PCI IP Core Specification

Authors: Miha Dolenc
mihad@opencores.org
Tadej Markovic
tadej@opencores.org

Rev. 0.3
May 22, 2001

Draft

Revision History

Rev.	Date	Authors	Description
0.0	5/1/01	Miha Dolenc Tadej Markovic	First Draft
0.1	5/8/01	Miha Dolenc Tadej Markovic	Waveforms added for WISHBONE slave
0.2	5/15/01	Miha Dolenc Tadej Markovic	Detailed description of FIFO added, Operation of PCI Target unit added, Waveforms added for PCI target
0.3	5/22/01	Miha Dolenc Tadej Markovic	FIFO structure changed

Table Of Contents

Table Of Figures

Table Of Tables

Table Of Examples

1 Introduction

1.1 What is PCI Bridge?

PCI bridges are used in applications and devices that want to use resources provided on PCI local bus. Most of applications use different internal buses and they need an interface from internal to PCI local bus. PCI bridges are intended to provide such interface.

1.2 PCI IP Core Introduction

The PCI IP core (PCI bridge) provides interface between WISHBONE SoC bus and PCI local bus. Actually the core consists of two independent units: One handles transactions originating on PCI bus, the other handles transactions originating on WISHBONE bus. They can be implemented together or separately.
The core is designed to offer as much flexibility as possible for all kinds of applications.

1.3 PCI IP Core Features

The following lists the main features of PCI IP core:

- 32-bit PCI interface
- Fully PCI 2.2 compliant (with 66 MHz PCI specification)
- Separated Initiator and Target functional blocks (which can also operate together)
- Supported Initiator commands and functions:
  - Memory Read, Memory Write
  - Memory Read Multiple (MRM)
  - Memory Read Line (MRL)
  - Memory Write and Invalidate (MWI)
  - I/O Read, I/O Write
  - Configuration Read, Configuration Write
  - Bus Parking
  - Special Cycles
  - Interrupt Acknowledge
  - Host Bridging
- Supported Target commands and functions:
  - Type 0 Configuration Space Header
    (Type 0 is used to configure agents on the same bus segment)
    (Type 1 is used to configure across PCI-ti-PCI bridges)
  - 3 or more Base Address Registers (I/O, MEM and MEM for device configuration registers - all with adjustable block sizes)

  - Parity Generation (PAR), Parity Error Detection (PERR# and SERR#)
  - Memory Read, Memory Write
  - Memory Read Multiple (MRM)
  - Memory Read Line (MRL)
  - Memory Write and Invalidate (MWI)
  - I/O Read, I/O Write
  - Configuration Read, Configuration Write
  - Target Abort, Target Retry, Target Disconnect
  - Full Command/Status Registers
- WISHBONE SoC Interconnection Rev. B compliant interface on processor side (Master with Target PCI and Slave with Initiator PCI interface)
- Configurable on-chip FIFOs

2

Architecture

2.1 Overview

PCI bridge consists of two units: PCI target unit and WISHBONE slave unit. Each unit consists of its own set of functions to support bridging operations from WISHBONE to PCI and from PCI to WISHBONE. WISHBONE slave unit acts as a slave on WISHBONE side of the bridge and initiates transactions as a master on PCI bus. PCI target unit acts as a target on PCI side of the bridge and as master on WISHBONE side. Both units' operations are independent of each other. PCI target unit implements target interface on PCI bus and master interface on WISHBONE bus, while WISHBONE slave unit implements slave interface on WISHBONE bus and master interface on PCI bus.
PCI interface is PCI specification 2.2 compliant, while WISHBONE is SoC Interconnection specification Rev. B compliant. The WISHBONE implementation implements a 32-bit bus width and does not support other bus widths.

Following figure shows the overview of the PCI bridge Core architecture.

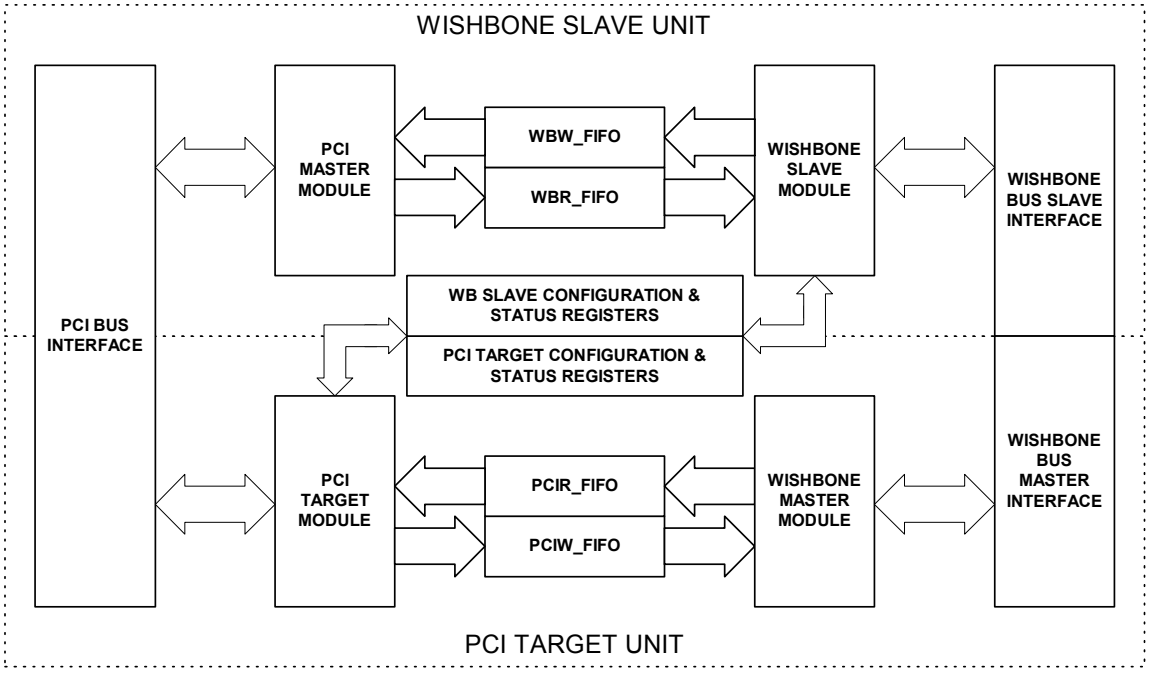

Figure 2-1: PCI bridge Core Architecture

2.2 WISHBONE Slave Unit

PCI bus is accessible from WISHBONE bus through WISHBONE slave unit. PCI address space is accessed through two to five images. Each image is selected with address provided on WISHBONE slave's module interface compared with base address masked with value of address mask stored in WB slave configuration registers. Each image can be mapped to memory or I/O space on PCI bus.

WB slave unit supports posted writes and pre-fetched reads in memory space, accesses to I/O space are handled as single read or write transactions.

Posted writes are stored in WISHBONE Write FIFO (WBW_FIFO), pre-fetched reads are stored in WISHBONE Read FIFO (WBR_FIFO).

WISHBONE slave unit connects to WISHBONE masters acting as a slave. This section describes the architecture of WISHBONE slave unit and is divided into subsections.

2.2.1 WISHBONE Slave Unit Architecture

This section describes architecture of WISHBONE slave unit. Detailed description is provided in following sections.

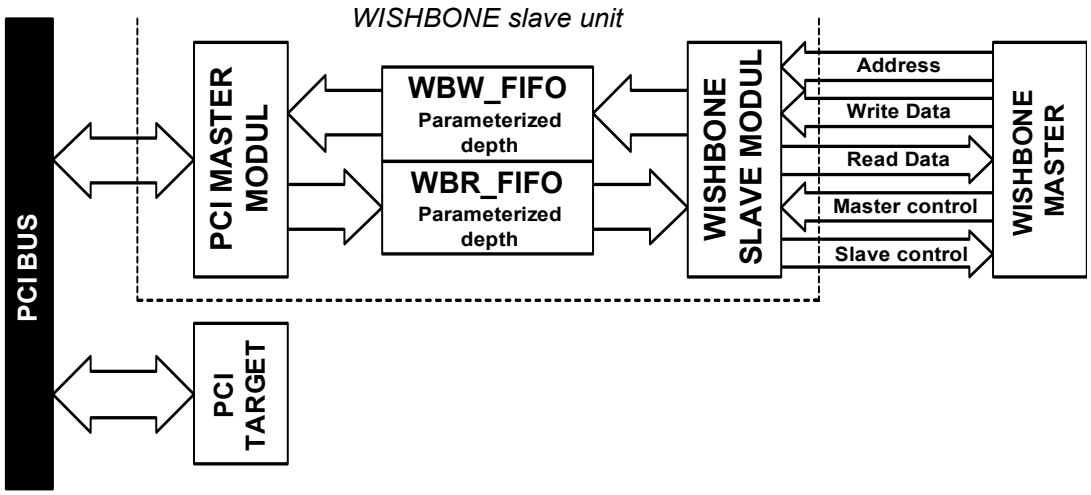

Figure 2-2: WISHBONE slave unit architecture

WISHBONE slave unit consists of a few functional parts allowing WISHBONE master to perform a Read/Write accesses to PCI bus.

2.2.1.1 WISHBONE Slave Module

WISHBONE slave module is 32-bit WISHBONE slave interface as defined in WISHBONE specification Rev. 1B. It includes three to six image units for address translation from WISHBONE bus and therefore it handles Read/Write cycles to images of PCI address space and configuration space accesses.

2.2.1.2 WBW_FIFO

WISHBONE slave module uses WBW_FIFO (WISHBONE WRITE FIFO) for posting memory and I/O writes performed by WISHBONE master. Parameterized depth of WBW_FIFO leaves system designer an option of defining it, regarding needs of particular application for posting more or less writes.
Writing to WBW_FIFO is adapted to WISHBONE bus speed, while writing to PCI bus from WBW_FIFO is adapted to PCI bus speed.

2.2.1.3 WBR_FIFO

WISHBONE slave module uses WBR_FIFO (WISHBONE READ FIFO) for storing data read from PCI targets.
Reading from PCI bus to WBR_FIFO is adapted to PCI bus speed, while reading from WBR_FIFO is adapted to WISHBONE bus speed.

2.2.1.4 PCI MASTER Module

PCI MASTER module is 32-bit/66MHz, PCI local bus specification Rev. 2.2 compliant initiator interface. The core requests PCI bus through its PCI MASTER module. PCI interface of the core is described in detail in Chapter 5-XY: PCI I/O interface description.

2.3 PCI Target Unit

WISHBONE bus is accessible to PCI agents through PCI target unit of the bridge. PCI target unit provides two to five images of WISHBONE side memory space. Each of these images is selected with address provided during address cycle on the PCI bus compared with base address masked with mask value stored in PCI configuration registers. Each image can be mapped into I/O or memory space.
Writes through PCI target unit are handled as posted writes while reads are pre-fetched. Posted writes are stored in PCIW_FIFO and pre-fetched reads are stored in PCIR_FIFO.

2.3.1 PCI Target Unit Architecture

This section describes architecture of PCI target unit. Detailed description is provided in following sections.

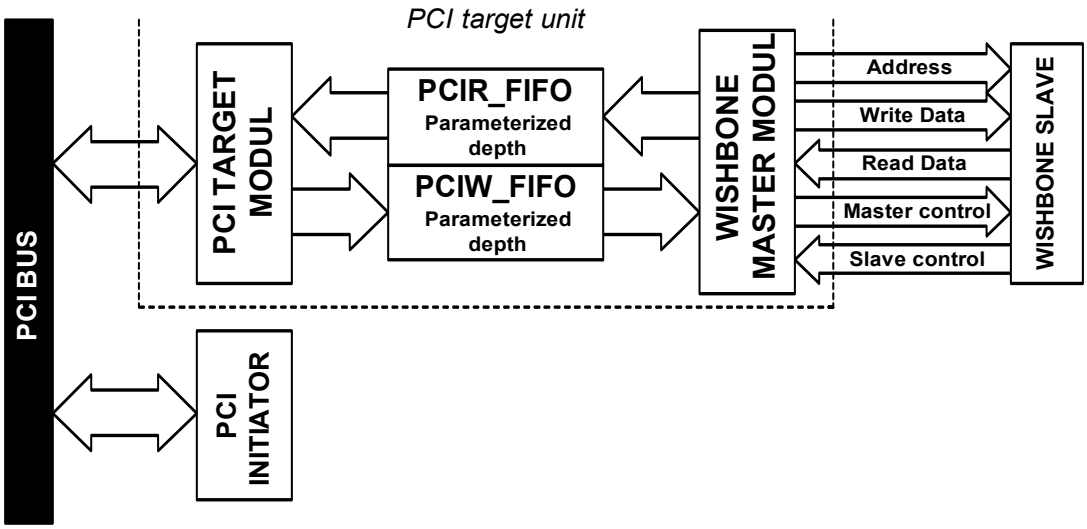

Figure 2-3: PCI target unit architecture overview

PCI target unit consists of a few functional parts allowing PCI initiators to perform a Read/Write accesses to WISHBONE bus.

2.3.1.1 PCI Target Module

PCI TARGET module is 32-bit/66MHz, PCI local bus specification Rev. 2.2 compliant target interface. It includes three to six image units for address translation from PCI bus and therefore it handles Read/Write cycles to images of WISHBONE address space and configuration space accesses.

2.3.1.2 PCIW_FIFO

PCI TARGET module uses PCIW_FIFO (PCI WRITE FIFO) for posting memory and I/O writes performed by PCI initiator. Parameterized depth of PCIW_FIFO leaves system

designer an option of defining it, regarding needs of particular application for posting more or less writes.
Writing to PCIW_FIFO is adapted to PCI bus speed, while writing to WISHBONE bus from PCIW_FIFO is adapted to WISHBONE bus speed.

2.3.1.3 PCIR_FIFO

WISHBONE master module uses PCIR_FIFO (PCI READ FIFO) for storing data read from WISHBONE slaves.
Reading from WISHBONE bus to PCIR_FIFO is adapted to WISHBONE bus speed, while reading from PCIR_FIFO is adapted to PCI bus speed.

2.3.1.4 WISHBONE Master Module

WISHBONE master module is 32-bit WISHBONE master interface as defined in WISHBONE specification Rev. 1B. The core requests WISHBONE bus through its WISHBONE MASTER module. WISHBONE interface of the core is described in detail in Chapter 5-XY: WISHBONE I/O interface description.

2.4 Clocks

The PCI core has two clock domains. One is from PCI bus and the other from WISHBONE bus. The adjustment of different bus clocks is made by FIFO with its interconnection logic. And there is no mater, which bus operates on higher frequency, what leads to the fact, that there is no difference between all four FIFOs.

2.5 FIFO

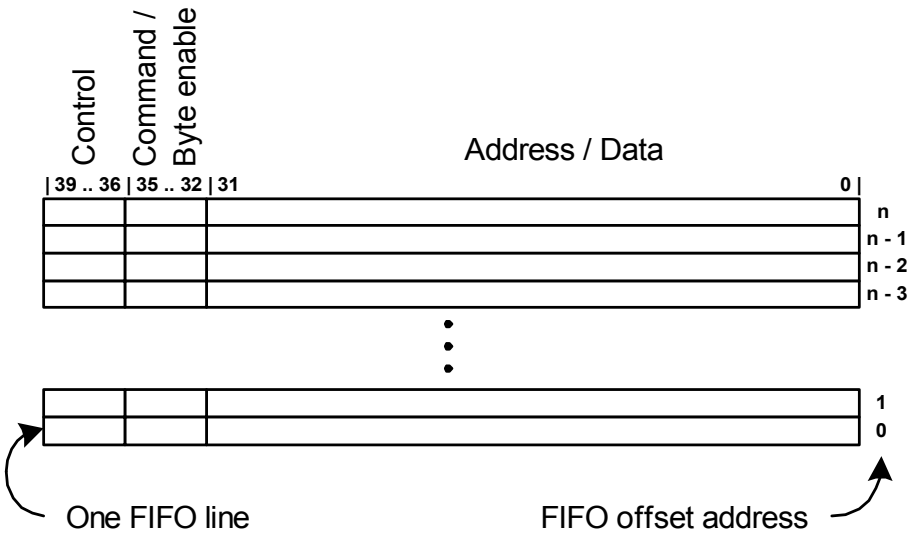

Figure 2-4: Detailed description of FIFO register lines

FIFO is structured from more than one FIFO lines. The number of FIFO lines is the number of FIFO's depth, and it can be configurable (how the number of FIFO depth is defined will be discussed in detail in design document and implementation notes). The structure of one FIFO line is well described in Figure 2-4. It consists of 4 control bits (how they are used, will be described in detail in design document, e.g. one bit is used to sign last data of the burst transfer etc.), 4 command or byte enable bits (coding will be described in detail in design document), 32 address or data bits.
FIFOs are implemented as circular data buffers between WISHBONE and PCI interfaces (Figure 2-5). As mentioned above, the FIFO with its interconnection logic adapts the different bus speeds.
There are data written to the FIFO on the input side with the input bus clock, which is also connected to the FIFO registers. The Input Pointer (input counter) stores the value of the input offset address of the first free FIFO line. Its clock frequency is the same as the frequency of the input bus side.

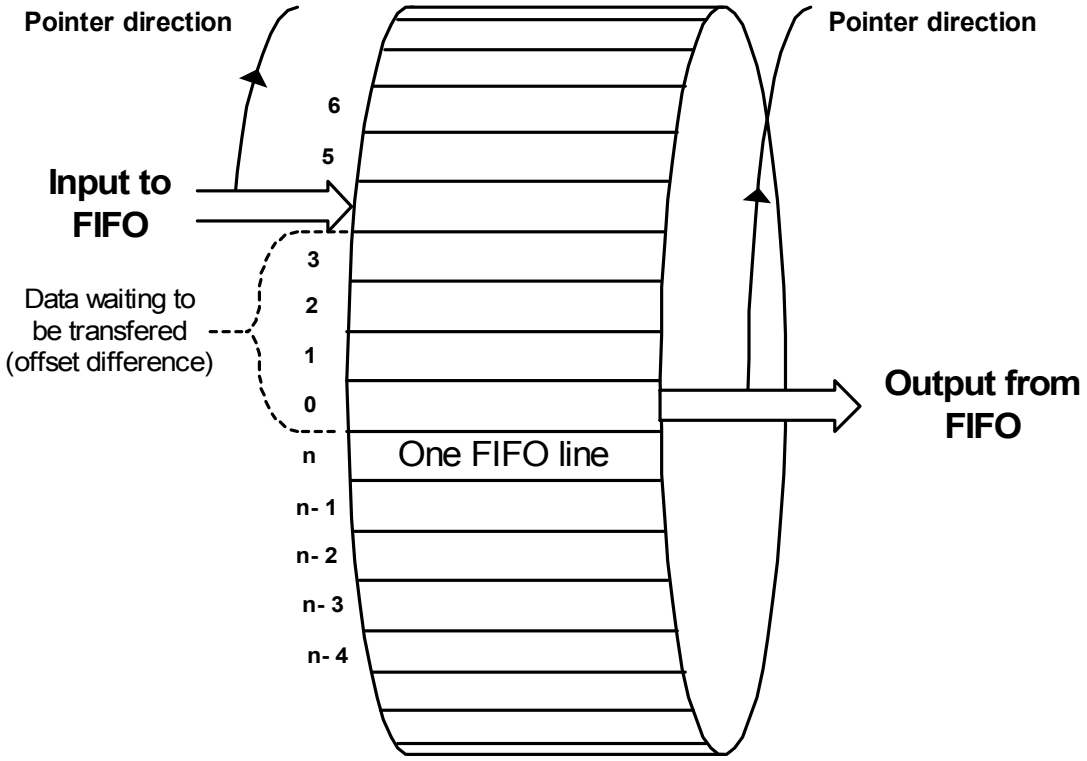

Figure 2-5: FIFO architecture

The Output Pointer (output counter) stores the value of the output offset address of the first FIFO line from which data is to be read. Its clock frequency is the same as the frequency of the output bus side, with which the data are read.
The comparator between both Pointers (counters) validates if there is any data waiting in the FIFO to be read (exact counter/comparator operation will be described in detail in design document). There is also a comparator between the counter, with a value of an Input Pointer incremented for one, and the Output Pointer. When equals, the FIFO is full.

2.6 Address Translation Logic

As mentioned, WISHBONE slave unit and PCI target unit incorporates several address space images. For each address space image there must be one address translation logic (Figure 2-6). Each address translation logic includes its own set of 32-bit registers:

- BASE ADDRESS register [31:0];
- ADDRESS MASK register [31:0];
- TRANSLATION ADDRESS register [31:0];
- IMAGE CONTROL register [31:0].

2.6.1 Description of Address Translation Logic

For description of address translation logic you must also see the Figure 2-6. All AND blocks and OR block are BIT oriented operators which stands for logic operations between bits with same weight (e.g. logic function between bit[n-2] of bus A and bit[n-2] of bus B).

The base address is written into the Base Address register. How many Most Significant bits are masked and replaced with translation address bits, depends on Address Mask register, which also defines the size of the image. There is one rule, how to set the Address Mask register. Address bits that can be masked, must start with MSbit (bit[31]) and follow till twelfth bit (bit[11]). All the allowed masked bits defines the smallest size of 4kBytes, that can be assigned. Between mask bits there must be no zeros, otherwise this image will have two base addresses with one base address register, what do not complies with PCI specification. Because of that, the output of Address Mask register is connected to Mask Correction unit.

To find out if address falls into the correct address range, the masked bits of an input address and of the base address must be compared (the number of masked bits defines the unchanging address of current address range, and with that the size of this image). Masking of the input and the base addresses is made with AND logic function between each one of the address and address mask.

To get the correct output address, there must be combined (with OR logic function) the unmasked bits of the input address and masked bits of the translation address. The unmasked bits of the input address can be achieved with masking (with AND logic function) the input address with negated mask bits.

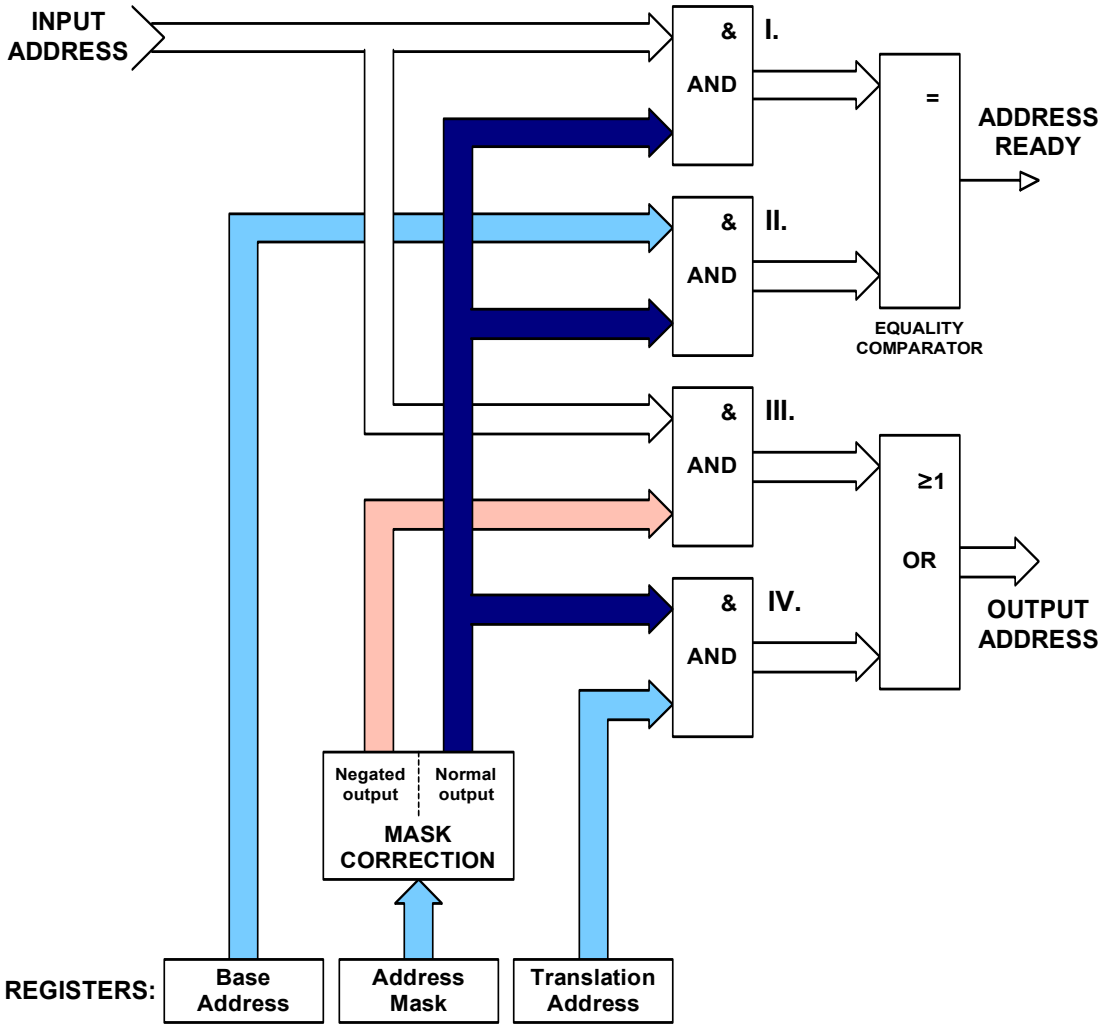

Figure 2-6: Block Diagram of Address Translation Logic

3

Operation

3.1 Configuration Space

Configuration space is accessible from PCI or WISHBONE bus depending on core implementation. If the core is implemented to be a host bus bridge, then WISHBONE slave unit has exclusive access to this space and PCI target unit has read only access. If the core is implemented as a guest (expansion bus bridge), then configuration space is accessible through PCI target unit, while read only access is provided for WISHBONE slave unit.

Configuration space has fixed block size of 4KB and is divided in two parts – one intended for configuration, control and status registers of WB slave unit and the other for PCI target unit registers. PCI configuration cycles can be generated through accessing specific registers in configuration space from WISHBONE bus, if the core is implemented as host bus bridge, otherwise configuration cycles must be performed by some other agent on PCI bus.

Configuration space is accessible only with single reads and single writes (e.g. cannot be accessed with bursts from PCI side).

All registers in core's configuration space are 32 bit with 8 bit granularity. All accesses must be DWORD aligned (e.g. two LS bits of address must be 00). PCI standard defines special encoding for these two bits for PCI bus accesses, while WISHBONE slave module will signal a bus error if any of these two bits is non-zero. Individual bytes can be accessed with appropriate value on BE# signals for PCI bus and SEL_O signals for WISHBONE bus access.

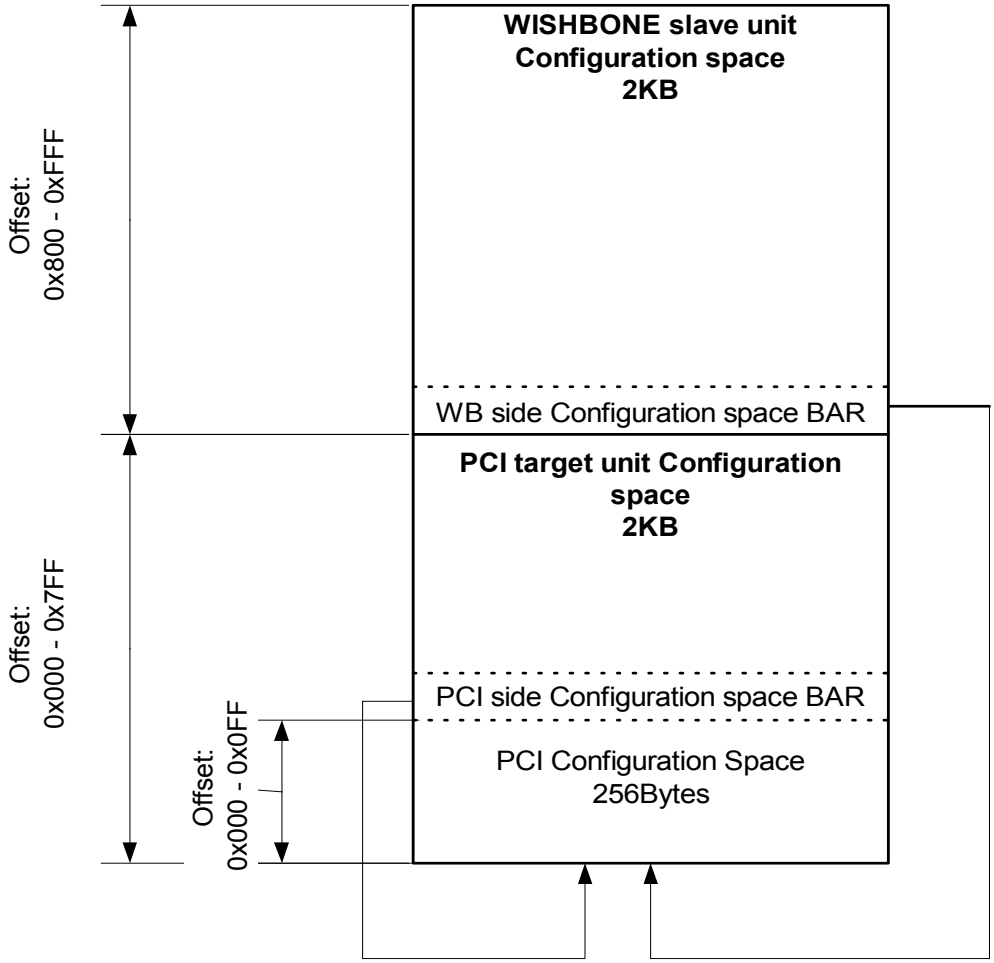

Figure 3-1: PCI bridge Configuration Space

3.1.1 Configuration Space Access for Host Bus Bridges

Host bus bridge implementation of the core provides Read/Write Access to configuration space for WISHBONE slave unit and Read Only Access for PCI target unit, thus WB Master takes full responsibility of configuring core's registers and any other PCI devices residing on PCI bus. WISHBONE side configuration space Base address is predefined and cannot be changed once the core is implemented (How and where base address is defined will be described in detail in design documentation).

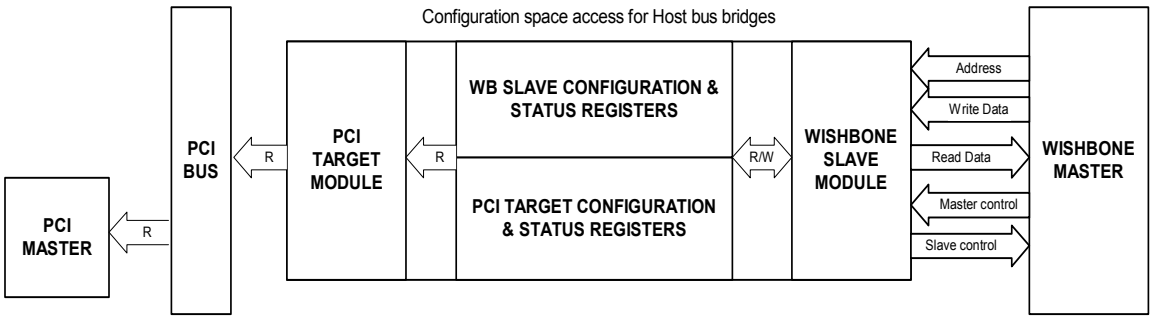

Figure 3-2: Configuration space access for Host bus bridges

Access to configuration space by WISHBONE MASTER can either be single read, single write or Read Modify Write (RMW). All of the configuration space is not implemented. If WISHBONE master attempts a write to non-implemented space, the cycle will be acknowledged by WISHBONE slave module, while reads to non-implemented space return all 0s.

PCI side configuration space Base Address can either be predefined or set by WISHBONE MASTER (How and where predefined Base Address option and address are set will be discussed in detail in design documentation). If PCI side configuration space Base Address is predefined, then this data is Read Only for WISHBONE master also and cannot be modified. If address is not predefined, WISHBONE master must perform a write to PCI side configuration space base address register to enable read only access to PCI agents. Read only access to configuration space from PCI bus is provided through PCI Target Module. The bus commands supported for access to configuration space from PCI bus are Memory Read and Memory Write, all other bus commands will be ignored by PCI Target Module. Memory write will have no effect on configuration registers. PCI Target module will signal Target Disconnect With Data to the initiator during the first Data Phase. Reads to non-implemented regions of configuration space will return all 0s, while writes will have no effect.

3.1.2 Configuration Space Access for Guest Bridges

Guest bridge (more commonly referred to as expansion bus bridge) implementation of the core provides Read/Write Access to configuration space for PCI target unit and Read Only Access for WISHBONE slave unit, thus other PCI agents take full responsibility of configuring core's registers and any other PCI devices residing on PCI bus. PCI side configuration space Base address is set by an agent on PCI bus (most commonly host bus bridge) by performing Type 0 configuration cycle and writing base address to PCI configuration space as stated in PCI local bus specification Rev. 2.2. PCI side configuration space base address register holds the same value as the first Base Address Register in PCI configuration space at offset 0x10. This enables device independent software to map the bridge's configuration space anywhere in memory address space. After base address has been set by Type 0 Configuration Cycle and bridge is in normal mode of operation, PCI agent can re-map configuration space anywhere in memory space by writing to PCI side configuration space base address register.

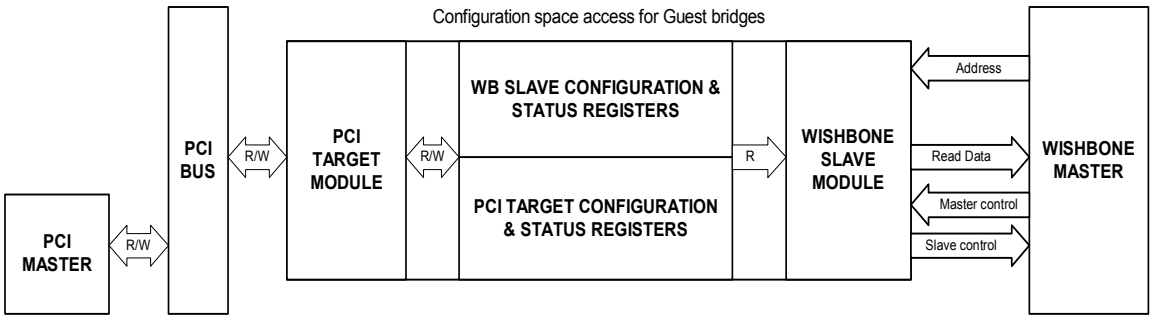

Figure 3-3: Configuration space access for Guest bridges

Access to configuration space by PCI initiator can be Memory Read or Memory Write. In case PCI initiator attempts configuration space access with some other bus command, PCI Target Module will not respond. All of the configuration space is not implemented. If PCI initiator attempts access (read or write) to non-implemented space, the initial data phase will be terminated gracefully with signaling Target Disconnect with data. Writes have no effect on configuration space while reads will return all 0s.
Read only access to configuration space from WISHBONE bus is provided through WISHBONE slave module. WISHBONE side configuration space base address is predefined and cannot be changed (How and where base address is defined will be described in detail in design documentation). WISHBONE slave module accepts Read or Write transfers to configuration space. Writes to configuration space have no effect on configuration space registers. If WISHBONE master attempts an access to non-implemented region, writes will be acknowledged with no effect on configuration space, while reads return all 0s.

3.1.3 Configuration Cycles

Configuration cycles are another way of accessing configuration space of the core. Only lower 256 bytes of configuration space are available for Read/Write Access with Type 0 Configuration cycles for Guest (expansion bus) implementation of the core. Host bus bridge implementation provides Configuration Read operation only (this feature probably will never be used, because host bus bridge normally generates configuration commands and PCI local bus specification doesn't require host bus bridge to respond to configuration cycles). Configuration writes will be accepted and acknowledged by host bus bridge, but will have no effect on configuration registers.
Addressing in configuration cycles is somewhat different from normal reads and writes on PCI bus (See PCI local bus specification Rev 2.2, chapter 3.1.1 – Command Definition).
There is only Type 00h predefined header portion implemented in lower 256 bytes of Configuration Space (also called in this document PCI Configuration Space). For Type 00h predefined header organization see PCI local bus specification Rev 2.2, chapter 6.1.

3.1.4 Generating Configuration Cycles

Host bus bridge implementation of the core provides a mechanism for generating configuration cycles on PCI bus. Configuration cycle generation is provided through accessing of CNF_ADDR register and CNF_DATA register.

- **Step 1:** WISHBONE master must write appropriate data to CNF_ADDR register, which holds information about register offset, function, device and bus number. TYPE bit in this register defines a type of configuration cycle that will be generated on PCI bus (0 = Type 0, 1 = Type 1). Offset field in CNF_ADDR register identifies a register offset to or from which WISHBONE master wishes to write or read. FUNCTION field is set to function number of multifunction device being a target of configuration cycle. DEVICE field is set to device number on PCI bus. This field identifies the address line that will be driven high for generating IDSEL signal for Type 0 configuration cycle. BUS field is set to bus number on which targeted device resides.
- **Step 2:** To actually begin configuration cycle on PCI bus, WISHBONE master must access CNF_DATA register. Accesses to CNF_DATA are treated as single delayed transactions. WISHBONE master's access to this register will be retried. If access to this register is a read, PCI master module arbitrates for PCI bus, performs configuration read command with byte enables provided by WISHBONE master (signals SEL_O(3..0)) and provides data on WISHBONE interface, when WISHBONE master retries the transaction. In case of write access to this register, PCI master module arbitrates for PCI bus, performs a write with provided byte enables (signals SEL_O(3..0) and acknowledges the transaction when WISHBONE master retries the transaction.

Driving of PCI bus AD lines during configuration cycle address phase depends on TYPE of configuration cycle. If WISHBONE master sets TYPE bit of CNF_ADDR to 1 (indicating Type 1 configuration cycle), the value of lines on PCI bus are driven with contents of CNF_ADDR register (AD[31..0] <= CNF_ADDR[31..0]) during address phase. If TYPE bit indicates TYPE 0 configuration cycle then AD[31..11] lines on PCI bus are driven according to following table (driving depends on DEVICE field in CNF_ADDR register):

DEVICE field value	Value on AD[31..11] lines during address phase of Configuration cycle
0000 0	0000 0000 0000 0000 0000 1
0000 1	0000 0000 0000 0000 0001 0
0001 0	0000 0000 0000 0000 0010 0
0001 1	0000 0000 0000 0000 0100 0
0010 0	0000 0000 0000 0000 1000 0
0010 1	0000 0000 0000 0001 0000 0
0011 0	0000 0000 0000 0010 0000 0
0011 1	0000 0000 0000 0100 0000 0
0100 0	0000 0000 0000 1000 0000 0
0100 1	0000 0000 0001 0000 0000 0

DEVICE field value	Value on AD[31..11] lines during address phase of Configuration cycle
0101 0	0000 0000 0010 0000 0000 0
0101 1	0000 0000 0100 0000 0000 0
0110 0	0000 0000 1000 0000 0000 0
0110 1	0000 0001 0000 0000 0000 0
0111 0	0000 0010 0000 0000 0000 0
0111 1	0000 0100 0000 0000 0000 0
1000 0	0000 1000 0000 0000 0000 0
1000 1	0001 0000 0000 0000 0000 0
1001 0	0010 0000 0000 0000 0000 0
1001 1	0100 0000 0000 0000 0000 0
1010 0	1000 0000 0000 0000 0000 0
1010 1	0000 0000 0000 0000 0000 0
1011 0	0000 0000 0000 0000 0000 0
1011 1	0000 0000 0000 0000 0000 0
1100 0	0000 0000 0000 0000 0000 0
1100 1	0000 0000 0000 0000 0000 0
1101 0	0000 0000 0000 0000 0000 0
1101 1	0000 0000 0000 0000 0000 0
1110 0	0000 0000 0000 0000 0000 0
1110 1	0000 0000 0000 0000 0000 0
1111 0	0000 0000 0000 0000 0000 0
1111 1	0000 0000 0000 0000 0000 0

Table 3-1: Value on AD[31:11] PCI bus lines during address phase of configuration cycle Type0

Specified driving of PCI bus lines AD[31..11] provides a mechanism for tying IDSEL signals of target devices directly to AD lines. This way device with number 0 would be connected with its IDSEL signal to AD[11], device with number 1 to AD[12] up until device with number 20 connected to AD[31]. A total of 21 targets can be accessed with configuration cycles through PCI bridge. Combinations of DEVICE field values of CNF_ADDR register 10101 through 11111 are valid and will terminate with Master Abort on PCI bus since none of the targets can respond to the cycle without its IDSEL signal being asserted. Configuration write data will discarded while reads will return all 1s on WISHBONE bus and transaction will be acknowledged as specified in PCI specification Rev. 2.2.
Other AD lines on PCI bus are driven during address phase of Type0 configuration cycle as described in PCI specification Rev. 2.2 with data stored in CNF_ADDR register.

3.1.5 Generating Interrupt Acknowledge Cycles

Special mechanism provides generating of Interrupt Acknowledge Cycles on PCI bus. WISHBONE master must perform a read to INT_ACK register. This read is treated as single delayed transaction, which will be retried until PCI Master module arbitrates for

PCI bus and fetches the data requested. Address and byte enables on PCI bus are exact copies of ADR_O(31..0) and SEL(3..0). Address has no meaning during interrupt acknowledge cycle while byte enables indicate the size of interrupt vector returned.
Reads of this register from PCI bus have no effect and return all 0s. Writes from WISHBONE or PCI side are accepted and have no effect.

3.2 WISHBONE Slave Unit

WISHBONE slave unit connects to WISHBONE masters acting as a slave. This section describes basic functionality of the WISHBONE slave unit and is divided into subsections, each of them describing what WISHBONE master needs to do to initiate WISHBONE to PCI transactions.

3.2.1 WISHBONE Slave Unit Functionality

This section describes functional overview of WISHBONE slave unit. Detailed description is provided in following sections.

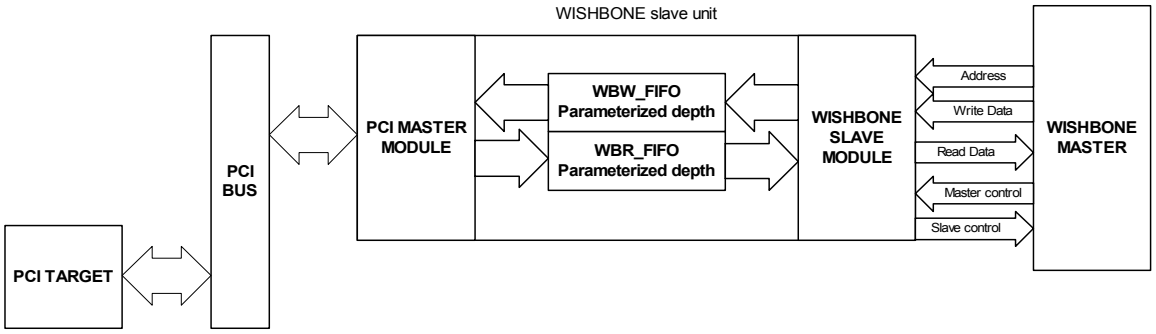

Figure 3-4: WISHBONE slave unit architecture overview

WISHBONE slave unit consists of a few functional parts allowing WISHBONE master to perform a Read/Write accesses to PCI bus.

3.2.1.1 WISHBONE Slave Module

WISHBONE slave module is 32 bit WISHBONE slave interface as defined in WISHBONE specification Rev. 1B. It handles Read/Write cycles to images of PCI address space and configuration space accesses.

3.2.1.2 WBW_FIFO

WISHBONE slave module uses WBW_FIFO (WISHBONE WRITE FIFO) for posting memory and I/O writes performed by WISHBONE master. WBW_FIFO also performs a different bus clock adaptation.

3.2.1.3 WBR_FIFO

WISHBONE slave module uses WBR_FIFO for storing data read from PCI targets. WBR_FIFO also performs a different bus clock adaptation.

3.2.1.4 PCI MASTER Module

PCI MASTER module is 32 bit/66MHz, PCI local bus specification Rev. 2.2 compliant initiator interface. The core requests PCI bus through its PCI MASTER module and performs bus operations as describes in following subsections. PCI interface of the core is described in detail in Chapter XY: PCI I/O interface description.

3.2.2 WISHBONE Slave Unit's Addressing and Images

As mentioned before, WISHBONE slave unit incorporates 2 to 5 configurable WISHBONE address space images (how the number of images is defined will be discussed in detail in design document and implementation notes) and one 4KB image used for configuration space accesses from WISHBONE bus with fixed base address. This fixed base address points to the starting address of the 4KB configuration space, but configuration space for WISHBONE slave unit starts at 0x800 offset address (only 2KB of configuration space are available for WISHBONE slave unit, see Figure 3-1). Base address for WISHBONE configuration space points at 0x0 offset address of whole configuration space and is different from the first BASE ADDRESS register in PCI header, which is also used for the same configuration space. The difference between this two base addresses is necessary, since WISHBONE bus has no configuration cycles, but configuration space has to be seen.
Each image's behavior is controlled by its WISHBONE BASE ADDRESS (W_BA1 – W_BA5), WISHBONE TRANSLATION ADDRESS (W_TA1 – W_TA5), WISHBONE IMAGE CONTROL (W_IMG_CTRL1 – W_IMG_CTRL5) and WISHBONE ADDRESS MASK (W_AM1 – W_AM5) registers. Statuses, errors and interrupts for each image are recorded in image's status registers described later in this document. WISHBONE slave module claims the cycle initiated by master on WISHBONE bus if one of WISHBONE images is selected and enabled. Image is enabled, if IMG_EN bit of its W_AM register is set to 1. Image is selected when address provided during initial cycle on WISHBONE bus falls within memory range of that image. Range is determined with values of W_BA and W_AM registers. Each image can represent from 4KB to 2GB of PCI address space.

Each image can be mapped to Memory or I/O space, which is determined by Address Space Mapping (ASM) bit of image's P_BAx register. If ASM bit is 0, then image maps to Memory space, otherwise it is mapped to I/O space.

How to specify an 1MB image of PCI address space with an addresses range of 0x10100000 - 0x101FFFFF ? Software must write a value of 0x10100XX0 to image's base address register (LSB of this register set to 0 to indicate a memory space mapping). This way base address is set at 0x10100000. Twelve LS bits are marked as don't cares – minimum block size is 4KB. Then software writes a value of 0xFFF00XXX into W_AM register of

corresponding image. MS bit is IMG_EN bit and this way it is set to a value of 1 (it's also used for address mask – that is how we limit a maximum image size to 2GB). Each bit in W_AM register corresponds to one address line – if bit is 1, then this address line is used in address comparison, otherwise it is not. A value of 0xFFF00000 in W_AM register means that ADDR_O(31..20) signals will be compared with W_BA[31..20] value. If values match, the image is selected. In this case, ADDR_O(19..0) lines define an offset in 1MB address range.

Example 3-1: Address range of WISHBONE slave image

If address translation is enabled for selected image (AT_EN bit of W_IMG_CTRLx is 1), then address translation between WISHBONE and PCI address is performed. Address translation is done by replacing masked part of WISHBONE address with corresponding bits from W_AT register. This provides a really flexible address mapping.

Let's make an assumption that base address and address mask are set as described in previous example. We want a WISHBONE address range of 0x10100000 – 0x101FFFFF to be mapped elsewhere on PCI bus, let's say 0x01000000 – 0x010FFFFF. For this to be done, we need addresses coming from WISHBONE master to be translated. We set AT_EN bit of corresponding W_IMG_CTRL register to a value of 1 and corresponding W_AT register to a value of 0x01000XXX. W_AM register is already set, so address translation will replace ADDR_O(31..20) provided by WISHBONE master with 0x010 value set in W_AT register for accesses on PCI bus. This way we have PCI address range of 0x01000000 – 0x010FFFFF accessible on WISHBONE bus in a range of 0x10100000 – 0x101FFFFF.

Example 3-2: Address translation

3.2.3 WISHBONE to PCI Writes

Previous section described, how WISHBONE slave unit knows, if it is a slave of current cycle initiated by WISHBONE master. Write accesses are described in detail in this section assuming that WISHBONE slave unit decoded an address to fall within a range of one of its enabled images.
WISHBONE slave module is capable of handling single and block write transfers through one of its WISHBONE slave images. Read modify write (RMF) cycles are not supported.

Note:
Serial block transfers (bursts) are still under discussion, because WISHBONE bus specification does not provide a mechanism of identifying them. Until serial block transfers are specified, block writes will be handled as single writes. Proposal: usage of internal signal hardwired to a value indicating non-burst transfers. When bursts are

defined in WISHBONE specification, we can use this signal to indicate whether block transfer is serial or not.

All writes from WISHBONE master to PCI bus will be handled as posted writes. Posted writes are acknowledged on WISHBONE bus immediately after receiving a request (before they are finished on PCI bus). They are stored in WBW_FIFO. Each image can be mapped to I/O or Memory space, which is determined by a value of Address Space Mapping (ASM) bit in W_BAx register of corresponding image. If image maps to I/O space, then serial block transfers are not possible and WISHBONE master gets an error signal. Normal block transfers are possible to I/O and memory space since every data beat in a block is treated as a single posted write.

A write to address range occupied by an image mapped into memory space must be DWORD aligned (e.g. ADDR_O(1:0) must be 00). Otherwise cycle will be terminated with an error on WISHBONE bus.
A write to address range occupied by an image mapped into I/O space can be byte aligned though some limitations on SEL_O(3:0) encoding are present. Following table describes valid SEL_O(3:0) encoding for different values on ADDR_O(1:0).

Value on ADDR_O(1:0) lines	Valid SEL_O(3:0) encoding
00	Any or all SEL_O(3:0) can be asserted
01	Any or all SEL_O(3:1) can be asserted
10	Any or both SEL_O(3:2) can be asserted
11	Only SEL_O(3) can be asserted

Table 3-2: Valid ADDR_O(1:0) and SEL_O(3:0) combinations for I/O mapped address space accesses

All other combinations are invalid. Invalid accesses will be terminated with ERROR on WISHBONE bus.

Supported lengths of serial block transfers are still to be defined when serial block transfers are defined on WISHBONE bus.

In some cases writes initiated by WISHBONE master cannot be accepted and are terminated with retry:

- WBW_FIFO is full or doesn't have enough space left to accommodate another serial block transfer (if master signaled serial block transfer).
- There is an uncompleted delayed read request still pending in a WISHBONE slave unit (Writes can't be posted until a read finishes on PCI bus).

PCI master module requests a PCI bus after a complete transaction is stored in WISHBONE slave's unit WBW_FIFO. After PCI bus has been granted to PCI master module, it initiates a transaction on PCI bus. Module will use Memory Write or I/O Write PCI bus commands, depending on a value of Address Space Mapping bit (0 = memory, 1

= I/O) of image's W_BAx register. In case serial block write was posted by WISHBONE master, then PCI MASTER module will perform a burst of the same length to PCI target. Single posted writes or non-serial block writes will be completed as single writes on PCI bus. If PCI bus arbiter revokes mastership from PCI master module (#GNT is deasserted), it will finish current cycle and release PCI bus. Then it will have to re-arbitrate for it, to continue any posted writes left in a WBW_FIFO. The core handles Retry and Target-Disconnect Terminations by retrying the transaction until it completes or some other Termination is signaled.

Because all writes are posted and therefore immediately acknowledged to WISHBONE master, there is an alternate way of communicating errors signaled on PCI bus when posted writes were actually written to their final destination. Error reporting mechanism is provided through Error reporting registers. Error reporting must be enabled by Errors Enable (ERR_EN) bit of WISHBONE Error control and status (W_ERR_CS) register. When error reporting is enabled, errors can generate interrupts when Error Interrupt Enable (EINT_EN) bit of W_ERR_CS register is 1. Each of error reporting registers stores a part of information about the posted write transaction on PCI that was terminated with an error.

- A value of 1 in bit Error Signaled (ERR_SIG) of W_ERR_CS register indicates that error was recorded. Field Bus Command (BC) of this register stores a bus command used for an access that terminated with an error, while field Byte Enables (BE) stores a value of byte enables during the transfer. Bit Error Source (ES) indicates a source of an error (1 = Master (Master Abort), 0 = Target (Target Abort)).
- W_ERR_ADDR stores 32 bit address that PCI MASTER module tried to access when error occurred.
- W_ERR_DATA stores 32 bits of data that were used in a transfer that terminated with an error.

While bit ERR_SIG of W_ERR_CS register is set to 1, all of operations in WISHBONE slave module are frozen and new requests by WISHBONE masters are retried (except configuration space accesses).
When this bit is cleared, posted writes are continued in an order they were received. Only write that generated an error is discarded.
If Error reporting is disabled (ERR_EN bit of W_ERR_CS register is 0), then the transaction that caused the error is discarded while other posted writes continue in the order they were received.

3.2.4 WISHBONE to PCI Reads

Reads initiated by WISHBONE master are handled as single delayed reads. Multiple delayed reads are not supported. Delayed transactions must be completed on PCI bus before they complete on WISHBONE bus. Section on addressing and images described, how WISHBONE slave unit decodes addresses to know, if it is a slave for a current cycle. Handling of read transactions is encoded in image's control register (W_IMG_CTRLx).

There are a few options to define a behavior of WISHBONE slave unit during read transactions for images mapped to memory space:

- PREF_EN bit indicates that image's address range is pre-fetchable. That means that bridge core can pre-fetch data from the target and store it in WBR_FIFO. This method increases system performance, since Delayed read transaction only knows starting address of the transfer.
- MRL_EN bit indicates that PCI MASTER module is free to use Memory Read Line bus command for burst Reads. This only applies if PREF_EN bit is also set, otherwise this bit has no meaning.

Images mapped to I/O space will handle any read transactions as single Delayed Read (no bursts). If WISHBONE master attempts a serial block read from I/O space mapped image, the cycle will be terminated with an error by WISHBONE slave module.

Bridge core performs pre-fetched reads only through images mapped to memory space. Pre-fetchable address space is assumed in following conditions:

1. PREF_EN bit of corresponding W_IMG_CTRLx register is set
   or
2. WISHBONE master signals serial block read (in this case read will be pre-fetched regardless of PREF_EN bit value).

Not pre-fetchable address space is assumed in following conditions:

1. Accesses to I/O mapped address space
2. WISHBONE master performs a single or block read and PREF_EN bit is cleared

When WISHBONE slave unit latches address and SEL(3:0) data of a read request, WISHBONE MASTER module requests for PCI bus mastership. When mastership is granted, WISHBONE MASTER module initiates a PCI read transaction. Bus command used for transaction depends on various parameters described in following table:

Address space mapping of image	Cycle initiated by WISHBONE master	PREF_EN bit value	MRL_EN bit value	Bus command used
I/O	Single or block read	X	X	I/O Read
Memory	Single or block read	0	X	Memory Read
		1	0	Memory Read
		1	1	Memory Read Line
	Serial block read	X	1	Memory Read Line
		X	0	Memory Read

Table 3-3: Bus command encoding for reads through PCI MASTER module

Non-pre-fetchable address space reads are performed in one data phase on PCI bus. After first data phase PCI MASTER module releases PCI bus.
All delayed reads from address space marked as pre-fetchable are performed in bursts. PCI MASTER module reads data from the target and puts it into WBR_FIFO. PCI MASTER module finishes a burst read and releases PCI bus if any of the following conditions is met:

1. WBR_FIFO is full
2. Target issues Target-Disconnect
3. Mastership of PCI bus is revoked by PCI arbiter (#GNT is de-asserted)

When WISHBONE master retries this read transaction, data will be ready and WISHBONE slave module will pull data out of the WBR_FIFO and provide it on the WISHBONE bus. When WBR_FIFO is empty or WISHBONE master issues a read of address that is not one DWORD higher than previous address within the same block transfer, then WISHBONE slave module latches information about new read request and terminates the cycle with retry.
Any data left in WBR_FIFO is flushed immediately.

Up until now, WISHBONE TO PCI reads were described as all of them were successfully completed, but it's common for PCI bus targets or masters to generate error terminations. Terminations from PCI bus must be propagated to WISHBONE bus in some way, to let WISHBONE master know what happened with transaction it initiated.

Following terminations are possible by PCI target:

- Retry
- Disconnect with data
- Disconnect without data
- Target abort

Retry termination is not propagated back to WISHBONE bus. Bridge core just retries the transaction.
Disconnect is valid termination for single reads. PCI master module will not retry this transactions – it will store data for single read and wait for WISHBONE master to fetch it. In case WISHBONE master requested a serial block read, this kind of termination will be retried by PCI MASTER module until whole length of serial block transfer is read or Target-Abort or Master Abort is signaled.
Target Abort is an error that will be signaled to WISHBONE MASTER. When it retries the transaction it will receive a bus error termination (WISHBONE slave module will assert ERR_I).
Master Abort is an error Termination. If MA_ERR_DIS bit is 0, then transaction on WISHBONE bus will be terminated with bus error, otherwise it will be acknowledged and read will return all 1s.

3.3 PCI Target Unit

PCI target unit connects to PCI initiators acting as a target. This section describes basic functionality of the PCI target unit and is divided into subsections, each of them describing what PCI initiator needs to do to initiate PCI to WISHBONE transactions.

3.3.1 PCI Target Unit Functionality

This section describes functional overview of PCI target unit. Detailed description is provided in following sections.

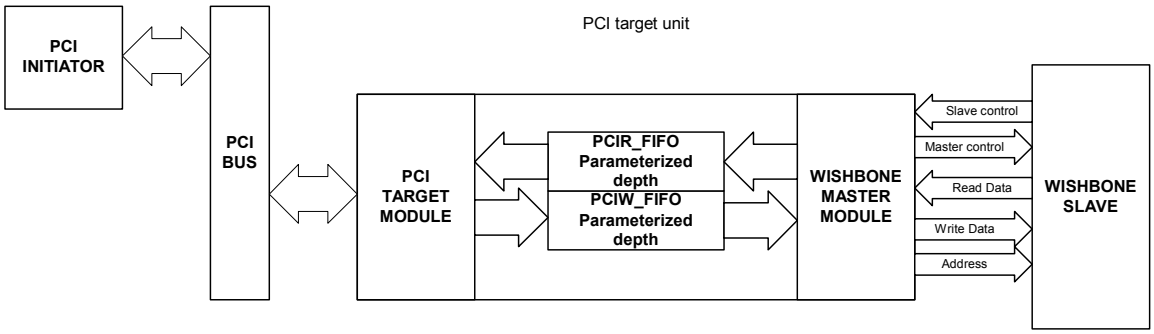

Figure 3-5: PCI target unit architecture overview

PCI target unit consists of a few functional parts allowing PCI initiators to perform a Read/Write accesses to WISHBONE bus.

3.3.1.1 PCI Target Module

PCI TARGET module is 32 bit/66MHz, PCI local bus specification Rev. 2.2 compliant target interface. It handles Read/Write cycles to images of WISHBONE address space and configuration space accesses.

3.3.1.2 PCIW_FIFO

PCI TARGET module uses PCIW_FIFO (PCI WRITE FIFO) for posting memory and I/O writes performed by PCI initiator. PCIW_FIFO also performs a different bus clock adaptation.

3.3.1.3 PCIR_FIFO

WISHBONE master module uses PCIR_FIFO (PCI READ FIFO) for storing data read from WISHBONE slaves. PCIR_FIFO also performs a different bus clock adaptation.

3.3.1.4 WISHBONE Master Module

WISHBONE master module is 32-bit WISHBONE master interface as defined in WISHBONE specification Rev. 1B. The core requests WISHBONE bus through its WISHBONE MASTER module. WISHBONE interface of the core is described in detail in Chapter 5-XY: WISHBONE I/O interface description.

3.3.2 PCI Target Unit's Addressing and Images

As mentioned before, PCI target unit incorporates 2 to 5 configurable PCI address space images (how the number of images is defined will be discussed in detail in design document and implementation notes) and one 4KB image used for configuration space accesses from PCI bus with configurable base address. This configurable base address points to the starting address of the 4KB configuration space, but only 2KB of configuration space are available for PCI target unit, see Figure 3-1. Configuration space for PCI target unit starts at 0x100 offset address (after a 256Bytes of PCI configuration header). Base address for PCI configuration space points at 0x0 offset address of whole configuration space and is actually a copy of first BASE ADDRESS register in PCI header, which is also used for configuration space.
Each image's behavior is controlled by its PCI BASE ADDRESS (P_BA1 – P_BA5), PCI TRANSLATION ADDRESS (P_TA1 – P_TA5), PCI IMAGE CONTROL (P_IMG_CTRL1 – P_IMG_CTRL5) and PCI ADDRESS MASK (P_AM1 – P_AM5) registers. Statuses, errors and interrupts for each image are recorded in image's status registers described later in this document. PCI target module claims the cycle initiated by initiator on PCI bus if one of PCI images is selected and enabled. Image is enabled, if IMG_EN bit of its P_AM register is set to 1. Image is selected when address provided during initial cycle on PCI bus falls within memory range of that image. Range is determined with values of P_BA and P_AM registers. Each image can represent from 4KB to 2GB of WISHBONE address space.

Each image can be mapped to Memory or I/O space, which is determined by Address Space Mapping (ASM) bit of image's P_BAx register. If ASM bit is 0, then image maps to Memory space, otherwise it is mapped to I/O space.

How to specify an 1MB image of WISHBONE address space with an addresses range of 0x10100000 - 0x101FFFFF ? Software must write a value of 0x10100XX0 to image's base address register (LSB of this register set to 0 to indicate a memory space mapping). This way base address is set at 0x10100000. Twelve LS bits are marked as don't cares – minimum block size is 4KB. Then software writes a value of 0xFFF00XXX into P_AM register of corresponding image. MS bit is IMG_EN bit and this way it is set to a value of 1 (it's also used for address mask – that is how we limit a maximum image size to 2GB). Each bit in P_AM register corresponds to one address line – if bit is 1, then this address line is used in address comparison, otherwise it is not. A value of 0xFFF00000 in P_AM register means that ADDR_O(31..20) signals will be compared with P_BA[31..20] value. If values match, the image is selected. In this case, ADDR_O(19..0) lines define an offset in 1MB address range.

Example 3-3: Address range of WISHBONE slave image

If address translation is enabled for selected image (AT_EN bit of P_IMG_CTRLx is 1), then address translation between PCI and WISHBONE address is performed. Address translation is done by replacing masked part of PCI address with corresponding bits from P_AT register. This provides a really flexible address mapping.

Let's make an assumption that base address and address mask are set as described in previous example. We want a PCI address range of 0x10100000 – 0x101FFFFF to be mapped elsewhere on WISHBONE bus, let's say 0x01000000 – 0x010FFFFF. For this to be done, we need addresses coming from PCI initiator to be translated. We set AT_EN bit of corresponding IMG_CTRL register to a value of 1 and corresponding P_AT register to a value of 0x01000XXX. P_AM register is already set, so address translation will replace AD(31..20) provided by PCI initiator with 0x010 value set in P_AT register for accesses on WISHBONE bus. This way we have WISHBONE address range of 0x01000000 – 0x010FFFFF accessible on PCI bus in a range of 0x10100000 – 0x101FFFFF.

Example 3-4: Address translation

3.3.3 PCI to WISHBONE Writes

Previous section described, how PCI target unit knows, if it is a target of current cycle initiated by PCI initiator. Write accesses are described in detail in this section assuming that PCI target unit decoded an address to fall within a range of one of its enabled images. PCI target module is capable of handling single and burst write transfers through one of its PCI target images.

Note:
Serial block transfers (bursts on WISHBONE bus) are still under discussion, because WISHBONE bus specification does not provide a mechanism of identifying them. Until serial block transfers are specified, all bursts from PCI bus are handled as block writes.

All writes from PCI master to WISHBONE bus will be handled as posted writes, what is also the reason, why Read Modify Write command on WISHBONE bus is not supported. Posted writes are claimed on the PCI bus immediately after receiving a request. They are stored in PCIW_FIFO. Each image can be mapped to I/O or Memory space, which is determined by a value of Address Space Mapping (ASM) bit in P_BAx register of corresponding image.

If image maps to the I/O space, then all 32 AD lines are used for a full byte address decoding and AD(1:0) lines are indicating the least significant valid byte for transaction. The byte enable lines BE#(3:0) indicate the size of the transfer within the DWORD and they must be consistent with AD(1:0) as seen in Table 3-5. All other combinations are invalid. Invalid accesses will be terminated with Target-Abort on the PCI bus.
All PCI bursts to the I/O space are treated as single posted writes and burst transfer is therefore broken into single transfers where the data phase will be terminated with

signaling Target Disconnect with data. The PCI Initiator will attempt next access with data following the first transfer. This is repeated until the "burst" transfer is finished.

Value on AD(1:0) lines	Starting Byte	Valid BE#(3:0) encoding
00	Byte 0	xxx0 or 1111
01	Byte 1	xx01 or 1111
10	Byte 2	x011 or 1111
11	Byte 3	0111 or 1111

Table 3-4: Valid AD(1:0) and BE# (3:0) combinations for I/O mapped address space accesses

If image maps to the memory space, then 30 AD lines (the AD(31:2) bus) provides a DWORD aligned address. The AD(1:0) lines are not part of the address decode, but they indicates the order in which the PCI Initiator is requesting the data to be transferred as seen in Table 3-5. Linear Incrementing burst mode is fully supported, while Cache-line Wrap mode will be broken into single transfers where the data phase will be terminated with signaling Target Disconnect with data. The PCI Initiator will attempt next access with data following the first transfer. This is repeated until the Cache-line Wrap mode burst transfer is finished.

Value on AD(1:0) lines	Burst Ordering encoding
00	**Linear Incrementing**
01	Reserved (disconnect after first data phase)
10	Cache-line Wrap mode
11	Reserved (disconnect after first data phase)

Table 3-5: BURST Ordering combinations for MEMORY mapped address space accesses

All other combinations are reserved (because of earlier version of PCI specification) and therefore accesses must be terminated with disconnect after first data phase, but there will be NO effect on requested memory address space.

There are some more reasons, why PCI Target TERMINATES current bus-cycle.
PCI Target unit terminates transfer with or after data was transferred on the initial data phase, when it is unable to respond within its subsequent latency requirement, with Disconnect with/without data:

- Target is not capable of doing a burst (as mentioned above);
- Target is temporarily unable to continue bursting, when PCIW_FIFO is already fulfilled with the current burst write;
- The burst crosses a resource boundary (or a resource boundary occurs).

PCI Target unit abnormally terminates transfer, when it detects a fatal error or it will never be able to complete the requested transfer, with Target-Abort:

- Master initiates a non-valid combination of AD(1:0) and BE#(3:0) when accessing the I/O mapped image space (as mentioned above).

PCI Target unit terminates transfer before any data is transferred, when it is busy and temporarily unable to process the transaction, with Retry:

- Internal resource conflict when PCIW_FIFO is full or doesn't have enough space left to accommodate another burst transfer;
- Target is locked by another master, when there is an uncompleted delayed read request still pending in a PCI target unit (Writes can't be posted until a read finishes on WISHBONE bus).

All PCI bus transfer terminations described above are PCI Target terminations, but Masters may also terminate transactions (described in chapters 3.2.3 and 3.2.4).

Regardless of MEMORY or I/O space image is mapped to, PCI Initiator or Target can insert wait cycles into the current write transfer.

PCI Target module must perform an address decoding every time PCI Initiator initiates a write transfer, to determine if this transfer is related to it. WISHBONE master module initiates a transaction on the WISHBONE bus after a complete transaction is stored in PCI target's unit PCIW_FIFO. Module will use single Write or block Write transfers, depending on a value of control bit in PCIW_FIFO line, which indicates burst from PCI bus. Block writes on WISHBONE bus are the same length as bursts from PCI. If a burst on the PCI bus was cut, because of smaller PCIW_FIFO depth, then the block size is as large as the size of the burst written into the PCIW_FIFO. When a PCI Initiator will complete burst write with next access, it will be treated as a new burst transfer written to PCIW_FIFO.

Because all writes are posted and therefore immediately claimed to PCI initiator and stored to PCIW_FIFO, there is an alternate way of communicating errors signaled on WISHBONE bus when posted writes were actually written to their final destination. Error reporting mechanism is provided through Error reporting registers. Error reporting must be enabled by Errors Enable (ERR_EN) bit of PCI Error control and status (P_ERR_CS) register. When error reporting is enabled, errors can generate interrupts when Error Interrupt Enable (EINT_EN) bit of P_ERR_CS register is 1. Each of error reporting registers stores a part of information about the posted write transaction on the WISHBONE bus that was terminated with an error.

- A value of 1 in bit Error Signaled (ERR_SIG) of P_ERR_CS register indicates that error was recorded. Field Bus Command (BC) of this register stores a bus command used on PCI bus for an access that terminated with an error on WISHBONE bus, while field Byte Enables (BE) stores a value of byte enables (SEL_O(3:0) lines) during the transfer.
- P_ERR_ADDR stores 32-bit address that WISHBONE MASTER module tried to access when error occurred.
- P_ERR_DATA stores 32-bits of data that were used in a transfer on WISHBONE bus that terminated with an error.

While bit ERR_SIG of P_ERR_CS register is set to 1, all of operations in PCI target module are frozen and new requests by PCI initiators are retried (except configuration space accesses).
When this bit is cleared, posted writes are continued in an order they were received. Only write that generated an error is discarded.
If Error reporting is disabled (ERR_EN bit of P_ERR_CS register is 0), then the transaction that caused the error is discarded while other posted writes continue in the order they were received.

3.3.4 PCI to WISHBONE Reads

Reads initiated by PCI initiator are handled as single delayed reads, what is also the reason, why Read Modify Write command on WISHBONE bus is not supported. Multiple delayed reads are not supported. Delayed transactions must be completed on WISHBONE bus before they complete on PCI bus. Section on addressing and images described, how PCI target unit decodes addresses to know, if it is a target for a current cycle. Handling of read transactions is encoded in PCI image's control register (P_IMG_CTRLx). There are a few options to define a behavior of PCI target unit during read transactions for images mapped to memory space:

- PREF_EN bit indicates that PCI image's address range is pre-fetchable. That means that bridge core can pre-fetch data from the slave and store it in WBR_FIFO. This method increases system performance, since Delayed read transaction only knows starting address of the transfer.

Images mapped to I/O space will handle any read transactions as single Delayed Read (no bursts). If PCI initiator attempts a burst read from I/O space mapped image, the cycle will be terminated with Disconnect with Data (see also chapter 3.3.3), so that initiator can continuo with reading the rest of the data (with disconnecting of the bursts).

Bridge core performs pre-fetched reads only through images mapped to memory space. Pre-fetchable address space is assumed in following conditions:

- PREF_EN bit of corresponding P_IMG_CTRLx register is set
  or
- PCI initiator signals burst read (in this case read will be pre-fetched regardless of PREF_EN bit value).

Not pre-fetchable address space is assumed in following conditions:

- Accesses to I/O mapped address space
- PCI initiator performs a single read and PREF_EN bit is cleared

The following table shows which PCI bus read commands are considered single or block transfer regarding PREF_EN bit:

Address space mapping of image	Bus command initiated by PCI initiator	PREF_EN bit value	Used cycle by WISHBONE master
I/O	I/O Read	X	Single read
Memory	Memory Read	0	Single read
	Memory Read	1	Block read
	Memory Read Line	X	Block read

Table 3-6: Bus command encoding for reads through PCI TARGET module

Non-pre-fetchable address space reads are performed in one data phase on WISHBONE bus. After first data phase WISHBONE MASTER module releases WISHBONE bus.
All delayed reads from address space marked as pre-fetchable are performed block reads. WISHBONE MASTER module reads data from the WISHBONE slave and puts it into PCIR_FIFO. WISHBONE MASTER module finishes a block read and releases WISHBONE bus if any of the following conditions is met:

- PCIR_FIFO is full
- WISHBONE slave issues Error or Retry.

When PCI initiator retries this read transaction, data will be ready and PCI target module will pull data out of the PCIR_FIFO and provide it on the PCI bus. When PCIR_FIFO is empty or PCI initiator issues a read of address that is not one DWORD higher than previous address within the same block transfer, then PCI target module latches information about new read request and terminates the cycle with Retry.
Any data left in PCIR_FIFO is flushed immediately.

Up until now, PCI TO WISHBONE reads were described as all of them were successfully completed, but it's common for WISHBONE bus slaves or masters to generate error terminations. Terminations from WISHBONE bus must be propagated to PCI bus in some way, to let PCI initiators know what happened with transaction it initiated.
Following terminations are possible by WISHBONE slaves:

- Retry, which is not propagated back to PCI bus. Bridge core just retries the transaction.
- Error, is a termination that will be signaled to PCI initiator. When PCI initiator retries the transaction it will receive a Target-Abort.

There are some more reasons, why PCI Target TERMINATES current bus-cycle.
PCI Target unit terminates transfer with or after data was transferred on the initial data phase, when it is unable to respond within its subsequent latency requirement, with Disconnect with/without data:

- Target is not capable of doing a burst (as mentioned above – reading from I/O mapped space);
- Target is temporarily unable to continue bursting, when PCIR_FIFO become empty with the current burst read;

- The burst crosses a resource boundary (or a resource boundary occurs).

PCI Target unit abnormally terminates transfer, when it detects a fatal error or it will never be able to complete the requested transfer, with Target-Abort:

- Master initiates a non-valid combination of AD(1:0) and BE#(3:0) when accessing the I/O mapped image space (see Table 3-4 in chapter 3.3.3).

PCI Target unit terminates transfer before any data is transferred, when it is busy and temporarily unable to process the transaction, with Retry:

- Internal resource conflict when PCIR_FIFO is empty.

All PCI bus transfer terminations described above are PCI Target terminations, but Masters may also terminate transactions (described in chapters 3.2.3 and 3.2.4).

Regardless of MEMORY or I/O space image is mapped to, PCI Initiator or Target can insert wait cycles into the current write transfer.
On the other side of the PCI Target module, on the WISHBONE master unit side, the WISHBONE slave can also insert wait cycles.

3.4 Transaction Ordering

In order to satisfy PCI transaction ordering rules, following functionality will be implemented:

1. When WISHBONE slave unit receives a read request and no other delayed read request is pending or waiting to be retried by WISHBONE master, it latches address and byte enable information and terminates the cycle with retry.
2. After receiving Read Request, WISHBONE slave unit locks out any non-configuration space accesses. (All requests to WISHBONE slave unit are terminated with retry).
3. Posted writes from WBW_FIFO are processed until WBW_FIFO is empty.
4. PCI master module completes a read on PCI bus.
5. After read is completed (e.g. becomes Delayed Read Completion), posted writes are accepted again in WBW_FIFO.
6. PCI target module retries all non-configuration space accesses from PCI bus.
7. All posted writes from PCIW_FIFO are completed on WISHBONE bus until PCIW_FIFO is empty.
8. WISHBONE slave unit will allow a read to complete on WISHBONE bus
9. If the read is not completed, then WISHBONE slave and PCI target unit allow posting of writes.

3.5 Parity

Parity monitoring and generation is required by all PCI agents according to PCI local bus specification. PCI MASTER module monitors PAR signal during reads and drives it during writes. PAR signal provides even parity through C/BE# [3:0] and AD [31:0] lines during address and data phase. If PCI MASTER performs a write, target is responsible for monitoring PAR and asserting PERR# if error is detected. During reads, PCI MASTER module monitors PAR and asserts PERR# if error is detected. If master detects parity

error during a read transaction or samples PERR# signal asserted during a write transaction, it must set Parity Error Detected bit in its Configuration space status register.
If Parity Error Response bit is set, PCI MASTER module must signal parity error by asserting PERR# signal during read transactions.
When PERR_INT_EN bit is set, then the core signals an interrupt request as additional response to parity errors as recommended by PCI bus specification. Parity error detection has no influence on PCI MASTER module – it continues the transaction until it is finished or until Target terminates it.

3.6 Interrupts

PCI IP core is capable of generating interrupts in response to different events. Interrupts are controlled through Interrupt Control and Interrupt Status registers. Interrupts are reported on PCI bus through assertion of INTA# pin if core is implemented as Guest bridge. Interrupts are reported on WISHBONE bus through assertion of INTA_O pin if core is implemented as Host.
Interrupt control register is used for enabling/disabling interrupts originating from different sources.
Interrupt status register is used to determine the source of interrupt and clearing interrupt requests. Software must locate and clear the source of interrupt request before clearing status bits in a bridge core.

4

Registers

This section describes all control and status register inside the PCI core. The *Width* field specifies the number of bits in the register, and *Access* specifies the valid access types for that register. R/W stands for read and write access and R stands for read only access.

4.1 Register list and description

Code	Offset	Width	Access	Description
PCI Configuration Space	0x000 – 0x0FF			PCI specification Rev2.2 Configuration space
PCI_CONF_SPC_BAR (Base for PCI bus)	0x100 and 0x010	32	R/W	PCI Configuration Space Base Address Register
P_IMG_CTRL1	0x104	32	R/W	PCI Image1 control register
P_BA1	0x014 and 0x108	32	R/W	PCI Image1 Base Address Register
P_AM1	0x10C	32	R/W	PCI Image1 Address Mask Register
P_TA1	0x110	32	R/W	PCI Image1 Translation Address Register
P_IMG_CTRL2	0x114	32	R/W	PCI Image2 control register
P_BA2	0x018 and 0x118	32	R/W	PCI Image2 Base Address Register
P_AM2	0x11C	32	R/W	PCI Image2 Address Mask Register
P_TA2	0x120	32	R/W	PCI Image2 Translation Address Register
P_IMG_CTRL3	0x124	32	R/W	PCI Image3 control register
P_BA3	0x01C and 0x128	32	R/W	PCI Image3 Base Address Register
P_AM3	0x12C	32	R/W	PCI Image3 Address Mask Register
P_TA3	0x130	32	R/W	PCI Image3 Translation Address Register
P_IMG_CTRL4	0x134	32	R/W	PCI Image4 control register

Code	Offset	Width	Access	Description
P_BA4	0x020 and 0x138	32	R/W	PCI Image4 Base Address Register
P_AM4	0x13C	32	R/W	PCI Image4 Address Mask Register
P_TA4	0x140	32	R/W	PCI Image4 Translation Address Register
P_IMG_CTRL5	0x144	32	R/W	PCI Image5 control register
P_BA5	0x024 and 0x148	32	R/W	PCI Image5 Base Address Register
P_AM5	0x14C	32	R/W	PCI Image5 Address Mask Register
P_TA5	0x150	32	R/W	PCI Image5 Translation Address Register
P_ERR_CS	0x154	32	R/W	PCI Error Control and Status Register
P_ERR_ADDR	0x158	32	R	PCI Erroneous Address Register
P_ERR_DATA	0x15C	32	R	PCI Erroneous Data Register
WB_CONF_SPC_BAR (Base for WISHBONE bus)	0x800	32	R	WISHBONE Configuration Space Base Address
W_IMG_CTRL1	0x804	32	R/W	WISHBONE Image1 control register
W_BA1	0x808	32	R/W	WISHBONE Image1 Base Address Register
W_AM1	0x80C	32	R/W	WISHBONE Image1 Address Mask Register
W_TA1	0x810	32	R/W	WISHBONE Image1 Translation Address Register
W_IMG_CTRL2	0x814	32	R/W	WISHBONE Image2 control register
W_BA2	0x818	32	R/W	WISHBONE Image2 Base Address Register
W_AM2	0x81C	32	R/W	WISHBONE Image2 Address Mask Register
W_TA2	0x820	32	R/W	WISHBONE Image2 Translation Address Register
W_IMG_CTRL3	0x824	32	R/W	WISHBONE Image3 control register
W_BA3	0x828	32	R/W	WISHBONE Image3 Base Address Register

Code	Offset	Width	Access	Description
W_AM3	0x82C	32	R/W	WISHBONE Image3 Address Mask Register
W_TA3	0x830	32	R/W	WISHBONE Image3 Translation Address Register
W_IMG_CTRL4	0x834	32	R/W	WISHBONE Image4 control register
W_BA4	0x838	32	R/W	WISHBONE Image4 Base Address Register
W_AM4	0x83C	32	R/W	WISHBONE Image4 Address Mask Register
W_TA4	0x840	32	R/W	WISHBONE Image4 Translation Address Register
W_IMG_CTRL5	0x844	32	R/W	WISHBONE Image5 control register
W_BA5	0x848	32	R/W	WISHBONE Image5 Base Address Register
W_AM5	0x84C	32	R/W	WISHBONE Image5 Address Mask Register
W_TA5	0x850	32	R/W	WISHBONE Image5 Translation Address Register
W_ERR_CS	0x854	32	R/W	WISHBONE Error Control and Status Register
W_ERR_ADDR	0x858	32	R	WISHBONE Erroneous Address Register
W_ERR_DATA	0x85C	32	R	WISHBONE Erroneous Data Register
CNF_ADDR	0x860	32	R/W	Configuration Cycle Generation Address Register
CNF_DATA	0x864	32	R/W	Configuration Cycle Generation Data Register
INT_ACK	0x868	32	R	Interrupt Acknowledge Register
ICR	0xFF8	32	R/W	Interrupt Control Register
ISR	0xFFC	32	R/W	Interrupt Status Register

4.1.1 WISHBONE Slave Unit Control & Status

WISHBONE slave unit registers start at offset 0x800 from base address. Base address is pre-defined at design time for WISHBONE bus accesses, base address for PCI bus is defined with configuration cycle for Guest implementation or with writing to this register by WISHBONE master for Host implementation.
See also the Chapter 3.1 (Configuration space).

4.1.1.1 WISHBONE Configuration Space BAR

Width	Access	Reset	Description
32	R	*	This register stores base address for accessing core's registers from WISHBONE bus. It is Read Only.

* - Value at reset is defined before implementation in parameter file

Table 4-1: WISHBONE Configuration space Base Address Register

Register layout:

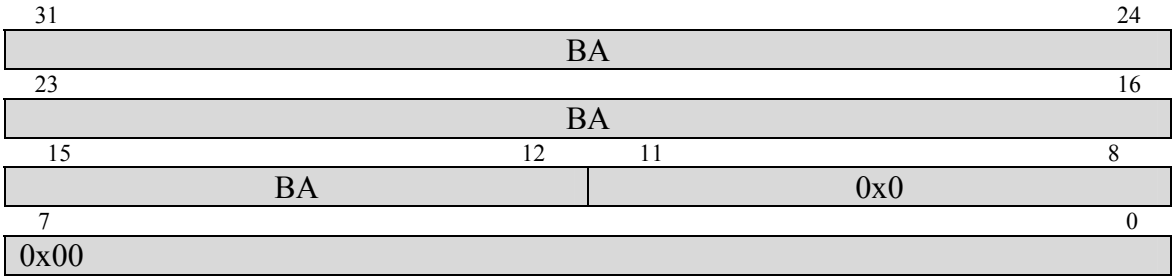

Figure 4-1: WISHBONE Configuration space Base Address Register layout

Register is Read Only. Bits 31 – 12 define WISHBONE configuration space Base Address. Bits 11 through 0 are always 0 because Configuration space always occupies 4KB of address space.

4.1.1.2 WISHBONE Image Control and Address Registers

There are five possible configurable WISHBONE slave images. Each of these images implements its own set of registers. Image Control and Address registers are the same through all five Images.

Image Control Registers – W_IMG_CTRL1 – W_IMG_CTRL5

Width	Access	Reset	Description
32	RW	0x00000000	Register value controls WISHBONE slave unit behavior when image is selected and enabled.

Table 4-2: WISHBONE Image Control Register

Register layout:

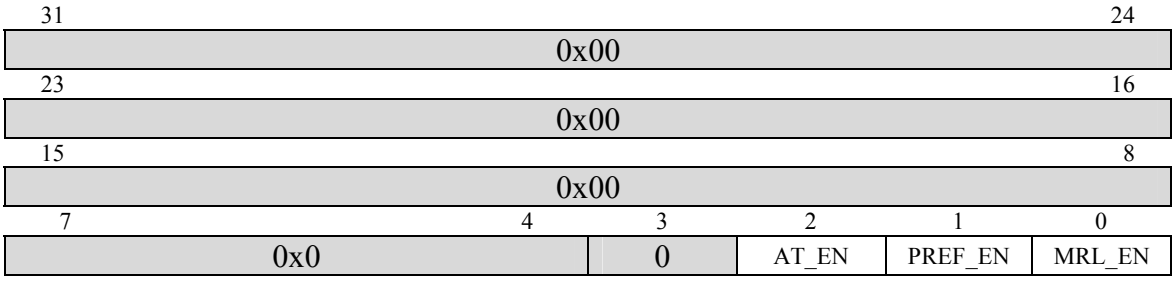

Figure 4-2: WISHBONE Image Control Register layout

Bits descriptions:

Bit number	Name	Description
31 – 3	N/A	Not used
2	Address Translation Enable	If this bit is set, address translation for corresponding image is enabled.
1	Pre-fetch enable	Marks address space occupied by an image as pre-fetchable
0	Memory Read Line Enable	If PREF_EN bit is also set or WISHBONE master signals a serial block read through WISHBONE slave unit, PCI master module will use Memory Read Line Command instead of Memory Read command on PCI bus.

Table 4-3: WISHBONE Image Control Register bits descriptions

Base address registers – W_BA1- W_BA5

Width	Access	Reset	Description
32	RW	0x00000000	Register value holds WISHBONE bus base address of an image

Table 4-4: WISHBONE Base Address Register

Register layout:

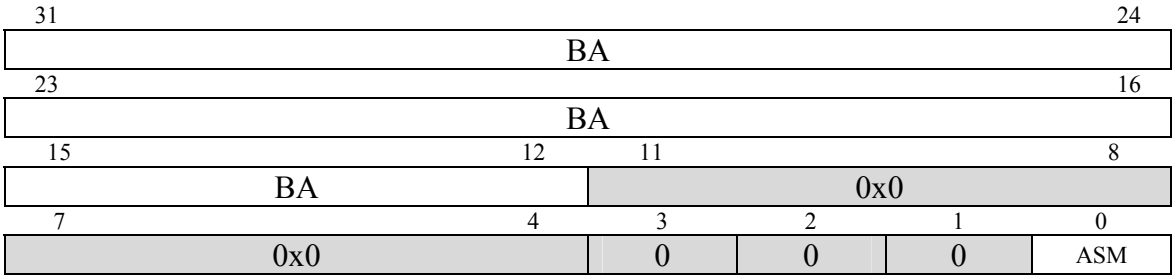

Figure 4-3: WISHBONE Base Address Register layout

Bits descriptions:

Bit number	Name	Description
31 – 12	Base Address	Image's base address. How many bits from this field are compared with ADDR_I(31:0) is defined in Address mask register
11-1	N/A	Because minimum block size is 4KB, this field is reserved
0	Address Space Mapping	Bit defines which address space on PCI bus image maps to. 0 – Memory space mapping 1 – I/O space mapping

Table 4-5: WISHBONE Base Address Register bits descriptions

Address mask registers – W_AM1 – W_AM5

Width	Access	Reset	Description
32	RW	0x00000000	Register value represents address mask. If corresponding bit is 1, address line in the same position is compared with value in Base address register. If bit is 0, corresponding address line is not compared with value in BA register.

Table 4-6: WISHBONE Address Mask Register

Register layout:

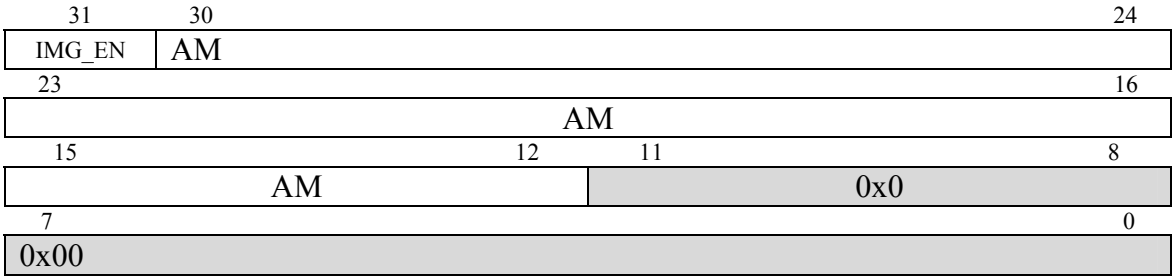

Figure 4-4: WISHBONE Address Mask Register layout

Bits descriptions:

Bit number	Name	Description
31	Image Enable & Address Mask(31)	Bit must be set for image to be enabled. If bit is 0, corresponding image is not enabled. This bit is also used in Address Masking – that's how a limit of 2GB per image is implemented (at least ADDR_I(31)) must be compared with BA for each image.
30 – 12	Address Mask	The remainder of Address Mask. If Bit(x) of address mask is 1, then ADDR_I(x) is compared with BA(x) bit in Base Address Register, otherwise it is not.
11-0	N/A	Because minimum block size is 4KB, this field is always 0x000 (lower twelve address lines are never compared with BA register value).

Table 4-7: WISHBONE Address Mask Register bits descriptions

Translation address registers – W_TA1 – W_TA5

Width	Access	Reset	Description
32	RW	0x00000000	If address translation is enabled, compared address lines from WISHBONE bus (specified with AM value) are replaced with corresponding values in this register for PCI bus Accesses.

Table 4-8: WISHBONE Translation Address Register

Register layout:

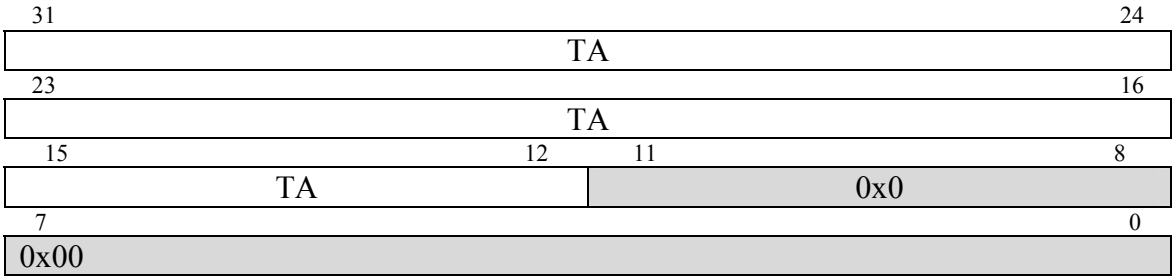

Figure 4-5: WISHBONE Translation Address Register layout

Bits descriptions:

Bit number	Name	Description
31 – 12	Translation Address	Register value is used when Address Translation is enabled. Each value on ADDR_I lines that is not masked by AM register setting is replaced with corresponding bit value of Translation address register for PCI bus accesses.
11-0	N/A	Because minimum block size is 4KB, this field is always 0x000 (lower twelve address lines are never replaced).

Table 4-9: WISHBONE Translation Address Register bits descriptions

4.1.2 PCI Target Unit Control & Status

Base address for PCI Target Configuration Header registers from PCI bus is defined with configuration cycle for Guest implementation or with writing to this register by WISHBONE master for Host implementation.
See also the Chapter 3.1 (Configuration space).
Besides PCI Target Configuration Header registers, there are also PCI Target Unit Configuration Registers.

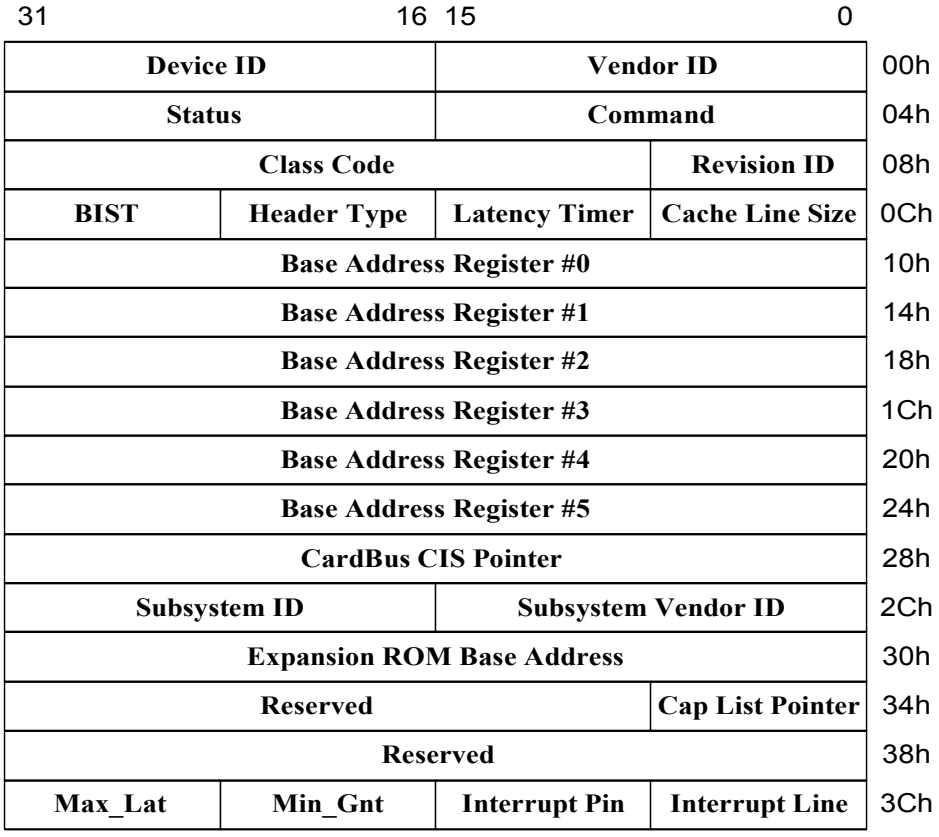

Figure 4-6: PCI Configuration Space Header (Header type 00h)

Vendor ID, Device ID, Command, Status, Revision ID, Class Code and Header Type must be supported by all PCI compliant devices. Header Type is type 00h, which defines the header structure on Figure 4-6.
Configuration space header used for device identification:

- **Vendor ID:** This field identifies the manufacturer of the device. Valid vendor identifiers are allocated by the PCI SIG to ensure uniqueness. 0FFFFh is an invalid value for Vendor ID.
- **Device ID:** This field identifies the particular device. This identifier is allocated by the vendor.
- **Revision ID:** This register specifies a device specific revision identifier. The value is chosen by the vendor. Zero is an acceptable value. This field should be viewed as a vendor defined extension to the Device ID.
- **Header Type:** This byte identifies the layout of the second part of the predefined header (beginning at byte 10h in Configuration Space) and also whether or not the device contains multiple functions. Bit 7 in this register 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. Bits 6 through 0 identify the layout of the second part of the predefined header.
- **Class Code:** The Class Code register is read-only and is used to identify the generic unction of the device and, in some cases, a specific register-level programming interface (for detailed description see PCI 2.2 Specification).

For device control functions, there is **Command Register**. When 0 is written to it, the device is logically disconnected from the bus (except for configuration accesses). Following table shows bits descriptions.

Bits descriptions:

Bit number	Implemented	Description
15 – 10	-	Reserved
9	NO	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. A value of 1 means the master is allowed to generate fast back-to-back transactions to different agents. A value of 0 means fast back-to-back transactions are only llowed to the same agent. State after **RST#** is 0.
8	√	SERR# enable. A value of 0 disables the **SERR#** driver. A value of 1 enables the **SERR#** driver. This bit's state after **RST#** is 0. Address parity errors are reported only if this bit and bit 6 are 1.
7	NO	Stepping control. This bit is used to control whether or not a device does address/data stepping. Devices that never do stepping must hardwire this bit to 0.

Bit number	Implemented	Description
6	√	Parity Error Response. This bit controls the device's response to parity errors. When the bit is set, the device must take its normal action when a parity error is detected. When the bit is 0, the device sets its Detected Parity Error status bit (bit 15 in the Status register) when an error is detected, but does not assert **PERR#** and continues normal operation. State after **RST#** is 0.
5	NO	VGA Palette Snoop. This bit controls how VGA compatible and graphics devices handle accesses to VGA palette registers. When this bit is 1, palette snooping is enabled (i.e., the device does not respond to palette register writes and snoops the data).
4	NO	Memory Write and Invalidate. This is an enable bit for using the Memory Write and Invalidate command. When this bit is 1, masters may generate the command. When it is 0, Memory Write must be used instead. State after **RST#** is 0..
3	NO	Special cycles. Controls a device's ability to act as a master on the PCI bus. A value of 0 disables the device from generating PCI accesses. A value of 1 allows the device to behave as a bus master. After RST# is 0.
2	√	Bus master. Controls a device's ability to act as a master on the PCI bus. A value of 0 disables the device from generating PCI accesses. A value of 1 allows the device to behave as a bus master. After RST# is 0.
1	√	Memory space. Controls a device's response to Memory Space accesses. A value of 0 disables the device response. A value of 1 allows the device to respond to Memory Space accesses. State after RST# is 0.
0	√	I/O space. Controls a device's response to I/O Space accesses. A value of 0 disables the device response. A value of 1 allows the device to respond to I/O Space accesses. After RST# is 0.

Table 4-10: Command register of PCI configuration header

For device status, there is **Status Register**. Reserved bits are read-only and return 0 after reading. A one bit is reset whenever is a 1 written to a corresponding bit location. Following table shows bits descriptions.

Bits descriptions:

Bit number	Implemented	Description
15	√	Detected Parity Error. This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit 6 in the Command register).
14	√	Signaled System Error. This bit must be set whenever the

Bit number	Implemented	Description
		device asserts **SERR**#.
13	√	Received Master Abort. This bit must be set by a master device whenever its transaction (except for Special Cycle) is terminated with Master-Abort. All master devices must implement this bit.
12	√	Received Target Abort. This bit must be set by a master device whenever its transaction is terminated with Target-Abort.
11	√	Signaled Target Abort. This bit must be set by a target device whenever it terminates a transaction with Target-Abort.
10 – 9	√	DEVSEL timing: 00 – fast; **01 – medium**; 10 – slow. These bits are read-only and must indicate the slowest time that a device asserts **DEVSEL#** for any bus command except Configuration Read and Configuration Write.
8	√	Master Data Parity Error. This bit is only implemented by bus masters. It is set when three conditions are met: 1) the bus agent asserted **PERR#** itself (on a read) or observed **PERR#** asserted (on a write); 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.
7	NO	Fast Back-to-Back Capable. 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.
6	-	Reserved
5	√	66 MHz capable. This optional read-only bit indicates whether or not this device is capable of running at 66 MHz. A value of 1 indicates that the device is 66 MHz capable.
4	NO	Compatibilities list. A value of zero indicates that no New Capabilities linked list is available. A value of one indicates that the value read at offset 34h is a pointer in Configuration Space to a linked list of new capabilities.
3 – 0	-	Reserved

Table 4-11: Status register of PCI configuration header

Following descriptions include only that Miscellaneous (device independent) registers, which are implemented.

Latency Timer register specifies in units of PCI bus clocks the value of the timer. After RST# the register value is 0.

Interrupt Line register tells which input of the system interrupt controller(s) the device's interrupt pin is connected to (how it is implemented, will be described in detail in Design document).

Interrupt Pin register tells which interrupt pin the device uses. A value of 1 corresponds to INTA# and so on. The values from 05h to FFh are reserved.

There are 6 **Base Address registers,** and each one of them consists of 28-bit base address for MEMORY map or 30-bit base address for I/O map. Other bits are control bits and described in the following table:

Bits descriptions:

Bit number	Description
31 – 4	Base address (only upper 20 bits are valid)
3	Pre-fetchable
2 –1	Type: **00 – 32-bit address space**; 01 – Reserved; 10 – 64-bit address space; 11 – Reserved.
0	Memory space indicator = **'0'** (always for MEMORY mapped space) !!!

Table 4-12: BASE ADDRESS register of PCI configuration header for MEMORY mapped space

Bits descriptions:

Bit number	Description
31 – 2	Base address(only upper 20 bits are valid)
1	Reserved.
0	I/O space indicator = **'1'** (always for I/O mapped space) !!!

Table 4-13: BASE ADDRESS register of PCI configuration header for I/O mapped space

Regardless which space (Memory or I/O) Base Address register is pointing to, its value is also mapped to the PCI Configuration space BAR (base address register).

4.1.2.1 PCI Configuration Space BAR

Width	Access	Reset	Description
32	RW	*	This register stores base address for accessing core's registers from WISHBONE bus.

* - Value at reset is defined before implementation in parameter file

Table 4-14: PCI Configuration space Base Address Register

Register layout:

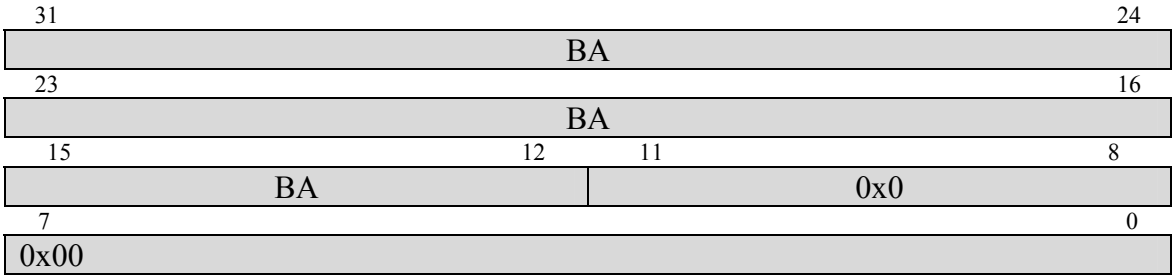

Figure 4-7: PCI Configuration space Base Address Register layout

Register is Read Only. Bits 31 – 12 define PCI configuration space Base Address. Bits 11 through 0 are always 0 because Configuration space always occupies 4KB of address space.

4.1.2.2 PCI Image Control and Address Registers

There are five possible configurable PCI slave images. Each of these images implements its own set of registers. Image Control and Address registers are the same through all five Images.

Image Control Registers – P_IMG_CTRL1 – P_IMG_CTRL5

Width	Access	Reset	Description
32	RW	0x00000000	Register value controls PCI target unit behavior when image is selected and enabled.

Table 4-15: PCI Image Control Register

Register layout:

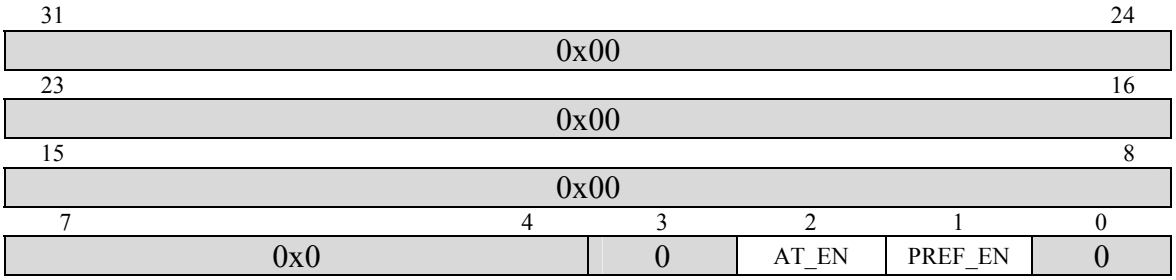

Figure 4-8: PCI Image Control Register layout

Bits descriptions:

Bit number	Name	Description
31 – 3	N/A	Not used
2	Address Translation Enable	If this bit is set, address translation for corresponding image is enabled.
1	Pre-fetch enable	Marks address space occupied by an image as pre-fetchable
0	N/A	Not used

Table 4-16: PCI Image Control Register bits descriptions

Base address registers – P_BA1- P_BA5

Width	Access	Reset	Description
32	RW	0x00000000	Register value holds PCI bus base address of an image

Table 4-17: PCI Base Address Register

Register layout:

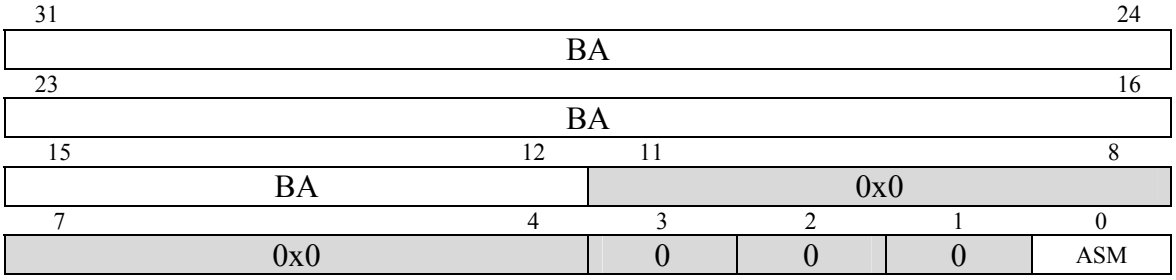

Figure 4-9: PCI Base Address Register layout

Bits descriptions:

Bit number	Name	Description
31 – 12	Base Address	Image's base address. How many bits from this field are compared with ADDR_I(31:0) is defined in Address mask register
11-1	N/A	Because minimum block size is 4KB, this field is reserved
0	Address Space Mapping	Bit defines which address space on PCI bus image maps to. 0 – Memory space mapping 1 – I/O space mapping

Table 4-18: PCI Base Address Register bits descriptions

Address mask registers – P_AM1 – P_AM5

Width	Access	Reset	Description
32	RW	0x00000000	Register value represents address mask. If corresponding bit is 1, address line in the same position is compared with value in Base address register. If bit is 0, corresponding address line is not compared with value in BA register.

Table 4-19: PCI Address Mask Register

Register layout:

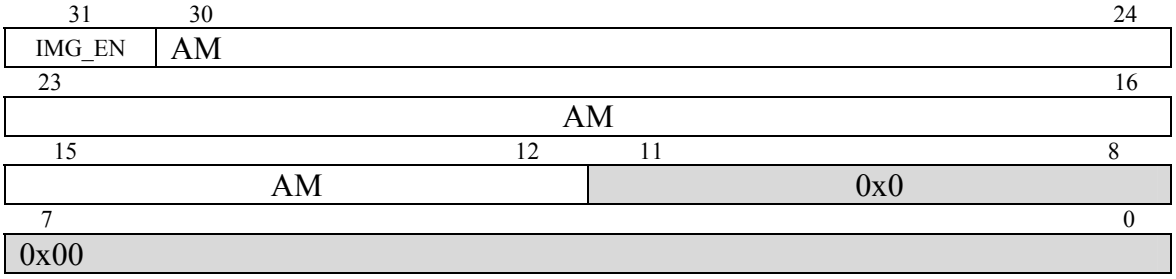

Figure 4-10: PCI Address Mask Register layout

Bits descriptions:

Bit number	Name	Description
31	Image Enable & Address Mask(31)	Bit must be set for image to be enabled. If bit is 0, corresponding image is not enabled. This bit is also used in Address Masking – that's how a limit of 2GB per image is implemented (at least ADDR_I(31)) must be compared with BA for each image.
30 – 12	Address Mask	The remainder of Address Mask. If Bit(x) of address mask is 1, then ADDR_I(x) is compared with BA(x) bit in Base Address Register, otherwise it is not.
11-0	N/A	Because minimum block size is 4KB, this field is always 0x000 (lower twelve address lines are never compared with BA register value).

Table 4-20: PCI Address Mask Register bits descriptions

Translation address registers – P_TA1 – P_TA5

Width	Access	Reset	Description
32	RW	0x00000000	If address translation is enabled, compared address lines from PCI bus (specified with AM value) are replaced with corresponding values in this register for WISHBONE bus Accesses.

Table 4-21: PCI Translation Address Register

Register layout:

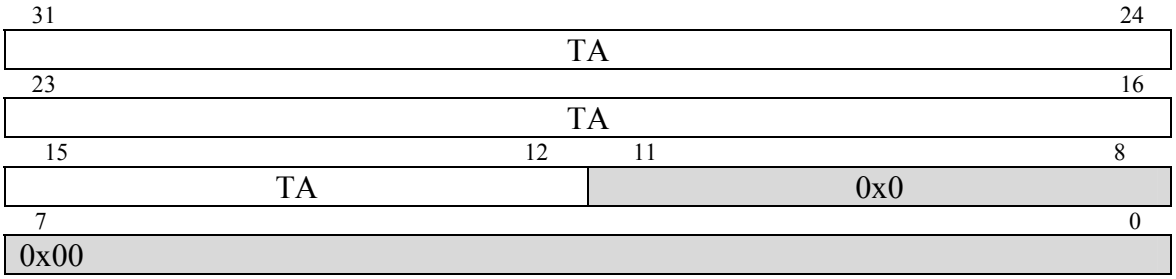

Figure 4-11: PCI Translation Address Register layout

Bits descriptions:

Bit number	Name	Description
31 – 12	Translation Address	Register value is used when Address Translation is enabled. Each value on ADDR_I lines that is not masked by AM register setting is replaced with corresponding bit value of Translation address register for WISHBONE bus accesses.
11-0	N/A	Because minimum block size is 4KB, this field is always 0x000 (lower twelve address lines are never replaced).

Table 4-22: PCI Translation Address Register bits descriptions

4.1.3 Error Reporting Registers

Error reporting registers are provided because of Posted Writes. Posted Writes are always acknowledged on WISHBONE bus before they actually complete on PCI bus and vice-versa, so errors detected on PCI or WISHBONE buses cannot be reported back to WISHBONE master or PCI initiator using standard bus protocol.

4.1.3.1 WISHBONE Slave Unit Error Reporting Registers

WISHBONE Error Control and Status Register – W_ERR_CS

Width	Access	Reset	Description
32	RW	0x00000000	Part of register is used for controlling Error Reporting mechanism, another part is used for reporting statuses and additional information about an error that occurred during completion of posted write on PCI bus.

Table 4-23: WISHBONE Error Control and Status Register

Register layout:

31 – 28	27 – 24	23 – 16	15 – 12	11	10	9	8
BE	BC	0x00	0x0	0	RTY_EXP	ES	ERR_SIG

7 – 4	3	2	1	0
0x0	0	0	0	ERR_EN

Figure 4-12: WISHBONE Error Control and Status Register layout

Bits descriptions:

Bit number	Name	Description
31 – 28	Byte Enables	Field value reports the state of BE# signals that was used in Posted Write which terminated with an error.
27-24	Bus Command	Field value reports bus command that was used for Posted Write which terminated with an error
16 – 11	N/A	Not used
10	Retry Counter Expired	Reports that posted write was Retried *MAX_RETRY* times.
9	Error Source	Set ES bit indicates that master terminated the transaction with Master Abort. Software can distinguish between two kinds of Master Abort terminations that PCI MASTER module performs – If RTY_EXP bit is cleared, then Master Abort was performed because no target claimed the transaction. If RTY_EXP is set, then target signaled too many Retry terminations. Cleared ES bit indicates that target of the transaction signaled Target Abort.

Bit number	Name	Description
8	Error Signaled	Set bit indicates that error is being reported. While this bit is set all WISHBONE slave unit's operation is frozen. Software must clear this bit to enable transactions to resume their path through WISHBONE slave unit. Bit is cleared by writing 1 to its location.
7-1	N/A	Not used
0	Error Enable	Setting this bit enables Error Reporting mechanism. Clear bit means that Error Reporting will not be performed – Transaction that caused an error is discarded, other transactions continue normally.

Table 4-24: WISHBONE Error Control and Status Register bits descriptions

WISHBONE Erroneous Address Register – W_ERR_ADDR

Width	Access	Reset	Description
32	R	0x00000000	When error reporting is enabled and error is signaled, this register stores address of transaction on PCI bus that caused an error.

Table 4-25: WISHBONE Erroneous Address Register

WISHBONE Erroneous Data – W_ERR_DATA

Width	Access	Reset	Description
32	R	0x00000000	When error reporting is enabled and error is signaled, this register stores data of transaction on PCI bus that caused an error.

Table 4-26: WISHBONE Erroneous Data Register

4.1.3.2 PCI Target Unit Error Reporting Registers

PCI Error Control and Status Register – P_ERR_CS

Width	Access	Reset	Description
32	RW	0x00000000	Part of register is used for controlling Error Reporting mechanism, another part is used for reporting statuses and additional information about an error that occurred during completion of posted write on WISHBONE bus.

Table 4-27: PCI Error Control and Status Register

Register layout:

31 – 28	27 – 24	23 – 16	15 – 12	11	10	9	8	7 – 4	3	2	1	0
BE	BC	0x00	0x0	0	RTY_EXP	0	ERR_SIG	0x0	0	0	0	ERR_EN

Figure 4-13: PCI Error Control and Status Register layout

Bits descriptions:

Bit number	Name	Description
31 – 28	Byte Enables	Field value reports the state of byte enable signals SEL_O(3:0) that was used in Posted Write which terminated with an error on WB bus.
27-24	Bus Command	Field value reports bus command from PCI bus that was used for Posted Write which terminated with an error on WB bus.
16 – 11	N/A	Not used
10	Retry Counter Expired	Reports that posted write was Retried *MAX_RETRY* times.
9	N/A	Not used
8	Error Signaled	Set bit indicates that error is being reported. While this bit is set all PCI target unit's operation is frozen. Software must clear this bit to enable transactions to resume their path through PCI target unit. Bit is cleared by writing 1 to its location.
7-1	N/A	Not used
0	Error Enable	Setting this bit enables Error Reporting mechanism. Clear bit means that Error Reporting will not be performed – Transaction that caused an error is discarded, other transactions continue normally.

Table 4-28: PCI Error Control and Status Register bits descriptions

PCI Erroneous Address Register – P_ERR_ADDR

Width	Access	Reset	Description
32	R	0x00000000	When error reporting is enabled and error is signaled, this register stores address of transaction on WISHBONE bus that caused an error.

Table 4-29: PCI Erroneous Address Register

PCI Erroneous Data – P_ERR_DATA

Width	Access	Reset	Description
32	R	0x00000000	When error reporting is enabled and error is signaled, this register stores data of transaction on WISHBONE bus that caused an error.

Table 4-30: PCI Erroneous Data Register

4.1.3.3 Configuration Cycle Generation Registers

Two registers are provided for generating configuration cycles on PCI bus. WISHBONE master initiates configuration cycle in two steps:
Writes appropriate value in CNF_ADDR register
Reads from or Writes to CNF_DATA register to generate configuration read or write cycle respectively.

Configuration Address – CNF_ADDR

Width	Access	Reset	Description
32	RW	0x00000000	Register stores all information needed to drive address lines during Address phase of Configuration Cycle (e.g. used within a host PCI device).

Table 4-31: Configuration Address Register

Register layout:

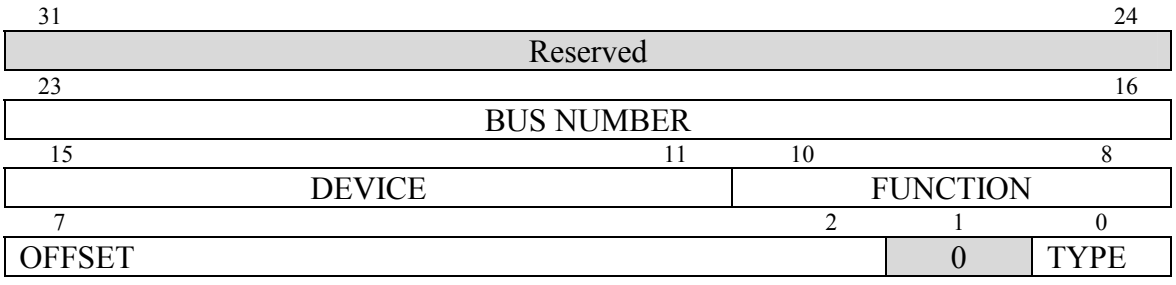

Figure 4-14: Configuration Address Register layout

Bits descriptions:

Bit number	Name	Description
31 – 24	N/A	Value in this field is ignored for any kind and type of Configuration Cycle
23 – 16	Bus number	Field holds a bus number on which a target of Configuration Space access resides. Only used in Type 1 Configuration Cycles (TYPE bit = 1).
15 – 11	Device number	Value in this field represents a device number. This field is driven directly to AD(15:11) lines during Address phase for Type1 (TYPE = 1) configuration cycle and is decoded for Type0 configuration cycle (See Table XY for Device number decoding).
10 – 8	Function number	Value in this field is function number for multifunction devices.
7 –2	Register number	Register offset for a device addressed with Configuration Cycle.
1	N/A	Not used – always 0
0	Type	Type of configuration cycle (0 – Type 0, 1 – Type 1)

Table 4-32: Configuration Address Register bits descriptions

Configuration Data – CNF_DATA
Read from or Write to this register will perform Configuration Cycle on PCI bus using information written to CNF_ADDR register.

Width	Access	Reset	Description
32	RW	0x00000000	Register stores read or write data for Configuration Cycles.

Table 4-33: Configuration Data Register

4.1.3.4 Interrupt Acknowledge Cycle Generation Register

Read from INT_ACK register generates interrupt acknowledge cycle on PCI bus.

Width	Access	Reset	Description
32	R	0x00000000	Register stores interrupt vector data returned during Interrupt Acknowledge cycle.

Table 4-34: Interrupt Acknowledge Register

4.1.4 Interrupt Control & Status Registers

Interrupt Control Register - ICR

Width	Access	Reset	Description
32	RW	0x00000000	Register is used to enable/disable interrupt request generation from various sources.

Table 4-35: Interrupt Control Register

Register layout:

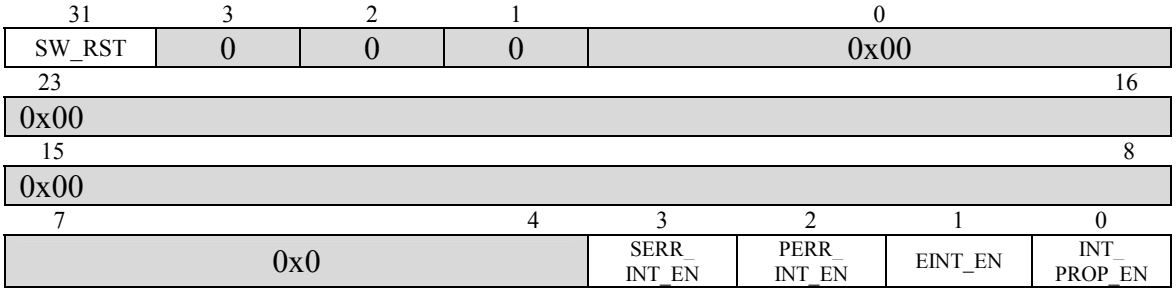

Figure 4-15: Interrupt Control Register layout

Bits descriptions:

Bit number	Name	Description
0	Interrupt Propagation Enable	For Guest bridge implementation this bit indicates, that INT_I line assertion on WISHBONE bus will generate interrupt request on PCI bus through assertion of INTA# pin. For Host bridge implementation this bit indicates that assertion of INTA# pin on PCI bus will generate interrupt request on WISHBONE bus through assertion of INT_O pin.
1	Error Interrupt Enable	If set, this bit enables interrupt request generation when error is signaled during execution of Posted Write through WISHBONE slave unit. Cleared bit disables this interrupts, but doesn't disable error reporting.
2	Parity Error Interrupt enable	Bit enables/disables interrupt request generation when parity error is detected by PCI MASTER module. This interrupt has meaning only on Host bridge implementation.*
3	System Error Interrupt Enable	Bit enables/disables interrupt request generation when system error (address parity error) is detected by PCI MASTER module. This interrupt has meaning only on Host bridge implementation.*

Bit number	Name	Description
31	Software Reset	Setting this bit causes software initiated reset on WISHBONE bus. It can be used to hold processor on add-in board in reset, while operating system or drivers are downloaded to add-in board's memory. This bit has meaning only in Guest bridge implementation.

* Interrupt triggering upon PERR# and SERR# detection for Guest implementation has no meaning because Guest implementation triggers interrupts on PCI bus. Agent that is responsible for routing interrupts to host processor may trigger interrupt when one of this errors is detected.

Table 4-36: Interrupt Control Register bits descriptions

Interrupt Status Register – ISR

Width	Access	Reset	Description
32	R(W*)	0x00000000	Register is used for identifying and clearing sources of interrupts.

* Interrupt status bits are cleared with writing ones to their location.

Table 4-37: Interrupt Status Register

Register layout:

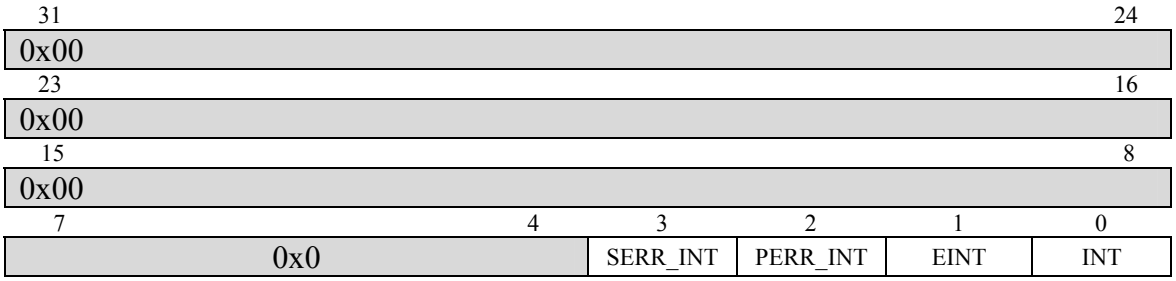

Figure 4-16: Interrupt Status Register layout

Bits descriptions:

Bit number	Name	Description
0	Interrupt	For Guest bridge implementation this bit indicates, that INT_I input on WISHBONE bus was asserted and it propagated to INTA# pin on PCI bus. That means some device on WISHBONE bus generated interrupt request to host processor. For Host bridge implementation this bit indicates that INTA# pin on PCI bus was asserted and it propagated to INT_O pin on WISHBONE bus. That means some device residing on PCI bus generated interrupt request to host processor.
1	Error Interrupt	If set, this bit indicates interrupt request from error reporting mechanism, which detected error during execution of Posted Write through WISHBONE slave unit.
2	Parity Error Interrupt	Bit indicates that interrupt request was generated due to Parity Error on PCI bus. This interrupt has meaning only on Host bridge implementation.*
3	System Error Interrupt Enable	Bit indicates that interrupt request was generated due to System Error (Address Parity) on PCI bus. This interrupt has meaning only on Host bridge implementation.*

* Interrupt triggering upon PERR# and SERR# detection for Guest implementation has no meaning because Guest implementation triggers interrupts on PCI bus. In Guest implementation this two bits will never be set.

Table 4-38: Interrupt Status Register bits descriptions

5

IO Ports

5.1 PCI Interface

PCI interface has required and also optional pins and all of them are organized in the functional groups. Required pins must be implemented. There is also a description of implemented optional pins (needed for requested features).

5.1.1 Required PCI Interface Pins

Name	Size	Direction	Description
AD	32	I/O	Multiplexed address and data bus (little endian).
C/BE#	4	I/O	Indicates PCI Command during address phase, indicates Byte Enables during data phases.
PAR	1	I/O	Parity bit.

Table 5-1: PCI address and data pins

Name	Size	Direction	Description
FRAME#	1	I/O	Start and end of a transaction.
IRDY#	1	I/O	Initiator ready (assertion of signal indicates that Initiator is ready to send or receive data).
DEVSEL#	1	I/O	Device selected (when target recognizes its address on the bus, it asserts this signal to claim the transaction).
TRDY#	1	I/O	Target ready (assertion of signal indicates that Target is ready to send or receive data).
STOP#	1	I/O	Used by target to signal various terminating conditions.
IDSEL	1	I	Individual device select (for configuration; unique IDSEL line per agent).

Table 5-2: PCI interface control pins

Name	Size	Direction	Description
PERR#	1	I/O	Parity error.
SERR#	1	I/O	System error.

Table 5-3: PCI error reporting pins

Name	Size	Direction	Description
REQ#	1	O	Asserted by Initiator to request for bus ownership.
GNT#	1	I	Asserted by Arbiter to grant the bus ownership.

Table 5-4: PCI arbitration pins (INITIATOR only)

Name	Size	Direction	Description
CLK	1	I	PCI input clock (signals are sampled on the rising edge of the clock).
RST#	1	I/O	Asynchronous reset (PCI device must tri-state all signals during reset).

Table 5-5: PCI system pins

5.1.2 Implemented Optional PCI Interface Pins

Name	Size	Direction	Description
INTA#	1	O	Asserted by Initiator to request an interrupt.

Table 5-6: PCI interrupt pin

Name	Size	Direction	Description
M66EN	1	I	Mode 66 MHz Enable (indicates to a device whether the bus segment is operating at 66 or 33 MHz).
CLKRUN#	1	I/O/Z	Clock running (The central resource request permission to stop or slow **CLK**. The central resource must provide the pullup for **CLKRUN**#.).
PME#	1	O	Power management Event (can be used by a device to request a change in the device or system power state. The assertion and deassertion of **PME#** is asynchronous to **CL**K.).

Table 5-7: PCI interface control pins

5.2 WISHBONE Interface

The SoC interface is a WISHBONE Rev B compliant interface. PCI IP core's WISHBONE slave unit is connected to WISHBONE bus as a slave while PCI target unit connects to WISHBONE bus as a master.

Name	Size	Direction	Description
ADDR_O	32	O	Address output
MDATA_I	32	I	Data input
MDATA_O	32	O	Data output
SEL_O	4	O	WE_O asserted – indicates valid bytes on MDATA_O bus. WE_O deasserted – indicates which bytes must be supplied by slave on MDATA_I bus.
WE_O	1	O	Write enable – indicates write cycle when asserted high and read cycle when low
CYC_O	1	O	Encapsulates a valid transfer cycle
STB_O	1	O	Indicates a valid transfer to the slave
ACK_I	1	I	Acknowledgment input – slave signals a normal cycle termination
ERR_I	1	I	Slave signals abnormal cycle termination.
RTY_I	1	I	Slave signals that the interface is not ready and that the master should retry the operation
CAB_O	1	O	Indicates to the slave that consecutive address block transfer is in progress.

Table 5-8: PCI target unit's WISHBONE interface (MASTER)

Name	Size	Direction	Description
ADDR_I	32	I	Address input
SDATA_I	32	I	Data input
SDATA_O	32	O	Data output
SEL_I	4	I	WE_O asserted – indicates valid bytes on MDATA_I bus. WE_O deasserted – indicates which bytes must be supplied on MDATA_O bus.
WE_I	1	I	Write enable – indicates write cycle when asserted high and read cycle when low
CYC_I	1	I	Encapsulates a valid transfer cycle
STB_I	1	I	Indicates a valid transfer to the slave
ACK_O	1	O	Acknowledgment output – slave signals a normal cycle termination
ERR_O	1	O	Slave signals abnormal cycle termination.

Name	Size	Direction	Description
RTY_O	1	O	Slave signals that the interface is not ready and that the master should retry the operation
CAB_I	1	I	Master signals consecutive address block transfer is in progress when 1

Table 5-9: WISHBONE slave unit's WISHBONE interface (SLAVE)

Name	Size	Direction	Description
CLK_I	1	I	Clock input (application side clock)
RST_I	1	I	Reset input (application side reset)
RST_O	1	O	Used for propagating RST# from PCI bus to application side of the bridge. Also used for initiating software reset.
INTA_O(*)	1	O	Interrupt output
INTA_I(*)	1	I	Interrupt input

(*) – these two signals will never be used at the same time. Guest implementation of the core will signal interrupts to PCI bus, so only INTA_I is used. Host bridge implementation will signal interrupts to WISHBONE bus, so INTA_O is used.

Table 5-10: WISHBONE common control and system Ios

6

Waveforms

6.1 Wishbone Slave Unit

This section describes basic waveforms of various accesses to core's configuration space and mapped PCI address space. Waveforms supplied have only informational purpose at this time.

6.1.1 WISHBONE Configuration Accesses

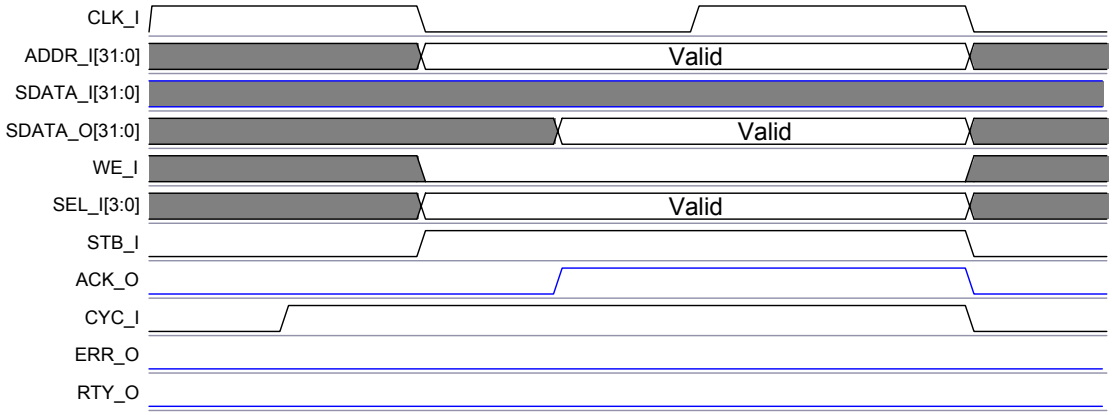

Figure 6-1: WISHBONE configuration read

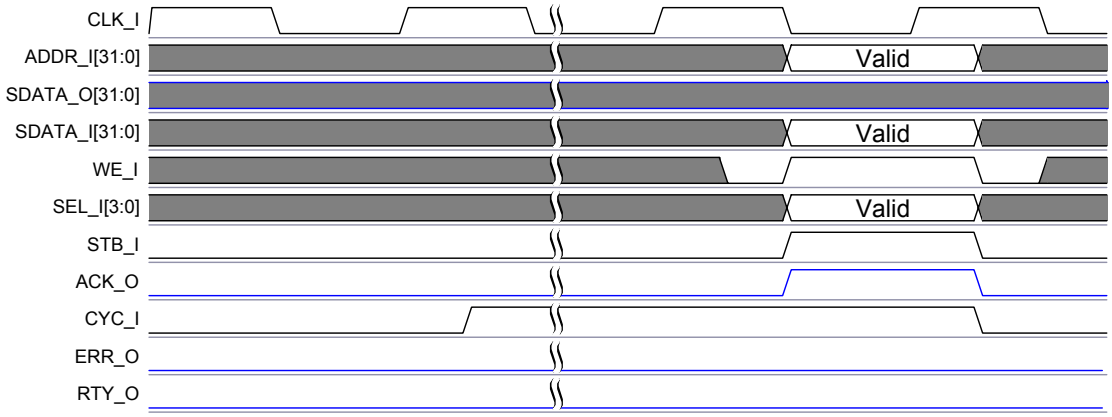

Figure 6-2: WISHBONE configuration write

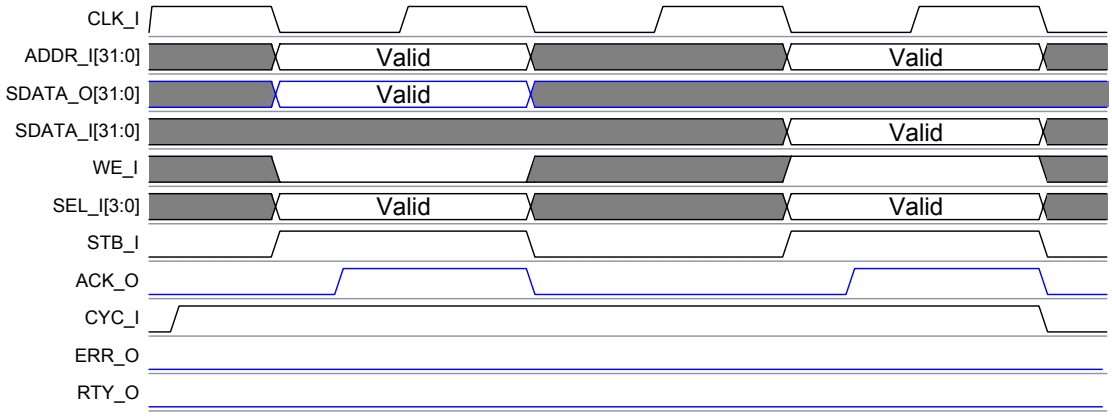

Figure 6-3: WISHBONE configuration RMW cycle

Wishbone masters will most commonly use single read cycles for accessing core's configuration space as shown in Figure 6-1. Write to core's register space by WISHBONE master is shown in Figure 6-2. Writes to unimplemented configuration space have no effect while reads return all 0s. RMW cycles to core's configuration space are also accepted like Figure 6-3 shows and are most commonly used for interrupt handling since RMW cycle is defined as atomic (indivisible) operation in WISHBONE bus specification.

6.1.2 WISHBONE to PCI Accesses

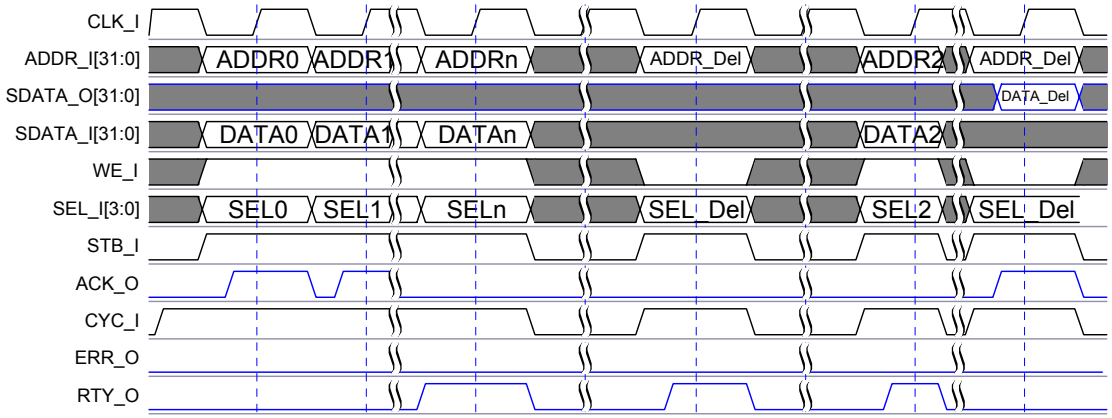

Figure 6-4: WISHBONE access to PCI address space

Figure 6-4 shows how WISHBONE master sees cycles intended for PCI address space traveling through WISHBONE slave unit of the core. The first cycle in the figure initiated by WISHBONE master is a block write. WISHBONE slave module accepts writes until WBW_FIFO is full. Subsequent writes in this block cycle are terminated with retry (RTY_O asserted on ADDRn, DATAn, SELn transfer). Second cycle in the figure is a read. Reads from PCI address space are retried immediately (RTY_O asserted on first ADDR_Del, SEL_Del transfer). Address, byte enable and CAB_I information is latched

by WISHBONE slave unit on the first rising edge of CLK_I where STB_I is asserted. The third cycle is a write to PCI address space and is also retried. In this case, there is more than one possibility for WISHBONE slave unit to signal a retry:

- WBW_FIFO is still full from previous transfers
- Delayed read latched in previous transfer has not completed on PCI bus yet
- There is delayed read completion present in PCI target unit and hasn't been completed on PCI bus yet

In 4th cycle, WISHBONE master retries a read request initiated and latched by WISHBONE slave module in 2nd cycle. Since PCI master module already performed a read on PCI bus and stored data in WBR_FIFO, WISHBONE slave module takes data from the FIFO and supplies it on WISHBONE bus. WISHBONE slave module can supply data for the master as long as WBR_FIFO contains any data and read addresses are serial and DWORD aligned.

6.1.3 PCI Cycles

WISHBONE slave unit incorporates a PCI master module, which is capable of initiating various types of PCI address space accesses.

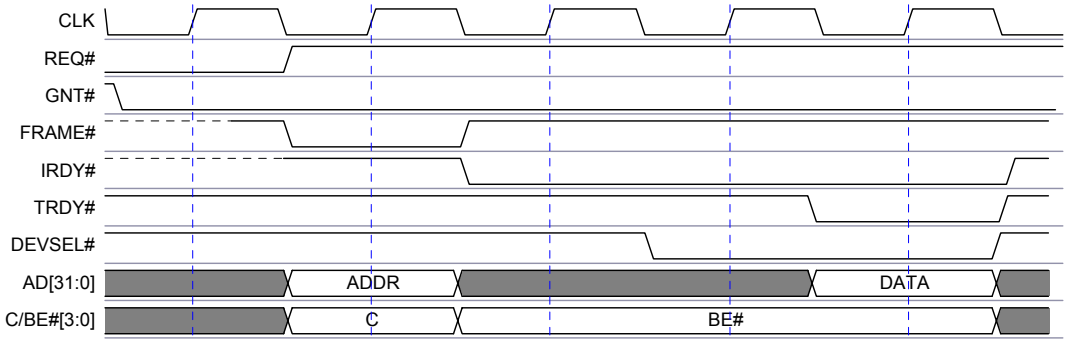

Figure 6-5: PCI single read

Figure 6-5 shows a single read on PCI bus performed by PCI master module. On the first clock edge PCI master module samples it's GNT# signal asserted and it claims the bus cycle by asserting FRAME# on the next rising edge of clock. 2nd clock cycle is also an address phase, so address and bus command information is provided on ADDR and C/BE# lines respectively. At the end of address phase master module de-asserts FRAME# and asserts IRDY# indicating that it only wishes to perform a single data phase. A device with medium decode was assumed for a diagram, so nothing happens on 3rd rising edge of clock. On 4th clock targeted device claims an access by asserting DEVSEL#. This clock cycle is used as Turnaround cycle (target starting to drive AD lines) inserted by delaying assertion of TRDY#. On 5th clock actual data transfer occurs indicated by TRDY# and IRDY# being asserted at the same time. Immediately after that master module de-asserts IRDY# indicating end of transfer.

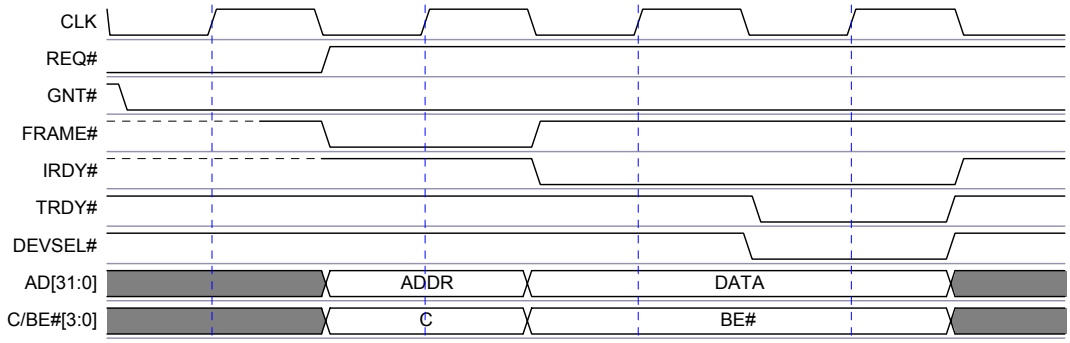

Figure 6-6: PCI single write

Figure 6-6 shows a single write on PCI bus performed by PCI master module. On the first clock edge PCI master module samples it's GNT# signal asserted and it claims the bus cycle by asserting FRAME# on the next rising edge of clock. 2nd clock cycle is also an address phase, so address and bus command information is provided on ADDR and C/BE# lines respectively. At the end of address phase master module de-asserts FRAME# and asserts IRDY# indicating that it only wishes to perform a single data phase. With assertion of IRDY#, write data and byte enables must be driven on AD and C/BE# lines respectively. A device with medium decode was assumed for a diagram, so nothing happens on 3rd rising edge of clock. On 4th clock targeted device claims an access by asserting DEVSEL#. On this clock actual data transfer occurs also, indicated by TRDY# and IRDY# being asserted at the same time. Immediately after that master module de-asserts IRDY# indicating end of transfer.

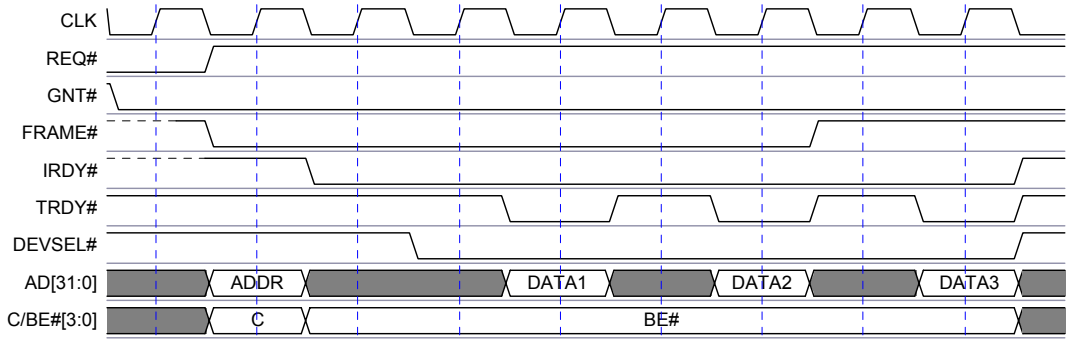

Figure 6-7: PCI burst read

Figure 6-7 shows how PCI master module performs burst read transactions. The mechanism for claiming the bus is the same as in previous diagrams. The first difference is, that in burst transfers FRAME# stays asserted until last data transfer. Medium decode target device is assumed for diagram which inserts Turnaround cycle on clock 4. Target also inserts one WS after each data phase. Byte enables don't change during bursts – they are always 0000. Last data phase is phase 3, which is indicated by FRAME# de-asserted and IRDY# asserted at the same clock edge. Immediately after master module latches data from the bus (clock edge when TRDY# is asserted), it de-asserts IRDY# to indicate end of transfer.

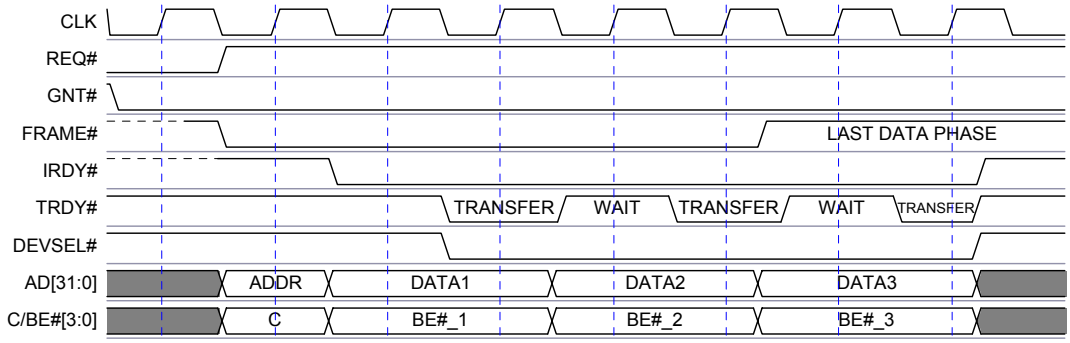

Figure 6-8: PCI burst write

Figure 6-8 shows PCI burst writes performed by PCI master module. The mechanism for claiming the bus is the same as in previous diagrams. FRAME# stays asserted until last data transfer. Medium decode target device is assumed for diagram which claims an access and latches first data beat on clock 4. Target also inserts one WS after each data phase. Last data phase is phase 3, which is indicated by FRAME# de-asserted and IRDY# asserted at the same clock edge. Immediately after target latches data from the bus (clock edge when TRDY# is asserted), master module de-asserts IRDY# to indicate end of transfer.

6.1.4 PCI Terminations

6.1.4.1 Master Initiated Terminations

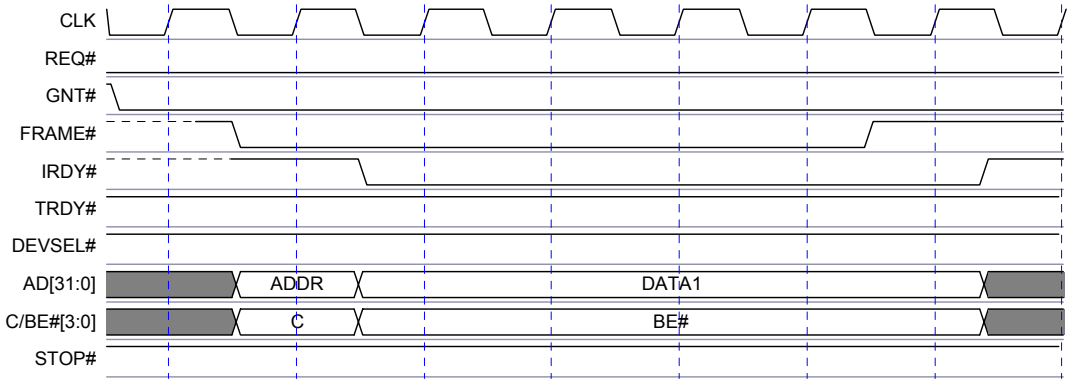

Figure 6-9: Master Abort termination

PCI master module terminates the transaction with Master Abort shown in Figure 6-9. What happens? Master initiates transaction with address phase and is waiting for target to respond by asserting DEVSEL#. Master is only required to wait for assertion of DEVSEL# for 4 clocks. DEVSEL# wasn't asserted until 4th clock (subtractive decode devices). Master de-asserts FRAME# and must hold IRDY# asserted for additional clock cycle indicating end of transaction.

If Error Reporting is enabled and transaction is a posted write, address, bus command, data and byte enables are stored in corresponding registers (See chapter 3.2.3) and WISHBONE slave unit locks out all but configuration space accesses until proper Error Status bit is cleared. Current transaction is discarded (pulled out of WBW_FIFO), while any other posted writes are not influenced by error.
If transaction is a read, the termination will be signaled to WISHBONE master with error on WISHBONE bus, when it retries a Read request.

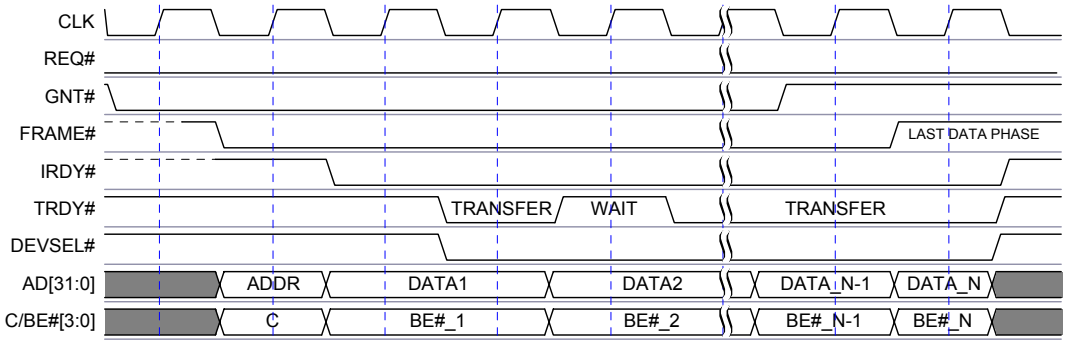

Figure 6-10: Timeout termination

Timeout termination is specified in PCI Local Bus Specification. It must be implemented in PCI master module. Timeout termination is not abnormal termination, it's just a mean of assuring access to PCI bus for other masters in reasonable amount of time. Master is supposed to complete the transaction when its latency timer expires and its GNT# has been removed by PCI arbiter. In other words – after master latency timer has expired, PCI master module must sample its GNT# on every rising edge of clock. If it samples it de-asserted, it must complete the transaction as soon as possible. In Figure 6-10, it is assumed that on data phase N-1 master's latency timer has expired and as drawn, its grant has been removed. Master module samples GNT# de-asserted, so it completes an access on the next clock cycle by de-asserting FRAME#.
Timeout terminations are not signaled to WISHBONE bus since PCI master module can resume transaction next time it gains bus mastership.
Timeout detection is implemented with Master Latency Timer register in PCI configuration space and a counter. Counter is enabled when PCI master module asserts FRAME# and is cleared and suspended as soon as FRAME# is de-asserted.

6.1.4.2 Target Terminations Handled by PCI Master Module

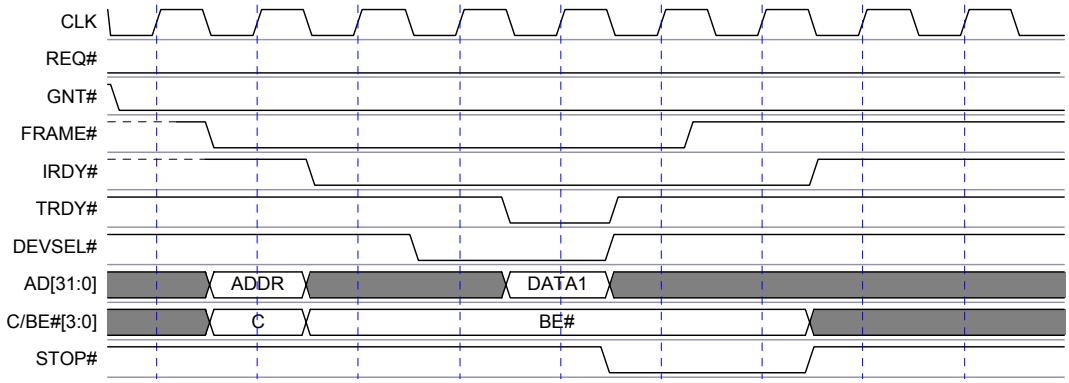

Figure 6-11: Target Abort

Target signals Target Abort termination to the master when it is and will be unable to complete the access initiated by the master. That means that master should not attempt to retry accesses terminated with Target Abort.
Posted writes terminated with Target Abort are discarded. If Error Reporting is enabled, WISHBONE slave unit reports an error and locks out any non-configuration space accesses until corresponding error status bit is cleared.
Target Abort termination during read cycles is signaled to WISHBONE master when it retries the request. Access to the address, which resulted in Target Abort, is terminated with an error on WISHBONE bus. If WISHBONE master never accesses address that resulted in Target Abort, termination won't be signaled in any way (Target Abort can be signaled because PCI master module reads over address space boundaries of specific target during pre-fetched read so WISHBONE master will never perform a read to that address).

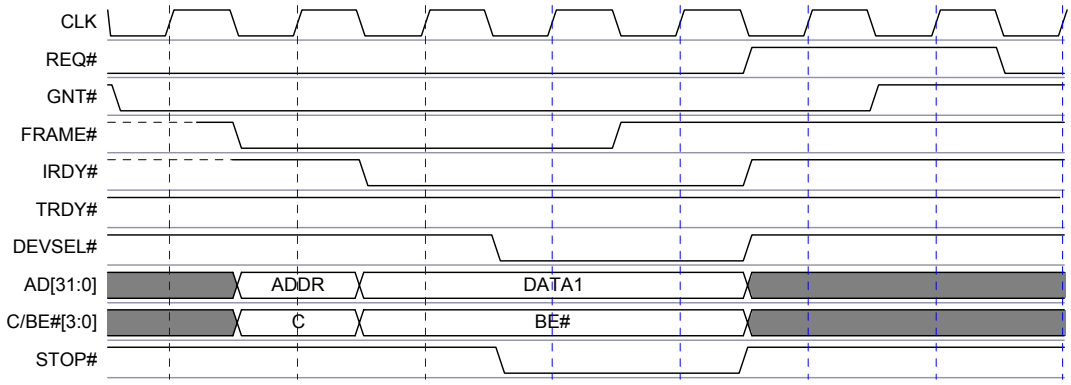

Figure 6-12: Target Retry

Target signals a Retry to the master when it is not ready to process the request. No data is transferred during Retry nevertheless, PCI master must still terminate normally – by de-asserting FRAME# and keeping IRDY# asserted for one PCI clock cycle indicating last data phase. Master must relinquish PCI bus for at least two cycles after it receives a Target Retry by de-asserting its REQ# line. It must also retry the same request at the later time.

Target Retry is not signaled on WISHBONE bus – PCI master module retries the transaction transparently on PCI bus.

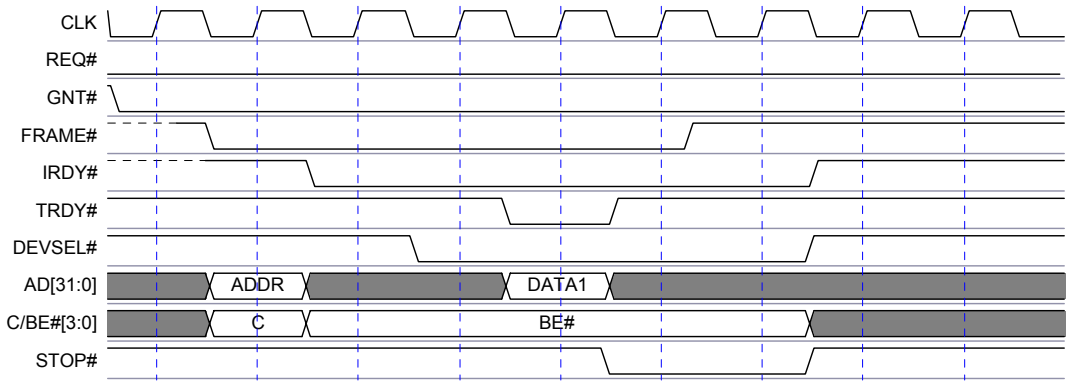

Figure 6-13: Target Disconnect without data

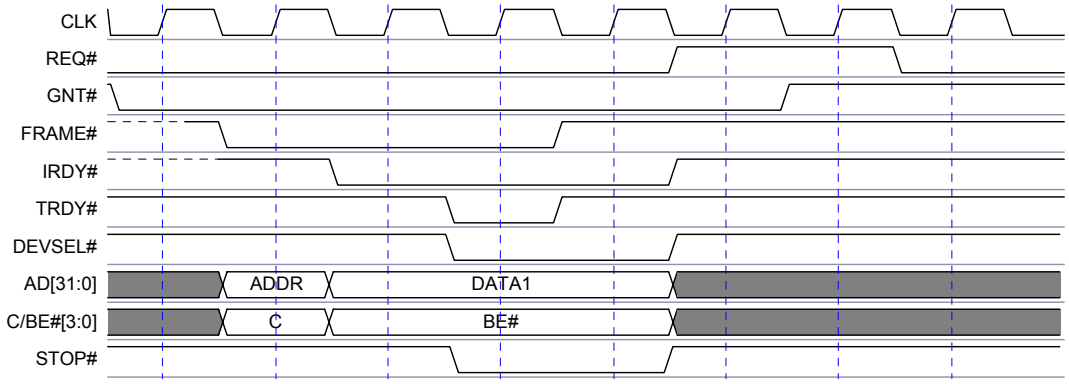

Figure 6-14: Target Disconnect with data

Target signals Target Disconnect to the master when it is not capable of receiving or supplying any more data from/to Master. Data must be transferred with (Disconnect with data) or before (Disconnect without data) Target signals disconnect. Master must terminate the transaction normally – de-asserting FRAME# and keeping IRDY# asserted for one clock cycle. If target signals Target Disconnect with data on the last data phase (FRAME# de-asserted, IRDY#, TRDY# and STOP# asserted) then termination is treated as normal Master termination. (e.g. STOP# is logical don't care for a Master when FRAME# is de-asserted and IRDY# and TRDY# are asserted).
Target Disconnect is not abnormal termination and it won't be signaled to WISHBONE master in any way.

6.2 PCI Target Unit

This section describes basic waveforms of various accesses to core's configuration space and mapped WISHBONE address space. Waveforms supplied have only informational purpose at this time.

6.2.1 PCI Configuration Accesses

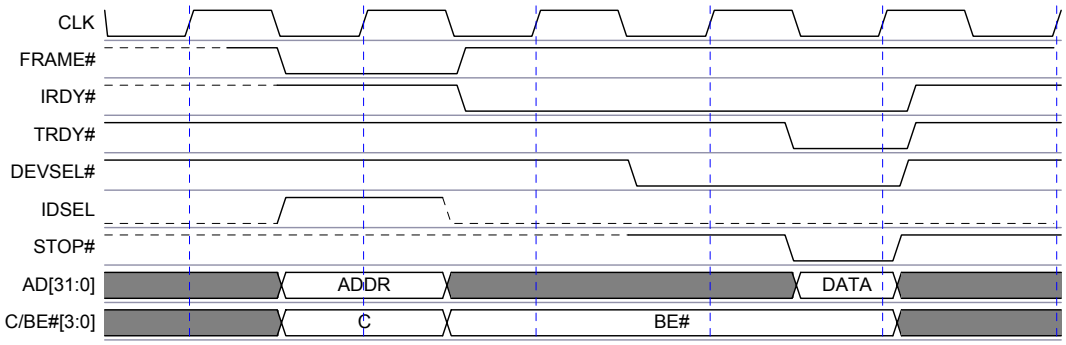

Figure 6-15: PCI configuration read

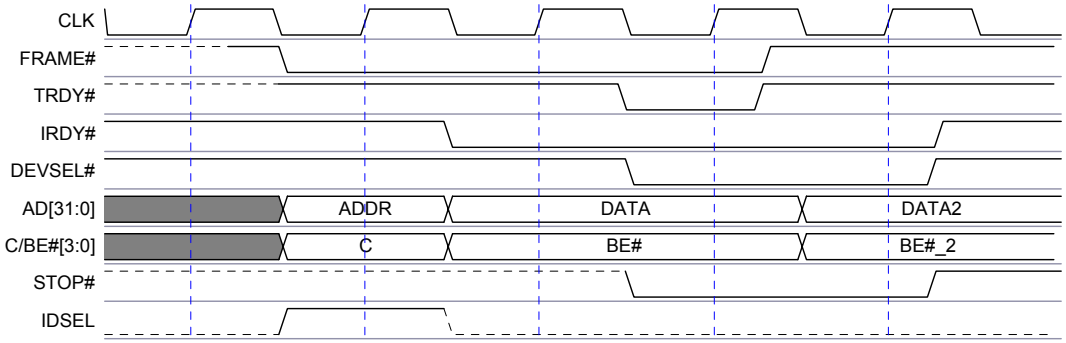

Figure 6-16: PCI configuration write

PCI initiators will most commonly use single read cycles for accessing core's configuration space as shown in Figure 6-15. Write to core's register space by PCI initiator is shown in Figure 6-16. Writes to unimplemented configuration space have no effect while reads return all 0s.

6.2.2 PCI to WISHBONE Accesses With WISHBONE Cycles

Following figures show how PCI initiator sees cycles intended for WISHBONE address space traveling through PCI target unit of the core. The first cycle in the Figure 6-17 initiated by PCI initiator is a delayed read command. PCI target module accepts read command. Subsequent reads in this cycle are terminated with retry. Next figure shows previous transaction transferred to WISHBONE bus. Second cycle in the first figure is a read from PCI master.

For reference, there are also bursts accesses from PCI through PCI target module (read and write) on Figure 6-19 and on Figure 6-20. Last follows a diagram of a write transfer on WISHBONE bus which was initiated by PCI initiator.

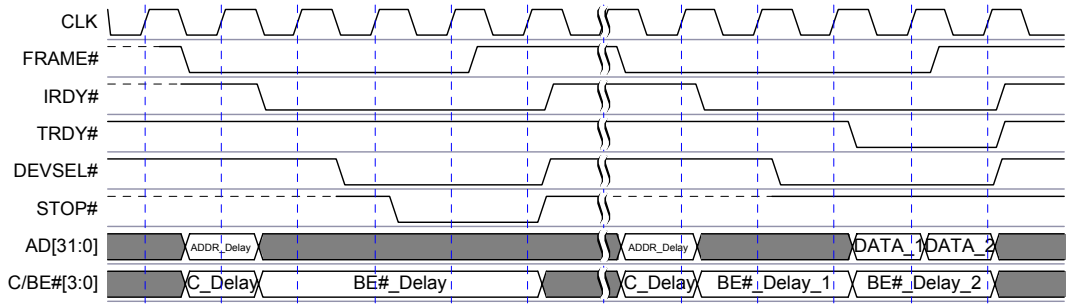

Figure 6-17: PCI target read

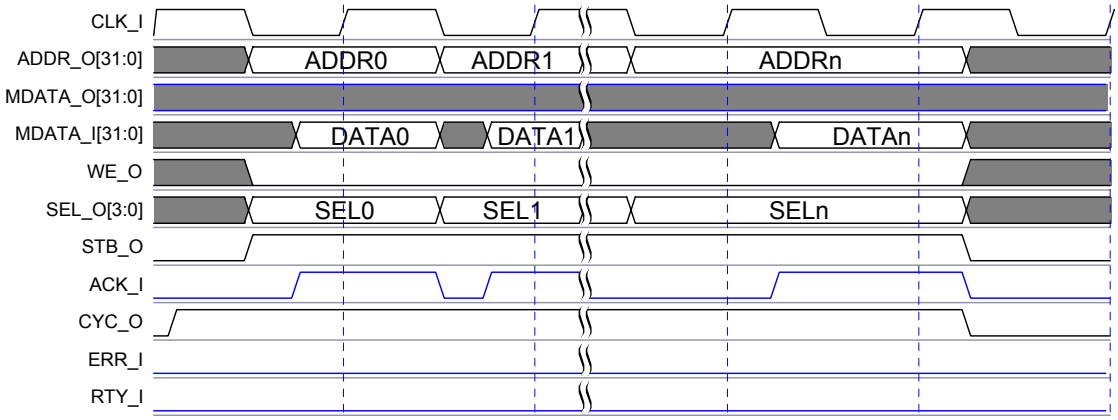

Figure 6-18: PCI to WISHBONE read

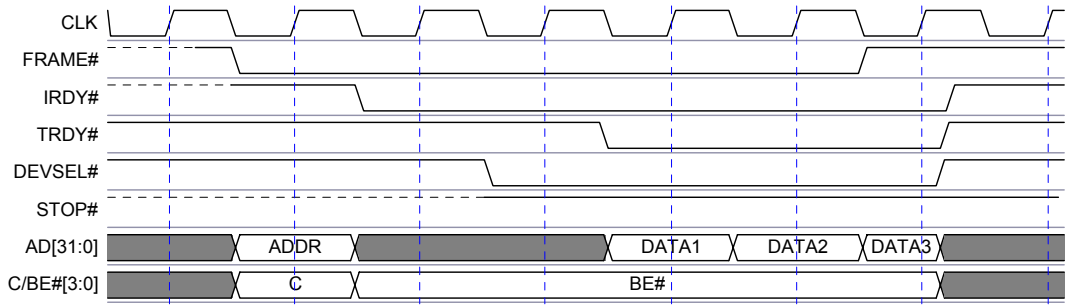

Figure 6-19: PCI initiator to target burst read

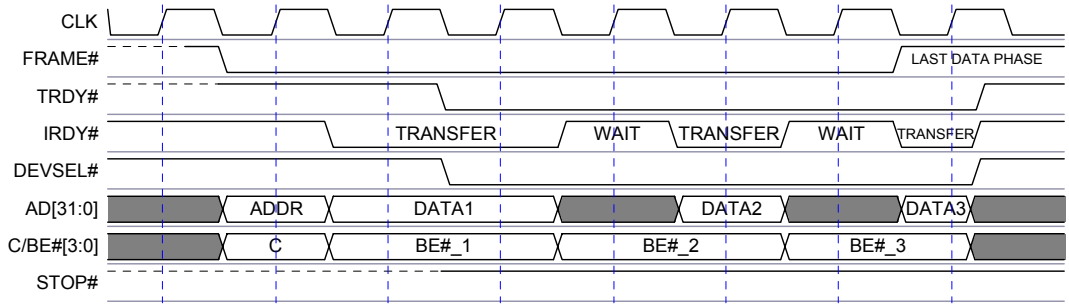

Figure 6-20: PCI initiator to target burst write

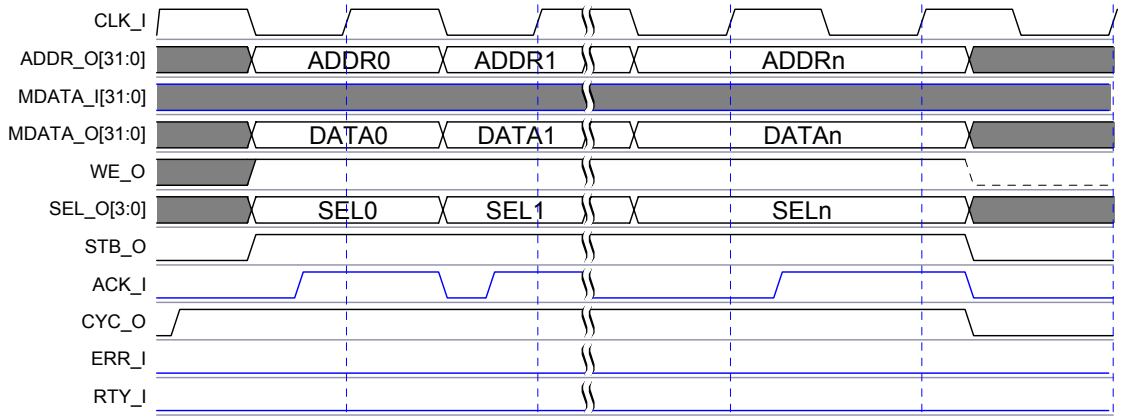

Figure 6-21: WISHBONE write transfer caused by PCI to WISHBONE write

6.2.3 WISHBONE Terminations

Terminations on WISHBONE bus are always performed by WISHBONE slaves. What a retry or error cause on WISHBONE bus is described in chapters 3.3.3 and 3.3.4.

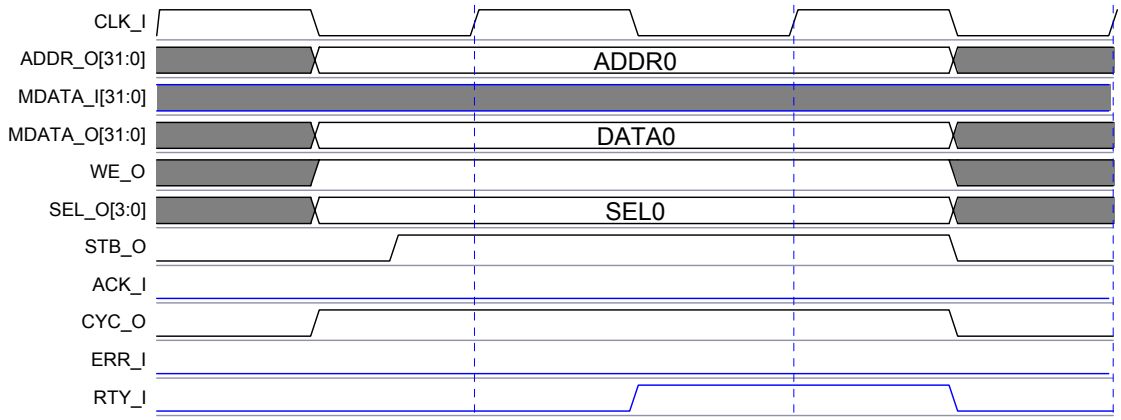

Figure 6-22: Retry on WISHBONE bus caused by PCI to WISHBONE transfer

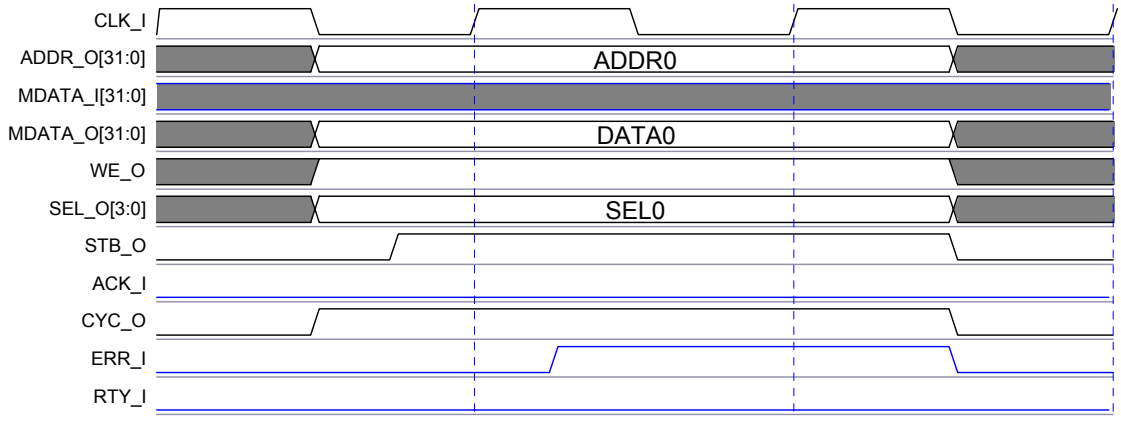

Figure 6-23: Error on WISHBONE bus caused by PCI to WISHBONE transfer

A

Core HW Configuration

This section describes parameters that are set by the user of the core and define configuration of the core. Parameters must be set by the user before actual use of the core in simulation or synthesis.
This chapter will be edited through the development phase.

A.1 HW Configuration parameters

Proposed parameters:

- FIFO depth – all four FIFOs have variable depth (WBW_DEPTH, WBR_DEPTH, PCIW_DEPTH, PCIR_DEPTH). FIFO depths depend on actual system implementation and are calculated by system designer
- HOST/GUEST – (`define HOST or `define GUEST). Defining one of these parameters has impact on configuration space accesses and routing of reset and interrupt pins. If GUEST is defined, Read/Write access to configuration space is available through PCI bus and Read Only access is provided for WISHBONE bus. WISHBONE bus interrupts are accepted on INTA_I pin and are propagated to PCI bus with assertion of INTA# pin (if this is enabled of course). Reset on PCI (RST# asserted) is propagated to WISHBONE bus with assertion of RST_O pin. If HOST is defined, rules are the other way around compared to definition of Guest: Read/Write to configuration space provided for WISHBONE bus and Read Only for PCI bus, interrupts are passed from PCI bus to WISHBONE bus (from INTA# pin on PCI to INTA_O pin on WISHBONE). Resets are passed from WISHBONE to PCI (from RST_I on WISHBONE to RST# on PCI).
- NUM_OF_PCI_IMGS – number of implemented images for PCI address space. This number can be from 1 to 5 (one is used for Configuration Space). Images are used for WISHBONE accesses to PCI bus.
- NUM_OF_WB_IMGS – number of implemented images for WISHBONE address space. Number can be from 1 to 5 (one is always used for Configuration Space). Images are used for PCI accesses to WISHBONE bus.
- WB_CONF_SPC_BA – WISHBONE configuration space Base Address – must be set and is stored as Read Only value in WISHBONE Configuration Space BAR.
- PCI_CONF_SPC_BA – PCI configuration space Base Address. This parameter only has meaning if HOST is defined also and sets the Base Address of Configuration Space on PCI side of the bridge. If parameter is not set, then software running on WISHBONE side system takes responsibility of writing BA into appropriate register to provide Read Only access of Configuration Space for

other PCI agents. If GUEST is defined, this Base Address is set by other PCI agent performing Type 0 configuration cycle.