up, the NV RAM interface sequencer controls the signals connected to the serial E^2PROM. The sequencer will download the E^2PROM contents and write them to the appropriate PCI Configuration Register and PCI to Add-on Address Mapping Register bits.

After the download it signals the rest of the PCI-ISA-001 to start normal operation. During normal operation the sequencer also passes control from a few register bits to the E^2PROM so that E^2PROM contents can be accessed by a PCI master under software control.

1.2 Key Benefits of PCI-ISA-001

PCI-ISA-001 is targeted to mimic the success of low-cost ISA adapter card market to the PCI bus architecture. This can be achieved by converting existing ISA adapter card designs to PCI using the PCI-ISA-001. Moreover, the simplicity of the ISA bus architecture presents a very attractive option for designing new adapter products that require throughput similar to the ISA bus.

The low-cost PCI-ISA-001 offers following benefits:

- It protects investment in ISA adapter card products while enabling PCI adapter card market with very low additional investment.
- It allows fast low-risk engineering effort for ISA to PCI adapter migration that enables faster PCI market entry.
- It allows multiple ISA card functions to be integrated in the same PCI adapter card to open up further market opportunities. The multi-function PCI-ISA-001 can support a maximum of three functions.
- It enables new designs that require low-medium data throughput to use PCI bus architecture without the overhead of a PCI learning curve.

2.0 Signal List

PCI-ISA-001 signal pins are catagorized as shown in Figure 2-1.

2.1 Signal Type Definition

The following signal type definitions are used as described in PCI 2.1 specification:

in **Input** is a standard input-only signal.

out **Totem Pole** Output is a standard active driver.

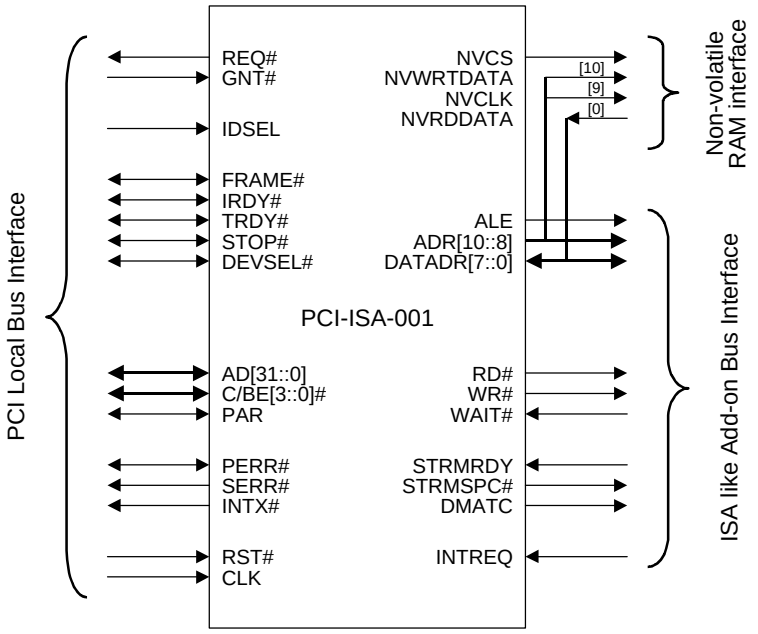
t/s **Tri-State** is a bi-directional, tri-state input/ output pin.

s/t/s **Sustained Tri-State** is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives an s/t/s pin low must drive it high for at least one clock before letting it float. A new agent cannot start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. A pullup is required to sustain the inactive state until another agent drives it, and must be provided by the central resource.

o/d **Open Drain** allows multiple devices to share as a wire-OR. A pull-up is required to sustain the inactive state until another agent drives it, and must be provided by the central resource.

Any signal name ending with # is an active low signal. Otherwise it is an active high signal.

Figure 2-1: PCI-ISA-001 Signal Pins



2.1.1 PCI System Pins

CLK	in	**Clock** provides timing for all transactions on PCI and is an input to PCI-ISA-001 controller chip. All other PCI signals, except RST# and INT# are synchronous to the rising edge of CLK. PCI-ISA-001 operates up to 33 MHz.
RST#	in	**Reset** is used to bring all PCI registers, PCI bus sequencers, PCI signals, DMA engine registers, ISA device interface sequencers, and the ISA device interface signals to a consistent state. RST# will also initiate the download of certain PCI configuration registers parameters, and the ISA device interface address mapping registers from the serial NV-ROM. Anytime RST# is asserted, all PCI output signals will be asynchronously tri-stated and open drain signals are floated. RST# may be asynchronous to CLK when asserted or deasserted. Although asynchronous, deassertion is required to be a clean, bounce-free edge.

2.1.2 PCI Address and Data Pins

AD[31::00]	t/s	**Address and Data** are multiplexed on the same PCI pins. A bus transaction consists of an address phase followed by one or more data phases. PCI supports both read and write bursts. The address phase is the clock cycle in which FRAME# is asserted. During the address phase AD[31::00] contain a physical address (32 bits). For I/O, this is a byte address; for configuration and memory, it is a DWORD address. During data phases AD[07::00] contain the least significant byte (lsb) and AD[31::24] contain the most significant byte (msb). Write data is stable and valid when IRDY# is asserted and read data is stable and valid when TRDY# is asserted. Data is transferred during those clocks where both IRDY# and TRDY# are asserted.
C/BE[3::0]#	t/s	**Bus Command and Byte Enables** are multiplexed on the same PCI pins. During the address phase of a transaction, C/BE[3::0]# define the bus command. During the data phase C/BE[3::0]# are used as Byte Enables. The Byte Enables are valid for the entire data phase and determine which byte lanes carry meaningful data. C/BE[0]# applies to byte 0 (lsb) and C/BE[3]# applies to byte 3 (msb). C/BE[3::0]# value and commands are described in Table 2-1.
PAR	t/s	**Parity** is even parity across AD[31::00] and C/BE[3::0]#. PAR is stable and valid one clock after the address phase. For data phases, PAR is stable and valid one clock after either IRDY# is asserted on a write transaction or TRDY# is asserted on a read transaction. Once PAR is valid, it remains valid until one clock after the completion of the current data phase. (PAR has the same timing as AD[31::00], but it is delayed by one clock.) The PCI master drives PAR for address and write data phases; the PCI target drives PAR for read data phases. The number of "1"s on AD[31::00], C/BE[3::0]#, and PAR equal an even number.

Table 2-1: PCI C/BE[3::0]# Command encodings

0000	**Interrupt Acknowledge.** Not implemented in PCI-ISA-001 controller chip.
0001	**Special Cycle.** Not implemented in PCI-ISA-001 controller chip.
0010	**I/O Read.** The I/O Read command is used to read data from an agent mapped in I/O Address Space. AD[31::00] provide a byte address. The byte enables indicate the size of the transfer and must be consistent with the byte address.
0011	**I/O Write.** The I/O Write command is used to write data to an agent mapped in I/O Address Space. The byte enables indicate the size of the transfer and must be consistent with the byte address.
0100	Reserved.
0101	Reserved.
0110	**Memory Read.** The Memory Read command is used to read data from an agent mapped in the Memory Address Space. AD[31::00] provide a byte address. The byte enables indicate the size of the transfer and must be consistent with the byte address.
0111	**Memory Write.** The Memory Write command is used to write data to an agent mapped in the Memory Address Space. The byte enables indicate the size of the transfer and must be consistent with the byte address.
1000	Reserved.
1001	Reserved.
1010	**Configuration Read.** The Configuration Read command is used to read the Configuration Space of each agent. An agent is selected during a configuration access when its IDSEL signal is asserted and AD[1::0] are 00. During the address phase of a configuration cycle, AD[7::2] address one of the 64 DWORD registers (where byte enables address the byte(s) within each DWORD) in Configuration Space of each device and AD[10::08] indicate which device of a multi-function agent is being addressed.
1011	**Configuration Write.** The Configuration Write command is used to transfer data to the Configuration Space of each agent. An agent is selected when its IDSEL signal is asserted and AD[1::0]are 00. During the address phase of a configuration cycle, AD[7::2] address one of the 64 DWORD registers (where byte enables address the byte(s) within each DWORD) in Configuration Space of each device and AD[10::08] indicate which device of a multi-function agent is being addressed.
1100	**Memory Read Multiple.** Not implemented in PCI-ISA-001 controller chip.
1101	**Dual Address Cycle.** Not implemented in PCI-ISA-001 controller chip.
1110	**Memory Read Line.** Treated same as Memory Read in this device.
1111	**Memory Write and Invalidate.** Treated same as Memory Write in this device.

2.1.3 PCI Interface Control Pins

FRAME#	s/t/s	**Cycle Frame** is driven by the current master to indicate the beginning and duration of an access. FRAME# is asserted to indicate a bus transaction is beginning. While FRAME# is asserted, data transfers continue. When FRAME# is deasserted, the transaction is in the final data phase or has completed.
IRDY#	s/t/s	**Initiator Ready** indicates the initiating agent's (bus master's) ability to complete the current data phase of the transaction. IRDY# is used in conjunction with TRDY#. A data phase is completed on any clock both IRDY# and TRDY# are sampled asserted. During a write, IRDY# indicates that valid data is present on AD[31::00]. During a read, it indicates the master is prepared to accept data. Wait cycles are inserted until both IRDY# and TRDY# are asserted together.
TRDY#	s/t/s	**Target Ready** indicates the target agent's (selected device's) ability to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#. A data phase is completed on the clock when both TRDY# and IRDY# are sampled as asserted. During a read, TRDY# indicates that valid data is present on AD[31::00]. During a write, it indicates the target is prepared to accept data. Wait cycles are inserted until both IRDY# and TRDY# are asserted together.
STOP#	s/t/s	**Stop** indicates the current target is requesting the master to stop the current transaction.
IDSEL	in	**Initialization Device Select** is used as a chip select to PCI-ISA-001 controller chip during configuration read and write transactions.
DEVSEL#	s/t/s	**Device Select**, when actively driven, indicates the PCI-ISA-001 controller chip has decoded its address as the target of the current access. As an input, DEVSEL# indicates whether any device on the bus has been selected.

2.1.4 PCI Bus Arbitration Pins

REQ#	t/s	**Request** indicates to the PCI arbiter that the PCI-ISA-001 controller chip desires use of the PCI bus for DMA transfers. Every PCI master has its own REQ# which is tri-stated while RST# is asserted.
GNT#	t/s	**Grant** indicates to the PCI-ISA-001 controller chip that access to the PCI bus has been granted. Every PCI master has its own GNT# which is ignored while RST# is asserted.

2.1.5 PCI Error Reporting Pins

PERR#	s/t/s	**Parity Error** is only for the reporting of data parity errors during all PCI transactions except a Special Cycle. The PERR# pin is a sustained tri-state signal and driven active by the PCI-ISA-001 controller chip when a data parity error is detected while receiving the data. It is driven active two clocks following the data that caused the data parity error. The minimum duration of PERR# is one clock for each data phase that a data parity error is detected. (If sequential data phases each have a data parity error, the PERR# signal will be asserted for more than a single clock.) PERR# will be driven high for one clock before being tri-stated.
SERR#	o/d	**System Error** is for reporting address parity errors, data parity errors on the Special Cycle command, or any other system error where the result will be catastrophic. SERR# is pure open drain and is actively driven for a single PCI clock by PCI-ISA-001 reporting the error.
INTX#	o/d	**Interrupt Request**. This device asserts its INTX# line when requesting attention from its device driver. The PCI-ISA-001 controller may optionally assert this when DMA engine inside PCI-ISA-001 reaches terminal count. This will also be asserted when INTREQ is asserted by the add-on device. Once the INTX# signal is asserted, it remains asserted until the device driver clears the pending request.

2.1.6 ISA-like Add-on Bus Interface Pins

STRMSPC#	out	**Stream Space Access.** Accesses to any Add-on devices can occur at two different address spaces: Memory-Register space, and Stream space. The Memory-Register space accesses need an address to be provided to the Add-on device. During such accesses, the location specified by the address is accessed by the PCI-ISA-001. On the other hand, the Stream space behaves similar to a FIFO interface and the PCI-ISA-001 does not need to provide any address. STRMSPC# indicates when a Stream space access is in progress. During a Memory-Register space access STRMSPC# is de-asserted and during Stream space access STRMSPC# is asserted by the PCI-ISA-001.
ALE	out	**Address Latch Enable**. An access at the Memory-Register space of an Add-on device by PCI-ISA-001 controller consists of an address phase followed by a data phase. During the address phase ALE is asserted by PCI-ISA-001 while Address[7::0] will be presented at DATADR[7::0] pins. ALE can be used latch the address for the Add-on device. During a Stream space access there will be no address phase and ALE will not be asserted.
ADR[10::8]	out	**Add-on Address**. Address[10::8] is presented during an Add-on device access at the Memory-Register space.
DATADR[7::0]	t/s	**Add-on Address and Data**. Address[7::0] and Data[7::0] are multiplexed on the same physical pins. The data transfer to an Add-on device Memory-Register space consists of an address phase followed by one data phase. A Stream space access consists of data phase only. During the address phase ALE is asserted and Address[7::0] will be driving the DATADR[7::0] pins. During the data phase, Data[7::0] will be driving the DATADR[7::0] pins if it is a write access. In a read access the Add-on device will be driving DATADR[7::0] pins during the data phase.
RD#	out	**Add-on Device Read** is asserted in data phase during a read access.
WR#	out	**Add-on Device Write** is asserted in data phase during a write access.
WAIT#	in	**Add-on Device Wait** indicates to the PCI-ISA-001 controller that the Add-on device is not ready to complete the access made by PCI-ISA-001. In response PCI-ISA-001 will delay the completion of the data phase till this signal is de-asserted.
STRMRDY	in	**Stream Data Ready** indicates to the PCI-ISA-001 controller that the Add-on device is now ready for a Stream Space access. Since the Stream space behaves similar to a FIFO interface, in case of an intended read this signal behaves like a FIFO-not-empty indication. Similarly, in case of an intended write this signal behaves like a FIFO-not-full indication.
DMATC	out	**DMA Terminal Count** indicates to the Add-on device that the DMA engine in the PCI-ISA-001 controller chip has completed transferring as much data as the engine was programmed by the device driver and reached terminal count.
INTREQ	in	**Add-on Device Interrupt Request**. The Add-on device asserts this to indicate it requires service from device driver. This signal in turn asserts PCI INTX# pin.

2.1.7 NV-RAM interface

NVCS	in	**Configuration NV-RAM Chip Select**. This pin acts as chip select for the serial E^2PROM device connected to PCI-ISA-001.
NVWRTDATA/ ADR[10]	out	**Configuration NV-RAM Write Data Out**. This is a shared pin. When NVCS is asserted, this provides the command, addreess, and write data to the serial E^2PROM.
NVCLK/ADR[9]	out	**Configuration NV-RAM Clock**. This is a shared pin. When NVCS is asserted, this provides the serial clock to the serial E^2PROM. PCI-ISA-001 holds NVWRTDATA valid at the rising edge of this signal.
NVRDDATA/ DATADR[0]	t/s	**Configuration NV-RAM Read Data In.** This is a shared pin. While NVCS is asserted, this obtains the read data from the serial E^2PROM.

3.0 PCI Configuration Registers

All PCI devices need to implement certain registers which are located in a 256-byte space called Configuration space. PCI-ISA-001 PCI Configuration registers can be accessed from the PCI Bus with a Type 0 Configuration cycle. As a PCI compliant device, PCI-ISA-001 Configuration space can support—

- Full device relocation, including interrupt binding;
- Installation, configuration, and booting without user intervention;
- System address map construction by device independent software;
- Effective reuse of existing device drivers with minimal or no change;

PCI-ISA-001 is a Multifunction PCI device and provides a Configuration Space for each function implemented as required by the PCI spec. The Configuration Space is accessible at all times, not just during system boot. System initialization software can scan the PCI bus to determine what devices are actually present.

Configuration Space is intended for configuration, initialization, and catastrophic error handling functions. Its use is restricted to initialization software and error handling software. The initialization software must map any Add-on device address space used by operational Software to some I/O and/or Memory space. All operational software must continue to use normal I/O and/or Memory Space accesses to manipulate Add-on device registers and DMA registers accessible through PCI-ISA-001.

PCI-ISA-001 treats Configuration Space write operations to reserved registers as no-ops— i.e., the access is completed normally on the bus and the data is discarded. Read accesses to reserved or unimplemented registers are completed normally and a data value of 0 returned.

Any multi-byte numeric fields in the Configuration Space registers follow little-endian ordering; that is, lower addresses contain the least significant parts of the field. Software must take care to deal correctly with bit-encoded fields that have some bits reserved for future use. On reads, software must use appropriate masks to extract the defined bits, and may not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved— i.e., the values of reserved bit positions must first be read, merged with the new values for other bit positions and the data then written back.

Figure 3-1 shows the layout of the implemented registers in the Configuration space by PCI-ISA-001. For full description of all the registers that are not implemented in PCI-ISA-001 or reserved, consult the PCI specification.

3.1 Vendor ID Register

This 16-bit field identifies the manufacturer of the device. Valid vendor identifiers are allocated by the PCI SIG to ensure uniqueness. When RST# is removed, this field is initialized from the external NV-RAM. The Vendor ID for ISS assigned by PCI SIG is 1549h. The users of PCI-ISA-001 will be allowed to use ISS vendor ID in conjunction with ISS assigned Device ID, if they do not have an assigned Vendor ID of their own. Such users need to obtain a Device ID from ISS. After the value is boot-loaded from the NV-RAM, Vendor ID register can be read from PCI

Figure 3-1: PCI-ISA-001 Configuration Register Mapping in the PCI Configuration Space

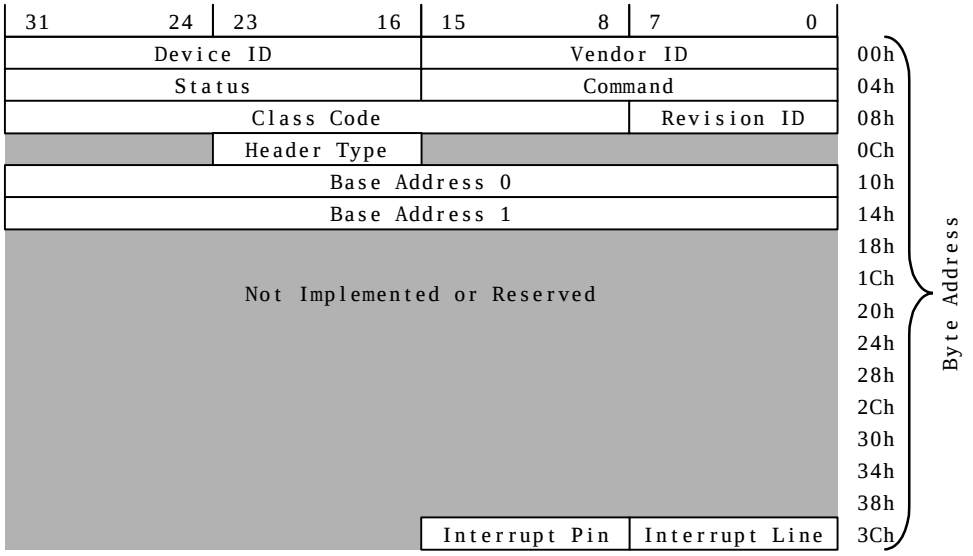


Figure 3-2: Vendor ID Register

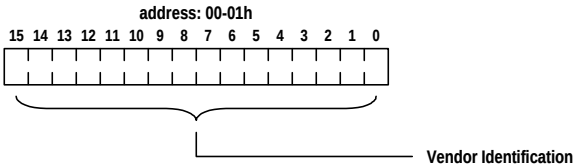


Figure 3-3: Device ID Register

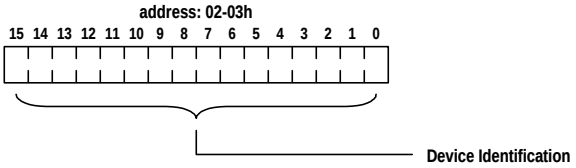


Figure 3-4: Revision ID Register

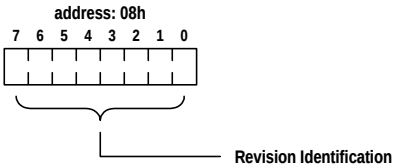


Figure 3-5: Header Type Register

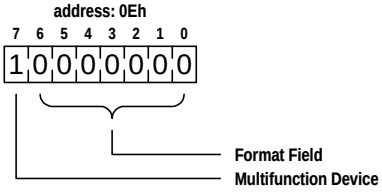
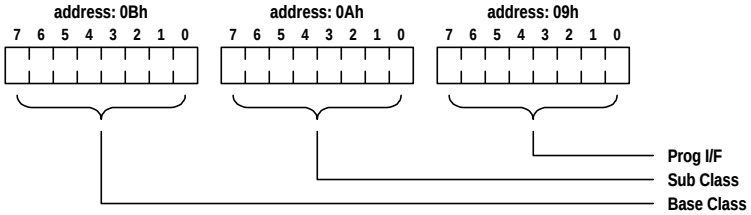


Figure 3-6: Class Code Register



side. Any write operation to this register will be ignored.

3.2 Device ID Register

This 16-bit field identifies the particular device. This identifier is allocated by the vendor. If the PCI-ISA-001 user has his own Vendor ID assigned by the PCI SIG the user will generate value for this field. If the user intends to use ISS's Vendor ID, the user needs to request ISS to generate an unique Device ID for the user. When RST# is removed, this 16-bit field is initialized from the external NV-RAM. After the value is boot-loaded from the NV-RAM, Device ID register can be read from PCI side. Any write operation to this register will be ignored.

3.3 Revision ID Register

This 8-bit register specifies a device specific revision identifier. Zero is an acceptable value. This field should be viewed as a vendor defined extension to the Device ID. The PCI-ISA-001 users are responsible for generating and maintaining this value. When RST# is removed, this 8-bit field is initialized from the external NV-RAM. After the value is boot-loaded from the NV-RAM, Revision ID register can be read from PCI side. Any write operation to this register will be ignored.

3.4 Header Type Register

This 8-bit register contains two read-only fields,

Bit 7 This bit is used to identify a multi-function device. If the bit is 0, then the device is single function. If the bit is 1, then the device has multiple functions. PCI-ISA-001 is a multifunction device and Bit 7 is hard-wired to 1.

Bit 6-0 Identifies the layout format of the registers between address 10h-3Fh in the Configuration Space. The layout for PCI-ISA-001 is indicated by a hard-wired encoding of 00h.

Any write operation to the Header Type register will be ignored.

3.5 Class Code Register

The Class Code register is used to identify the generic function of the device and, in some cases, a specific register-level programming interface. The register is broken into three byte-size fields. The upper byte at offset 0Bh is a base class code which broadly classifies the type of function the device performs. The middle byte at offset 0Ah is a sub-class code which identifies more specifically the function of the device. The lower byte at offset 09h identifies a specific register-level programming interface if needed, so that device independent soft-

ware can interact with the device. Table 3-1 shows the PCI defined encodings.

PCI-ISA-001 is a multifunction device and supports function 0 through 2. Therefore PCI-ISA-001 maintains three independent Class Code registers, one for each function, namely, Class Code #F0, Class Code #F1, and Class Code #F2 for function 0, function 1, and function 2 respectively.

When RST# is removed, all three Class Code registers are initialized from the external NV-RAM. After the value is boot-loaded from the NV-RAM, Class Code registers can be read from the PCI side. Any write operation to these registers will be ignored.

Table 3-1: Class Code Register Encodings

Base Class	Sub-Class	Intf.	Description
Device was built before Class Code definitions were finalized.			
00h	00h	00h	All currently implemented devices except VGA-compatible devices.
00h	01h	00h	VGA-compatible device.
Mass storage controller			
01h	00h	00h	SCSI bus controller.
01h	01h	xxh	IDE controller.
01h	02h	00h	Floppy disk controller.
01h	03h	00h	IPI bus controller.
01h	04h	00h	RAID controller.
01h	80h	00h	Other mass storage controller.
Network controller			
02h	00h	00h	Ethernet controller.
02h	01h	00h	Token Ring controller.
02h	02h	00h	FDDI controller.
02h	03h	00h	ATM controller.
02h	80h	00h	Other network controller.
Display controller			
03h	00h	00h	VGA-compatible controller. Memory addresses 0A0000h thru 0BFFFFh. I/O addresses 3B0h to 3BBh and 3C0h to 3DFh and all aliases of these addresses.
03h	00h	01h	8514-compatible controller -- 2E8h and its aliases, 2EAh-2EFh.
03h	01h	00h	XGA controller.
03h	80h	00h	Other display controller.
Multimedia device			
04h	00h	00h	Video device.
04h	01h	00h	Audio device.
04h	80h	00h	Other multimedia device.
Memory controller			
05h	00h	00h	RAM.
05h	01h	00h	Flash.
05h	80h	00h	Other memory controller.
Bridge device			
06h	00h	00h	Host bridge.
06h	01h	00h	ISA bridge.
06h	02h	00h	EISA bridge.
06h	03h	00h	MCA bridge.
06h	04h	00h	PCI-to-PCI bridge.
06h	05h	00h	PCMCIA bridge.
06h	06h	00h	NuBus bridge.
06h	07h	00h	CardBus bridge.
06h	80h	00h	Other bridge device.

Base Class	Sub-Class	Intf.	Description
Simple communication controllers			
07h	00h	00h	Generic XT-compatible serial controller.
07h	00h	01h	16450-compatible serial controller.
07h	00h	02h	16550-compatible serial controller.
07h	01h	00h	Parallel port.
07h	01h	01h	Bi-directional parallel port.
07h	01h	02h	ECP 1.X compliant parallel port.
07h	80h	00h	Other communications device.
Base system peripherals			
08h	00h	00h	Generic 8259 PIC.
08h	00h	01h	ISA PIC.
08h	00h	02h	EISA PIC.
08h	01h	00h	Generic 8237 DMA controller.
08h	01h	01h	ISA DMA controller.
08h	01h	02h	EISA DMA controller.
08h	02h	00h	Generic 8254 system timer
08h	02h	01h	ISA system timer.
08h	02h	02h	EISA system timers (two timers).
08h	03h	00h	Generic RTC controller.
08h	03h	01h	ISA RTC controller.
08h	80h	00h	Other system peripheral.
Input devices			
09h	00h	00h	Keyboard controller.
09h	01h	00h	Digitizer (pen).
09h	02h	00h	Mouse controller.
09h	80h	00h	Other input controller.
Docking stations			
0Ah	00h	00h	Generic docking station.
0Ah	80h	00h	Other type of docking station.
Processors			
0Bh	00h	00h	386.
0Bh	01h	00h	486.
0Bh	02h	00h	Pentium.
0Bh	10h	00h	Alpha.
0Bh	20h	00h	PowerPC.
0Bh	40h	00h	Co-processor.
Serial bus controllers			
0Ch	00h	00h	FireWire (IEEE 1394).
0Ch	01h	00h	ACCESS.bus.
0Ch	02h	00h	SSA.
0Ch	03h	00h	Universal Serial Bus (USB).
0Ch	04h	00h	Fibre Channel.
Class code 0Dh - FEh is Reserved			
Other Devices			
FFh			Device does not fit in any defined classes.

3.6 The Command Register

This 16-bit register provides coarse control over a PCI device's ability to generate and respond to PCI cycles. When a 0 is written to this register, the PCI-ISA-001 is logically disconnected from the PCI bus for all accesses except configuration accesses. The definition of the bit-fields follows:

Bit 0 I/O Space. Controls a device's response to I/O Space accesses. A value of 0 disables the PCI-ISA-001 response. A value of 1 allows the PCI-ISA-001 to respond to I/O Space accesses. State after RST# is 0.

Bit 1 Memory Space. Controls a device's response to Memory Space accesses. A value of 0 disables the PCI-ISA-001 response. A value of 1 allows the PCI-ISA-001 to respond to Memory Space accesses. State after RST# is 0.

Bit 2 Bus Master. Controls PCI-ISA-001's ability to act as a master on the PCI bus. A value of 0 disables the PCI-ISA-001 from generating PCI accesses. A value of 1 allows the PCI-ISA-001 to become bus master during DMA operation. State after RST# is 0.

Bit 3 Special Cycle. Controls a device's action on Special Cycle operations. PCI-ISA-001 does not support Special Cycle operations and this bit is hard-wired to 0.

Bit 4 Memory Write and Invalidate. This is an enable bit for using the Memory Write and Invalidate command. PCI-ISA-001 does not generate Memory Write and Invalidate command and this bit is hard-wired to 0.

Bit 5 VGA Palette Snoop. This bit controls how VGA compatible and graphics devices handle accesses to VGA palette registers. PCI-ISA-001 does not respond to palette register writes and snoops the data. It treats palette accesses like all other accesses and this bit is hard-wired to 0.

Bit 6 Parity Error Response. This bit controls the PCI-ISA-001's response to parity errors. When the bit is set, the PCI-ISA-001 takes its normal action when a parity error is detected. When the bit is 0, the PCI-ISA-001 ignores any parity errors that it detects and continue normal operation. This bit's state after RST# is 0. PCI-ISA-001 still generates parity even if parity checking is disabled.

Bit 7 Wait Cycle Control. This bit is used to control whether or not a device does address/data stepping. PCI-ISA-001 never does stepping and this bit is hard-wired to 0.

Bit 8 SERR# Enable. This bit is an enable bit for the SERR# driver. A value of 0 disables the SERR# driver. A value of 1 enables the SERR# driver. This bit's state after RST# is 0. This bit (and bit 6) must be on to report address parity errors.

Bit 9 Fast Back-to-back Enable. This optional read/write bit controls whether or not a master can do fast back-to-back transactions to different devices. PCI-ISA-001 does not generate fast back-to-back transactions and this bit is hard-wired to 0.

Bit 10-15 Reserved.

3.7 The Status Register

This 16-bit register is used to record status information for PCI bus related events. Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. *A bit is reset whenever the register is written, and the data in the corresponding bit location is a 1.* For instance, to clear bit 14 and not affect any other bits, write the value 0100_0000_0000_0000b to the register. The definition of the bit-fields follows:

Bit 0-4 Reserved.

Bit 5 66 Mhz Capable. PCI-ISA-001 is not 66 MHz capable and has this bit hard-wired to 0.

Figure 3-7: Command Register

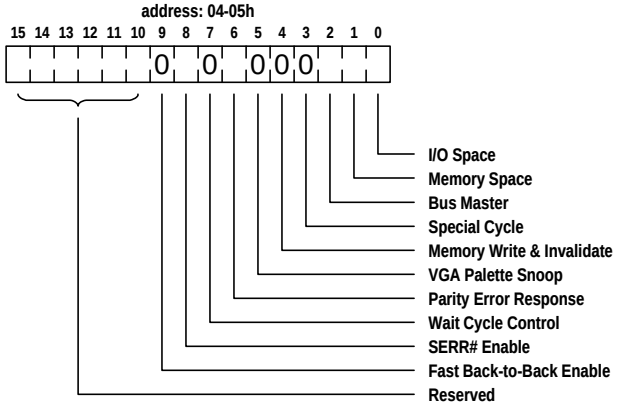


Figure 3-8: Status Register

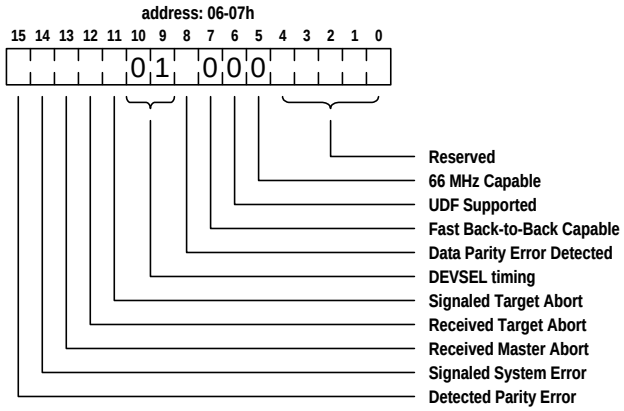


Figure 3-9: Interrupt Line Register

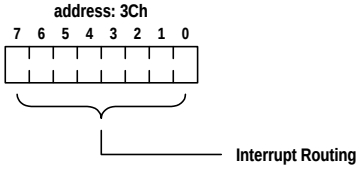
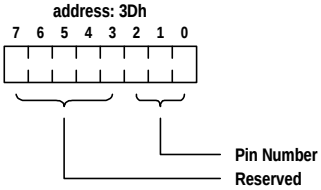


Figure 3-10: Interrupt Pin Register



Bit 6 UDF Supported. PCI-ISA-001 does not support UDF and has this bit hard-wired to 0.

Bit 7 This optional read-only bit indicates whether or not the target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent. PCI-ISA-001 has this bit hard-wired to 0.

Bit 8 Data Parity Error Detected. This bit is only implemented by bus masters. It is set when three conditions are met: 1) the bus agent asserted PERR# itself or observed PERR# asserted; 2) the agent setting the bit acted as the bus master for the operation in which the error occurred; and 3) the Parity Error Response bit (Command register) is set.

Bit 9-10 DEVSEL Timing. These bits encode the timing of DEVSEL#. These are encoded as 00b for fast, 01b for medium, and 10b for slow (11b is reserved). PCI-ISA-001 has these bits hard-wired to 01b.

Bit 11 Signaled Target-Abort. This bit is set by PCI-ISS-001 acting as a target device whenever it terminates a transaction with Target-Abort.

Bit 12 Received Target-Abort. This bit is set by PCI-ISA-001 acting as a master device whenever its transaction is terminated with Target-Abort.

Bit 13 Received Master-Abort. This bit is set by PCI-ISA-001 acting as a master device whenever its transaction (except for Special Cycle) is terminated with Master-Abort.

Bit 14 Signaled System Error. This bit is set whenever PCI-ISA-001 asserts SERR#.

Bit 15 This bit is set by PCI-ISA-001 whenever it detects a parity error, even if parity error handling is disabled by writing a 0 to the bit 6 location in the Command Register.

3.8 Interrupt Line Register

This is an 8-bit read/write register used to communicate interrupt line routing information. POST software will write the routing information into this register as it initializes and configures the system. The value in this register tells which input of the system interrupt controller(s) the PCI-ISA-001's interrupt pin is connected to. The PCI-ISA-001 itself does not use this value, rather it is used by device drivers and operating systems. Device drivers and operating systems can use this information to determine priority and vector information. Values in this register are system architecture specific.

3.9 Interrupt Pin Register

This 8-bit register tells which PCI interrupt pin is connected to PCI-ISA-001 INTX# pin. A value of 1 corresponds to INTA#. A value of 2 corresponds to INTB#. A value of 3 corresponds to INTC#. A value of 4 corresponds to INTD#. Add-on applications that do not use any PCI interrupt pin must put a 0 in this register.

3.10 Base Address Registers

One of the most important functions for enabling superior configurability and ease of use is the ability to relocate PCI devices in the address spaces. At system power-up, device independent configuration software determines what devices are present and then builds a consistent address map before booting the machine to an operating system. This means it has to determine how much memory is in the system, and how much memory and I/O address space the PCI devices in the system require. After determining this information, power-up software can map the PCI devices into reasonable locations and proceed with system boot. In order to do this mapping in a device independent manner, the 32-bit Base Address Registers are used.

Base Address Registers specify how much space it requires by making a number of lower bits read-only to the power-up configuration software. For instance if an Add-on device to the PCI-ISS-001 contains 256 bytes of registers which need to be accessed by the host device driver, the Base Address Register assigned to this space will have lower 8 bits read only so that $2^8 = 256$. Power-up software can determine how much address space the device requires by determining how many lower bits are read-only. This implies that all address spaces used are a power of two in size, and are naturally aligned. Now the power-up configuration software will determine the PCI address location where this 256-byte space can be mapped. This location must be a 256-byte aligned location, which implies the lower 8-bits of this location will be zero, and will act as the Base Address to the 256-byte space once mapped. The power-up configuration software will write this address to the Base Address Register. For example if the Base Address is 0ABCD1200h, then an access to the range 0ABCD1200h-0ABCD12FFh will be intended for those 256-byte registers on the Add-on device.

During normal operation when PCI-ISA-001 detects an access on the PCI bus, it compares the access address against all the Base Address Registers. For the comparison only the upper read-write bits in each Base Address Register are used. If there is match against any of the Base Addresses, then the access is claimed by the PCI-ISA-001.

Bit 0 of a Base Address Register is used to determine whether the register maps into Memory or I/O Space. Bit 0 of a Base Address Register that maps to Memory space is indicated by a value of 0. Bit 0 of a Base Address register that maps to I/O space is indicated by a value of 1.

Base Address Registers that map into I/O space has bit 1 as reserved and the minimum I/O space size that can be specified is 4 bytes. Base Address Registers that map into memory space use bits 3-1 for indicating additional space and access attributes applicable to the space requested. Therefore the minimum memory space size that can be specified is 16 bytes.

PCI-ISA-001 Add-on bus address space is limited to 11 bits. Therefore the maximum memory or I/O space that can be claimed by in a Base Address Register is $2^{11} = 2048$ bytes. Therefore, a subset or all of the low 11 bits of a Base Address Register used for mapping devices on the Add-on bus can be made read-only by PCI-ISA-001. Which bit are to behave as read-only is specified in the E^2PROM. The values for the read-only bits are also specified in the E^2PROM and boot-loaded into PCI-ISA-001 during RST#.

Figure 3-11: Memory Base Address Register

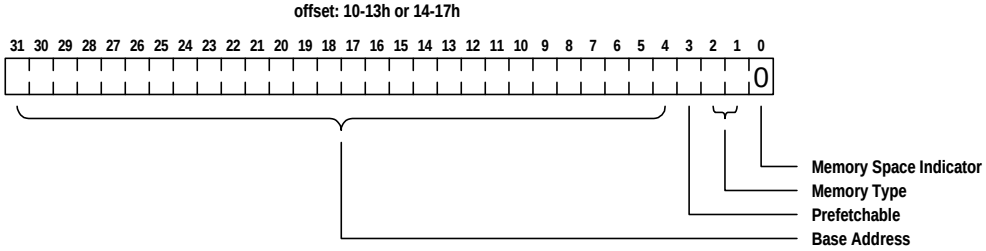
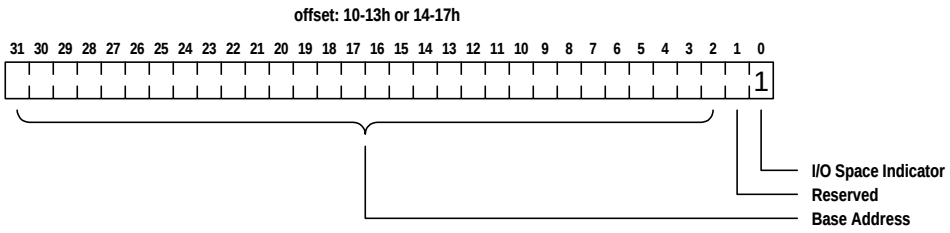


Figure 3-12: I/O Base Address Register



The definition of the bit-fields for a Base Address Register mapped to memory space follows:

Bit 0 Memory Space Indicator. Mapped to memory space is indicated by a value of 0.

Bit 1-2 Memory Type field determines how the space is mapped in the host PCI memory space. Table 3-2 shows the encoding used for this field.

Bit 3 Prefetchable. Set to 1 if there are no side effects on reads, the device returns all bytes on reads regardless of the byte enables, and host bridges can merge processor writes into this range without causing errors. This bit must be set to 0 otherwise.

Bit 4-31 Base Address. Programmable bits to hold the memory base address as well as the size indication for the space mapped at the base address.

The definition of the bit-fields for a Base Address Register mapped to I/O space follows:

Bit 0 I/O Space Indicator. Mapped to I/O space is indicated by a value of 1.

Bit 1 Reserved.

Bit 2-31 Base Address. Programmable bits to hold the I/O base address as well as the size indication for the space mapped at the base address.

There are total of 6 Base Address Registers in PCI-ISA-001. As a multifunction device PCI-ISA-001 supports function 0, function 1, and function 2. Each function is allocated 2 Base registers at address 10h and 14h. The Base Address registers are identified as,

Function 0 – 10h Base Address 0 #F0 Register.

Function 0 – 14h Base Address 1 #F0 Register.

Function 1 – 10h Base Address 0 #F1 Register.

Function 1 – 14h Base Address 1 #F1 Register.

Function 2 – 10h Base Address 0 #F2 Register.

Function 2 – 14h Base Address 1 #F2 Register.

Table 3-2: Base Address Register Bits 2-1 Encoding mapped to memory space.

00	Base register is 32 bits wide and mapping can be done anywhere in the 32-bit Memory Space.
01	Base register is 32 bits wide but must be mapped below 1M in Memory Space.
10	PCI-ISA-001 does not support 64 bit addressing and this value should never be used by the user.
11	Reserved.

All the Base Address Registers except Base Address Register 1 #F2 can be used to map any space in the Add-on device connected to PCI-ISA-001. The Base Address1 #F2 Register is dedicated to map the DMA Registers on PCI-ISA-001. As a result, the read-only bits of the Base Address 1 #F2 Register is hardwired, and can only be mapped to memory space. Rest of the Base Address Registers can be programmed from E^2PROM during RST# and the map will depend on which bits are made read-only and their values.

The definition of the bit-fields for the Base Address 1 #F2 Register as mapped to memory space follows:

Bit 0 Memory Space Indicator = 0.

Bit 1-2 Memory Type = 00.

Bit 3 Prefetchable = 0.

Bit 4-31 Base Address. These bit are all R/W. The mapped memory space size will be 16 bytes.

4.0 DMA Control Registers

PCI-ISA-001 can transfer data back and forth between PCI and the ISA-like interface under DMA control with the assistance of the DMA engine inside PCI-ISA-001. The DMA engine performs the transfer in accordance with the parameters written in the DMA Control Registers by the controlling software such as a device driver. The DMA Control Registers are accessible at the memory address mapped by the Base Address Register 1 for function 2 from the PCI side. For example, if the Base Address specified in Base Address 1 #F2 Register is 0ABCD1230H by the configuration software during PCI-ISA-001 initialization, a PCI access to address 0ABCD1234H will access the DMA Control Register located at byte address offset 4. *DMA Control Registers can only be mapped to memory space* and can not be mapped to the I/O space.

There are a total of four DMA Control Registers, each of which is 32-bits wide. They are,

Byte offset 00h – Mode Register.
Byte offset 04h – PCI Address Register.
Byte offset 08h – Add-on Address Register.
Byte offset 0Ch – Transfer Count Register.

The following sections explain each of the DMA Control Registers in detail.

4.1 Mode Register

This 32-bit register provides various controls necessary to perform a DMA operation in various addressing modes. After the device driver writes to the PCI Address Register, Add-on Address Register, and the Transfer Count Register, this register will be written to initiate the intended DMA transfer.

Part of the Mode Register (bit 9-12) is also used to control the In-Circuit update of the E^2PROM or read the contents of the E^2PROM. Device driver software should be careful not to change the contents of bit 9-11 while programming the rest of the bits of the Mode Register for DMA operation.

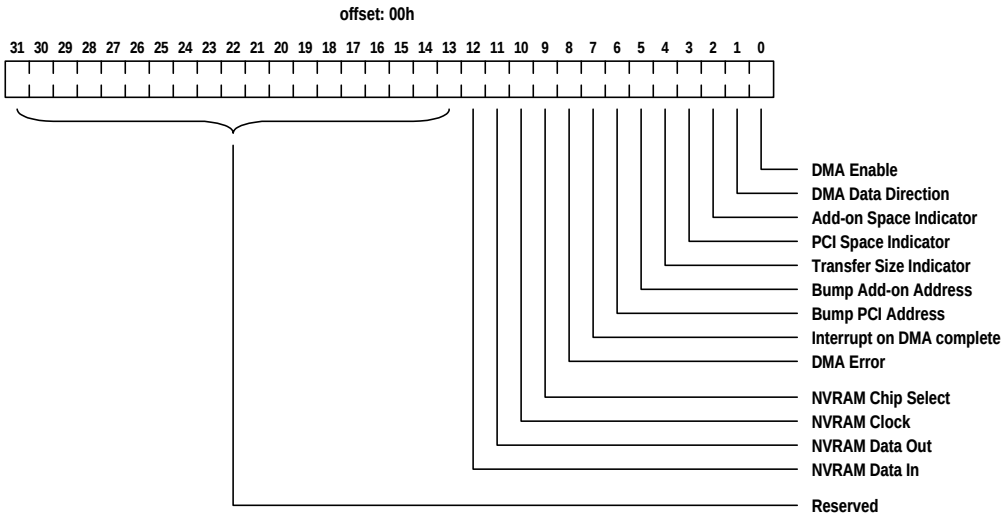
The definition of the bit-fields for the Mode Register follows:

Bit 0 DMA Enable. When this bit is set to 1 it starts the DMA transfer cycle. As the DMA transfer continues this bit remains asserted during the transfer process. When the Terminal Count is reached or DMA transfer failed due to some error, this bit is set back to 0. Device driver software can also write a 0 in this bit location to abort the current DMA transfer cycle.

Bit 1 DMA Data Direction. This bit indicates the DMA data transfer direction. When this is set to 0, the DMA engine will read data from the PCI address space and write the data back to the Add-on device or logic. If this bit is set to 1 the DMA engine will read data from the Add-on device or logic and write the data back to the PCI address space.

Bit 2 Add-on Space Indicator. There are two address spaces supported on the ISA like Add-on interface in PCI-ISA-001. The first is the Memory-Register space and the second is the Stream space. The Stream space behavior is similar to a FIFO operation and no address is provided to the Add-on device. If the Add-on Space Indicator bit is set to 0 then the DMA engine accesses data at the Stream space during the DMA transfer. At this point, the DMA engine will start watching the STRMRDY pin to see when the Add-on device asserts STRMRDY indicating it is ready to source or sink data at the Stream

Figure 4-1: Mode Register



space. As a result, the PCI-ISA-001 DMA engine will assert STRMSPC# to indicate it is transferring the current data while the Add-on device can get ready for the next data. When this bit is set to 1 then the DMA engine accesses data at Memory-Register space during the DMA transfer. The Add-on Address Register provides the Memory-Register address on the Add-on bus interface when the transfer takes place.

Bit 3 PCI Space Indicator. There are two address spaces supported on the PCI bus. The first is the memory space and the second is the I/O space. If this bit is set to 0 then the DMA engine accesses data at the PCI memory space during the DMA transfer. The PCI Address Register provides memory address on the PCI bus when the transfer takes place. When this bit is set to 1 then the DMA engine accesses data at the PCI I/O space during the DMA transfer. The PCI Address Register provides I/O address on the PCI bus when the transfer takes place.

Bit 4 Transfer Size Indicator. PCI-ISA-001 supports two different byte-size transfer modes at the PCI bus during a DMA cycle-Single-byte transfer mode, and Four-byte transfer mode. During Single-byte transfers, PCI-ISA-001 does a byte-wide PCI data transfer for each byte data transfer occurs on the Add-on bus for the DMA cycle. For a Four-byte transfer a DWORD (32-bit) transfer occurs on PCI bus for every four bytes transferred on the Add-on bus for the DMA cycle. If this bit is set to 0 then Single-byte transfer mode will be enabled. When this bit is set to 1, Four-byte transfer mode will be enabled.

Bit 5 Bump Add-on Address. If this bit is set to 0, the address stored in the Add-on Address Register will remain the same during the entire DMA transfer cycle. When this bit is 1, the address stored in the Add-on Address Register is incremented by one every time a byte is transferred at the Add-on bus by the DMA engine.

Bit 6 Bump PCI Address. If this bit is set to 0, the address stored in the PCI Address Register will remain the same during the entire DMA transfer cycle. When this bit is 1 during Single-byte transfer mode, the address stored in the PCI Address Register is incremented by one every time a byte is transferred at the PCI bus by the DMA engine. When this bit is 1 during Four-byte transfer mode, the PCI Address Register is incremented by four every time a DWORD is transferred at the PCI bus by the DMA engine.

Bit 7 Interrupt on DMA Complete. When the Transfer Count Register reaches a values of zero, PCI-ISA-001 is capable of generating a PCI interrupt to notify the device driver that the DMA transfer is complete. Similarly, if the DMA transfer is aborted due to any error condition PCI interrupt can also be generated. This bit enables the interrupt generation under such conditions. If this bit is set to 0, no interrupt will be generated. When this bit is set to 1, interrupt will be generated upon terminal count or DMA error. The interrupt service routine can write 0 at this bit location to clear the interrupt

Bit 8 DMA Error. This bit will be set by the DMA engine if it is unable to continue the intended DMA cycle. Such a condition occurs when DMA engine detects a master-abort or target-abort while trying to access data from the PCI bus. The device driver can clear this bit by writing a 0 at this bit location.

Bit 9 NV RAM Chip Select. This bit controls the value of the NVCS pin during normal operation. At RST# the NV RAM contents are downloaded to PCI-ISA-001 and the NV Sequencer directly controls the NVCS pin. During normal operation, NVCS pin has the same value as this bit. After the boot-load, this bit is set by PCI-ISA-001 to a value 0, and as a result the NVCS pin is also set to 0. This bit remains 0, unless NV RAM access is performed under device driver or other system configuration software control. In such a case this bit will be set to 1.

Bit 10 NV RAM Clock. This bit controls the value of the NVCLK pin during normal operation. At RST# the NV RAM contents are downloaded to PCI-ISA-001 and the NV Sequencer directly controls the NVCLK pin. During normal operation, NVCLK pin has the same value as this bit whenever Bit 9 of the Mode Register is 1. Device driver or other system configuration software can control this bit to read or write the NV RAM contents.

Bit 11 NV RAM Data Out. This bit controls the value of the NVWRTDATA pin during normal operation. At RST# the NV RAM contents are downloaded to PCI-ISA-001 and the NV Sequencer directly controls the NVWRTDATA pin. During normal operation, NVWRTDATA pin has the same value as this bit whenever Bit 9 of the Mode Register is 1. Device driver or other system configuration software can control this bit to read or write the NV RAM contents.

Bit 12 NV RAM Data In. This read-only bit has the same value as the NVRDDATA pin. Writing to this bit has no effect. Device driver or other system configuration software can read this bit to access the NV RAM contents.

Figure 4-2: PCI Address Register

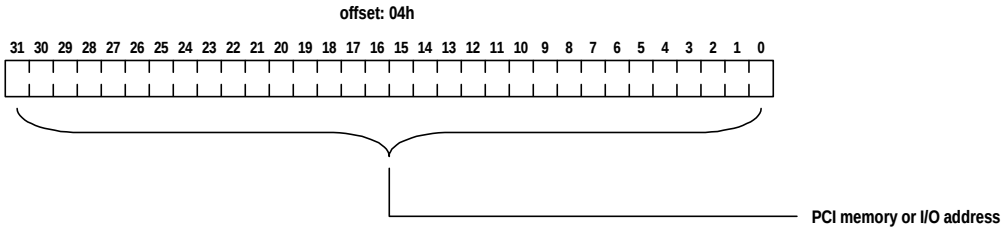


Figure 4-3: Add-on Address Register

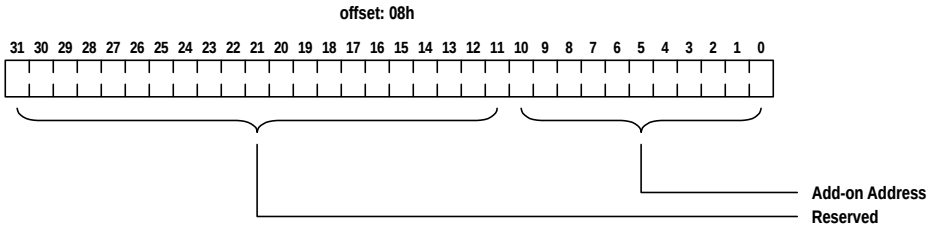
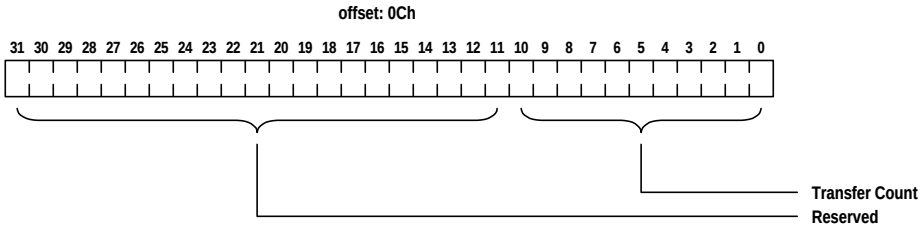


Figure 4-4: Transfer Count Register



4.2 PCI Address Register

This register contains a 32-bit address which is used on the PCI bus to read or write the DMA data at the PCI bus during DMA cycle. This address can be either a PCI memory address or a PCI I/O address. If Bit 3 of the Mode Register is 0, then the content of this register is used as a memory address. When Bit 3 of the Mode Register is 1, the content of this register is used as an I/O address. When bit 6 of the Mode Register is set to 0, the address stored in the PCI Address Register will remain the same during the entire DMA transfer cycle. If bit 6 of the Mode Register is 1 and bit 4 of the Mode Register is 0 (Single-byte transfer mode), the address stored in the PCI Address Register is incremented by one every time data is transferred at the PCI bus by the DMA engine. In such a case the data will always be a single byte. When both bit 6 and bit 4 of the Mode Register is 1 (Four-byte transfer mode), the address stored in the PCI Address Register is incremented by four every time data is transferred at the PCI bus by the DMA engine which will always be a DWORD.

When bit 4 of the Mode Register is 1 which is the Four-byte transfer mode, the initial address stored in the PCI Address Register by the device driver must be DWORD aligned to ensure correct operation on PCI-ISA-001.

4.3 Add-on Address Register

This register contains a 11-bit address which is used on the Add-on bus to read or write the DMA data in the Memory-Register space to the Add-on device or logic during DMA cycle. When bit 5 of the Mode Register is set to 0, the address stored in the Add-on Address Register will remain the same during the entire DMA transfer cycle. If bit 5 of the Mode Register is 1, the address stored in the Add-on Address Register will then be incremented by one each time data is transferred at the Add-on bus by the DMA engine. The data size at the Add-on bus is always a single-byte.

4.4 Transfer Count Register

This register contains an 11-bit value which keeps track of how many data transfers occurred at the PCI bus during a DMA cycle. the DMA engine decrements the value stored in the Transfer Count Register by one every time it transfers data at the PCI bus. In the Single-byte transfer mode, as indicated by bit 4 of the Mode Register = 0, the initial transfer count value stored in the Transfer Count Regis-

ter by the device driver should be the number of bytes intended to be transferred. During the Four-byte transfer mode, as indicated by bit 4 of the Mode Register = 1, the initial transfer count value stored in the Transfer Count Register by the device driver should be the number of DWORDS intended to be transferred.

When the Transfer Count Register value becomes zero the DMA engine stops transferring data, and if bit 7 of the Mode Register is 1 at this time a PCI interrupt will be generated. Also, if bit 2 of the Mode Register is 0, DMATC will be asserted at this time.

5.0 PCI to Add-on Address Mapping Resisters

PCI-ISA-001 allows a PCI master to access any Add-on device memory or registers so that they appear on the PCI memory or I/O address space. The PCI master can perform a memory or I/O access and PCI-ISA-001 will respond to the access and conduct the data transfer as a PCI target. The events that occur in this PCI bus transaction, are in the following order:

1. The PCI Master puts the memory or I/O access command and the corresponding address on the PCI bus.
2. All PCI targets including PCI-ISA-001 will see the address on the bus. As a PCI target PCI-ISA-001 will latch and compare the PCI address against each of the PCI Base Address Registers, and determine if it matches with any of the base addresses. In the case of a match, PCI-ISA-001 will claim the request and proceed with the access.
3. The Latched PCI address will be translated to Add-on address and an access is made to the Add-on device memory or register with that address. If it is a read operation the data read at the Add-on bus is sent back to the PCI bus for the PCI master to obtain it. For a write operation the write data sent from the PCI master is taken from the PCI bus and presented at the Add-on bus for the write to be performed.

The PCI address to the Add-on address translation is performed with the help of a set of two registers. PCI-ISA-001 contains one such set for each of the five PCI Base Address Register that is available for use for the Add-on device or logic access. The registers in the set are,

- Base Address Read-only Mask Register.
- Add-on Base Address Register.

Table 5-1 shows these registers along with the available PCI Base Address Registers. These registers can not be accessed directly from PCI. At RST# their values are boot-loaded from the E^2PROM and never change during normal operation.

Figure 5-1 illustrates how PCI-ISA-001 performs the Address translation process with the help of these registers. As required by PCI, a number of rightmost bits of a PCI Base Address Register need to be made read-only. The number of the bits, N, as shown in the figure, where 2^N is the number of bytes from the PCI addressing space need to be assigned for the Add-on memory or registers to be mapped. To accomplish this, the Base Address Read-only Mask Register is used, which is 11-bit wide. The rightmost N bits are boot-loaded with ones and the remaining leftmost 11-N bits are boot-loaded with zeros from the E^2PROM during RST#. If all the Base Address Read-only Mask Register bits are loaded with zeros then the corresponding PCI Base Address Register is inhibited. Any bit that is loaded with one in the Base Address Read-only Mask Register will make PCI-ISA-001 configure

Figure 5-1: PCI to Add-on Address translation.

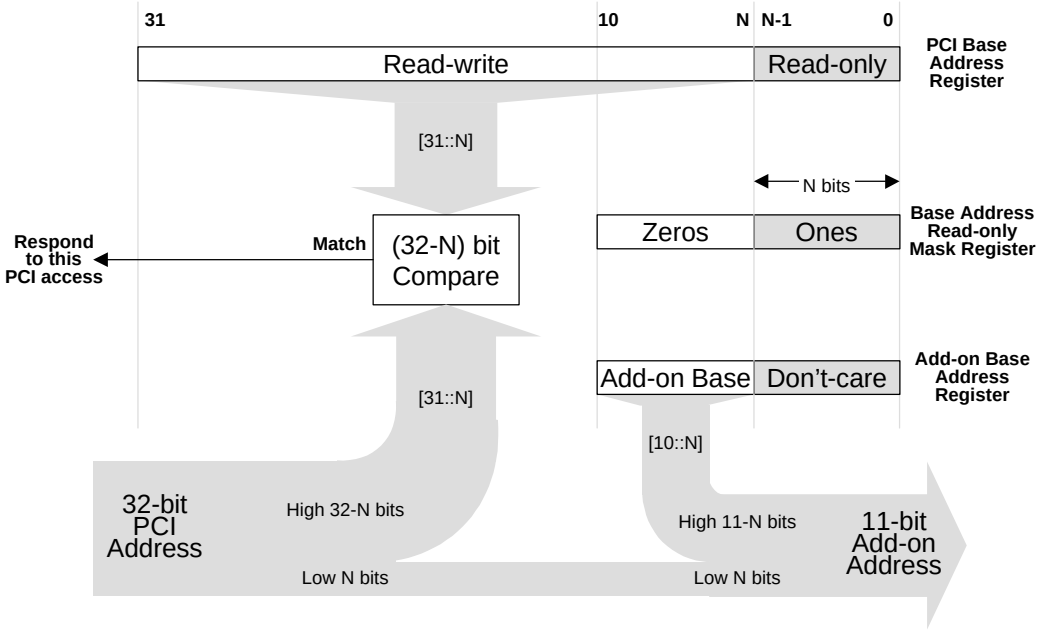
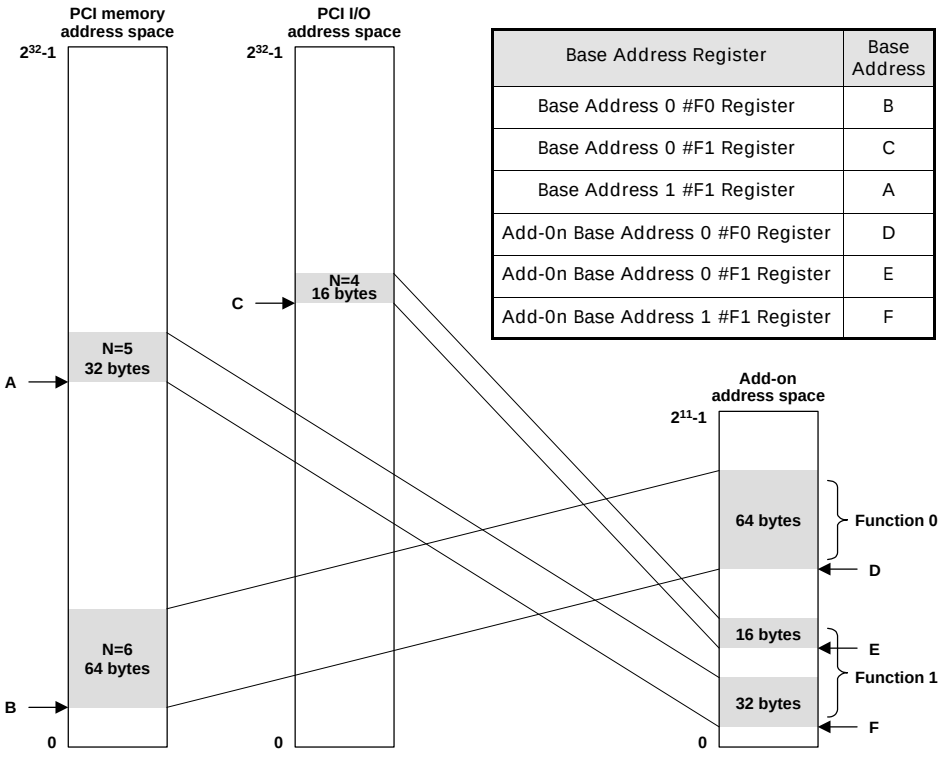


Figure 5-2: An address mapping scenario

Read-only Mask Register	N	Value
Base Address Read-only Mask 0 #F0 Register	6	000_0011_1111b
Base Address Read-only Mask 0 #F1 Register	4	000_0000_1111b
Base Address Read-only Mask 1 #F1 Register	5	000_0001_1111b



Base Address Register	Base Address
Base Address 0 #F0 Register	B
Base Address 0 #F1 Register	C
Base Address 1 #F1 Register	A
Add-0n Base Address 0 #F0 Register	D
Add-0n Base Address 0 #F1 Register	E
Add-0n Base Address 1 #F1 Register	F

the corresponding bit in the PCI Base Address Register as read-only. The initial read-only value for the right-most N bits in the PCI Base Address Register is also boot-loaded from the E^2PROM. During a PCI access, the 32-bit PCI address is latched by PCI-ISA-001, while the leftmost 32-N bits of the address is compared with the leftmost 32-N bits of the PCI Base Address Register. A match indicates the access is made within the PCI address range allocated for the corresponding PCI Base Address Register and PCI-ISA-001 will claim the access. The right-most N bits of the latched PCI address, which is the offset from the PCI base address is separated and concatenated with the leftmost 11-N bits of the Add-on Base Address Register. The resulting 11-bit value is used as the Add-on address to access the Add-on device memory or register. The Add-on Base Address Register is 11-bit wide and bit 11-2 value is boot-loaded from E^2PROM during RST# and bit 1-0 value is hard-wired to zeros.

Table 5-1: Address Mapping Registers to Map Add-on Devices

PCI Base Address Register	Base Address Read-only Mask Register	Add-on Base Address Register
Base Address 0 #F0 Register	Base Address Read-only Mask 0 #F0 Register	Add-on Base Address 0 #F0 Register
Base Address 1 #F0 Register	Base Address Read-only Mask 1 #F0 Register	Add-on Base Address 1 #F0 Register
Base Address 0 #F1 Register	Base Address Read-only Mask 0 #F1 Register	Add-on Base Address 0 #F1 Register
Base Address 1 #F1 Register	Base Address Read-only Mask 1 #F1 Register	Add-on Base Address 1 #F1 Register
Base Address 0 #F2 Register	Base Address Read-only Mask 0 #F2 Register	Add-on Base Address 0 #F2 Register

Figure 5-2 shows the effect of the address translation within both address spaces in an example scenario. In this scenario, the Add-on interface has two add-on devices. The registers on the first device are located at address D and occupy 64 bytes in the Add-on address space where D must be a 64-byte aligned address. The second device has two sets of registers. The first set is located at address E and occupies 16 bytes in the Add-on address space where E must be a 16-byte aligned address. Similarly the second set is located at address F and occupies 32 bytes in the Add-on address space where F must be a 32-byte aligned address.

This scenario maps the first device with the function 0 base registers and the second device with the function 1 base registers. During boot-load the Add-on Base Address Registers and the Base Address Read-only Mask Registers will be loaded with values shown in Figure 5-2. Also, the read-only values of the PCI Base Address Register are loaded at this time so that the first device maps to the PCI memory space, the first set of registers in the second device maps to the PCI I/O space, and the second set to the PCI memory space. Later, during the POST operation the PCI Base Address Registers will be written with values shown in Figure 5-2 by power-on configuration software using PCI configuration accesses to the PCI-ISA-001 before normal operation begins.

6.0 PCI Bus Operation

This section describes the PCI bus operation as it relates with PCI-ISA-001. However, this section is not intended to be a complete reference for the PCI bus protocol and the PCI specification should be consulted when a detailed explanation of the PCI bus operation is needed.

6.1 Arbitration

PCI is a synchronous bus and a central clock fed to all the PCI devices clocks any operation that occurs on the PCI bus. PCI bus architecture allows a PCI device to initiate data transfer to and from another PCI device in a PCI bus transaction. In order to initiate data transfer, a PCI device first arbitrates for the bus ownership. The PCI device sends request for the bus ownership to a central arbiter by asserting it's REQ# pin. The arbiter grants the request by asserting the GNT# pin of the requesting PCI device. At this point the requesting device can become the bus master to proceed with the bus transaction and is called a PCI Master. The Master can access any of the remaining PCI devices and those PCI devices are called acting as PCI Targets. A PCI device must be able to act as a Target. However, some PCI devices may not be capable of becoming a Master. Figure 6-1 shows the arbitration signals during this process. After completion of the PCI bus transaction, next arbitrating device may become the Master when it receives its GNT#. Between two transactions the bus enters the benign state called Idle. In Idle state FRAME# and IRDY# signals remain at logic zero.

6.2 Normal PCI Transactions

A Master conducts any PCI bus transaction in two phases. The Address phase and the Data phase. The Master starts the transaction by asserting FRAME# while it puts the bus command such as read or write on the C/BE[3::0]# pins and the PCI address of the transaction on the AD[31::0] pins. This state of the bus is called the address phase. Table 2-1 lists all the PCI bus commands supported by PCI-ISA-001. Address phase is only one clock long and PCI-ISA-001 latches the PCI address during this phase. Following the Address phase, Master asserts the IRDY# pin to begin the Data phase. During the Data phase, first, all the Targets examine the PCI address to determine if it should respond to the transaction. A Target can respond to the transaction by asserting the DEVSEL# signal. If no Target responds within the next four clocks, Master terminates the transaction, known as Master-Abort. Figure 6-2 shows a Master-Abort. The dotted lines indicate floating state for sustained tri-state signals. The actual data transfer occurs during the remainder of the Data phase. At this time AD[31::0] will carry the data bits.

The Target asserts #TRDY as soon as it is ready for the intended data transfer. In case of a write transaction, the Master will continue to drive the AD[31::0] during the data phase. In case of a read, the Master will stop driving the AD[31::0] right after the address phase. In such a case, Target will start driving AD[31:0] at least a clock later to allow the bus to turn around. The data transfer occurs anytime both IRDY# and TRDY# are asserted in the same clock. A Master can extend the Data phase by

Figure 6-1: PCI Bus Arbitration Request by PCI-ISA-001

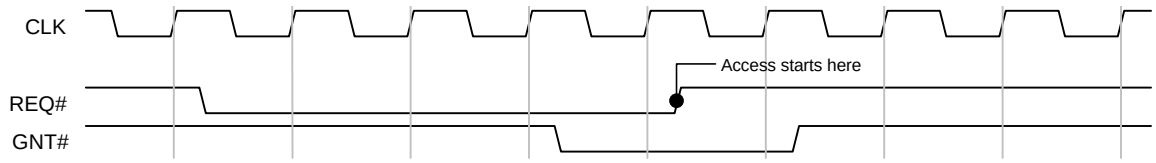


Figure 6-2: PCI Master-Abort

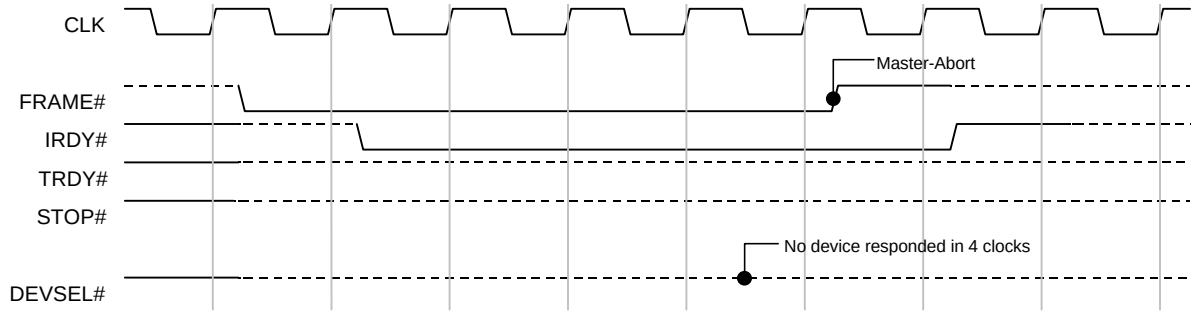


Figure 6-3: PCI Read by PCI-ISA-001

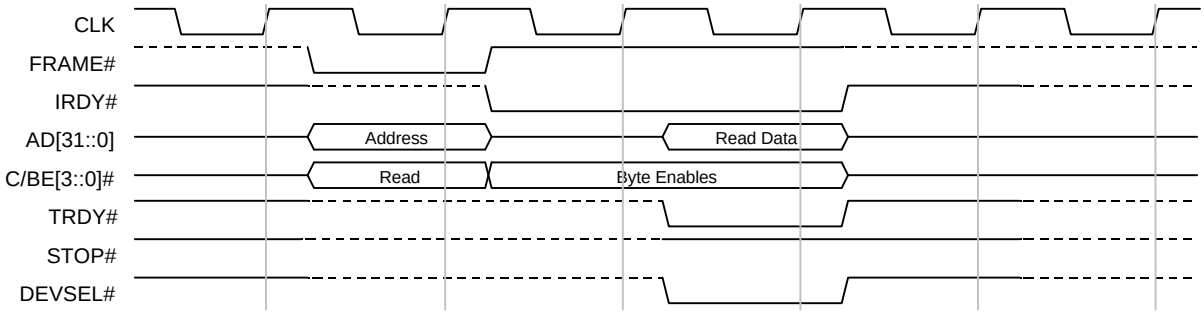


Figure 6-4: PCI Write by PCI-ISA-001

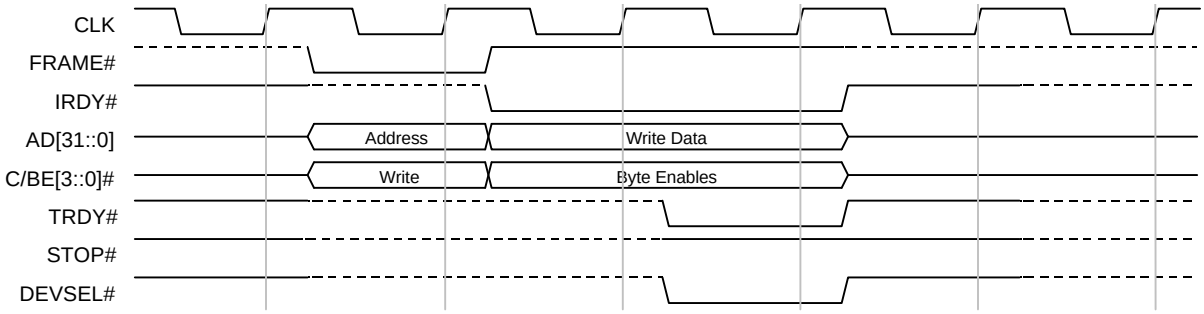
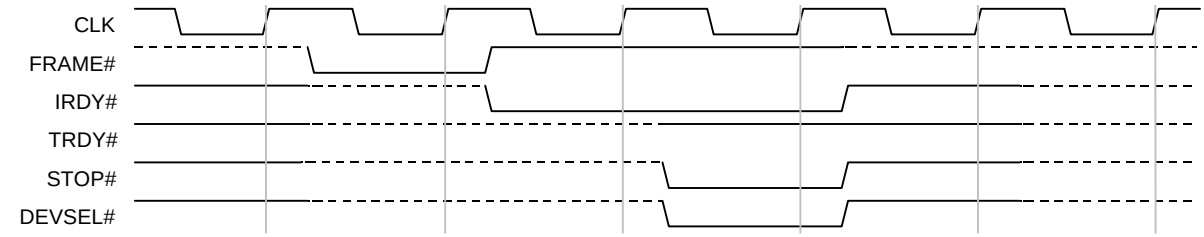


Figure 6-5: PCI Target-Retry



de-asserting IRDY#. Similarly a Target can extend the Data phase by de-asserting TRDY#. During data transfer, Master drives C/BE[3::0] indicating which bytes on the AD[31::0] has meaningful data.

The normal termination of the data phase happens when the Master completes the bus transaction. During data phase Master may de-assert the FRAME# while asserting #IRDY indicating its intention of the final data transfer. At this time when the Target asserts TRDY#, the final data transfer occurs. At the next clock, the Master de-asserts the IRDY# and the Target de-asserts TRDY# and #DEVSEL#. This concludes the transaction while the bus enters into the Idle state. PCI-ISA-001 always performs single data transfers and will de-assert the #FRAME in the very first clock it enters the Data Phase as a Master. Figure 6-3 shows a PCI read operation and Figure 6-4 shows a PCI write operation while PCI-ISA-001 is acting as a Master.

6.3 Incomplete PCI Transactions

A Target can also initiate the termination of the data phase before Master completes the transaction. Target can do so by asserting STOP# during the data phase to signal Master to terminate. Data transfer occurs only when both IRDY# and TRDY# are asserted in the same clock at the time of termination. Target may choose to assert or de-assert TRDY# when it asserts STOP# depending on whether it can perform the data transfer or not. If there is data transfer, TDRY# will be de-asserted. Master will de-asserts FRAME# in response to STOP# assertion, and will de-assert IRDY# a clock later, to enter the bus into the Idle state. Such a termination is known as Target-Disconnect. The Master may later on chose to retry the unfinished part of the transfer in a separate transaction. If the Target Asserts STOP# at the very first clock of the Data Phase without asserting TRDY#, it indicates

Figure 6-6: Target-Disconnect by PCI-ISA-001

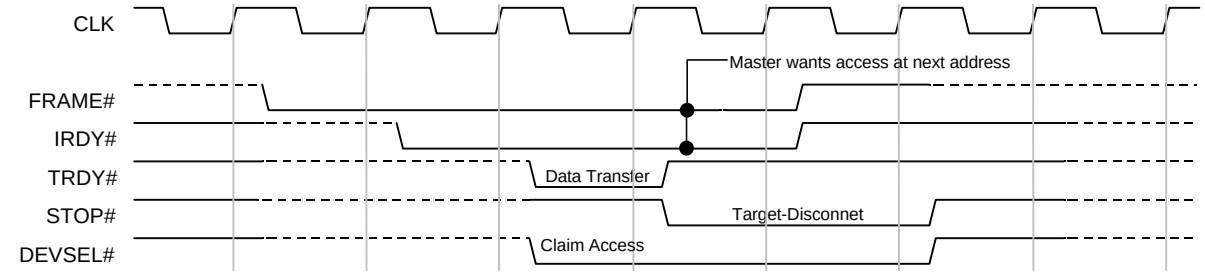
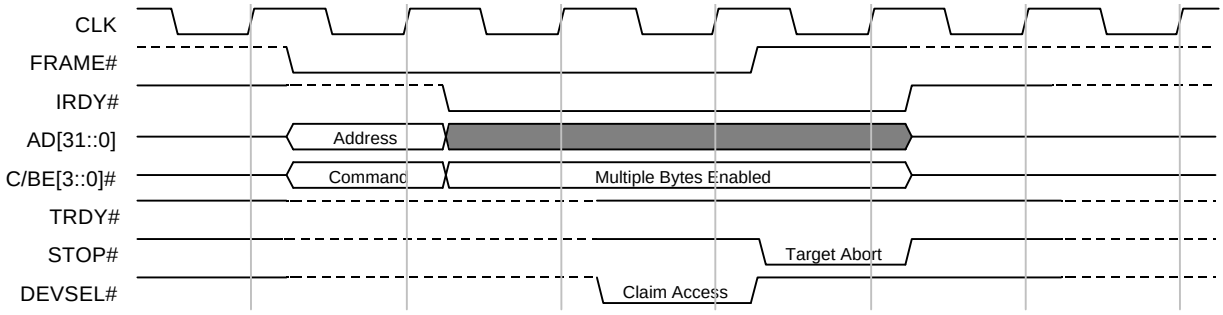


Figure 6-7: Target-Abort by PCI-ISA-001 while a Master is accessing Add-on device



that Target can not perform the data transfer at this time. This is known as Target-Retry and Master must retry the transaction at a later time. If PCI-ISA-001 acting as a Master detects a Target-Retry, it automatically retries the transaction later. Figure 6-5 shows an example of Target-Retry. PCI-ISA-001 always performs single data transfers and a PCI-ISA-001 acting as a Target will always assert #STOP without asserting #TRDY to signal Target-Disconnect if a Master tries to transfer more than one data. Figure 6-6 shows how PCI-ISA-001 performs a Target-Disconnect.

If Target can not provide the data and does not want the Master to retry the data later, the Target asserts STOP#, while de-asserting the #TRDY, and DEVSEL#. In this case the transaction terminates the same way as Target-Retry or Target-Disconnect. However, Master will treat this as an error condition and not retry. This type of termination is called a Target-Abort. PCI-ISA-001 is capable of generating an interrupt upon detection of a Target-Abort when it is acting as a Master. The only time PCI-ISA-001 acts as a Master is when it performs a DMA operation and detection of a Target-Abort will fail the DMA operation.

6.4 Accesses to PCI-ISA-001

When a Master tries to access the Add-on device, PCI-ISA-001 acting as a Target expects those accesses as single byte access. PCI-ISA-001 will terminate an access to the Add-on device of a different size with a Target-Abort. However, if a Master tries to access the configuration space and the DMA register, all sizes are supported by PCI-ISA-001. Figure 6-7 shows how PCI-ISA-001 performs a Target-Abort.

7.0 Add-on Bus Operation

The ISA-like Add-on bus is designed such that the devices built for ISA adapter cards can be integrated with the PCI-ISA-001 device without any difficulty. This enables immediate low-cost migration of existing ISA adapter cards to the PCI architecture.

PCI-ISA-001 always acts as the master on the Add-on Bus and add-on device act as a slave. PCI-ISA-001 controls any accesses that occur to the add-on devices. The Add-on Bus architecture supports two different address spaces for accesses to any add-on device. They are,

- Memory-Register space.
- Stream space.

The Memory-Register space accesses need an address to be provided to the Add-on device. In such a case, PCI-ISA-001 accesses the add-on device location specified by this address. The Stream space behaves much like a FIFO interface. During an access in the Stream space, PCI-ISA-001 does not provide any address. The Add-on device indicates when it is ready for the Stream access by asserting STRMRDY. In response PCI-ISA-001 will perform the access to the Stream space. STRMSPC# indicates when a Stream space access is in progress. During a Memory-Register space access STRMSPC# is deasserted and during Stream space access STRMSPC# is asserted by PCI-ISA-001.

7.1 Memory-Register Space Access

A Memory-Register space access consists of an address phase followed by a data phase. The Address[10::8] is presented at ADR[10::8] pins and stays the same during both phases. Address[7::0] and Data[7::0] are multiplexed on the same physical pins, DATADR[7::0].

During the address phase ALE is asserted and Address[7::0] is presented at DATADR[7::0] pins by PCI-ISA-001. ALE can be used latch the address for the Add-on device. In case of a write access, the 8-bit write data is presented at the DATADR[7::0] pins while WR# is asserted by PCI-ISA-001 during the data phase. In case of a read access, RD# is asserted by PCI-ISA-001 and the Add-on device drives DATADR[7::0] with the 8-bit read data during the data phase.

Figure 7-1 shows a Memory-Register space read operation and Figure 7-2 shows a Memory-Register space write operation.

A Memory-Register space access can be caused by the host processor, while the host processor is doing a read or write operation at the PCI bus. The Memory-Register space accesses can also be caused by the DMA engine inside PCI-ISA-001 as a result of a programmed DMA operation.

7.2 Stream Space Access

A Stream space access contains data phase only and does not contains any address phase. The Add-on device asserts STRMRDY when it is ready for a Stream Space access.

As it was mentioned earlier, Stream space behavior is similar to a FIFO interface. In case of an intended read, STRMRDY behaves like a FIFO-not-empty indication. In this case, RD# is asserted by PCI-ISA-001 and the Add-on device drives DATADR[7::0] with the 8-bit read data. Similarly, in case of an intended write, STRMRDY behaves like a FIFO-not-full indica-

Figure 7-1: Memory-Register Space Read

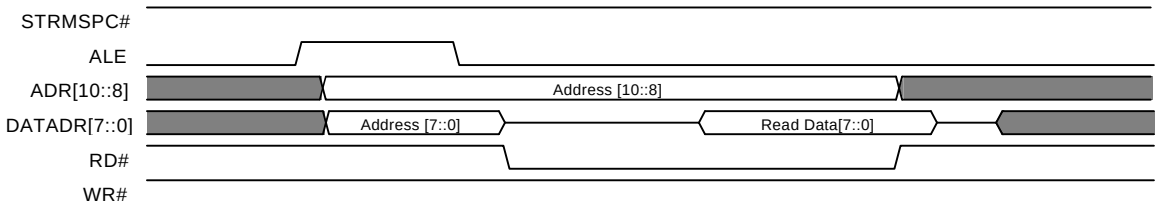
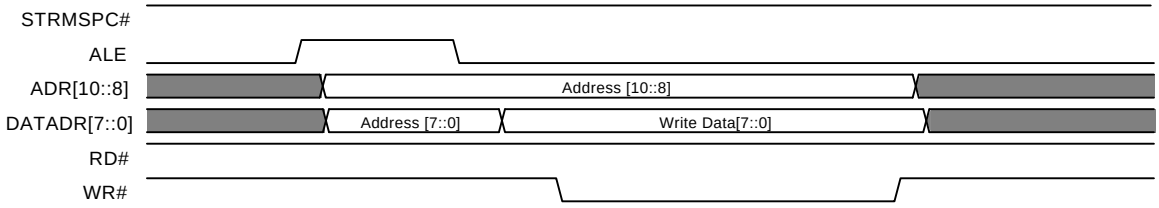


Figure 7-2: Memory-Register Space Write



tion. In this case, the 8-bit write data is presented at the DATADR[7::0] pins while WR# is asserted by PCI-ISA-001.

Figure 7-3 shows a Stream space read operation and Figure 7-4 shows a Stream space write operation. The Stream space accesses are always driven by the DMA engine inside PCI-ISA-001. DMA engine performs Stream space data transfers until the DMA engine reaches terminal count. The last data transfer occurs at this time and the DMATC is asserted by the PCI-ISA-001.

7.3 Extending the Data Phase

During the data phase of an access the Add-on device can assert the WAIT# to extend the data phase. The data phase will continue to be extended until the time WAIT# is de-asserted.

Figure 7-5 shows how a read can be extended. Similarly, Figure 7-6 shows how a write can be extended.

The Add-on devices must use caution to extend data phases. If the data phase is caused by a PCI

Figure 7-3: Stream Space Read

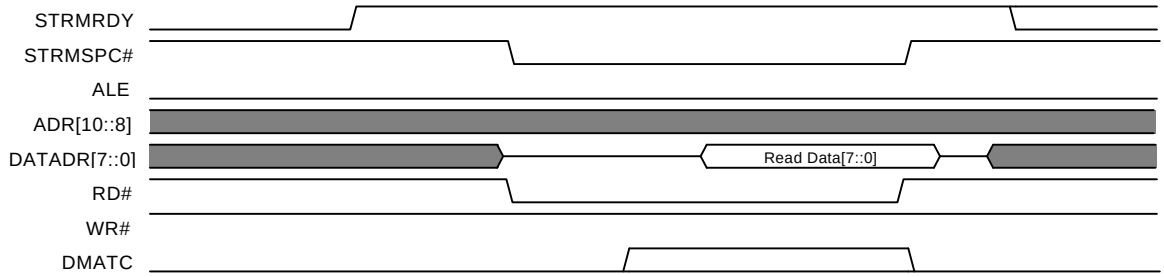


Figure 7-4: Stream Space Write

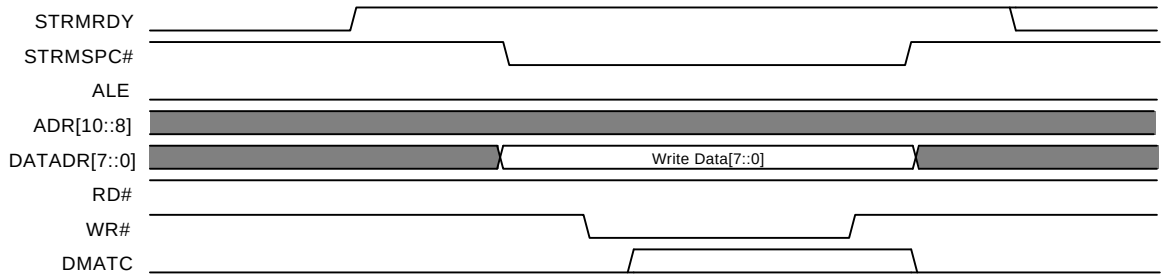


Figure 7-5: Extending a Read

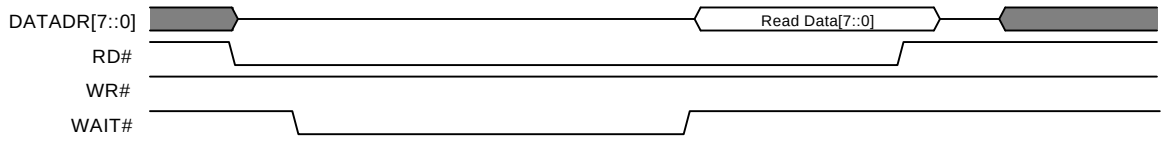
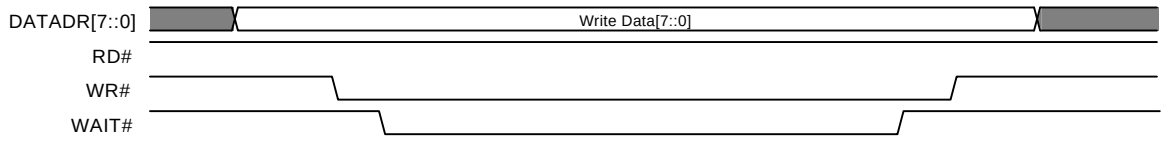


Figure 7-6: Extending a Write



access, the PCI access will also be extended in a such case. Extending PCI accesses beyond a certain point may seriously degrade system performance. Refer to PCI specification regarding details of extending PCI cycles and their limits.

7.4 ISA Adapter Card Migration to PCI Architecture

Low-cost PCI-ISA-001 is the ideal solution to enable existing ISA card vendors to migrate to PCI with minimal effort. The PCI-ISA-001 Add-on Bus signals make a natural fit for the devices that are built for ISA adapter cards in the industry.

Figure 7-7 shows an ISA adapter card with the signals that are likely to be seen with the devices that are used in such adapter cards. Figure 7-8 show how this ISA adapter board design can be transformed into a PCI adapter card with PCI-ISA-001 controller device and very minimal glue logic. This offers fast design turn-around time with reduced risk, since the PCI adapter card would leverage on an existing proven ISA adapter card design.

Figure 7-7: An Existing ISA Adapter Card

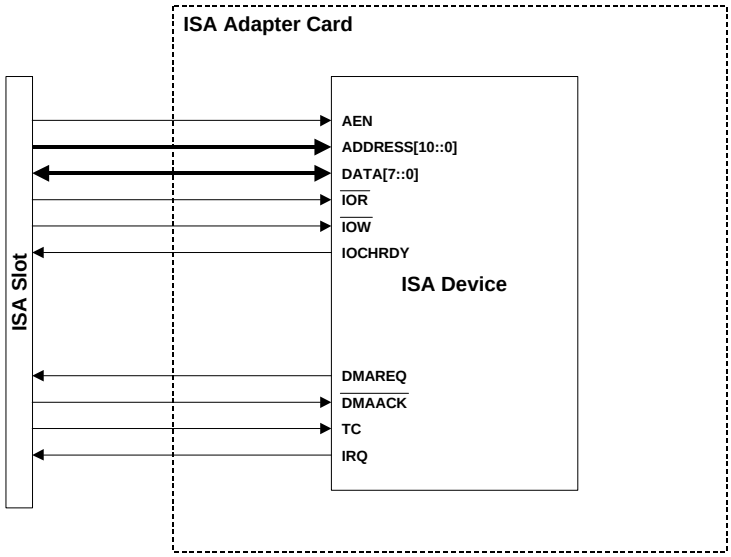
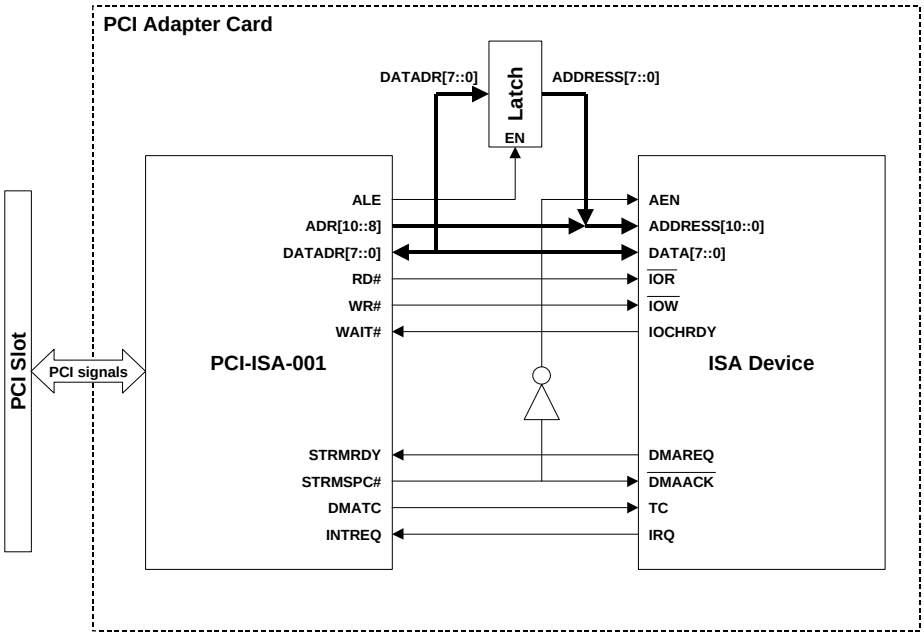


Figure 7-8: Transformed PCI Adapter Card



8.0 Non-Volatile RAM Operation

The PCI-ISA-001 expects to boot-load its personalization data from an external, standard low-cost off–the-shelf, 1K-bit serial E^2PROM that uses the MICROWIRE™ compatible protocol. Connection to the E^2PROM is accomplished using a standard 4-wire serial E^2PROM interface. The PCI-ISA-001 uses 16-bit E^2PROM data organization. Any data transfer is synchronized using the NVCLK. NVCLK meets most industry standard E^2PROM requirement of maximum clock frequency of 1 MHz. Data on NVRDDATA and NVWRTDATA are latched on the rising edge of NVCLK. NVCLK is valid when the E^2PROM is accessed by asserting NVCS, otherwise NVCLK value is undetermined because this pin is shared, and should be ignored by the E^2PROM.

PCI-ISA-001 initiates read access to the E^2PROM after a RST# by asserting NVCS. PCI-ISA-001 will not respond to any PCI accesses until the E^2PROM bits have been read and placed inside PCI-ISA-001. Once the boot-load read sequence is complete, NVCS will be de-asserted and PCI accesses will be allowed.

During normal operation after the boot-load, E^2PROM access can be gained by using the low-level control on the standard 4-wire serial E^2PROM interface. This low-level control is provided through bits 9-12 of the Mode Register. Prior to any operation, NVCS needs to be set high. All E^2PROM read/write operations follow a sequence of: a start bit, some op-code, address and data bits. During write operation the start bit, write op-code, write address, and the write data is transferred to the E^2PROM serially through NVWRTDATA, which is synchronized by

Table 8-1: NV-RAM image map

Bit-address location of MSB	Bit-address location of LSB	Field Size	Destination PCI-ISA-001 Register-field
0	15	16 bits	Non-Volatile RAM signature. Must be equal to A55A hex. Any other value will inhibit the boot-load sequence.
16	31	16 bits	Vendor ID Register [15::0].
32	47	16 bits	Device ID Register [15::0].
48	55	8 bits	Revision ID Register [7::0].
56	63	8 bits	Class Code #F0 Register, base class code [7::0].
64	71	8 bits	Class Code #F0 Register, sub-class code [7::0].
72	79	8 bits	Class Code #F0 Register, programming interface [7::0].
80	90	11 bits	Base Address 0 #F0 Register [10::0].
91	101	11 bits	Base Address Read-only Mask 0 #F0 Register [10::0].
102	110	9 bits	Add-on Base Address 0 #F0 Register [10::2].
111	121	11 bits	Base Address 1 #F0 Register [10::0].
122	132	11 bits	Base Address Read-only Mask 1 #F0 Register [10::0].
133	141	9 bits	Add-on Base Address 1 #F0 Register [10::2].
142	149	8 bits	Class Code #F1 Register, Base class code [7::0].
150	157	8 bits	Class Code #F1 Register, sub-class code [7::0].
158	165	8 bits	Class Code #F1 Register, programming interface [7::0].
166	176	11 bits	Base Address 0 #F1 Register [10::0].
177	187	11 bits	Base Address Read-only Mask 0 #F1 Register [10::0].
188	196	9 bits	Add-on Base Address 0 #F1 Register [10::2].
197	207	11 bits	Base Address 1 #F1 Register [10::0].
208	218	11 bits	Base Address Read-only Mask 1 #F1 Register [10::0].
219	227	9 bits	Add-on Base Address 1 #F1 Register [10::2].
228	235	8 bits	Class Code #F2 Register, Base class code [7::0].
236	243	8 bits	Class Code #F2 Register, sub-class code [7::0].
244	251	8 bits	Class Code #F2 Register, programming interface [7::0].
252	262	11 bits	Base Address 0 #F2 Register [10::0].
263	273	11 bits	Base Address Read-only Mask 0 #F2 Register [10::0].
274	282	9 bits	Add-on Base Address 0 #F2 Register [10::2].
283	285	3 bits	Interrupt Pin Register [2::0]

NVCLK. A read operation is performed by serially transferring the start bit, read op-code, and read address on NVWRTDATA. Following that the read data is returned on NVRDDATA serially to PCI-ISA-001. For exact details on the E^2PROM operation and the timing requirements consult the vendors' datasheet. PCI-ISA-001 can use Microchip Technology's 93LC46B or any identical E^2PROM device that supports sequential read operation.

8.1 The Non-Volatile RAM image

The boot-loadable PCI-ISA-001 register-field images in the NV-RAM bit-address space are shown in Table 8-1. The bit-address locations used in the image are computed as,

bit-address = byte-address*8 + bit-number.

In this convention, the first bit that shifts out from E^2PROM during a byte read command is considered as bit 0 and the last bit is considered as bit 7. Notice that this convention could be different from E^2PROM vendor's datasheet and in which case an appropriate translation will be required. The total image size is 286 bits which occupies bit-address locations 0-285.

Users that do not have their own Vendor ID to be used in the NV-RAM image are encouraged to use the Vendor ID assigned to Interconnect Systems Solution (ISS) by PCI SIG. In such case, they are required to obtain the Device ID from ISS by sending the Device ID request form provided in the next page to ISS. Once the Device ID is obtained you may write down the Device ID at the space provided below for your convenience.

Device ID assignment from ISS

Application: _____

Vendor ID: 1549 hex (Interconnect Systems Solution)

Device ID: _____

Assigned on (date): _____

PCI Device ID Request Form

Please fill-in the information below,

Name
Title
Company
Address
City
State
Country
Postal Zip Code
Telephone
FAX
Email
Application (describe your board application)

Mail to:

Interconnect Systems Solution
22691 Lambert Street, Suite 503
Lake Forest, California 92630, USA.

Allow 2-3 weeks for the Device ID assignment. ISS assigned Device ID is to be used in conjunction with ISS vendor ID 1549 hex.

9.0 Package and Pinout

PCI-ISA-001 is currently available only in 100 pin TQFP package. The pin assignments are shown in Figure 9-1 and the physical dimensions are shown in Figure 9-2. The physical dimensions are give in millimeter unit. Contact ISS for availability of PCI-ISA-001 in other packages.

Figure 9-1: Pin assignments to 100 pin TQFP package

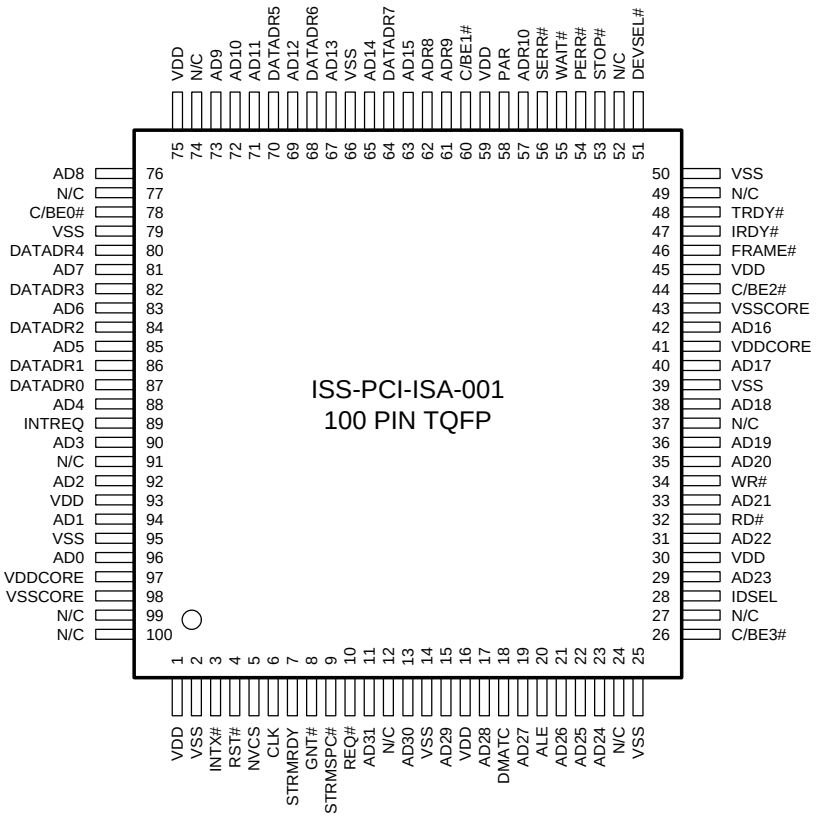
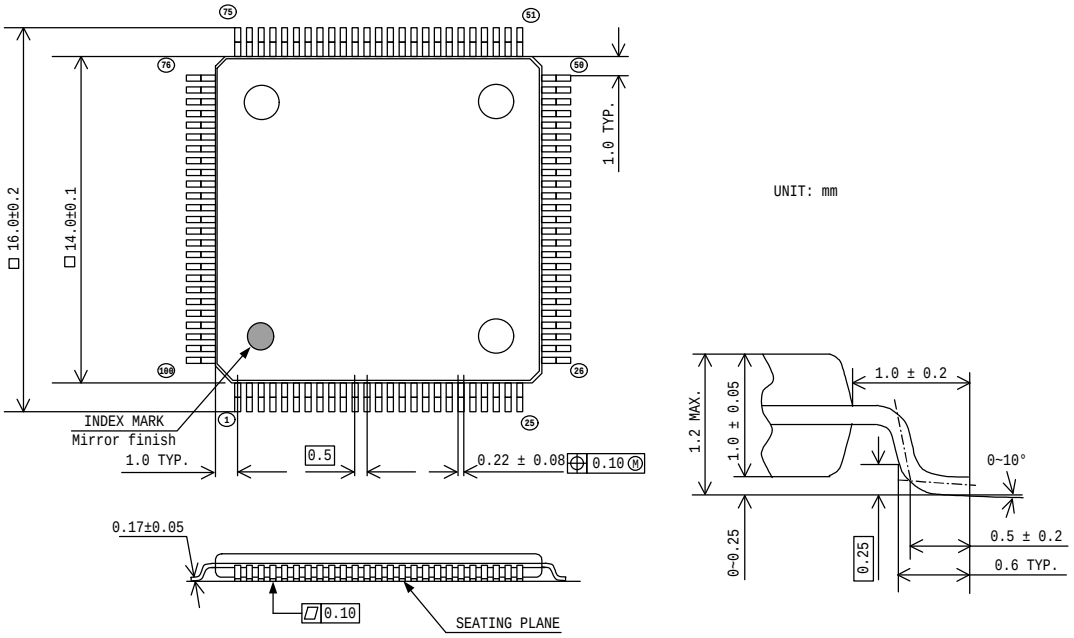


Figure 9-2: 100 pin TQFP Package dimensions



10.0 Electrical Specification

10.1 DC Characteristics

ABSOLUTE MAXIMUM RATINGS

Absolute Maximum Ratings are those values beyond which damage to the device may occur. Unless otherwise specified all voltages are referenced to VSS.

Supply Voltage (VDD, VDDCORE)	–0.5V to +7.0V
Input Voltage (VIN)	–0.5 to VDD+0.5V
Output Voltage (VOUT)	–0.5 to VDD+0.5V
Power Dissipation (PD)	1W
Storage Temperature (TSTG)	–65ºC to +165ºC
Lead Temperature (TL), Soldering, 10 seconds	+260ºC

RECOMMENDED OPERATING CONDITIONS

Supply Voltage (VDD, VDDCORE)	3.3V±5%
Operating Temperature (TA)	0ºC to 70ºC

Table 10-1: DC Characteristics Under Recommended Operating Conditions

Symbol	Description	Conditions	Min	Max	Units
ICC	VDD Average Supply Current	VDD = 3.3V+5%		150	mA
VIH	Input High Voltage		2.0	5.0	V
VIL	Input Low Voltage		–0.5	0.8	V
VOH	Output High Voltage	IOH = –4.0 mA	2.4	VDD	V
VOH	Output Low Voltage	IOH = 4.0 mA	0	0.4	V
VOHP	PCI Output High Voltage	IOH = –2.0 mA	2.4	VDD	V
VOHP	PCI Output Low Voltage	IOH = 6.0 mA	0	0.55	V
IIL	Input Leakage Current	VSS ≤ VIN ≤ VDD	–10	+10	µA
IOZ	Tri-State Output Leakage Current	VSS ≤ VIN ≤ VDD	–10	+10	µA

10.2 AC Characteristics

Figure 10-1: PCI Bus Timing Diagram

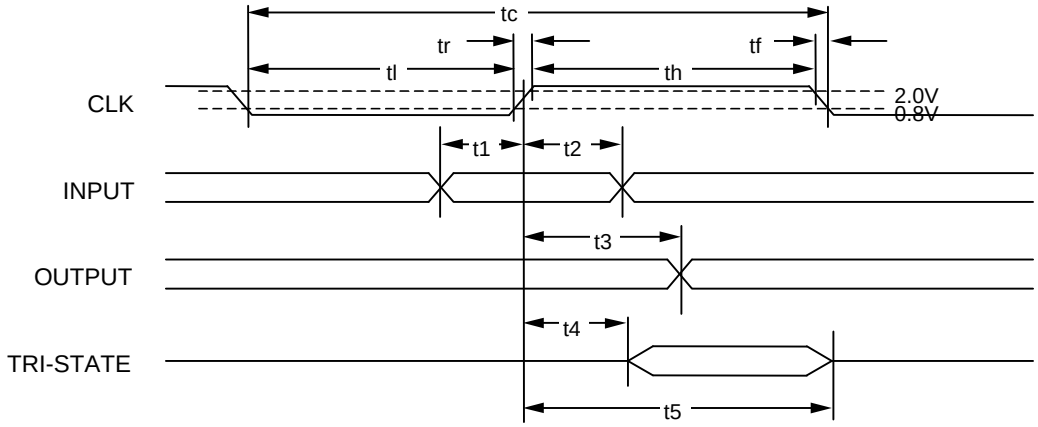


Table 9-10: PCI Bus Timing Values (expressed in ns, 50pf load on outputs)

Symbol	Description	Min	Max
tc	Clock cycle time	30	
th	Clock high time	12	
tl	Clock low time	12	
tr	Clock rise time		3
tf	Clock fall time		3

Symbol	Description	Min	Max
t1	Input setup time (bussed) Input setup time (GNT#, IDSEL)	7 10	
t2	Input hold time	0	
t3	Output valid time (bussed) Output valid time (REQ#)	2 2	11 12
t4	Float to active time	2	
t5	Active to float time		28

Figure 10-2: Add-on Bus Timing Diagrams

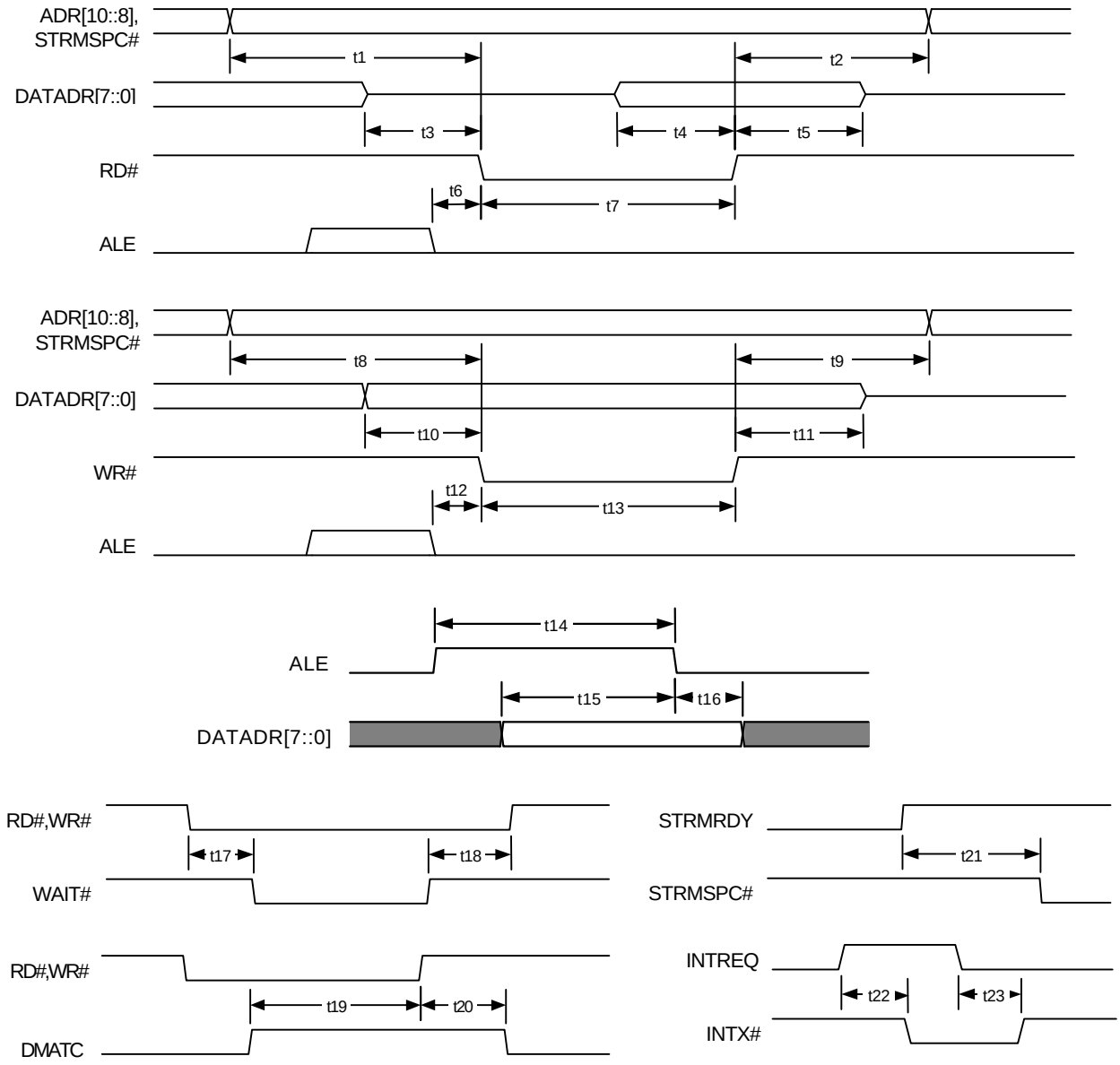


Table 10-3: Add-on Bus Timing Values (expressed in ns, 20pf load on outputs)

Symbol	Description	Min	Max
t1	Address to read active delay	26	
t2	Read address hold time	30	
t3	Data float to read active delay	0	
t4	Read data setup time	26	
t5	Read data hold time	0	
t6	ALE inactive to read active delay	0	
t7	Read active width	120	
t8	Address to write active delay	26	
t9	Write address hold time	30	
t10	Data valid to write active delay	0	
t11	Write data hold time	30	
t12	ALE inactive to write active delay	0	

Symbol	Description	Min	Max
t13	Write active width	90	
t14	ALE width	30	
t15	Address to ALE setup time	20	
t16	Address to ALE hold time	2	
t17	Wait active delay		65
t18	Wait inactive delay		35
t19	DMATC setup time	60	
t20	DMATC hold time	0	
t21	STRMRDY to STRMSPC# delay	70	
t22	INTREQ active to INTX# delay		55
t23	INTREQ inactive to INTX# delay		55

11.0 Sales Information

ISS worldwide sales office is located at,

Interconnect Systems Solution
22691 Lambert Street, Sutie 503
Lake Forest, CA 92691-1614, USA
Tel: 949-587-0628
email: sales@iss-us.com

ISS-PCI-001 is distributed directly by ISS sales office at the time of this document release. Please contact ISS sales office for a current list of distributors.

Notes