COMMUNICATIONS
CONTROLLERS

Marvell Solutions

MV64360

Tuning the Performance of the MV64360

Doc. No. MV-S300237-00, Rev. A
December 1, 2002

MOVING FORWARD
FASTER®

MARVELL®

Document Status	
Advanced Information	This document contains design specifications for initial product development. Specifications may change without notice. Contact Marvell Field Application Engineers for more information.
Preliminary Information	This document contains preliminary data, and a revision of this document will be published at a later date. Specifications may change without notice. Contact Marvell Field Application Engineers for more information.
Final Information	This document contains specifications on a product that is in final release. Specifications may change without notice. Contact Marvell Field Application Engineers for more information.
Revision Code:	
Preliminary	Technical Publication:

Marvell
700 First Avenue
Sunnyvale, CA 94089
Phone: (408) 222 2500
Sales Fax: (408) 752 9029
Email commsales@marvell.com

Table of Contents

List of Tables

List of Figures

Section 1. Introduction

The MV64360 is an integrated system controller for high-performance embedded control applications. This document discusses the MV64360 performance, in general, and how to tune it for maximum performance, specifically.

This document provides:

- Performance data describing performance test conditions and results.
- Performance tuning guidelines for configuring the MV64360 for maximum performance.
- Waveforms demonstrating the MV64360 performance.

Note

This document is written as a supplement to the MV64360 datasheet. Refer to the MV64360 datasheet for a full description of features and configuration register settings.

1.1 About the Performance Testing

Unless otherwise stated, all measurements were taken on a DB64360 Rev 2.0 system with the following configuration.

CPU bus configuration:

- SysClk = 133 MHz
- Motorola MPC7450 CPU
- MPX mode
- Single CPU mode (bit 11 in CPU configuration register set to ’1’)
- Pipelining enabled (bit 13 in CPU configuration register set to ’1’)

DDR SDRAM configuration:

- DDR SDRAM running at the core clock frequency (Sync mode)
- Open pages enabled
- Unregistered DIMMs
- CAS latency = 2.5 @ 133 MHz, 2 @ 100 MHz
- No ECC
- Snoop pipeline enabled (bit 24 in Dunit Control High register set to ’1’)

PCI/PCI-X configuration:

- PClk = 133 MHz for PCI-X mode or 66 MHz for conventional PCI mode
- 64-bit PCI bus
- PCI <Mburst> set to 128B, <RdSize> set to 256B

Note

The PCI and PCI-X performance measurements were taken using an Agilent E2920 PCI Analyzer/ Exerciser and Agilent E2929B PCI-X Analyzer/Exerciser.

Table 1 shows the MV64360 register settings used for the performance testing.

Table 1: Register Settings for Performance Tests

Register Name	Offset	Value	Comment
CPU Configuration	0x0	0x040a28ff	• Single CPU mode • Pipelining enabled
SDRAM Configuration	0x1400	0x58200400	• Unregistered DRAM • ECC disabled • interleaving enabled
SDRAM Timing Low	0x1408	0x11511220	
SDRAM Timing High	0x140c	0x00000009	
SDRAM Open Pages Control	0x1414	0x00000000	Open pages enabled
Dunit Control Low	0x1404	0x24110051	DDR SDRAM running at core clock frequency
Dunit Control High	0x1424	0x0b00f777	Snoop pipeline enabled
SDRAM Mode	0x141c	0x00000022	CAS latency=2 clks (or 2.5 clks @ 133 MHz)
PCI Access Control Base 0 Low	0x1e00	0x00000e01	• <MBurst> = 128B • <RdSize> = 256B • non-coherent **NOTE:** If coherency is required, enable cache coherency and set <MBurst> to 32B, see the MV64360 datasheet.

Section 2. Performance Data

This section discusses performance data captured on a DB64360 system.

2.1 CPU Bus Performance

The MV64360 provides a high bandwidth CPU bus interface supporting enhanced PowerPC bus features, including streaming and data intervention in MPX mode. Streaming saves one clock cycle on consecutive address tenures and one clock cycle on data tenures. As a result, the bus utilization measured can be close to 100%.

Table 2 shows the bandwidth measured on CPU accesses to DDR SDRAM.

Table 2: Maximum CPU Bus Bandwidth (MPX Bus Mode)[1]

Transaction Type	Bandwidth	MPX Bus Utilization
Reads from DDR SDRAM	946 MBps	89%
Writes to DDR SDRAM	1064 MBps	100%

1. Bandwidth was measured by counting the number of data cycles and dividing by the total number of cycles.

2.2 PCI-X Performance

The MV64360 has a high bandwidth PCI-X interface that is capable of delivering 767 MB on a 4 KB read and absorbing 852 MB on a 4 KB write. When running in PCI-X mode, the MV64360 can provide much higher performance than in conventional PCI mode. The PCI-X mode offers protocol and frequency advantages over the conventional PCI mode.

Table 3 shows the PCI-X bandwidth that was measured for 4 KB reads and 4 KB writes to DDR SDRAM by an external PCI-X agent (HP2920). The measurements were taken on accesses to both cache coherent and non-coherent memory regions. The highest read bandwidth was achieved on an access to a non-coherent memory region with <MBurst> set to 128B (0x10).

Table 3: Maximum PCI-X Bandwidth on DDR SDRAM Accesses

Transaction Type	MBurst	Cache Coherent Region[1]	Bandwidth	Bus Utilization
Read	128	No	767 MBps	78%
Read	32	No	554 MBps	58%
Read	32	Yes	434 MBps	46%
Write	128	No	852 MBps	94%

Table 3: Maximum PCI-X Bandwidth on DDR SDRAM Accesses (Continued)

Transaction Type	MBurst	Cache Coherent Region[1]	Bandwidth	Bus Utilization
Write	32	No	668 MBps	88%
Write	32	Yes	668 MBps	88%

1. If "Yes", then the access results in snoop transactions on the CPU bus.

If coherency is required, <MBurst> must be set to 32B (0x00) and the data must be snooped on the CPU bus. However, since the MV64360 is able to pipeline these snoop transactions (if the snoop pipeline is enabled), the performance impact of maintaining coherency is minimized.

Note

For information on enabling the snoop pipeline, see 3.1.5 "Snoop Pipeline" on page 12.

Figure 1 and Figure 2 illustrate PCI-X read and write bandwidth, respectively, for transactions of various sizes.

Figure 1: PCI-X Read Bandwidth from DDR SDRAM

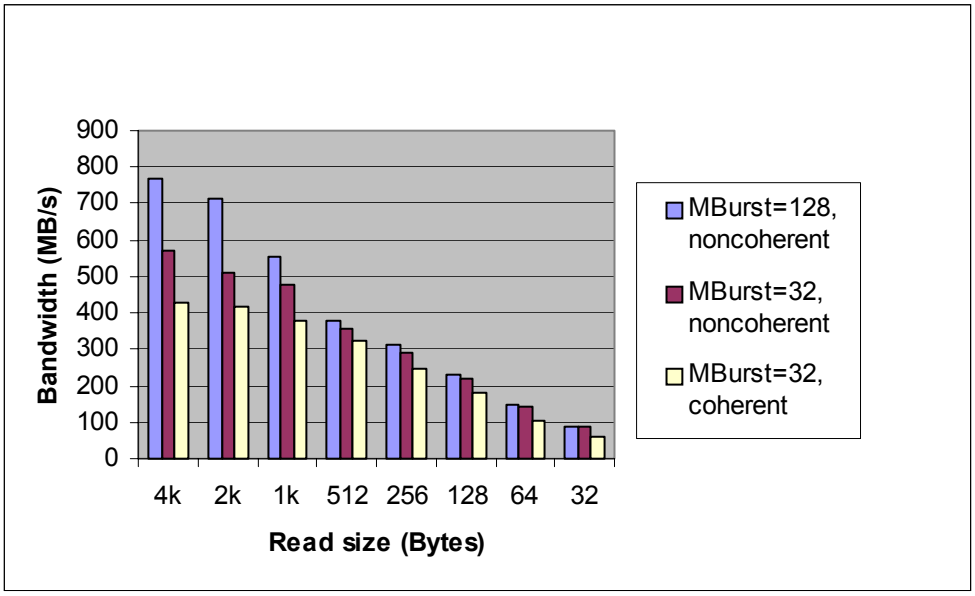
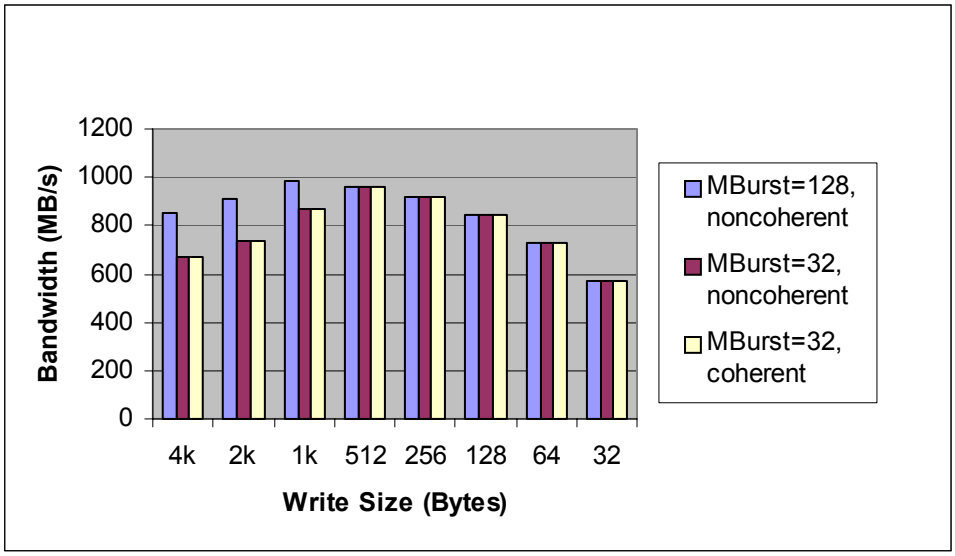


Figure 2: PCI-X Write Bandwidth to DDR SDRAM



2.3 DDR SDRAM Performance

Table 6 shows the SDRAM bus bandwidth on accesses from both PCI-X interfaces and IDMA to the SDRAM bus. These results were simulated in Synchronous mode at 133 MHz.

Table 4: SDRAM - Bus Bandwidth on Both PCI-X Interfaces and IDMA Accesses

Transaction Type	MBurst	Bandwidth	PCI-X Bus Utilization
Read	128	2000 MBps	94%
Write	128	2000 MBps	94%
Read/Write Mix	128	1787 MBps	84%[1]

1. This is 100% of the maximum bandwidth available due to the inherent turnaround cycles in the protocol.

Section 3. Performance Tuning Guide

This section discusses how to configure the MV64360 for maximum performance.

3.1 Optimizing the DDR SDRAM Interface

3.1.1 Open Pages

Enable open pages to increase performance. After a specific page is accessed it will stay open. The page will not be precharged and closed. The next access to this same page will not require an RAS cycle and will save three clock cycles in all following accesses.

Register: SDRAM_Open_Pages_Control (0x1414) <OPEn> bits [15:0] set to ’0’.

3.1.2 SDRAM Timing

Set the timing parameters to the minimum allowable values.

Register: SDRAM_Timing_Low (0x1408) and SDRAM_Timing_High (0x140C)

3.1.3 SDRAM Clocking

Use the Sync mode, where both the DDR SDRAM run at the core clock frequency. In most cases, this setting provides for maximum performance.

The DDR SDRAM interface can run in Async mode (the SDRAM clock frequency is higher than the MV64360 core clock frequency) or Sync mode (both the SDRAM and the MV64360 run off of the same clock). While it is possible to achieve higher memory bandwidth in Async mode, the synchronization between the two clock domains adds more latency. It is recommended to use Async mode only if the additional bandwidth is required and utilized.

Register: Dunit_Control_(Low) (0x1404) <ClkSync> bit [0] set to ’0’.

Note

At reset, setting DevAD[18] to ’1’ means that the SDRAM is running with the core clock frequency.

3.1.4 Unregistered SDRAM

For best performance, use unregistered SDRAM.

Register: SDRAM_Configuration (0x1400) <RegDRAM> bit [17] set to ’0’.

3.1.5 Snoop Pipeline

Enable the snoop pipeline for best performance on accesses to cache coherent memory regions.

Register: `Dunit_Control (High)` (0x1424) `<SnoopPipe>` bit [24] set to ’1’.

3.1.6 Bank Interleaving

Interleaving increases performance when multiple interfaces attempt to access memory at the same time. For example, if the CPU and a PCI devices simultaneously attempt to access memory with interleaving disabled, the second access must wait for the first access to be completed.

Alternatively, if both accesses target different physical or virtual SDRAM banks with interleaving enabled, the two accesses are interleaved. This means the second access can start before the first access ends. This reduces the total time it takes for both accesses to complete.

To take advantage of interleaving, enable interleaving and the address control register must be tuned to increase the probability that interleaving occurs.

Registers: `SDRAM_Configuration` (0x1400) `<PInter>` and `<VInter>` bits [15:14] set to ’0’.
`SDRAM_Address_Control` (0x1410).

Note

See the MV64360 datasheet for further information on interleaving.

3.1.7 Priority Queues

Different interfaces can be assigned higher priority when interfacing the DDR SDRAM. For instance, the CPU can be assigned higher priority on the transaction queues and the read queues. When competing with other interfaces, the CPU access is generated towards the DDR SDRAM before the other requesters. The data read from the DDR SDRAM also receives a "clear path" back to the CPU interface

Register: `Dunit_Control_(Low)` (0x1404) `<Prio>` bits [15:10] set to ’0’ (normal priority) or ’1’ (high priority).

- [10] = CPU
- [11] = PCI_0
- [12] = PCI_1
- [13] = MPSC
- [14] = IDMA
- [15] = Gb

3.2 Optimizing the CPU Interface

3.2.1 Pipelining

Enable pipelining for maximum bandwidth.

Register: CPU_Configuration (0x000) <Pipeline> bit [13] set to '1'.

3.2.2 MPX Bus Mode

If the CPU supports the MPX mode, enable this mode for maximum bandwidth.

The MPX bus protocol allows for higher bandwidth than the 60x protocol. It allows for address and data streaming and out-of-order completion. However, in some cases, the MPX bus protocol can add an additional latency cycle.

At reset, set DevAD[7:6] to '01'. This means that the CPU bus configuration is set to MPX mode.

3.2.3 Internal SRAM

The MV64360 has 2 Mb of internal SRAM that can be used for different purposes, such as maintaining the Gigabit Ethernet descriptors. The Gigabit packets are controlled via descriptors. Placing these descriptors in the SRAM gives very low latency to the Gigabit Ethernet unit, avoids using the DDR interface, and results in lower latency from the CPU when it is handling these descriptors.

Note

For further details, see the MV64360 Datasheet.

3.2.4 Single CPU Mode

The Single CPU mode can save one cycle of latency. Refer to the datasheet for details on when this mode can be used. All measurements in this document were taken with Single CPU mode enabled.

Register: CPU_Configuration (0x000) <SingleCPU> bit [11] set to '1'.

3.3 Optimizing PCI and PCI-X Performance

3.3.1 MBurst and RdSize

Set <MBurst> to its maximum value of 128B for maximum PCI and PCI-X bandwidth on accesses to non-coherent memory regions.

Note

<MBurst> must be set to 32B for accesses to cache coherent memory regions.

The <RdSize> parameter affects PCI performance, only. Set <RdSize> to its maximum value of 256B for maximum bandwidth. However, this setting adds additional latency before the first data is driven on the PCI bus.

Note

Avoid setting <MBurst> and <RdSize> to their maximum values if the additional bandwidth is not needed. This avoids wasting the available DDR SDRAM bandwidth.

Register: PCI_Access_Control_Base_0/1_(Low) (0x1E000, 0x1E10) <MBurst> bits [9:8] set to ’10’; <RdSize> set to ’11’.

3.3.2 Combining

Enable read/write combining to maximize the performance of the MV64360 PCI_0/1 master.

Note

Only applicable in conventional PCI mode.

Registers: PCI_0/1_Command (0xC00, 0xC80) <MWrCom> and <MRdCom> bit [5:4] set to ’1’.

3.3.3 Fast Back-to-Back Transactions

Enable the PCI ability to generate fast back-to-back transactions.

Note

Only applicable in conventional PCI mode.

Registers: PCI_0/1_Status_and_Command (0x04, 0x84) <FastBTBEn> set to ’1’.

3.4 Bandwidth Allocation

As a result of the crossbar architecture, many interfaces may simultaneously require access to the memory interface. For latency sensitive interfaces, the system is designed to ensure lower latency versus lower priority/less sensitive interfaces.

To accommodate this need, the MV64360 implements a "pizza arbiter". This feature provides control over the memory bandwidth and allows the user to allocate additional bandwidth to selected interfaces.

For example, the DMA controller and a PCI master may try to simultaneously access memory. The PCI interface can be allocated more available DDR SDRAM bandwidth by programming the DDR SDRAM crossbar control to allocate more "pizza slices" to the PCI master than to the DMA controller.

The following performance test demonstrates this feature. In this test, four large DMA transactions to and from memory occur simultaneously with a large PCI write to memory. The PCI write bandwidth was measured under various settings of the DDR SDRAM Crossbar Control Registers.

The test shows that the PCI write bandwidth increased when the PCI interface was given relatively more crossbar control slices than the DMA controller. There are 16 slices available in the SDRAM crossbar control which can be allocated to the various interfaces. If only one slice was allocated to the PCI (each ’3’ in the register value represents one slice allocated to the PCI) and 14 slices were allocated to the DMA (each ’6’ in the register value repre-

sents one slice for DMA), the PCI bandwidth measured is 174 MBps. However, when there were 14 slices for the PCI and only one for the DMA, the PCI bandwidth increased to 332 MBps (and consequently, the DMA bandwidth was reduced).

Table 5 summarizes the results of this performance test. It shows the measured PCI write bandwidth under various DDR SDRAM crossbar control settings.

Table 5: Bandwidth Allocation Example

SDRAM Crossbar Control Register Low (0x1430)[1]	SDRAM Crossbar Control Register High (0x1434)	PCI Bandwidth	PCI Bus Utilzation
0x66666632	0x66666666	174 MBps	41%
0x00360032	0x00060032	187 MBps	45%
0x33363332	0x33333333	332 MBps	78%

1. Key: Each ’3’ in the register value indicates an arbitration slice allocated to PCI. A ’6’ represents DMA, and ’2’ represents the CPU. A value of ’0’ is null and does not affect bandwidth allocation.

Section 4. Waveforms

4.1 CPU Interface

Figure 3: Pipelined CPU Reads from DDR SDRAM

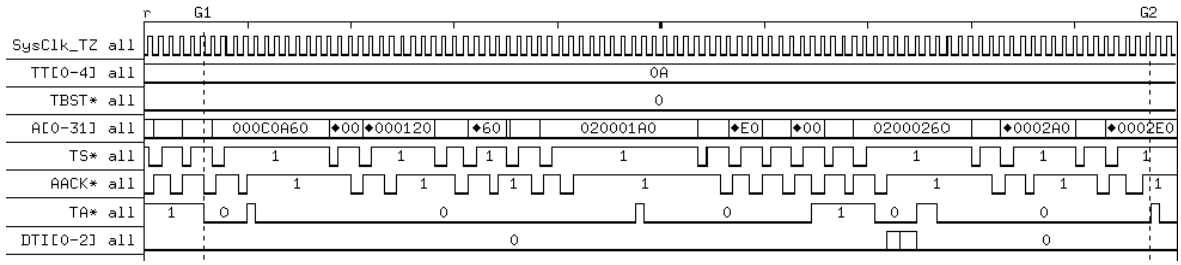
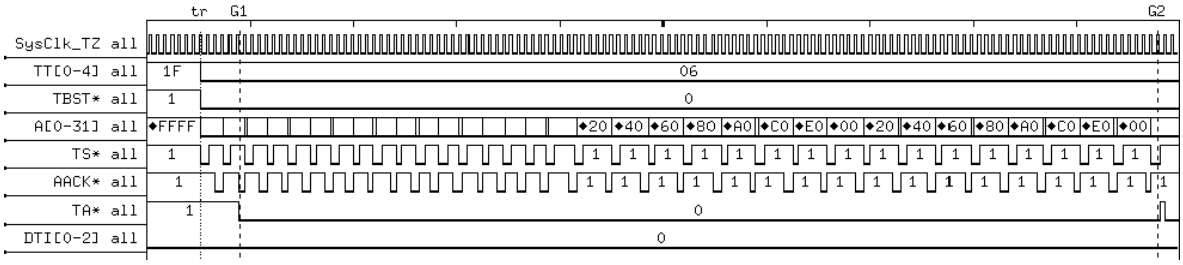


Figure 4: Pipelined CPU Writes to DDR SDRAM



4.2 PCI-X Interface

Figure 5: 4KB PCI-X Read from Non-coherent DDR SDRAM

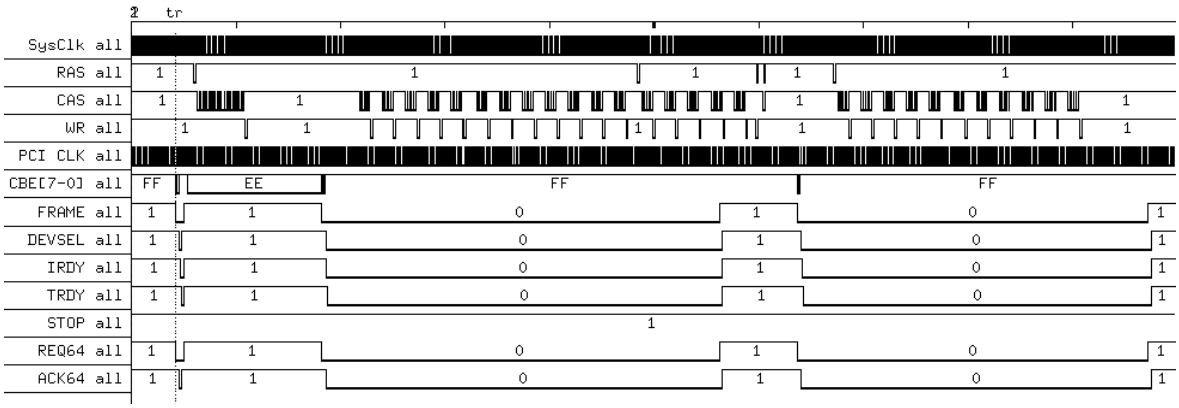


Figure 6: 1KB PCI-X Write to Non-coherent DDR SDRAM