AMD Alchemy™ Solutions Au1500™ Processor PCI Bus Performance

Application Note

Revision: **30275A**
Issue Date: **April 2003**

Contacts

www.amd.com pcs.support@amd.com

Trademarks

1.0 Introduction

This document describes the performance characteristics of the PCI 2.2 bus controller integrated into the Au1500™ processor. This document assumes the reader is familiar with the PCI 2.2 specification as found in the *PCI Local Bus Specification Rev. 2.2* (see 6.0 "References").

2.0 PCI Bus Controller Overview

The Au1500™ processor features an integrated PCI 2.2 bus controller for connecting to external peripherals. The PCI bus controller is designed to support a 32-bit wide interface at 33MHz or 66MHz. The PCI controller can initiate master cycles and can also serve as a target for device-initiated master cycles into the SDRAM memory of the Au1500 processor.

The PCI bus controller supports a maximum of four loads and arbitration for five PCI devices including the Au1 core.

Two separate Au1500 application notes describe clocking schemes for the PCI bus controller and software techniques for utilizing the PCI bus (see 6.0 "References").

The general arrangement of the PCI bus controller is depicted below in Figure 1: "Au1500™ Processor's PCI Bus controller".

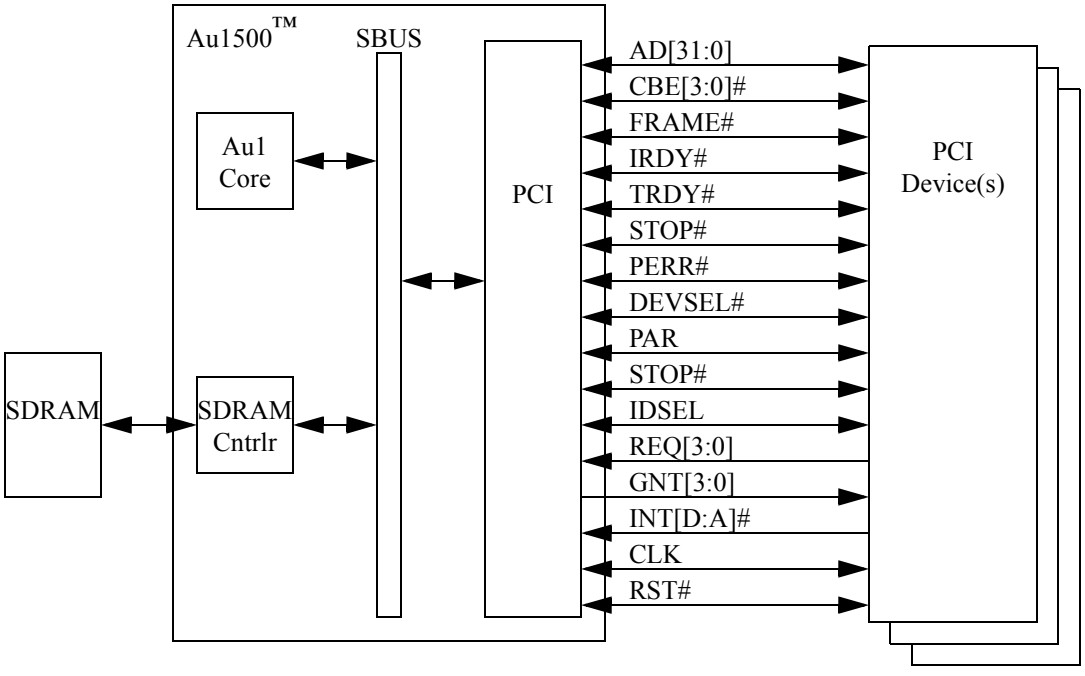

Figure 1: Au1500™ Processor's PCI Bus controller

PCI supports a variety of bus cycle types. This document focuses on the following six types:

- Au1 core initiated single-beat read from PCI
- Au1 core initiated single-beat write to PCI
- PCI device initiated single-beat read from Au1500 SDRAM
- PCI device initiated single-beat write to Au1500 SDRAM
- PCI device initiated burst read from Au1500 SDRAM
- PCI device initiated burst write to Au1500 SDRAM

The performance of the PCI bus is measured as throughput, the amount of data that can be transferred to and from the PCI bus in a given time period. These six PCI bus cycle types represent the significant majority of PCI bus cycles that occur in a running system. Thus, the throughput estimates of each cycle type can be combined to provide an overall estimate of PCI bus throughput.

3.0 PCI Bus Cycles Performance

In the examples below, the Au1 core is assumed to be operating at 396MHz, the system bus (SBUS) at 198MHz (5.1ns per SBUS clock), the PCI bus at 66MHz (15.2ns per PCI clock), and the SDRAM interface at 99MHz (10.1ns per SDRAM clock). The operating frequencies of the Au1 core, system bus, and SDRAM are controlled by the sys_cpupll and sys_powerctrl registers.

> **Note:** Further information on the SDRAM timings used in this document are available in the “Au1x00 SDRAM Performance” application note.

3.1 Au1 Core Initiated Single-Beat Read From PCI

The Au1 core initiates PCI single-beat reads with a load access, typically from PCI device registers or memory. Software frequently uses single-beat read accesses while managing the operation of a PCI device.

Au1 core accesses to PCI are non-cacheable, which in turn initiate single beat accesses.

> **Note:** The Au1 core can initiate cacheable load accesses to PCI address space via the cacheable memory window controlled by pci_cmem. Cacheable load accesses to PCI address space will initiate a burst read from PCI, not a single beat read.

An Au1 core initiated access traverses both the system bus and the PCI bus. The timing diagram of a PCI single-beat read is provided in “Figure 3-5: Basic Read Operation”, page 47, of the *PCI Local Bus Specification Rev. 2.2*. The activity of the buses during such an access is depicted below in Figure 2: “Au1 Core Initiated Single-Beat Read from PCI”.

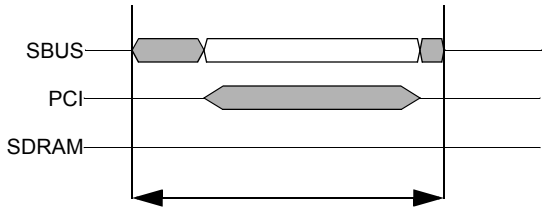

Figure 2: Au1 Core Initiated Single-Beat Read from PCI

The minimum time necessary for a PCI single-beat read is the sum of the active SBUS and PCI bus times: 5 SBUS clocks + 11 PCI clocks + 3 SBUS clocks (5 SBUS clocks for synchronization and arbitration, 6 PCI clocks for internal state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, 1 PCI clock for turnaround, 1 PCI clock for data, 1 PCI clock for state machine and 3 SBUS clocks to return data to the Au1 core and to complete the access). This yields 5*5.1ns + 11*15.2ns + 3*5.1ns or 208.0ns for an Au1 core initiated single-beat read access.

At 208.0ns per single-beat read, the theoretical maximum number of single-beat reads possible is 4,807,692 per second, which yields a theoretical maximum throughput of **19.2MB/s** (4,807,692 * 4 bytes). In reality, the timing of a single-beat read usually exceeds the minimum time outlined above for the reasons provided in 4.2 “Detrimental Influences on PCI Bus Performance”, in particular the assertion of DEVSEL# and TRDY#.

Furthermore, a PCI read must stall the Au1 core until data is returned. As illustrated above, the time can be considerable depending upon the ability of the PCI device to return data in a timely fashion.

3.2 Au1 Core Initiated Single-Beat Write To PCI

The Au1 core initiates PCI single-beat writes with a store access, typically to PCI device registers or memory. Software frequently uses single-beat write accesses while managing the operation of a PCI device. For graphics devices, write accesses dominate all other accesses due to the drawing of the frame buffer contents.

Au1 core accesses to PCI are non-cacheable, which in turn initiate single beat accesses.

Note: The Au1 core can initiate cacheable store accesses to PCI address space via the cacheable memory window controlled by pci_cmem. Cacheable store accesses to PCI address space do not initiate a burst write immediately; a burst write to PCI is initiated after the data cache casts out the corresponding cache line(s), or the corresponding cache line(s) are flushed.

Note: By programming the TLB with CCA=7 when mapping PCI memory spaces, the write buffer can gather Au1 core stores which in turn leads to more efficient burst writes into PCI memory space.

An Au1 core initiated access traverses both the system bus and the PCI bus. The timing diagram of a PCI single-beat write is provided in “Figure 3-6: Basic Write Operation”, page 48, of the *PCI Local Bus Specification Rev. 2.2*. The activity of the buses during such an access is depicted below in Figure 3: “Au1 Core Initiated Single-Beat Write to PCI”.

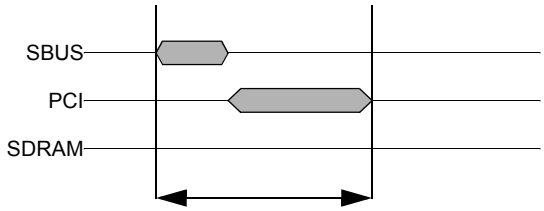

Figure 3: Au1 Core Initiated Single-Beat Write to PCI

The minimum time necessary for a PCI single-beat write is the sum of the active SBUS and PCI bus times: 5 SBUS clocks + 9 PCI clocks (5 SBUS clocks for synchronization, arbitration and start of PCI write access, 6 PCI clocks for state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, and 1 PCI clock for data). This yields 5*5.1ns + 9*15.2ns or 162.3ns for an Au1 core initiated single-beat write access.

At 162.3ns per single-beat write, the theoretical maximum number of single-beat writes possible is 6,161,429 per second, which yields a theoretical maximum throughput of **24.6MB/s** (6,161,429 * 4 bytes).

In reality, the timing of a single-beat write usually exceeds the minimum time outlined above for the reasons provided in 4.2 “Detrimental Influences on PCI Bus Performance”, in particular the assertion of DEVSEL# and TRDY#. However, the PCI write FIFO can reduce the single-beat timing, see 4.1.3 “PCI Write FIFO”.

3.3 PCI device initiated single-beat read from Au1500™ Processor SDRAM

A PCI device initiates single-beat reads of Au1500 processor SDRAM during its operation, typically while processing a DMA ring buffer or similar data structure.

A PCI device access to Au1500 SDRAM traverses the PCI bus, the system bus, and the SDRAM interface. The timing diagram of a PCI single-beat read is provided in “Figure 3-5: Basic Read Operation”, page 47, of the *PCI Local Bus Specification Rev. 2.2*. The activity of the buses during the access is depicted below in Figure 4: “PCI Device Initiated Single-Beat Read from Au1500™ Processor SDRAM”.

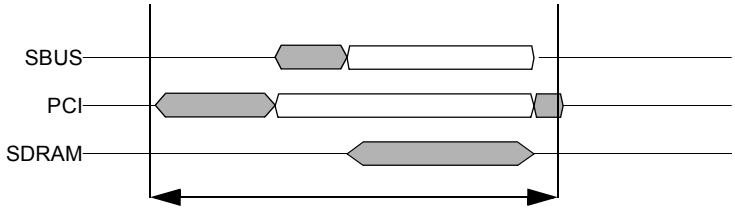

Figure 4: PCI Device Initiated Single-Beat Read from Au1500™ Processor SDRAM

The minimum time necessary for a PCI device single-beat read is the sum of the active PCI bus, SBUS and SDRAM times: 3 PCI clocks + 5 SBUS clocks + 6 SDRAM clocks + 1 PCI clock (1 PCI

clock for arbitration, 1 PCI clock for address/command, 1 PCI clock for turnaround, 5 SBUS clocks for synchronization and arbitration, 6 SDRAM clocks for a single-beat read, and 1 PCI clock for data to complete the access). This yields 4*15.2ns + 5*5.1ns + 6*10.1ns or 146.9ns for PCI device initiated single-beat read access to Au1500 SDRAM.

At 146.9ns per single-beat read, the theoretical maximum number of single-beat reads possible is 6,807,351 per second, which yields a theoretical maximum throughput of **27.2MB/s** (6,807,351 * 4 bytes). In reality, the timing of the access usually exceeds the minimum time outlined above for the reasons provided in 4.2 “Detrimental Influences on PCI Bus Performance”, in particular the arbitration for the system bus and PCI retries.

3.4 PCI device initiated single-beat write to Au1500™ Processor SDRAM

A PCI device initiates single-beat writes of Au1500 Processor SDRAM during its operation, typically while processing/updating a DMA ring buffer or similar data structure.

A PCI device access to Au1500 SDRAM traverses the PCI bus, the system bus, and the SDRAM interface. The timing diagram of a PCI single-beat write is provided in “Figure 3-6: Basic Write Operation”, page 48, of the *PCI Local Bus Specification Rev. 2.2*. The activity of the buses during the access is depicted below in Figure 5: “PCI Device Initiated Single-Beat Write to Au1500™ Processor SDRAM”.

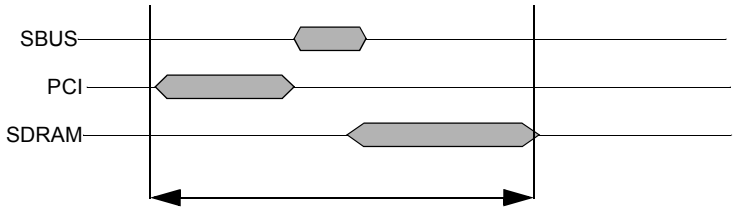

Figure 5: PCI Device Initiated Single-Beat Write to Au1500™ Processor SDRAM

The minimum time necessary for a PCI device single-beat write is the sum of the active PCI bus, SBUS and SDRAM times: 6 PCI clocks + 5 SBUS clocks + 6 SDRAM clocks (3 PCI clocks for state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, 1 PCI clock for data, 5 SBUS clocks for synchronization and arbitration, and 6 SDRAM clocks for a single-beat write). This yields 6*15.2ns + 5*5.1ns + 6*10.1ns or 177.3ns for PCI device initiated single-beat write access to Au1500 SDRAM.

At 177.3ns per single-beat write, the theoretical maximum number of single-beat writes possible is 5,640,157 per second, which yields a theoretical maximum throughput of **22.6MB/s** (5,640,157 * 4 bytes). In reality, the timing of the access usually exceeds the minimum time outlined above for the reasons provided in 4.2 “Detrimental Influences on PCI Bus Performance”, in particular the arbitration for the system bus and PCI retries.

Once the PCI data has been moved onto the system bus (on its way to the SDRAM), the next PCI bus cycle can be initiated, but if the access is to the Au1500 processor SDRAM, the cycle stalls until the

previous write completes. For example, PCI-to-PCI bus cycles can continue while data is transferred to Au1500 SDRAM.

3.5 PCI device initiated burst read from Au1500™ Processor SDRAM

A PCI device initiates burst reads of Au1500 processor SDRAM during its operation, typically while utilizing PCI bus-mastering DMA to transmit network packets or writing disk blocks. The burst transfers permit efficient movement of data and thus better performing I/O.

A PCI device access to Au1500 SDRAM traverses the PCI bus, the system bus, and the SDRAM interface. The timing diagram of a PCI burst read is provided in "Figure 3-5: Basic Read Operation", page 47, of the *PCI Local Bus Specification Rev. 2.2*. The activity of the buses during the access is depicted below in Figure 6: "PCI Device Initiated Burst Read from Au1500™ Processor SDRAM".

Figure 6: PCI Device Initiated Burst Read from Au1500™ Processor SDRAM

The minimum time necessary for a PCI device burst read of eight words from Au1500 SDRAM is the sum of the active PCI bus, SBUS and SDRAM times: 6 PCI clocks + 5 SBUS clocks + 12 SDRAM clocks + 8 PCI clocks (3 PCI clocks for state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, 1 PCI clock for turnaround, 5 SBUS clocks for synchronization and arbitration, and 12 SDRAM clocks for a burst read, and 8 PCI clocks for eight words of data). This yields 14*15.2ns + 5*5.1ns + 12*10.1ns or 359.5.1ns for PCI device initiated burst read access to Au1500 processor SDRAM.

At 359.5ns per eight word burst, the theoretical maximum number of burst reads possible is 2,781,641 per second, which yields a theoretical maximum throughput of **89.0MB/s** (2,781,641 * 32 bytes). In reality, the timing of the access usually exceeds the minimum time outlined above for the reasons provided in 4.2 "Detrimental Influences on PCI Bus Performance", in particular the arbitration for the system bus and PCI retries.

3.6 PCI device initiated burst write to Au1500™ Processor SDRAM

A PCI device initiates burst writes of Au1500 processor SDRAM during its operation, typically while utilizing PCI bus-mastering DMA to receive network packets or reading disk blocks. The burst transfers permit efficient movement of data and thus better performing I/O.

A PCI device access to Au1500 SDRAM traverses the PCI bus, the system bus, and the SDRAM interface. The timing diagram of a PCI burst write is provided in "Figure 3-6: Basic Write Operation",

page 48, of the *PCI Local Bus Specification Rev. 2.2*. The activity of the buses during the access is depicted below in Figure 7: “PCI Device Initiated Burst Write to Au1500™ Processor SDRAM”.

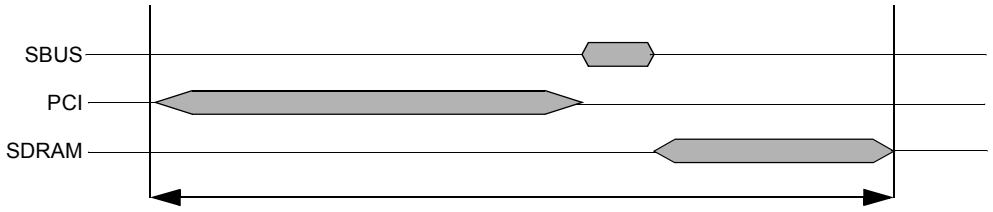

Figure 7: PCI Device Initiated Burst Write to Au1500™ Processor SDRAM

The minimum time necessary for a PCI device burst write of eight words to Au1500 SDRAM is the sum of the active PCI bus, SBUS and SDRAM times: 13 PCI clocks + 5 SBUS clocks + 12 SDRAM clocks (3 PCI clocks for state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, 5 SBUS clocks for synchronization and arbitration, and 12 SDRAM clocks for a burst write, and 8 PCI clocks for eight words of data). This yields 13*15.2ns + 5*5.1ns + 12*10.1ns or 344.3ns for PCI device initiated burst write access to Au1500 SDRAM.

At 344.3ns per eight word burst, the theoretical maximum number of burst writes possible is 2,904,443 per second, which yields a theoretical maximum throughput of **92.9MB/s** (2,904,443 * 32 bytes). In reality, the timing of the access usually exceeds the minimum time outlined above for the reasons provided in 4.2 “Detrimental Influences on PCI Bus Performance”, in particular the arbitration for the system bus and PCI retries.

Once the PCI data has been moved onto the system bus (on its way to the SDRAM), the next PCI bus cycle can be initiated, but if the access is to the Au1500 processor SDRAM, the cycle stalls until the previous write completes. For example, PCI-to-PCI bus cycles can continue while data is transferred to Au1500 SDRAM.

4.0 PCI Bus Performance

The overall throughput of the PCI bus interface dominated by the six PCI access cycles described previously. The maximum throughput of the PCI bus controller is approximated by this equation:

```
TP =(AUSBRTP * AUSBRR) + (AUSBWTP * AUSBWR) +
   (PDSBRTP * PDSBRR) + (PDSBWTP * PDSBWR) +
      (PDBRTP * PDBRR) + (PDBWTP * PDBWR)
```

where

- AUSBRTP is the Au1 core initiated single-beat read maximum throughput
- AUSBWTP is the Au1 core initiated single-beat write maximum throughput
- PDSBRTP is the PCI device initiated single-beat read from Au1500 processor SDRAM maximum throughput

- PDSBWTP is the PCI device initiated single-beat write to Au1500 processor SDRAM maximum throughput
- PDBRTP is the PCI device initiated burst read from Au1500 processor SDRAM maximum throughput
- PDBWTP is the PCI device initiated burst write to Au1500 processor SDRAM maximum throughput

All of these variables are the maximum throughput values identified in the discussion of each PCI access cycle type. The remaining variables are:

- AUSBRR is the ratio of Au1 core initiated single-beat reads
- AUSBWR is the ratio of Au1 core initiated single-beat writes
- PDSBRR is the ratio of PCI device initiated single-beat reads from Au1500 processor SDRAM
- PDSBWR is the ratio of PCI device initiated single-beat writes to Au1500 processor SDRAM
- PDBRR is the ratio of PCI device initiated burst reads from Au1500 processor SDRAM
- PDBWR is the ratio of PCI device initiated burst writes to Au1500 processor SDRAM

The sum of the ratios must equal 1.0 to represent 100% PCI bus utilization. The actual ratios for a given system depend upon the types of devices connected to the PCI bus.

This equation is only an approximation and tends to yield a realistic upper-bound of PCI bus throughput. The dynamic nature of I/O and the types of devices connected to the PCI bus often result in less than optimal throughput on the PCI bus. A few examples are provided at the end of this discussion.

The variety of the PCI devices and the interaction with the overall system influences the PCI bus throughput.

4.1 Positive Influences on PCI Bus Performance

The following items act to improve the PCI bus throughput.

- Utilizing the fast back-to-back capabilities of the PCI device and the Au1500 processor's PCI controller, PCI arbitration cycles are reduced thus shortening the PCI access time.
- The Au1500 processor's PCI controller features a coherency setting (pci_config[NC]=0) whereby PCI requests of Au1500 SDRAM are snooped by the data cache. If the request hits in the data cache, the data cache fulfills the request immediately, thus avoiding the need to access external SDRAM.
- The Au1500 processor's PCI controller features a cacheable window into the PCI memory address space (the pci_cmem register). Accesses to this window initiate burst transfers rather than single-beat transfers.

- By programming the TLB with CCA=7 when mapping PCI memory spaces, the write buffer can gather Au1 core stores which in turn leads to more efficient burst writes into PCI memory space.
- To improve performance, the Au1500 processor implements a write FIFO between the system bus and PCI. This FIFO effectively shortens the Au1 core write cycle to just the system bus time, if the FIFO has an available slot.

The pci_cmem feature, the use of CCA=7, and the write FIFO items warrant additional discussion as these performance features can make a significant, positive improvement in PCI bus throughput.

4.1.1 PCI Cacheable Memory Window

The Au1500™ processor's PCI controller features a cacheable window into PCI memory space via the pci_cmem register. The PCI cacheable window is used to map a pre-fetchable PCI memory space, enabling the Au1 core to cache the PCI memory window contents. This has two mutually beneficial effects:
1) the Au1 core caches the space for improved processing performance, and
2) the data cache initiates burst transfers to and from PCI for better PCI bus throughput.

The TLB mapping that covers pci_cmem must use CCA=4. This CCA encoding fetches word 0 first (as opposed to critical word first), to match the PCI specification. Also note that CCA=4 is a non-coherent configuration, therefore the data cache does not snoop PCI memory space accesses. For example, consider a PCI memory space that is both mapped via pci_cmem and either the source and/or destination of a a PCI target-to-target transfer, in this scenario the target-to-target transfer is contained solely within the PCI bus, so the Au1 core cache can not snoop the transfer, and as a result, either the PCI target or the Au1 data cache might contain stale data.

On an Au1 core read from this window, the data cache initiates a burst read transfer. The timing of the burst read access is depicted here in Figure 8: "Au1 Core Initiated Burst Read from PCI".

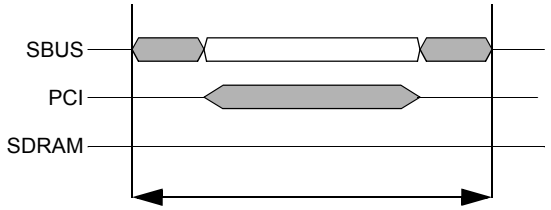

Figure 8: Au1 Core Initiated Burst Read from PCI

The minimum time necessary for a PCI burst read is the sum of the active SBUS and PCI bus times: 5 SBUS clocks + 18 PCI clocks + 10 SBUS clocks (5 SBUS clocks for synchronization and arbitration, 6 PCI clocks for internal state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, 1 PCI clock for turnaround, 8 PCI clocks for data, 1 PCI clock for state machine and 10 SBUS clocks to return data to the Au1 core and to complete the access). This yields 350.1ns for an Au1 core initiated bust read access. This yields 91.4MB/s throughput, vastly improved compared to the single-beat read 19.2MB/s throughput.

On a cast-out of a dirty cache line, the data cache initiates a burst write transfer. The timing of the burst write access is depicted here in Figure 9: “Au1 Core Initiated Burst Write to PCI”.

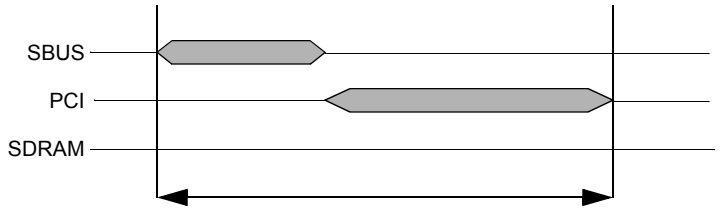

Figure 9: Au1 Core Initiated Burst Write to PCI

The minimum time necessary for a PCI burst write is the sum of the active SBUS and PCI bus times: 12 SBUS clocks + 17 PCI clocks (5 SBUS clocks for synchronization, arbitration and burst write access, 6 PCI clocks for state machine synchronization, 1 PCI clock for arbitration, 1 PCI clock for address/command, and 8 PCI clocks for data). This yields 258.4ns for an Au1 core initiated burst write access. This yields 123.8MB/s throughput, greatly improved compared to the single-beat write 24.6MB/s throughput. The PCI write buffer can improve performance as well, see discussion in 4.1.3 “PCI Write FIFO”.

If the software environment/application permits, the cacheable window allows the Au1 core to cache PCI memory for improved PCI bus throughput and overall system performance.

4.1.2 PCI CCA=7

The TLB mappings for PCI must use a non-cacheable setting (the exception being the pci_cmem window previously discussed). Typically the TLB CCA value is 2 (non-cached, non-mergeable, non-gatherable), but by programming the TLB with CCA value of 7, the write buffer can merge and gather Au1 core stores into more efficient burst writes into PCI memory space.

The throughput advantage of burst writes to PCI is discussed previously, the actual effect on overall system performance is positive but difficult to determine due to the dynamic nature of the run-time system. The PCI write buffer can further improve performance, see discussion in 4.1.3 “PCI Write FIFO”.

4.1.3 PCI Write FIFO

To improve performance, the Au1500 processor implements a write FIFO between the system bus and the PCI bus. This FIFO effectively shortens the Au1 core write cycles to just the system bus time, if the FIFO has an available slot to accept the write. That is, from the Au1 core perspective, the write completes as soon as the FIFO accepts it, rather than waiting until the write to the PCI target completes.

The FIFO can accept any combination of two single-beat write accesses or two burst write accesses. A third write access stalls the Au1 core (and the system bus) until a slot is available.

The PCI write FIFO improves overall system performance by buffering write accesses to PCI, thus

enabling the Au1 core and system bus to continue with other activities while the writes to PCI complete.

4.2 Detrimental Influences on PCI Bus Performance

The following items may reduce the PCI bus throughput.

- The PCI bus runs asynchronously to the Au1 core and system bus. As a result, on each access several clock cycles are consumed synchronizing the different clock domains.
- If one 33MHz PCI device is connected to the PCI bus, then the entire PCI bus operates at 33MHz, even for devices that can operate at 66MHz. All the examples above were calculated for a 66MHz bus, reducing the bus to 33MHz will have a detrimental impact on PCI bus throughput.
- The PCI clock may not be exactly 66MHz. When using an internally generated clock, a 64MHz (or 32MHz) PCI clock is common.
- The system bus is shared by a number of masters (Au1 core, PCI, USB, Ethernet, DMA). As such, PCI bus cycles that use the system bus to access SDRAM may experience increased latency until the PCI bus wins arbitration of the system bus. Furthermore, accesses to the static bus controller (e.g. Flash, PCMCIA, etc.) by the Au1 core or DMA occupy the system bus and can add tens or hundreds of nanoseconds of latency to arbitration.
- Au1 core initiated reads of PCI space prevent a PCI device that is attempting to access Au1500 SDRAM from winning arbitration on the system bus (because the Au1 core won arbitration for the system bus).
- The DEVSEL# timing for a given PCI device can increase the access time, which in turn decreases the PCI bus throughput. PCI devices assert DEVSEL# fast, medium or slow (see PCI Configuration space Status register).
- Many PCI devices are unable to satisfy a read or write request immediately. The TRDY# signal is de-asserted by the device to insert wait states into the access.
- For PCI bus cycles that access Au1500 processor SDRAM, if the PCI controller is unable to win system bus arbitration, then a PCI retry is signalled. In this situation, the access time to Au1500 SDRAM can be extended up to 16 PCI clocks while the PCI controller attempts to win system bus arbitration.
- The discussions above assume 4-bytes of data for a single-beat access, and 32-bytes of data for a burst access. In reality, not all accesses will transfer 4 or 32-bytes in the access; it can be less and thus will further decrease PCI bus throughput.
- The discussions above ignored other PCI cycles types (e.g. C/BE encoding). Initiating the other PCI cycles types further decreases the PCI bus bandwidth available to the six data movement PCI cycles types.

Of the above factors, the PCI bus clock, the PCI target TRDY# timing, and the number of active system bus masters are the most detrimental influences on PCI bus throughput.

4.3 PCI Bus Performance Examples

Here are two typical examples to illustrate PCI bus throughput. In a real design, the system should be profiled in order to determine more accurate ratios and thus a better estimate of the PCI bus throughput.

4.3.1 Network Device Example

A high performance PCI-based networking device uses PCI bus-mastering DMA to transfer packets to/from Au1500 SDRAM. The device also uses ring buffers in Au1500 processor SDRAM to provide queues of incoming and outgoing packets. In this environment, the ratios are similar to the following:

- AUSBRR is 0.10 for managing device operation, servicing interrupts, etc.
- AUSBWR is also 0.10 for managing device operation, servicing interrupts, etc.
- PDSBRR is 0.10 for reading ring buffer contents
- PDSBWR is 0.10 for updating ring buffer status
- PDBRR is 0.30 for transmitting outgoing packets
- PDBWR is 0.30 for receiving incoming packets

In substituting all the values, the throughput becomes:

$$TP = (19.2MB/s * 0.10) + (24.6MB/s * 0.10) + (27.2MB/s * 0.10) + (22.6MB/s * 0.10) + (89.0MB/s * 0.30) + (92.9MB/s * 0.30)$$

$$TP = 63.9MB/s$$

This example is also indicative of a high performance disk controller.

4.3.2 Graphics Device Example

A high performance PCI-based graphics device is programmed with drawing commands to perform the drawing locally (i.e. hardware acceleration), and drawing is also performed by the Au1 core directly writing into the frame buffer memory. In this environment, the ratios are similar to the following:

- AUSBRR is 0.20 for moderate read-modify-write pixel operations (blits)
- AUSBWR is also 0.80 for drawing commands, frame buffer updates (blits)
- PDSBRR is 0.00
- PDSBWR is 0.00
- PDBRR is 0.00

- PDBWR is 0.00

In substituting all the values, the throughput becomes:

$$TP = (19.2MB/s * 0.20) + (24.6MB/s * 0.80) + (27.2MB/s * 0.00) + (22.6MB/s * 0.00) + (89.0MB/s * 0.00) + (92.9MB/s * 0.00)$$

$$TP = 23.5MB/s$$

The PCI bus throughput capable with the Au1500 processor enables good motion video decode. For instance, full motion video decode typically requires 30 frames per second which is achievable with the Au1500 processor:

- A video clip with resolution of 800x600 at 16bpp requires 960,000 bytes per frame. This yields a frame rate of 24 frames per second (23.5MB/s / 960,000 bytes/frame).
- A video clip with resolution of 640x480 at 16bpp requires 614,400 bytes per frame. This yields a frame rate of 38 frames per second (23.5MB/s / 614,400 bytes per frame).

This throughput translates into very good graphics experiences.

5.0 Conclusion

The PCI controller integrated into the Au1500™ processor is capable of handling common PCI based peripherals such as networking, graphics and disk controllers. The actual PCI bus performance is dependent upon the devices connected to the PCI bus and can be estimated using the equation provided.

6.0 References

1. *The Alchemy™ Au1500™ Internet Edge Processor Data Book*, Alchemy Semiconductor, 2001.
2. *PCI Local Bus Specification Rev. 2.2*, PCI Special Interest Group, 1998.
3. *PCI Clock Generation on the Alchemy™ Au1500™ Processor from AMD - Application Note*, AMD, 2002.
4. *PCI Bus Software Support for Alchemy™ Au1500™ Processor from AMD - Application Note*, AMD, 2002.
5. *AMD Alchemy™ Solutions Au1000™, Au1100™ and Au1500™ Processors SDRAM Performance - Application Note*, AMD, 2003.