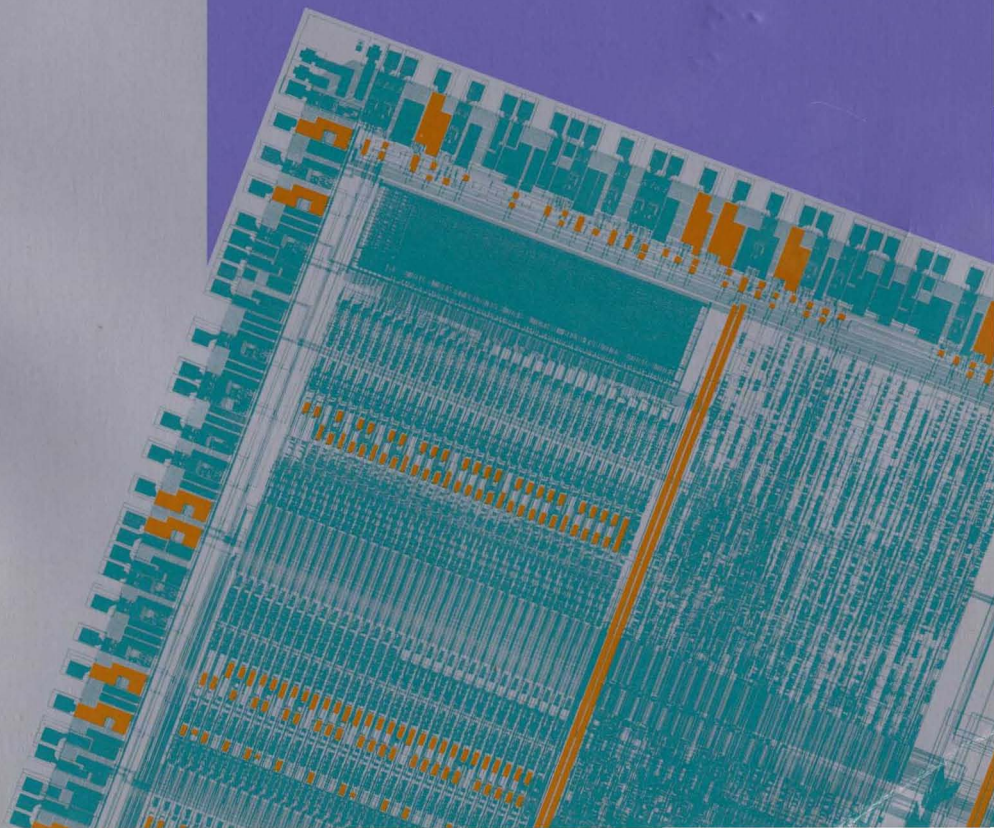
LSI LOGIC

CW33000 MIPS Embedded Processor User's Manual

A CoreWare™ Product

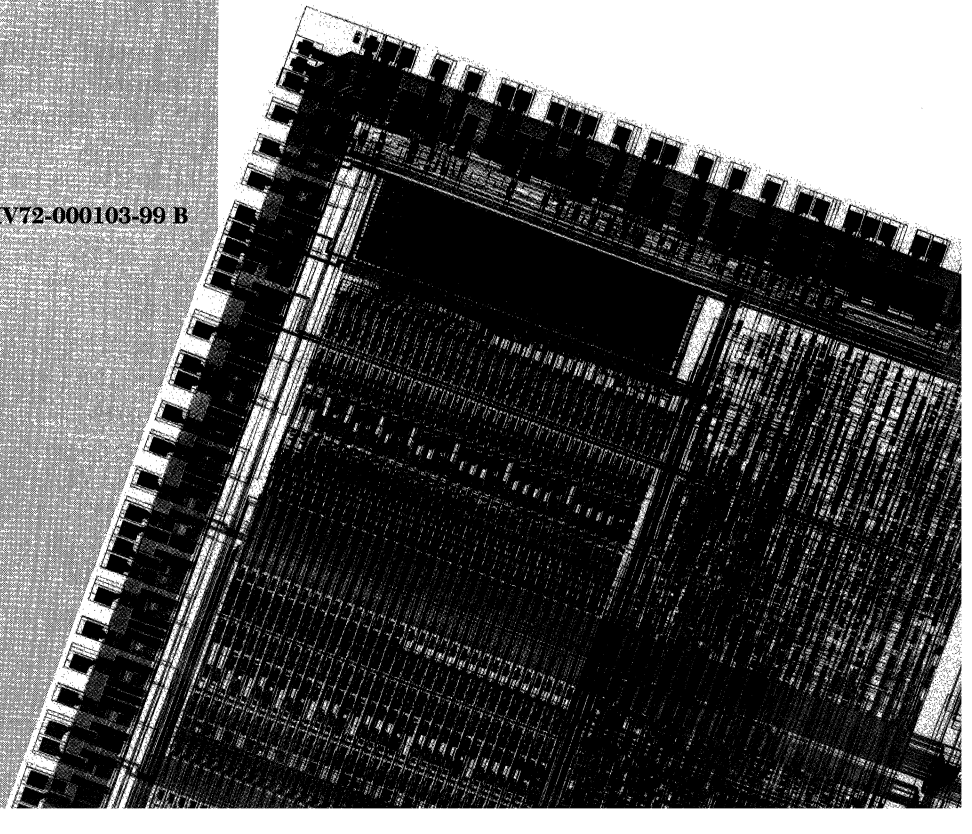


LSI LOGIC

CW33000 MIPS Embedded Processor User's Manual

A CoreWare™ Product

MV72-000103-99 B



Second Edition

This document contains data derived from functional simulations and performance estimates. LSI Logic has not verified either the functional descriptions, or the electrical and mechanical specifications using production parts.

Publications are stocked at the address given below. Requests should be addressed to:

LSI Logic Corporation
Literature Distribution, M/S D-102
1551 McCarthy Boulevard
Milpitas CA 95035
Fax 408.433.6802

LSI Logic Corporation reserves the right to make changes to any products herein at any time without notice. LSI Logic does not assume any responsibility or liability arising out of the application or use of any product described herein, except as expressly agreed to in writing by LSI Logic; nor does the purchase or use of a product from LSI Logic convey a license under any patent rights, copyrights, trademark rights, or any other of the intellectual property rights of LSI Logic or third parties.

TRADEMARK ACKNOWLEDGMENT
LSI Logic logo design, Modular Design Environment, and MDE are registered trademarks and CoreWare, C-MDE, Self-Embedding, and Right-First-Time are trademarks of LSI Logic Corporation. R3000 is a registered trademark and MIPS is a trademark of MIPS Computer Systems, Inc. All other product names, trademarks, and registered trademarks are the property of their respective owners.

Preface

This book is the primary reference and user's manual for the CW33000 MIPS Embedded Processor building block. It contains a functional description and physical and electrical specifications for the CW33000. Note that the *LR33000 Self-Embedding™ Processor User's Manual* is a required companion document for this manual; refer to the Related Publications section of this preface for information on how to acquire additional copies of that manual.

Audience

This book assumes that you have some familiarity with microprocessors and related support devices. The people who benefit from this book are:

- Engineers and managers who are evaluating the CW33000 building block for possible use in a system design
- Engineers who are designing the building block into a system-on-a-chip

Organization

This book has the following chapters:

- Chapter 1, **Introduction**, provides an overview of the CW33000 building block and the CoreWare program.
- Chapter 2, **CW33000 and LR33000: Similarities and Differences**, discusses the differences between the CW33000 building block and the LR33000 standard part, and guides the reader to relevant chapters of the *LR33000 Self-Embedding Processor User's Manual.*
- Chapter 3, **Signal Definitions**, describes the signals that comprise the CW33000's bit-level interface to other devices.
- Chapter 4, **Direct CPU Data Bus Interface**, describes in detail the Direct CPU Data Bus Interface, including functional waveforms.
- Chapter 5, **I/O Interface Operation**, discusses and illustrates the CW33000 I/O interface.
- Chapter 6, **AC Specifications**, provides both the AC timing and loading information for the CW33000 megacell.

Related Publications

LR33000 Self-Embedding Processor User's Manual, Order Number J14001.A

LR3000 and LR3000A MIPS RISC Microprocessor User's Manual, Order Number J14000.A

CW64500 Central Processing Unit User's Manual, Order Number C14003

CFY3900A SCSI Controller User's Manual, Order Number C14004

Conventions Used in this Manual

The following signal naming conventions are used throughout this manual:

Signal names for active HIGH signals end in P (positive) and signal names for active LOW signals end in N (negative).

Any signal with an overbar over its name is active LOW.

Hexadecimal numbers are indicated by the prefix "0x" before the number—for example, 0x32CF.

The first time a word or phrase is defined in this manual, it is *italicized.*

Contents

Tables

Chapter 1
Introduction

This document describes the LSI Logic CW33000 MIPS Embedded Processor building block. This building block, derived from LSI Logic's LR33000 Self-Embedding™ Processor standard part, also implements the MIPS Reduced Instruction Set Computer (RISC) architecture and executes the MIPS-1 instruction set. The CW33000 is part of LSI Logic's CoreWare™ library of leading-edge microprocessors, floating-point processors, and peripheral functions. Designers can combine the elements from this library with other LSI Logic library elements to implement highly integrated systems in silicon for a wide variety of applications.

This chapter provides an overview of LSI Logic's CoreWare program and the CoreWare library. Then it introduces the CW33000. Finally, it discusses the issue of CW33000 compatibility with the R2000 and R3000 microprocessors. The sections in Chapter 1 are:

- CoreWare Program
- CW33000 Overview
- Compatibility with the R2000 and R3000 Microprocessors

1.1 CoreWare Program

The CoreWare program offers a new approach to system design. Through the CoreWare program, LSI Logic gives customers the ability to combine leading-edge microprocessors, floating-point processors, and peripheral functions on a single chip to create products uniquely suited to the customer's applications. The approach—which combines high-performance building blocks, sophisticated design software, and expert support—provides unparalleled design flexibility and allows designers to create high-quality, leading-edge products for a wide range of markets.

The CoreWare program consists of three major components:

- CoreWare Building Blocks
- Design Environment
- Expert Support

CoreWare Building Blocks

The CoreWare building blocks include elements based on the LSI Logic high-performance standard products as well as other, industry-standard products. These standard products are leading-edge implementations of RISC microprocessors, floating-point processors, and peripherals. The CoreWare building blocks, ranging from an embedded MIPS processor and a 64-bit floating-point processor to a SPARC SBus interface and a SCSI controller, are fully supported library elements for use in the LSI Logic hardware development environment. Note that the building blocks include simulation models with timing information, so designers can accurately simulate device performance and trade off various implementation options.

Design Environment

LSI Logic's Modular Design Environment® (MDE®) design and analysis tools and the new C-MDE™ (concurrent MDE) design system provide a complete framework for device and system development and simulation. The MDE toolset and LSI Logic's advanced ASIC technologies consistently produce Right-First-Time™ silicon. The C-MDE system puts even more power in the hands of the designer by providing enhanced MDE capabilities in a concurrent design environment.

Expert Support

LSI Logic's in-house experts support the CoreWare program with high-level design and market experience in a wide variety of application areas. They provide design support from system architecture definition through chip layout and test vector generation. They help determine how much functionality to integrate on a single chip, trading off functionality versus cost to find the most cost-effective solution. When the trade-offs are complete, the designer and LSI Logic's applications engineers implement and test the design using MDE or C-MDE and the CoreWare building blocks.

1.2 CW33000 Overview

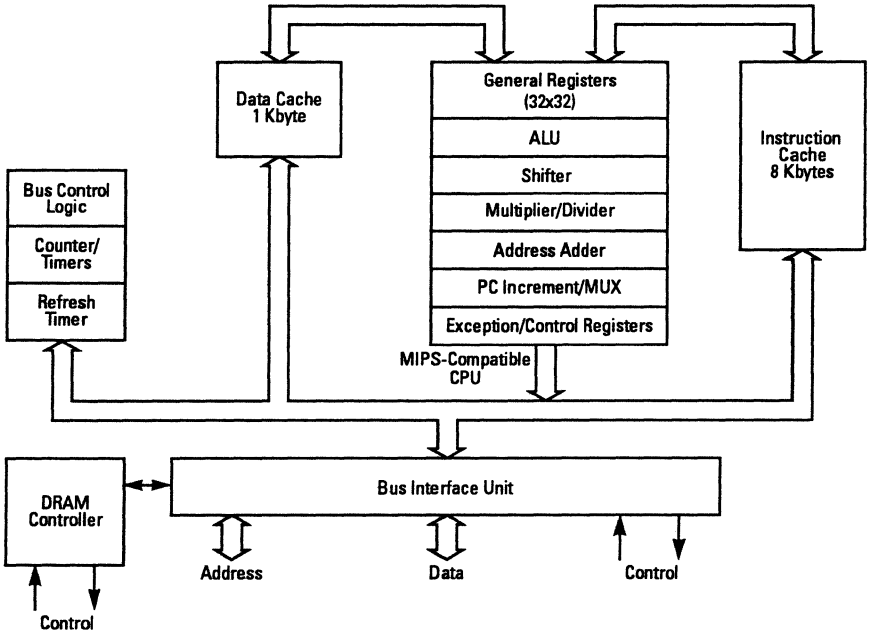
This section introduces the CW33000. First it discusses the architecture and general features of the building block; then it briefly discusses the CPU

registers, the instruction set, the system control coprocessor (CP0), and operating modes.

Architecture and General Features

The CW33000 building block design is based on the LSI Logic LR33000 Self-Embedding Processor, a standard product. The LR33000 is a high-performance, single-chip microprocessor optimized for embedded control applications. It includes a MIPS-compatible CPU with on-chip instruction and data caches, a DRAM Controller, a sophisticated bus interface unit, and three counter/timers. Figure 1.1 shows a block diagram of the LR33000 standard product.

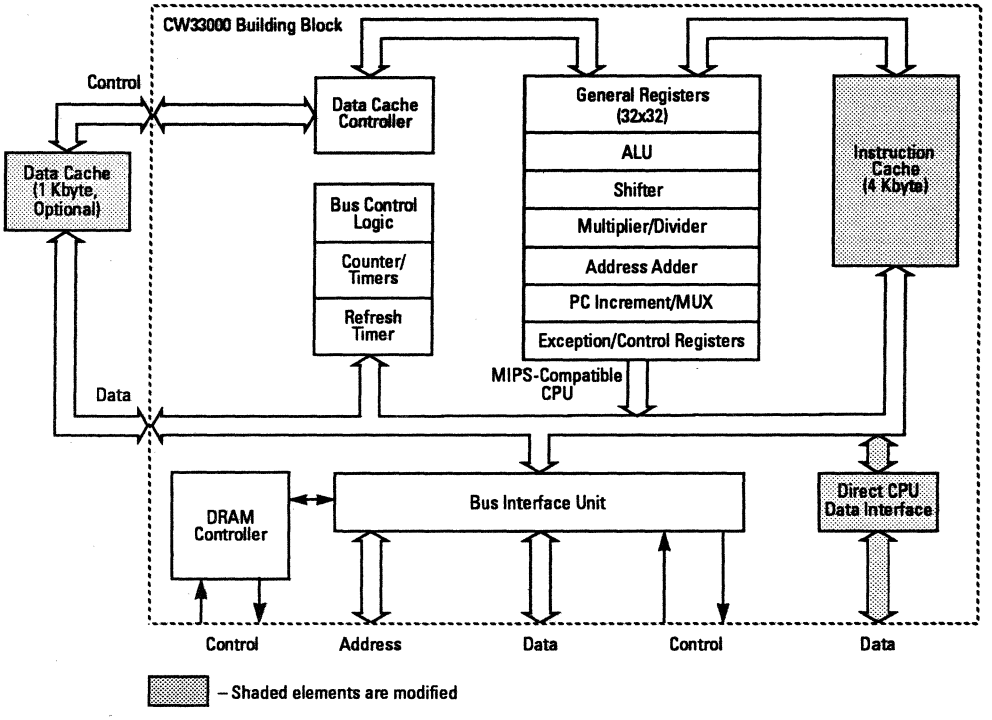
Figure 1.1 LR33000 Standard Product Block Diagram



The CW33000 building block is fundamentally the same processor with a few modifications, as shown in Figure 1.2. The differences, which allow application-specific configuration of the processor, concern the instruction and data caches and communication with the CPU. Note that the CW33000 building block is particularly suited for embedded control applications where introducing additional on-chip functionality enhances system performance, reduces the system chip count and board area, and lowers the system cost.

The shared features of and the differences between the CW33000 and the LR33000 are presented next and discussed in more detail in Chapter 2.

Figure 1.2
CW33000 Building Block Diagram



MIPS CPU – The CW33000 implements the high-performance MIPS-1 architecture found in the LR33000, with these features:

- **Instruction Set Compatibility** – The CW33000 maintains full binary compatibility with the R2000, R3000, and R3000A microprocessors.
- **Efficient Pipelining** – The CPU's five-stage pipeline design assists in obtaining an execution rate approaching one instruction per cycle. Pipeline stalls and exceptional events are handled precisely and efficiently.
- **Full 32-Bit Operation** – The CW33000 contains thirty-two 32-bit registers. All instructions and addresses are 32 bits wide to support a 4-Gbyte address space.

- **Enhanced Debug Features** – The CW33000 incorporates a hardware breakpoint on program counter and data address registers to ease software debugging, and it includes program trace support logic.

On-Chip Cache Memory – The CW33000 and LR33000 on-chip cache configurations differ. The LR33000 contains an 8-Kbyte instruction cache and a 1-Kbyte data cache. It can access both caches during a single clock cycle, which allows the processor to execute one instruction per cycle when executing instructions out of cache memory. Both caches employ direct address mapping and support bus snooping to maintain cache coherency. The data cache uses a write-through technique to maintain main memory coherency.

The CW33000 building block offers a 4-Kbyte instruction cache and an optional data cache. The CW33000 contains a data cache controller but does not include the 1-Kbyte data cache. The designer may add the data cache SRAMs to a design based on the CW33000 building block, or the designer may choose not to use a data cache at all and utilize the chip area for another purpose.

On-Chip DRAM Control – The CW33000 has an integral DRAM Controller that supports many popular DRAMs. Its features include support for page-mode DRAMS, CAS-before-RAS refresh, and DMA by separate I/O devices.

Bus Interface Unit (BIU) – The CW33000 BIU provides a sophisticated memory interface that connects easily to I/O devices, supports both 8- and 32-bit PROMs, and includes a programmable wait-state generator. The BIU supports optional parity generation and checking. Checking may be suspended on a per-memory-access basis, and the CW33000 provides the parity error output, PERRP, to signal parity errors to external logic.

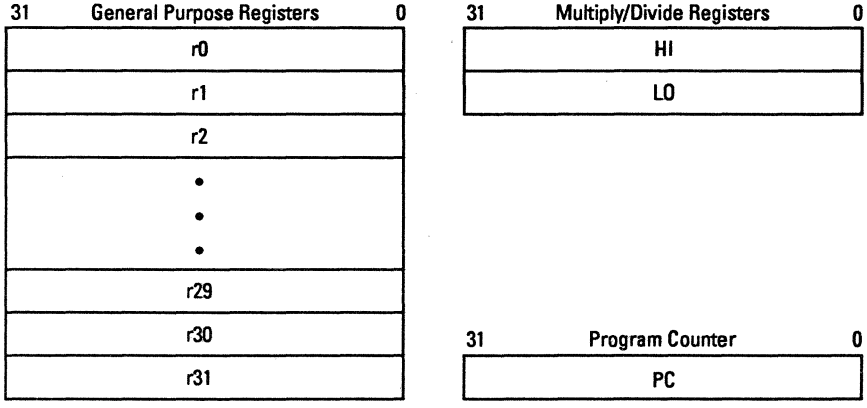
On-Chip Counter/Timers – The CW33000 includes three counter/timers: two 24-bit general-purpose timers and a 12-bit refresh timer that can support an on-chip or off-chip DRAM Controller.

Direct CPU Data Bus Interface – Although this interface is not included in the LR33000, the CW33000 includes the Direct CPU Data Bus Interface to provide fast, single-cycle data (operand) transfer between the CPU and dedicated SRAM or ROM that is on-chip but not part of the CW33000 building block.

CPU Registers

Like the LR33000, the CW33000 CPU provides 32 general-purpose 32-bit registers, a 32-bit program counter, and two 32-bit multiply/divide registers, HI and LO, that hold the results of integer multiply and divide operations. The CPU registers are shown in Figure 1.3 and are discussed in Section 2.1, "Data and Registers." Notice that the figure does not contain a Program Status Word (PSW) register; the functions traditionally provided by a PSW register are instead provided by the *Status* and *Cause* registers incorporated within the system control coprocessor (CP0).

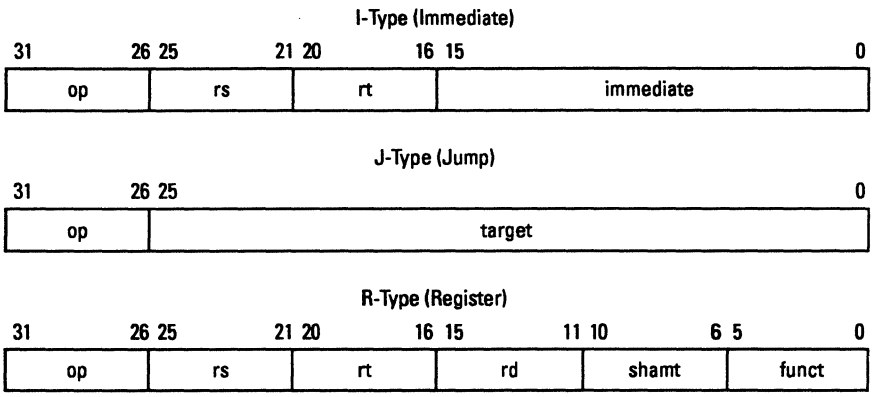
Figure 1.3
LR33000 CPU Registers



Instruction Set Overview

The CW33000 executes the same instruction set as the LR33000 standard part. All CW33000 instructions are 32 bits long. As shown in Figure 1.4, instructions have three basic formats: I-type (immediate), J-type (jump), and R-type (register). Instruction decoding is simplified by this approach. The compiler can synthesize more complex and less frequently used operations and addressing modes using sequences of simple RISC-type instructions. The assembler recognizes some CISC-type instructions for programming ease, but translates them into sequences of simple instructions.

Figure 1.4
CW33000/LR33000
Instruction Formats



The CW33000 instruction set is divided into the following groups:

- **Load/Store** instructions are the only instructions that move data between memory and general registers. These instructions are I-type, because the only addressing mode supported is base register plus a 16-bit, signed immediate offset.
- **Computational** instructions perform arithmetic, logical, and shift operations on values in registers. These instructions are either R-type (both operands and the result are registers) or I-type (one operand is a 16-bit immediate).
- **Jump** and **Branch** instructions change the control flow of a program. Jumps are always to a paged, absolute address either formed by combining a 26-bit target with four bits of the program counter (J-type format, for subroutine calls), or from the contents of a 32-bit byte address register (R-type, for returns and dispatches). Branches have 16-bit offsets relative to the program counter (I-type).
- **Coprocessor** instructions specify coprocessor operations. Most of these instructions are not implemented by the CW33000. See Section 1.3, "Compatibility with the R2000 and R3000 Microprocessors," for details.
- **Coprocessor 0** instructions specify operations on the system control coprocessor (CP0) registers to manipulate the exception-handling facilities of the processor.
- **Special** instructions perform a variety of tasks including movement of data between special and general registers, system calls, and breakpoint. These instructions are always R-type.

Table 1.1 lists the instruction set of the CW33000 processor. A more detailed summary is provided in Section 2.2, "Instruction Set Summary."

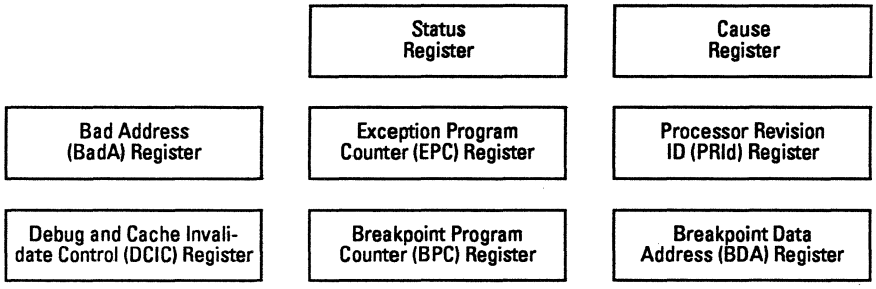
Table 1.1
CW33000/LR33000
Instruction Summary

Op	*Description*
Load/Store Instructions	
LB	Load Byte
LBU	Load Byte Unsigned
LH	Load Halfword
LHU	Load Halfword Unsigned
LW	Load Word
LWL	Load Word Left
LWR	Load Word Right
SB	Store Byte
SH	Store Halfword
SW	Store Word
SWL	Store Word Left
SWR	Store Word Right
Arithmetic Instructions (ALU Immediate)	
ADDI	Add Immediate
ADDIU	Add Immediate Unsigned
SLTI	Set on Less Than Immediate
SLTIU	Set on Less Than Immediate Unsigned
ANDI	AND Immediate
ORI	OR Immediate
XORI	Exclusive OR Immediate
LUI	Load Upper Immediate
Arithmetic Instructions (3-Operand, Register-Type)	
ADD	Add
ADDU	Add Unsigned
SUB	Subtract
SUBU	Subtract Unsigned
SLT	Set on Less Than
SLTU	Set on Less Than Unsigned
AND	AND
OR	OR
XOR	Exclusive OR
NOR	NOR
Shift Instructions	
SLL	Shift Left Logical
SRL	Shift Right Logical
SRA	Shift Right Arithmetic
SLLV	Shift Left Logical Variable
SRLV	Shift Right Logical Variable
SRAV	Shift Right Arithmetic Variable
Multiply/Divide Instructions	
MULT	Multiply
MULTU	Multiply Unsigned
DIV	Divide
DIVU	Divide Unsigned
MFHI	Move From HI
MTHI	Move To HI
MFLO	Move From LO
MTLO	Move To LO
Jump and Branch Instructions	
J	Jump
JAL	Jump And Link
JR	Jump Register
JALR	Jump And Link Register
BEQ	Branch on Equal
BNE	Branch on Not Equal
BLEZ	Branch on Less than or Equal to Zero
BGTZ	Branch on Greater Than Zero
BLTZ	Branch on Less Than Zero
BGEZ	Branch on Greater than or Equal to Zero
BLTZAL	Branch on Less Than Zero And Link
BGEZAL	Branch on Greater than or Equal to Zero And Link
Special Instructions	
SYSCALL	System Call
BREAK	Breakpoint
Coprocessor Instructions	
BCzT	Branch on Coprocessor z True
BCzF	Branch on Coprocessor z False
System Control Coprocessor (CP0) Instructions	
MTC0	Move To CP0
MFC0	Move From CP0
RFE	Restore From Exception

System Control Coprocessor (CP0)

The CW33000 system control coprocessor (CP0) supports exception-handling functions of the CW33000. Figure 1.5 summarizes the register set. Refer to Section 2.3, "Exception Processing," for descriptions of the exception-processing registers.

Figure 1.5
The CP0 Exception-Handling Registers



Operating Modes

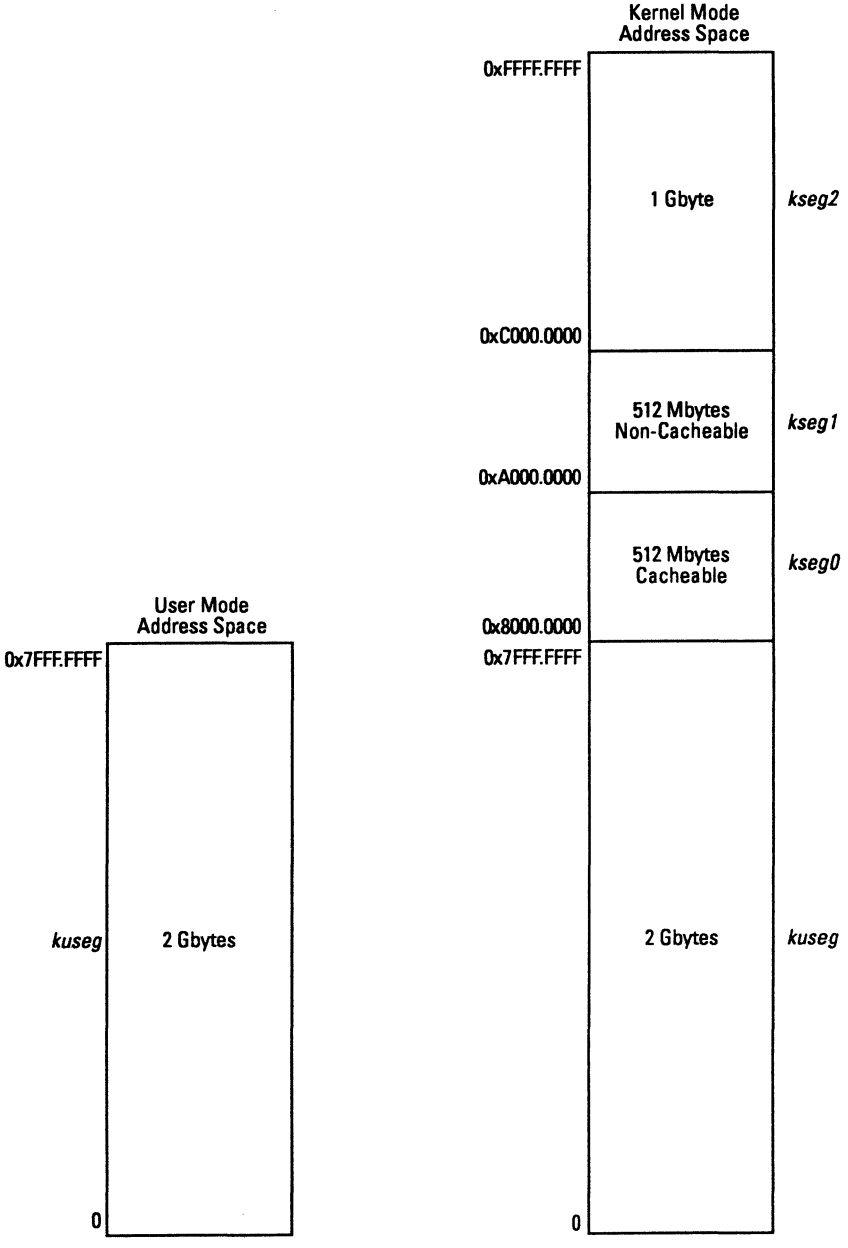
To facilitate the separation of user and supervisory software, the CW33000 has two operating modes: *user* and *kernel.* Normally, the processor operates in the user mode until an exception is detected, which forces it into the kernel mode. It remains in the kernel mode until a Restore From Exception (RFE) instruction is executed. Figure 1.6 shows the address space for the two operating modes.

User Mode – In user mode, a single, uniform address space (kuseg) of 2 Gbytes is available.

Kernel Mode – Four separate segments are defined in kernel mode:

- *kuseg* – References to this 2-Gbyte segment are treated just like user mode references, thus streamlining kernel access to user data.
- *kseg0* – References to this 512-Mbyte segment use cache memory and are hard-mapped to the first 512 Mbytes of memory.
- *kseg1* – References to this 512-Mbyte segment do not use the cache and are hard-mapped into the same 512-Mbyte segment of memory space as *kseg0.*
- *kseg2* – References to this 1-Gbyte segment are not mapped and the cacheability of data and instructions is controlled on a per access basis by the CACHDN signal.

Figure 1.6
CW33000/LR33000
Address Space



1.3 Compatibility with the R2000 and R3000 Microprocessors

The CW33000 MIPS Embedded Processor provides instruction set compatibility with the R2000 and R3000A microprocessors. Therefore, software developed for the R2000 and R3000A can be easily ported to systems based on the CW33000, significantly reducing software development time and resource requirements.

However, like the LR33000, the CW33000 does not support external coprocessors and does not implement the R2000/R3000A memory management unit (MMU). In addition, the internal cache memories are managed differently and two hardware breakpoint registers have been added to facilitate software debugging. These differences are further described under "The System Control Coprocessor CP0," below.

Coprocessor Support

The CW33000 does not support external coprocessors. To prevent unpredictable results should a coprocessor instruction be decoded, all external coprocessors should be disabled by setting the appropriate coprocessor usability bits in CP0's Status Register to zero. When the coprocessors are disabled in this way, the detection of a coprocessor instruction causes a Coprocessor Unusable Exception. Table 1.2 lists the unsupported coprocessor instructions.

Table 1.2 Unimplemented Instructions

Op	*Description*
Coprocessor Instructions	
LWCz	Load Word from Coprocessor
SWCz	Store Word to Coprocessor
MTCz	Move To Coprocessor[1]
MFCz	Move From Coprocessor[1]
CTCz	Move Control To Coprocessor
CFCz	Move Control From Coprocessor
COPz	Coprocessor Operation
System Control Coprocessor (CP0) Instructions	
TLBR	Read indexed TLB entry
TLBWI	Write Indexed TLB entry
TLBWR	Write Random TLB entry
TLBP	Probe TLB for matching entry

1. The CW33000 implements this instruction for CP0 when in kernel mode.

The CW33000 provides four coprocessor condition signals that software can test using the Branch on Coprocessor z True and Branch on Coprocessor z False (BCzT and BCzF) instructions. This facility allows software to

directly monitor hardware that is connected to the condition signals. The corresponding coprocessor must be enabled to use the test instruction. See Section 2.2, "Instruction Set Summary," for a description of the BCzT and BCzF instructions and Section 3.4, "Peripheral Interface Signals," for a description of the coprocessor condition signals.

Memory Management Unit Support

The CW33000 does not implement the R2000/R3000A memory management unit, which includes the Translation Lookaside Buffer (TLB). Consequently, the CW33000 does not implement any of the instructions designed to maintain and use the TLB. If the processor detects a TLB instruction, it causes a Reserved Instruction Exception. Refer to Table 1.2 for a list of the unimplemented instructions. Note that the CW33000 does not implement the TLB registers in CP0: EntryHi, EntryLo, Index, Random, and Context; references to those registers cause a Reserved Instruction Exception.

The System Control Processor CP0

Although it does not include the TLB registers, the CW33000 implementation of CP0 does include the additional debug and control registers summarized below. Section 2.3, "Exception Processing," contains more information on CP0.

Debug Registers – The CW33000 includes program counter and data address breakpoint registers to simplify software debugging. These registers have been incorporated into CP0, and software accesses them using Move from Coprocessor and Move to Coprocessor (MFC0 and MTC0) instructions.

Debug and Cache Invalidate Control Register – This register contains the enable and status bits for the CW33000 debug mechanism, as well as the control bits for the Cache Controller's invalidate mechanism. The CW33000 does not implement the R2000/R3000 cache control bits, which are located in the R2000/R3000 Status Register.

Chapter 2
CW33000 and LR33000: Similarities and Differences

This chapter discusses some major aspects of the CW33000 in terms of their similarities and differences with corresponding aspects of the LR33000 standard part. The chapter covers:

- Data and Registers
- Instruction Set Summary
- Exception Processing
- Memory System Architecture
- Cache Operation
- Counter/Timers
- Signal Definitions
- DRAM Controller Operation
- I/O Interface Operation
- Specifications

Note that the order of these topics parallels the chapter organization in the *LR33000 Self-Embedding Processor User's Manual*, for ease of reference.

2.1 Data and Registers

The CW33000 data formats and addressing scheme, as well as general register configuration, are identical to those in the LR33000; read Chapter 2, "Data and Registers," and Appendix B, "Memory-Mapped Register Summary," in the *LR33000 Self-Embedding Processor User's Manual* for complete details. Note that all general registers must be initialized during simulation, whether or not they are used in the simulation.

2.2 Instruction Set Summary

The CW33000 executes the same instruction set as the LR33000. Refer to the *LR33000 Self-Embedding Processor User's Manual* Chapter 3, "Instruction Set Summary," Appendix A, "Instruction Set Details," and

Appendix C, "Machine Language Programming Tips," for more information.

2.3 Exception Processing

The CW33000 and the LR33000 handle exceptions identically except for CW33000-based designs that include on-chip memory. Section 4.3 of this manual describes how to handle instruction exceptions that involve the CW33000 Direct CPU Data Bus Interface and user-designed on-chip memory. Otherwise, for general information on exception handling, refer to Chapter 4, "Exception Processing," in the *LR33000 Self-Embedding Processor User's Manual.*

Note that if the CW33000-based design does not include a data cache, then the function of the D (Data Invalidate Enable) bit, bit 13 in the Debug and Cache Invalidate Control Register (described in Chapter 4 of the LR33000 manual), alters slightly. In systems with a data cache, software sets the D bit to one to enable the data cache invalidate mechanism; then, when the software performs a store instruction, the data cache controller invalidates the data cache line at the effective store address. In a system with no data cache, if software sets the bit and tries to perform the store, neither the data cache invalidate nor the store to memory occurs.

2.4 Memory System Architecture

The CW33000 memory system architecture does deviate from that of the LR33000 in these three significant ways:

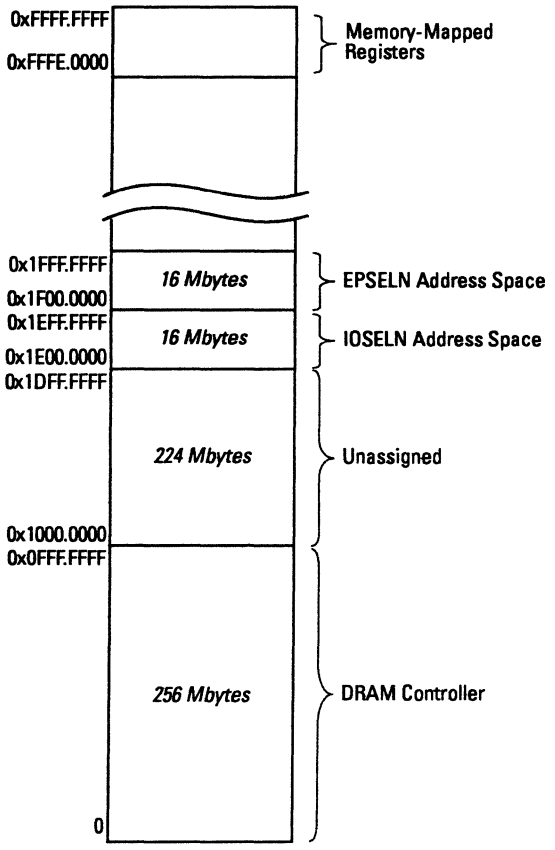
- The CW33000 offers a 4-Kbyte instruction cache (I-cache), instead of an 8-Kbyte I-cache.
- The 1-Kbyte data cache (D-cache) is optional; the user may choose to disable the D-cache controller and not include the data cache SRAM in a design.
- The user may include an on-chip memory in a design.

The impact of the I-cache and D-cache changes are described in more detail in the next section of this chapter, "Cache Operation."

Any CW33000-based design that includes an on-chip memory (described in Chapter 4 of this manual) must provide dedicated memory address space for the on-chip memory. Figure 2.1 shows the CW33000 memory map. The user should select the address space location appropriately, depending on whether on-chip memory will be used for user or kernel mode operations. Note that the user should position the on-chip memory address space

at either the bottom or top of the appropriate address space to leave the maximum amount of contiguous unassigned address space for other uses.

Figure 2.1
The CW33000 Memory Map



2.5 Cache Operation

As mentioned previously, the CW33000 I-cache and D-cache configurations deviate somewhat from those of the LR33000; these differences are discussed below. Fundamental I-cache operation is, however, as described in Chapter 6 of the *LR33000 Self-Embedding Processor User's Manual*; Chapter 6 also contains a description of the D-cache operation. In addition to the fundamental cache operation information presented in Chapter 6, please note the following information concerning cache operation during block fetch transactions.

The CW33000 automatically caches the data (if the system includes a D-cache) or instructions fetched during a block fetch transaction as follows. The CW33000 requests a block fetch transaction by asserting BFREQP

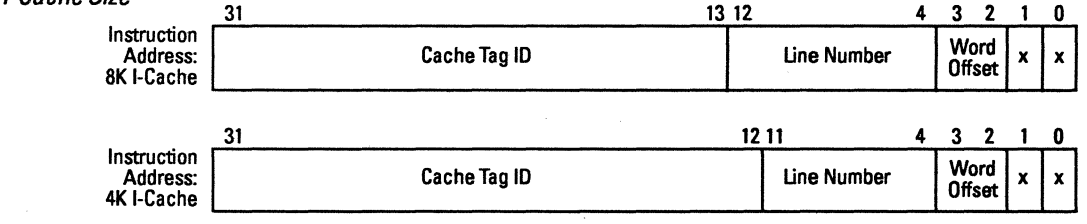
whenever a cache miss occurs. If the fetch is from the CW33000 DRAM Controller address space and the Controller is configured for block fetch transactions (that is, the BFD bit in the Configuration Register is set to zero), then the Controller acknowledges the request by asserting BFTCHN. The CW33000 then caches the fetched instruction or data, regardless of the state of the CACHDN signal. When the cache miss is in non-DRAM address space, the external memory system asserts BFTCHN to acknowledge the request and the CW33000 then caches the fetched instruction or data, regardless of the state of the CACHDN signal. If either the internal or external memory system does not acknowledge the block fetch request, the CW33000 does not perform the block fetch; instead, it uses the state of CACHDN to determine whether to cache the fetched word. If the memory access us in kseg1 (non-cacheable address space), the CW33000 does not request a block fetch and the fetched word is never cached.

I-Cache Differences

The CW33000 I-cache size is 4 Kbytes. Changing the size of the I-cache impacts the Instruction Address fields as explained below. Note that the I-cache word width does not change.

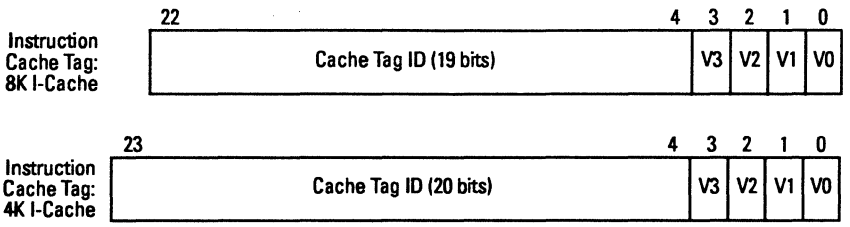
Figure 2.2 shows the Instruction Address formats for the 8K and 4K I-caches; notice that the Cache Tag ID gets larger and the Line Number smaller as the cache size decreases. The Line Number is smaller because the 4K I-cache has half the number of lines of the 8K I-cache. The Cache Tag ID gets larger because the CW33000 address space has twice as many lines, relative to the 4K I-cache, as it did relative to the 8K I-cache.

Figure 2.2 Instruction Address Format as a Function of I-Cache Size



Because the Instruction Cache Tag consists of both the Cache Tag ID and the valid bits, its format depends on the I-cache size as well. The Instruction Cache Tag changes as shown in Figure 2.3.

Figure 2.3 Instruction Cache Tag Format as a Function of I-Cache Size



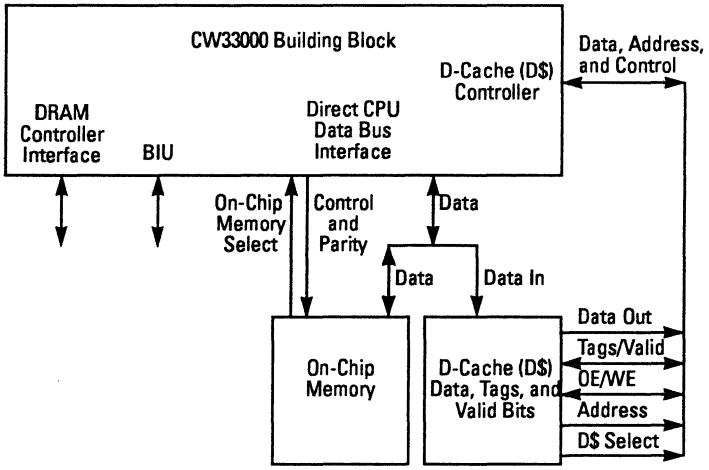
Implementation Note

Note that the I-cache tag and valid bits must be initialized during CW33000 simulation, whether or not they are used in the simulation.

D-Cache Differences

The D-cache, as mentioned previously, is optional in CW33000-based designs. Figure 2.4 shows the details of the D-cache interconnection to the CW33000.

Figure 2.4 CW33000 D-Cache Configuration



Note that at reset, hardware sets the DCD (Data Cache Disable) bit, bit 22 in the Configuration Register, to zero (D-cache enabled). Therefore, the Reset Handler *must set DCD to one* to disable all D-cache operations when no D-cache is present in a CW33000-based design. When DCD equals one, the CW33000 loads/stores all data directly from/to the addressed memory (off-chip memory or, if present, on-chip memory). Also, when DCD equals one, the DBS (Data Cache Block Size) bits, 18 and 19 in the Configuration Register, are ignored.

In systems with no D-cache, the state of the CACHDN (Cacheable Data) signal only affects the cacheability of data for the I-cache. In addition, the

DCTSTP (Data Cache Access) signal must be tied LOW in systems with no D-cache.

Implementation Note

If the system design includes a D-cache, then the D-cache tag and valid bits must be initialized during CW33000 simulation, whether or not they are used in the simulation.

2.6 Counter/Timers

The CW33000 counter/timer operation is identical to that in the LR33000, as described in Chapter 7 of the *LR33000 Self-Embedding Processor User's Manual.*

2.7 Signal Definitions

The CW33000 signal names differ from those of the LR33000. Although the signal names are similar, the CW33000 signal names do not use overbars; instead they append an N to all active-LOW signal names and a P to all active-HIGH signal names. In addition, many of the LR33000 bidirectional signals are separated into input and output signals in the CW33000. Refer to Chapter 3 of this manual for complete signal descriptions.

2.8 DRAM Controller Operation

The CW33000 DRAM Controller operation is identical to that in the LR33000, as described in Chapter 9 of the *LR33000 Self-Embedding Processor User's Manual.*

2.9 I/O Interface Operation

The I/O interface operation differs from that in the LR33000. As mentioned above, most of the LR33000 bidirectional signals are separate input and output signals on the CW33000. In addition, the CW33000 includes the Direct CPU Data Bus Interface (DCDBI), while the LR33000 does not. Refer to Chapter 5 of this manual for a complete I/O interface description.

2.10 Specifications

The AC timing for the CW33000 is somewhat similar to that of the LR33000, primarily because many of the signal dependencies are the same in the standard part and the building block. Changes at the periphery of the CW33000 building block, however, including the lack of input, output, and bidirectional buffers and the addition of the DCDBI, have altered the timing. Chapter 6 of this document contains complete AC timing for the CW33000, including the DCDBI timing, as well as specific loading information for all inputs and outputs.

Chapter 3
Signal Definitions

This chapter describes the signals that comprise the CW33000's bit-level interface to other building blocks. The signals are grouped by function; within each functional group, the signals are described in alphabetical order by mnemonic. Each signal definition contains the mnemonic and the full signal name. Note that in the signal descriptions, the verb *assert* means to drive TRUE or active. The verb *deassert* means to drive FALSE or inactive.

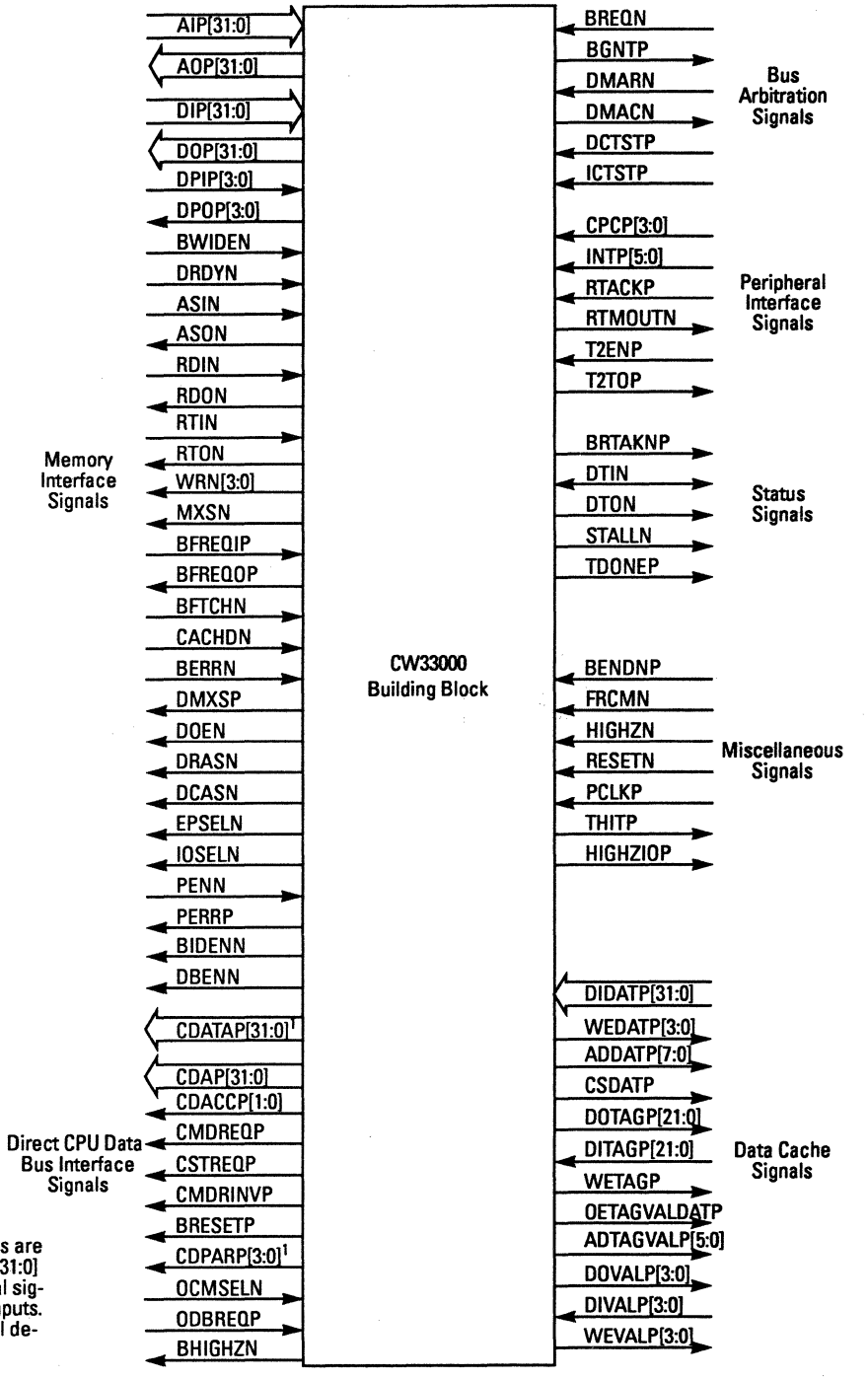
The signal names used in this document are those used in the design database. The signal naming convention is that signal names for active HIGH signals end in P (positive) and signal names for active LOW signals end in N (negative).

The chapter is divided into these sections:

- Memory Interface Signals
- Direct CPU Data Bus Interface Signals
- Bus Arbitration Signals
- Peripheral Interface Signals
- Status Signals
- Miscellaneous Signals
- Data Cache Signals

Figure 3.1 shows the logic symbol for the CW33000, and Table 3.1 lists the CW33000 signal names and their equivalent signal names on the LR33000. Note that various signals that are bidirectional on the LR33000 (for example, the data bus D[31:0] and the address bus A[31:0]) are separate input and output signals on the CW33000 (DIP[31:0] and DOP[31:0], AIP[31:0] and AOP[31:0]). A CoreWare-based design should include bidirectional buffers on all of these separate inputs and outputs, controlled by the appropriate enable control signals. Figure 3.2 shows some sample bidirectional buffer configurations. Table 3.2 summarizes the buffer enable control signals and the I/O signals they control.

Figure 3.1
CW33000 Logic Symbol



Note 1: Though these signals are shown as outputs, CDATAP[31:0] are functionally bidirectional signals and CDPARP[3:0] are inputs. Refer to the individual signal descriptions for details.

Table 3.1
CW33000/LR33000 Signal Naming

CW33000 Signal Names	*LR33000 Equivalent*	*Signal Summary*
Memory Interface		
AIP[31:0], AOP[31:0]	A[31:0]	Input and Output Address Buses
ASIN, ASON	$\overline{\text{AS}}$	Input and Output Address Strobe
BERRN	$\overline{\text{BERR}}$	Bus Error
BFREQIP, BFREQOP	BFREQ	Input and Output Block-Fetch Request
BFTCHN	$\overline{\text{BFTCH}}$	Block Fetch
BIDENN	—	Enable (Low) or 3-State (High) MXSN, WRN[3:0], AOP[31:0], ASON, RTON, DTON, BFREQOP
BWIDEN	$\overline{\text{BWIDE}}$	Byte-Wide Port
CACHDN	$\overline{\text{CACHD}}$	Cacheable Data
DBENN	—	Enable (Low) or 3-State (High) DOP[31:0], DPOP[3:0]
DCASN	$\overline{\text{DCAS}}$	DRAM Column Address Strobe
DIP[31:0], DOP[31:0]	D[31:0]	Input and Output Data Bus
DMXSP	DMXS	DRAM Mux Select
DOEN	$\overline{\text{DOE}}$	DRAM Output Enable
DPIP[3:0], DPOP[3:0]	DP[3:0]	Input and Output Data Parity
DRASN	$\overline{\text{DRAS}}$	DRAM Row Address Strobe
DRDYN	$\overline{\text{DRDY}}$	Data Ready
EPSELN	$\overline{\text{EPSEL}}$	EPROM Select
IOSELN	$\overline{\text{IOSEL}}$	I/O Select
MXSN	$\overline{\text{MXS}}$	Memory Transaction
PENN	$\overline{\text{PEN}}$	Parity Enable
PERRP	PERR	Parity Error
RDIN, RDON	$\overline{\text{RD}}$	Input and Output Read Strobe
RTIN, RTON	$\overline{\text{RT}}$	Input and Output Read Transaction
WRN[3:0]	$\overline{\text{WR}}$[3:0]	Byte-Write Strobes
Data Cache Data, Tags, and Valid Bits		
ADDATP[7:0]	—	Data Address for the D-Cache
ADTAGVALP[5:0]	—	Address for the Cache Tags and Valid Bits
CSDATP	—	Chip Select for the D-Cache

(Sheet 1 of 3)

Table 3.1 (Continued)
CW33000/LR33000 Signal Naming

CW33000 Signal Names	*LR33000 Equivalent*	*Signal Summary*
Data Cache Data, Tags, and Valid Bits (continued)		
DIDATP[31:0]	—	Data Input from the D-Cache
DITAGP[21:0]	—	Cache Tags Input from the D-Cache
DIVALP[3:0]	—	Valid Bits Input from the D-Cache
DOTAGP[21:0]	—	Cache Tags Output to the D-Cache
DOVALP[3:0]	—	Valid Bits Output to the D-Cache
OETAGVALDATP	—	Output Enable for the D-Cache Data, Tags, and Valid Bits
WEDATP[3:0]	—	Data Write Enable (byte-wise) for the D-Cache
WETAGP	—	Tag Write Enable (byte-wise) for the D-Cache
WEVALP[3:0]	—	Valid Bit Write Enable (byte-wise) for the D-Cache
Bus Arbitration		
BGNTP	BGNT	Bus Grant
BREQN	$\overline{\text{BREQ}}$	Bus Request
DCTSTP	D$TST	D-Cache ($) Access
DMACN	$\overline{\text{DMAC}}$	DMA Cycle
DMARN	$\overline{\text{DMAR}}$	DMA Request
ICTSTP	I$TST	I-Cache ($) Access
Peripheral Interface		
CPCP[3:0]	CPC[3:0]	Coprocessor Condition
INTP[5:0]	INT[5:0]	Interrupt
RTACKP	RTACK	Refresh Timer Acknowledge
RTMOUTN	$\overline{\text{RTO}}$	Refresh Timeout
T2ENP	T2EN	Timer Two Enable
T2TOP	T2TO	Timer Two Timeout
Direct CPU Data Bus Interface		
BHIGHZN	—	Force CDATAP[31:0] and CDPARP[3:0] to High Impedance
BRESETP	—	Reset
CDACCP[1:0]	—	CPU Data Access Type
CDAP[31:0]	—	CPU Address Bus
CDATAP[31:0]	—	CPU Data Bus (bidirectional)

(Sheet 2 of 3)

Table 3.1 (Continued)
CW33000/LR33000 Signal Naming

CW33000 Signal Names	*LR33000 Equivalent*	*Signal Summary*
Direct CPU Data Bus Interface (continued)		
CDPARP[3:0]	—	CPU Data Bus Parity Bits
CMDREQP	—	CPU Memory Data Request
CMDRINVP	—	CPU Memory Data Request Invalidate
CSTREQP	—	CPU Store Request
OCMSELN	—	Direct CPU Data Bus Interface Selected (for OCMem)
ODBREQP	—	OCMem Data Bus Request
Status		
BRTAKNP	BRTKN	Branch Taken
DTIN, DTON	$\overline{\text{DT}}$	Input and Output Data Transaction
STALLN	$\overline{\text{STALL}}$	Stall
TDONEP	TDONE	Test Mode Transaction Done
Miscellaneous		
BENDNP	BENDN	Big Endian
FRCMN	$\overline{\text{FRCM}}$	Force a Cache Miss
HIGHZN	$\overline{\text{HIGHZ}}$	Force All Outputs to High Impedance
HIGHZIOP	—	Enable (Low) or 3-state (High) BGNTP, BRTAKNP, DCASN, DMACN, DMXSP, DOEN, DRASN, EPSELN, IOSELN, PERRP, RESETN, RTMOUTN, STALLN, TDONEP, THITP, T2TOP
RESETN	$\overline{\text{RESET}}$	Reset
PCLKP	SYSCLK	System Clock
THITP	THIT	Test Hit in the I-Cache or D-Cache

(Sheet 3 of 3)

Figure 3.2
Sample Buffer Configurations

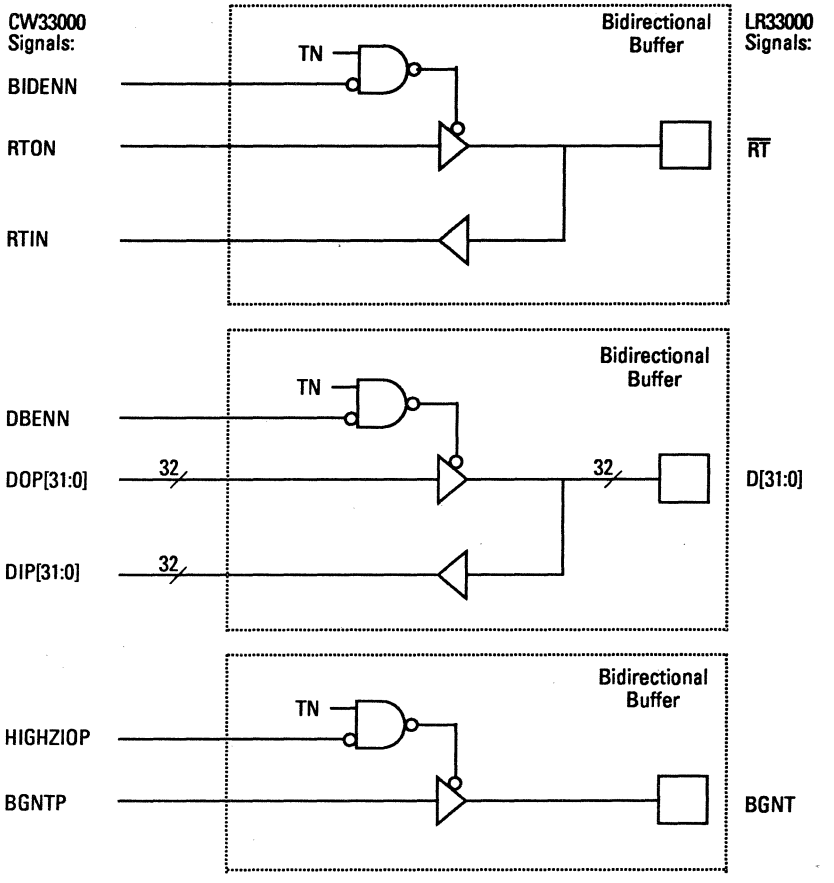


Table 3.2
Enable Control Signals

Enable Control Signal	*Affected Signals*
BIDENN	AOP[31:0], ASON, BFREQOP, DTON, MXSN, RDON, RTON, WRN[3:0]
DBENN	DOP[31:0], DPOP[3:0]
HIGHZIOP	BGNTP, BRTAKNP, DCASN, DMACN, DMXSP, DOEN, DRASN, EPSELN, IOSELN, PERRP, RTMOUTN, STALLN, TDONEP, THITP, T2TOP

3.1 Memory Interface Signals

This section describes the CW33000's memory interface, which consists of a 32-bit Address Bus, a 32-bit Data Bus, and the control signals required to regulate transactions with memory and I/O subsystems. The memory interface also contains the signals required to control DRAM arrays directly. The Direct CPU Data Bus Interface signals, which provide fast communication between the CW33000 CPU and a user-designed on-chip memory, are described in the next section.

AIP[31:0], AOP[31:0] Address Bus [31:0] Input, Output

When it is bus master, the CW33000 uses AOP[31:0] to transmit instruction and data addresses to the memory and peripherals. When another device owns the bus, the CW33000 snoops addresses generated by the bus master on AIP[31:0] (if CACHDN is asserted).

ASIN, ASON Address Strobe Input, Output

The address strobe signals indicate that a valid address is present on the Address Bus. During memory transactions as bus master, the CW33000 asserts ASON one-half cycle after it asserts MXSN.

When the CW33000 is not bus master, it monitors ASIN to determine when a valid address is present on the Address Bus. The CW33000 uses the addresses generated by other bus masters for cache snooping and invalidation, and for DRAM Controller operations.

BERRN Bus Error Input

External logic asserts this signal to indicate that an exceptional condition has occurred. When BERRN is asserted, the CW33000 terminates the current memory transaction and takes a Bus Error Exception. Note that the external logic should assert BERRN until the CW33000 deasserts ASON. Also, when the CW33000 is byte-gathering and a bus error occurs, the external system must assert BERRN during only the first byte transaction (as it would during a single byte fetch); it must not assert BERRN during the second, third, or fourth byte transactions.

BFREQIP, BFREQOP Block-Fetch Request Input, Output

The CW33000 asserts BFREQOP to indicate that it wants to perform a block-fetch transaction when a cache miss occurs, cache is enabled, and the data is in cacheable space. An external bus master asserts BFREQIP when it needs to use the CW33000 memory controller to perform a block-fetch transaction.

BFTCHN **Block Fetch** **Input**

When the CW33000 asserts BFREQOP, external memory control logic asserts BFTCHN to indicate that the CW33000 can expect a block transfer from memory. BFTCHN must be asserted for the duration of the block-fetch transaction, that is until ASON is deasserted.

When the CW33000's DRAM Controller is used to manage memory, the DRAM Controller acknowledges the BFREQOP signal internally and automatically. Unless there is an external device that is capable of block-fetch transactions, BFTCHN should be tied to VDD when using the internal DRAM Controller.

BIDENN **Bidirectional Enable** **Output**

This signal is the bidirectional enable control signal for these CW33000 outputs: AOP[31:0], ASON, BFREQOP, DTON, MXSN, RDON, RTON, WRN[3:0]. When LOW, it enables them; when HIGH, it 3-states them.

BWIDEN **Byte-Wide Port** **Input**

External logic asserts BWIDEN to indicate that the memory interface is eight bits wide. When BWIDEN is asserted, the CW33000 executes four memory cycles with sequential byte addresses, beginning with the effective address. (The byte-wide feature supports partial-word accesses, but always performs four memory cycles.) If the effective address is not modulo four, the CW33000 wraps around to get all of the bytes in the word in which the effective address falls.

For byte-wide transactions, the CW33000 always takes data from bits D[7:0]. During write transactions, the CW33000 ignores BWIDEN.

CACHDN **Cacheable Data** **Input**

External logic asserts this signal to indicate that the instruction or data fetched during the current cycle may be stored in the CW33000's on-chip cache. If CACHDN is not asserted, the CW33000 does not cache the instruction or data fetched during the current transaction unless the transaction is a block fetch. Instructions or data (in systems with a D-cache) fetched during a block fetch are always cached, regardless of the state of CACHDN.

When the CW33000 is *not* bus master, it monitors this signal to determine whether to perform bus snooping. When CACHDN and ASIN are asserted and RTIN is deasserted, the CW33000 monitors the Address Bus and invalidates the cached word specified by the address. When CACHDN is deasserted, the CW33000 does not perform bus snooping.

DBENN **Data Bus Enable** **Output**

The DBENN signal is the bidirectional enable control for the Data Bus and Data Parity Bus. When LOW, it enables them; when HIGH, it 3-states them.

DCASN **DRAM Column Address Strobe** **Output**

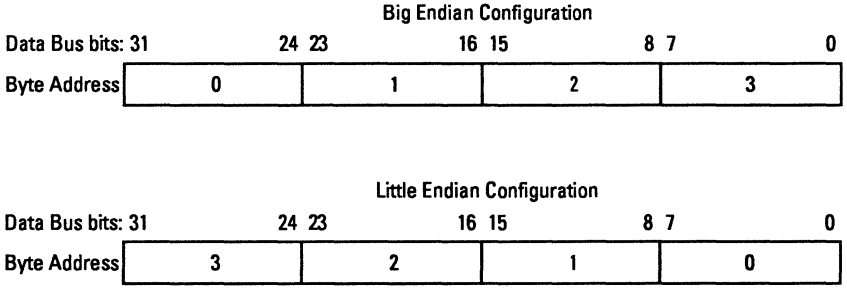
During a DRAM memory transaction, the CW33000's DRAM Controller asserts this signal to indicate that the address presented to the DRAM array is the column address.

DIP[31:0], DOP[31:0] Data Bus [31:0] **Input, Output**

The CW33000 uses the Data Bus to transfer data and instructions to (output) and from (input) memory and peripherals. When another device owns the bus, it may drive or sample these signals.

The CW33000's byte-ordering mode (endianness configuration) determines the relationship of the Data Bus lines to byte addresses. Figure 3.3 shows the relationship for both big and little endian modes.

Figure 3.3 Relationship of Data Bus Signal to Byte Addresses



The relationship of Data Bus signals to byte addresses must be observed when connecting byte-wide devices to the Bus. Byte-wide devices may reside on bits [31:24], [23:16], [15:8], or [7:0]. When using the CW33000's byte-wide port, however, the device must reside on bits [7:0] regardless of endianness.

DMXSP **DRAM Mux Select** **Output**

The CW33000's DRAM Controller outputs DMXSP for the Data Select input of the multiplexers required to multiplex the row and column address from the Address Bus on to the DRAMs' address bus. When HIGH, DMXSP indicates that the row address should be selected. When LOW, DMXSP indicates that the column address should be selected.

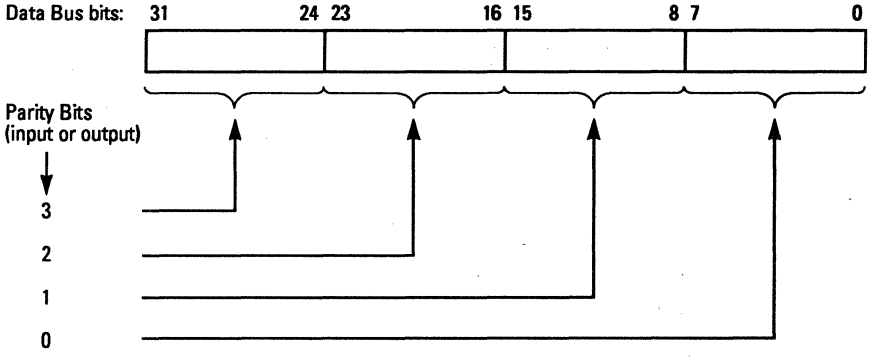
DOEN **DRAM Output Enable** **Output**

The CW33000's DRAM Controller asserts this signal to enable DRAM data outputs.

DPIP[3:0], DPOP[3:0] Data Parity [3:0] **Input, Output**

The data parity bits allow the CW33000 to check (input) and transmit (output) parity bits for the four data bus bytes as shown in Figure 3.4. The CW33000 uses even parity.

Figure 3.4
Data Bus Parity Bits



The CW33000 always generates parity for write data, and it checks parity for read data when PENN is asserted. When the CW33000 detects a parity error, it asserts the PERRP signal and posts the error in the CP0 Status Register.

DRASN **DRAM Row Address Strobe** **Output**

During a DRAM memory transaction, the CW33000's DRAM Controller asserts this signal to indicate that the address presented to the DRAM array is the row address.

DRDYN **Data Ready** **Input**

External logic asserts this signal to indicate that it can accept or is providing data. The assertion of DRDYN terminates the memory transaction. Because the CW33000 must wait for external logic to assert DRDYN before it can complete a memory transaction, slow devices can use DRDYN to stall the processor. However, when the automatic wait-state generator is enabled, the CW33000 asserts an internal version of DRDYN to end transactions.

EPSELN **EPROM Select** **Output**

The CW33000 asserts this signal to indicate that it is accessing the EPSELN address space (0x1F00.0000 to 0x1FFF.FFFF). EPSELN can be used as a chip select. If automatic wait-state generation is

enabled, the CW33000 generates an internal DRDYN signal a programmable number of cycles after the transaction begins. The external PROM logic may override the internal wait-state generator by asserting the DRDYN signal before the programmed number of wait states have passed.

IOSELN **I/O Select** **Output**

The CW33000 asserts this signal to indicate that it is accessing the IOSELN address space (0x1E00.0000 to 0x1EFF.FFFF). IOSELN can be used as a chip select. If automatic wait-state generation is enabled, the CW33000 generates its own DRDYN signal a programmable number of cycles after the transaction begins. The external I/O logic may override the internal wait-state generator and use the DRDYN signal to control the number of wait states.

MXSN **Memory Transaction Start** **Output**

The CW33000 asserts MXSN for one clock cycle at the beginning of a memory transaction to indicate the start of the transaction.

PENN **Parity Enable** **Input**

When asserted by external logic during a read transaction, the CW33000 performs byte-wise checking for even parity on the Data and Data Parity Buses. When deasserted, the CW33000 does not check parity. External logic that is connected to the CW33000's byte-wide port should never assert PENN.

PERRP **Parity Error** **Output**

The CW33000 asserts PERRP to indicate that it has detected a parity error during the just-completed memory transaction. PERRP can be tied to a interrupt input to cause a parity error exception.

RDIN, RDON **Read Strobe** **Input, Output**

The CW33000, as bus master, asserts RDON to indicate that memory may drive data onto DIP[31:0] during the current read transaction. An external bus master that is using the DRAM Controller asserts RDIN to signal a DMA read and causes the DRAM Controller to assert DOEN.

RTIN, RTON **Read Transaction** **Input, Output**

The CW33000 asserts RTON to indicate that the current memory cycle is a read transaction. The CW33000 deasserts RTON during write transactions. When the CW33000 is not the bus master, the CW33000 monitors RTIN for cache snooping and invalidation.

When the CW33000 is in cache test mode, an external bus master asserts RTIN to request DRAM access, to read or write the CW33000 internal cache.

WRN[3:0] Byte Write Strobes [3:0] **Output**

The CW33000 asserts these signals to indicate that there is valid data on the corresponding segment of the Data Bus. Figure 3.5 shows the relationship of the WRN signals to the Data Bus.

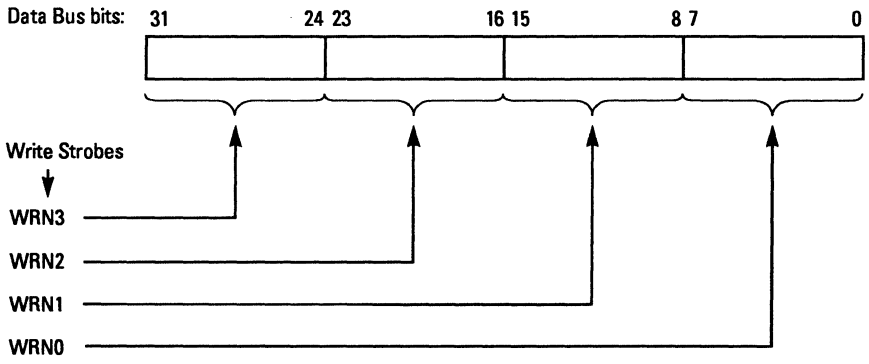


Figure 3.5 Relationship of the Write Strobes to the Data Bus

3.2 Direct CPU Data Bus Interface Signals

This section provides the signal definitions for the CW33000 Direct CPU Data Bus Interface (DCDBI). Because of the timing-critical nature of this interface to a user-designed on-chip memory, the signals are classified by when and how they are active, as follows.

Signal classification specifies when and how a signal occurs: in particular, during which clock phase the signal is active and whether it is a level-significant or edge-significant signal. Understanding the classification terminology requires a review of the CW33000 instruction execution pipeline.

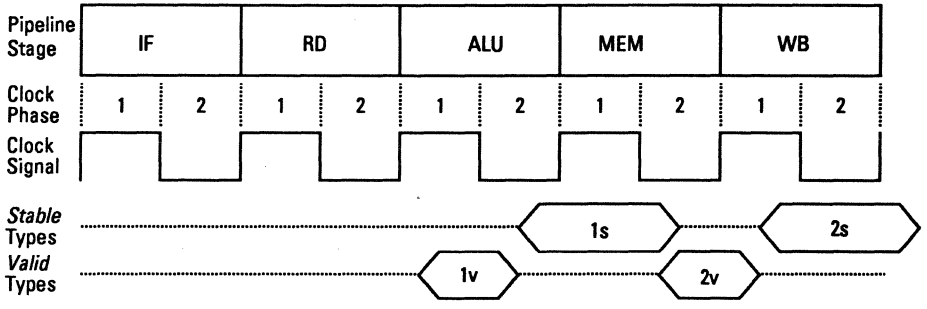
The pipeline consists of these five stages:

- IF - Instruction Fetch
- RD - Read operands and Decode instructions
- ALU - Perform ALU operations and compute data address
- MEM - Access MEMory
- WB - Write Back the results

Direct CPU Data Bus Interface accesses occur during the ALU and MEM stages. The DCDBI does not, however, need to know the state of the CPU pipeline.

As illustrated in Figure 3.6, each pipeline stage is one clock cycle in duration. Each stage encompasses two phases of the clock, where *phase 1* describes the half clock cycle during which the clock is HIGH, and *phase 2* refers to the half cycle when the clock is LOW.

Figure 3.6
Signal Types



Signals are classified as *1s* or *2s* if they are driven to a known or *stable* (*s*) state for the duration of either phase 1 or phase 2 of the clock, respectively. Similarly, signals are classified as 1v or 2v if they become *valid* (*v*), with some setup, at the end of either phase 1 or phase 2 of the clock. In the following signal descriptions, signals are classified according to this system.

BHIGHZN **3-state Enable Out** **Output**

The CW33000 BIU asserts this signal to force the on-chip memory controller off the CPU Data Bus by 3-stating the on-chip memory output drivers.

BRESETP **Reset Out** **Output**

The CW33000 asserts this RESET signal for 5 cycles following a LOW-to-HIGH transition of RESETN.

CDACCP[1:0] **CPU Data Access Type** **Output**

These two bits define the CPU data access type or the size of the data item for the load or store—word, tribyte, halfword, or byte—as follows. (2v)

CDACCP[1:0]	*Data Size*
0 0	Byte
0 1	Halfword
1 0	Tribyte
1 1	Word

CDAP[31:0] **CPU Data Address Bus** **Output**

CDAP[31:0] is the 32-bit, CPU data address bus. (2v)

CDATAP[31:0] CPU Data Bus **3-State Bidirect**

The 32-bit, bidirectional CPU data bus transfers data operands to and from the CPU. It must be driven only when the CW33000 is the bus master. Note that this bus is shown in the CW33000 design database as a 3-state output, although functionally it is a 3-state bidirectional bus. See Section 4.2, "Functional Description," for more details. (2v)

CDPARP[3:0] CPU Data Parity **3-State Input**

These four bits are the data parity bits for the shared (bidirectional) bus. The CPU does not monitor data parity. However, when the on-chip memory is the bus master it must drive these signals to some known value, zero in this case, because they cannot be left floating; otherwise, they should be 3-stated. Note that these signals are shown in the CW33000 design database as 3-state outputs, although functionally they are 3-state bidirectional. Refer to Section 4.2, "Functional Description," for more details. (2v)

CMDREQP CPU Memory Data Request **Output**

The memory data request signal validates the memory data address on the CDA bus. (2s)

CMDRINVP CPU Memory Data Request Invalidate **Output**

When the CPU asserts the CPU memory data request invalidate signal when an instruction exception occurs. If the exception occurs during the MEM cycle of an access of the on-chip memory, data writes must be disabled to maintain precise handling of instruction cancellation due to exceptions. (1s)

CSTREQP CPU Store Request Direction **Output**

The store request direction signal indicates whether the CPU intends to write information to on-chip memory (a CPU store) or read information from on-chip memory (a CPU load). (2s)

OCMSELN On-Chip Memory Selected **Input**

The on-chip memory system selected signal tells the BIU that the address on the CDA bus has selected the on-chip memory and that the BIU should ignore the transaction. (2v)

ODBREQP On-Chip Memory Data Bus Request **Input**

The on-chip memory system data bus request signal tells the CW33000 bus interface unit (BIU) that the Direct CPU Data Bus Interface will drive data onto the data bus in phase 2 of the next clock cycle. (2v)

3.3 Bus Arbitration Signals

This section describes the CW33000 bus arbitration signals. The CW33000 memory interface functions as a general-purpose bus, and these signals allow intelligent peripherals to request bus mastership to perform DMA operations or access the CW33000's on-chip cache memories.

BGNTP **Bus Grant** **Output**

When the CW33000 grants the bus to a requesting device, it asserts BGNTP, BIDENN, and DBENN. When BGNTP is asserted, the requesting bus master can control the signals that are 3-stated or enabled by BIDENN and DBENN.

The CW33000 asserts BGNTP only when there is no on-going bus transaction. If BREQN is asserted during a transaction, the CW33000 asserts BGNTP after the transaction has completed. The requesting logic, therefore, should continue to assert BREQN until the CW33000 grants the bus by asserting BGNTP. If both external logic and the CW33000 request the bus simultaneously, external logic has priority.

When the CW33000 has released the bus to another master, it *snoops*, or monitors bus transactions, to determine if cache entries need to be invalidated. Note that after granting the bus, the CW33000 continues to run. If the CW33000 then needs to use the bus and it is granted to another device, the CW33000 stalls until the bus is available.

BREQN **Bus Request** **Input**

External bus masters assert this signal to request control of the Address and Data Buses and the relevant control signals.

DCTSTP **Data Cache Access** **Input**

When this signal is asserted, external logic can read and write the CW33000's data cache, if present. In data cache access mode, the external device controls the AIP[31:0], DIP[31:0], ASIN, and RTIN signals.

DMACN **DMA Cycle** **Output**

The CW33000 asserts this signal to acknowledge the DMARN and to indicate that the DRAM Controller has started the access for the requesting device. During such DMA accesses, the DRAM Controller manages the DRASN, DCASN, and DMXSP signals. The DMA device must drive the DRAM write strobes, address bus, and data bus; note that this configuration requires multiport RAMs.

During DMA operations by other devices, the CW33000 continues to control and use the Address and Data Buses and data parity bits. If both external logic and the CW33000 need to use the DRAM Controller, they take turns, switching control at the end of each memory transaction.

DMARN **DMA Request** **Input**
External logic asserts this signal to request use of the CW33000's DRAM Controller for a DMA access. The DMARN and DMACN signals are not related to the BREQN and BGNTP signals.

ICTSTP **Instruction Cache Access** **Input**
When this signal is asserted, external logic can read and write the CW33000's instruction cache. In instruction cache access mode, the external device controls the AIP[31:0], DIP[31:0], ASIN, and RTIN signals.

3.4 Peripheral Interface Signals

The CW33000 contains three counter/timers. Two of the three timers can be controlled via the peripheral interface signals described in this section.

CPCP[3:0] **Coprocessor Condition [3:0]** **Input**
The four Coprocessor Condition signals, CPCP[3:0], carry condition inputs that the CW33000 can separately test using coprocessor branch instructions. Note that the corresponding coprocessor usable bit (Cu[3:0]) in the Status Register must be set in order to test a condition input. Refer to the *LR33000 Self-Embedding Processor User's Manual*, Chapter 4, "Exception Processing," for a description of the Status Register and the Cu bits.

Although the CPCP inputs are tested using coprocessor branch instructions, the CW33000 hardware makes no assumptions about the actual source of these signals. The use of these condition inputs is, therefore, application-dependent.

INTP[5:0] **Interrupt [5:0]** **Input**
External logic asserts these signals to cause the CW33000 to take an interrupt exception. The state of these signals is reflected in the Cause Register's IP[5:0] field, which the interrupt exception routine uses to identify the source of the interrupt. Consequently, the interrupting logic should continue to assert the INTP signal until the exception routine has serviced the interrupt.

New interrupts are not recognized by the processor during an interrupt exception. However, the IP[5:0] field in Cause Register continues to reflect the state of INTP[5:0], so software can determine

whether external logic has requested any additional interrupts since the start of the interrupt exception.

The interrupt inputs can be individually disabled or *masked* by setting the appropriate bit in the CW33000's Status Register, as described in the *LR33000 Self-Embedding Processor User's Manual*, Chapter 4, "Exception Processing."

RTACKP **Refresh Timer Acknowledge** **Input**

External logic asserts RTACKP to indicate that it has completed the refresh operation initiated by the CW33000's assertion of RTMOUTN. The assertion of RTACKP causes the CW33000 to deassert RTMOUTN. External logic must assert RTACKP for two cycles to guarantee the Refresh Timer reset.

RTMOUTN **Refresh Timeout** **Output**

The CW33000 asserts this signal to indicate that the Refresh Timer has counted down from its preset initial value (RCOUNT) to zero. The CW33000 continues to assert RTMOUTN until external logic asserts RTACKP, but the Timer does not stop running; instead, it loads the RCOUNT value and continues to count.

When the DRAM Controller is enabled, the CW33000 asserts RTMOUTN to indicate that a refresh cycle is in progress. When the DRAM Controller is disabled, external DRAM control logic can use the assertion of RTMOUTN to start a refresh cycle.

Because the user cannot disable the Refresh Timer, there is no way to guarantee that RTMOUTN will stay deasserted. To guarantee, however, that RTMOUTN will be either asserted (LOW) or toggling, set RCOUNT to zero and drive the RTACKP signal LOW or HIGH. The following table summarizes the RTMOUTN states vis-a-vis RTACKP.

RCOUNT	*RTACKP*	*RTMOUTN*
ZERO	LOW	LOW
ZERO	HIGH	Toggle

T2ENP **Timer 2 Enable** **Input**

External logic asserts this signal to enable the operation of Timer 2. Timer 2 counts as long as T2ENP is asserted.

T2TOP **Timer 2 Timeout** **Output**
The CW33000 toggles T2TOP to indicate that the CW33000's Timer 2 has counted down from its preset value to zero. T2TOP is set LOW on cold and warm starts.

3.5 Status Signals

The CW33000 generates four signals that indicate its internal state. Those signals are described in this subsection.

BRTAKNP **Branch Taken** **Output**
When asserted, this signal indicates that the condition has been met for a branch instruction in the CW33000 pipeline. The CW33000 signals a branch during the ALU stage of the pipeline, which is when the control transfer occurs.

DTIN, DTON **Data Transaction** **Input, Output**
When bus master, the CW33000 asserts DTON to indicate that the current bus transaction is a data read or write. The CW33000 deasserts this signal to indicate that the current bus transaction is an instruction fetch.

When an external device is bus master (BGNTP asserted), the external device uses DTIN to select the block size field for block transactions. When DTIN is asserted, the DRAM Controller uses the Data Cache Block Size (DBS) to determine the number of words to transfer. When DTIN is deasserted, the DRAM Controller uses the Instruction Cache Block Size (IBS) field. For a complete description of the block size fields, refer to the Configuration Register subsection in Chapter 6 of the *LR33000 Self-Embedding Processor User's Manual.*

STALLN **CPU Pipeline Stall** **Output**
The CW33000 asserts this signal to indicate that it is in a stall state. Note that when the CPU asserts STALLN, the on-chip memory must follow the CPU stall protocol as described in Chapter 4 of this manual. Note also that STALLN is a performance-critical signal, so the designer must minimize the external load on STALLN to assure maximum performance. (1s)

TDONEP **Test Mode Transaction Done** **Output**
The CW33000 asserts this signal to indicate that a cache access mode read or write operation is complete.

3.6 Miscellaneous Signals

The CW33000 has six miscellaneous inputs, described next.

BENDNP **Big Endian Byte Ordering** **Input**

When RESETN is asserted, the BENDNP signal specifies the byte ordering for the CW33000: big-endian if BENDNP is tied HIGH, or little-endian if BENDNP is tied LOW. If the CW33000-based chip includes an on-chip memory, then this signal must also input to the on-chip memory. Refer to Section 4.4, "Data Formats and Addressing," for a complete description of the use of BENDNP to specify data formats. Note that BENDNP is a static signal; do not change its state when RESETN is deasserted.

FRCMN **Force Cache Miss** **Input**

When external logic asserts FRCMN, the CW33000 detects a cache miss on the current instruction fetch. This signal is intended for test and in-circuit emulation purposes.

HIGHZN **High Impedance** **Output**

This HIGH-Z signal controls the 3-state enable signals DBENN, BIDENN, and HIGHZIOP. In addition, HIGHZN selects between warm and cold processor starts after reset. When HIGHZN and RESETN are asserted together, the CW33000 executes a cold start after RESETN is deasserted. When RESETN is asserted alone, the CW33000 executes a warm start. The DRAM Controller is *not* initialized on a warm start and the DRAM contents remain valid.

HIGHZIOP **High Impedance I/O Enable** **Output**

This signal is the bidirectional enable control signal for these CW33000 outputs: BGNTP, BRTAKNP, DCASN, DMACN, DMXSP, DOEN, DRASN, EPSELN, IOSELN, PERRP, RESETN, RTMOUTN, STALLN, TDONEP, THITP, T2TOP. When HIGH, HIGHZIOP 3-states these signals; when LOW, it enables them.

RESETN **Reset** **Input**

A LOW-to-HIGH transition of RESETN initializes the processor and initiates a non-maskable RESETN exception. Driving RESETN LOW does not halt the processor or cause any other action. Only a LOW-to-HIGH transition of RESETN initializes the processor.

The RESET Exception should vector the processor to a pre-defined bootstrap routine (typically, PROM-resident) that initializes the system. The RESET jump address is different from that used by interrupts, and RESETN (unlike interrupts) forces the CW33000 out of any wait loops. Refer to the *LR33000 Self-Embedding Processor*

User's Manual, Chapter 4, "Exception Processing," for a detailed description of the CW33000's response to the RESET Exception. Note that RESETN need not be synchronized with PCLKP, but PCLKP must be active for RESETN to take effect.

PCLKP **Clock Signal** **Input**
PCLKP is the CW33000 system clock.

THITP **Test Hit** **Output**
The CW33000 asserts THITP during a read of its instruction or data caches in cache access mode to indicate that the cache tags matched the address of the read. This signal is intended for test purposes.

3.7 Data Cache Signals

This section describes the signals that interface the CW33000 with the optional data cache (D-cache). The D-cache consists of three parts: the data, the tags, and the valid bits. The signals for each part utilize these naming conventions:

- AD in a signal name refers to the D-cache Address
- DI refers to data in to the CW33000 from the D-cache
- DO refers to data out from the CW33000 to the D-cache
- OE is an output enable
- WE is a write enable
- DAT is appended to data signals
- TAG is appended to tag signals
- VAL is appended to valid bit signals

The signals are described briefly below. Recall that the data input to the D-cache is through the Direct CPU Data Bus Interface, while the D-cache data output is to the D-cache controller.

ADDAT[7:0] **Data Address** **Output**
These eight bits are the data address from the CW33000 for the D-cache.

ADTAGVAL[5:0] **Cache Tag and Valid Bit Address** **Output**
These six bits are the address outputs from the CW33000 to the D-cache for the Cache Tags and Valid Bits.

CSDAT **D-Cache Select** **Output**
This chip select signal selects the D-cache for access.

DOTAG[21:0] Cache Tag Out **Output**

These 22 bits provide the Cache Tags to the D-cache.

DOVAL[3:0] Valid Bit Out **Output**

These four signals provide the Valid Bits for the D-cache.

DIDAT[31:0] Data Bus In **Input**

These signals are the data bus inputs from the D-cache.

DITAG[21:0] Cache Tag In **Input**

These D-cache Cache Tags signals are input from the D-cache.

DIVAL[3:0] Valid Bit In **Input**

These four Valid Bits are input from the D-cache.

WEDAT[3:0] Data Write Enable **Output**

These four bits provide the byte-wise write enable for the data part of the D-cache.

OETAGVALDAT Tag and Valid Bit Write Enable **Output**

This signal is the output enable for the Cache Tags and Valid Bits in the D-cache.

WETAG Cache Tag Write Enable **Output**

This signal is the write enable for the Cache Tags part of the D-cache.

WEVAL[3:0] Valid Bit Write Enable **Output**

These four signals provide the write enable for the Valid Bits part of the D-cache.

Chapter 4
Direct CPU Data Bus Interface

The CW33000 provides two interfaces to instruction and/or data operands. The Bus Interface Unit (BIU) offers access to a wide variety of instruction and data sources of various speeds, including fast or slow ROM or DRAM, as well as the flexibility to program the timing parameters required to access each source. The Direct CPU Data Bus Interface (DCDBI) is a fast, dedicated interface for data operands only. It is much less flexible than the BIU, but it is far faster. This chapter describes the Direct CPU Data Bus Interface and explains its use.

Sections in this chapter are:

- Overview
- Functional Description
- DCDBI Timing
- Data Formats and Addressing

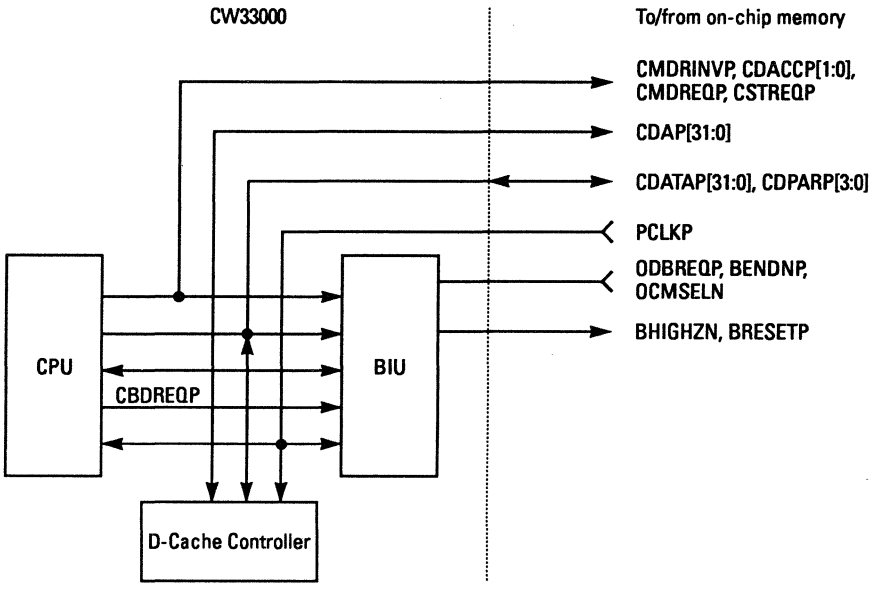
4.1 Overview

The CW33000 BIU offers access for both data and instruction operands from SRAM, DRAM, PROM, and I/O, as well as flexibility in programming the timing components for each type of data access. However, accessing data through the BIU requires more than one CPU cycle and therefore generates CPU pipeline stalls.

The Direct CPU Data Bus Interface provides direct communication between the CPU inside the CW33000 and a customer-designed on-chip memory that is physically on-chip but not part of the CW33000, without involving the BIU. This dedicated interface enables the CPU to access on-chip memory data in a single cycle with no CPU pipeline stalls, at the expense of data source selection and timing flexibility.

Figure 4.1 shows a block diagram of the Direct CPU Data Bus Interface in the CW33000.

Figure 4.1
Direct CPU Data Bus Interface



The important characteristics of the Direct CPU Data Bus Interface are:

- The Direct CPU Data Bus Interface can supply only data operands to the CPU, even though the CW33000 CPU has separate address and data buses for both data operands and instruction operands.
- The user must assign an address space to the on-chip memory. This on-chip memory address space and the address space that is accessible through the BIU must be separate.
- The access time for the user-designed on-chip memory must be less than the clock cycle time of the CW33000.
- The on-chip memory system must decode its addresses from the CPU Data Address Bus, CDAP[31:0], and assert the on-chip memory selected signal, OCMSELN, to signal the CW33000 when an address match occurs.
- The CPU asserts STALLN to notify the on-chip memory of a CPU pipeline stall. If the CPU stalls during an access to the on-chip memory, then the on-chip memory must stay synchronized with the stalled CPU pipeline.
- The on-chip memory system must recognize and handle exceptions in a precise manner.

- The on-chip memory system must handle 32-bit words and must also have byte-write capability. The CW33000 specifies both the data type (byte, halfword, tribyte, or word) to the on-chip memory as well as whether the data is big-endian or little-endian.

4.2 Functional Description

This section provides a functional overview of the Direct CPU Data Bus Interface (DCDBI). It discusses these topics:

- Memory Accesses
- Reset Handling
- CPU Stall Handling
- Exception Handling

The remaining sections in this chapter provide a more detailed description of the DCDBI.

Memory Accesses

Recall the CPU pipeline stages:

IF	RD	ALU	MEM	WB

Recall also that the on-chip memory must be assigned an address space, as discussed in Section 2.4, "Memory System Architecture." This address space and the remaining address space that is accessible through the BIU must be separate.

During the ALU stage of the pipeline, the CPU computes and presents the address of data operands for access on CDAP[31:0]. The on-chip memory decodes the memory address, and the CPU and the on-chip memory present the appropriate control signals:

- CDACCP[1:0], from the CPU, specifies the data type: byte, halfword, tribyte, or word.
- CSTREQP, from the CPU, defines whether the access is a load or store.
- CMDREQP, from the CPU, validates the memory data address.
- Either CDBREQP (an internal signal from the CPU to the BIU) or ODBREQP (from the on-chip memory) notifies the BIU that a bus master will drive the data bus on the following cycle.

- OCMSELN, from the on-chip memory, notifies the BIU that the on-chip memory has been addressed. Because the BIU is unaware of the on-chip memory's address space, it attempts to handle any memory request for data. When the on-chip memory asserts OCMSELN, the BIU does not respond to the CPU's memory data request.

During the MEM stage, either the CPU drives data onto the data bus (CDATAP[31:0]) for a store to the on-chip memory or the on-chip memory drives the data for a CPU load. The current bus master must also drive the data parity bus, CDPARP[3:0], during this stage. The CPU does not check data parity; however, because data parity is a 3-state bus, it must be driven to some known value. In particular, when the on-chip memory is the data bus master during a CPU load from on-chip memory, it must drive CDPARP[3:0] at the same time it drives CDATAP[31:0]. At the end of phase 2 of the MEM stage, either the CPU or on-chip memory latches the data on the data bus.

If a stall occurs during the ALU stage, the CPU asserts the stall signal STALLN as a 1s signal of the following MEM stage; STALLN prevents the CPU and on-chip memory from latching the data. One or more stall cycles then occur, and the CPU and the on-chip memory must re-drive all of the above control signals for the duration of the stall.

If an exception occurs during the ALU stage, the CPU asserts a memory data request invalidate signal (CMDRINVP) as a 1s signal in the MEM stage to inhibit the latching of CPU data into the on-chip memory.

If a stall occurs during the MEM stage, the stall signal is asserted as a 1s signal of the following WB stage, and both the data bus and the data parity bus must be re-driven for the duration of the stall.

Refer to Section 4.3 for details on CPU store/load operations to/from the on-chip memory.

Note that all data transfers between the CPU and the on-chip memory must be 32-bits wide. The data access bits, CDACCP[1:0], and the two low-order address bits, CDAP[1:0], define which bytes are valid in the 32-bit word, as discussed in Section 4.4.

Implementation Note

CDATAP[31:0] and CDPARP[3:0] are pictured in Chapter 2 and shown in the CW33000 design database as 3-state outputs, although functionally CDATAP[31:0] is a 3-state bidirectional bus and CDPARP[3:0] are 3-state inputs. This approach is required because in designs with no on-chip mem-

ory and no D-cache these signals are left unconnected, and leaving 3-state inputs or bidirects unconnected generates errors in the LCMP program and causes the LLINK program to abort. Follow these guidelines to avoid problems. For systems with on-chip memory and D-cache, (that is, systems that use the signals), connect the signals to 3-state drivers. The LSED program will report *WARNING – net drives no inputs* (which is not a problem), the LCMP program will report *WARNING – multiple output connections* (which is also not a problem), and the LLINK program has no problem with this solution. For systems that do not use on-chip memory or D-cache, leave the signals unconnected. Because the signals are 3-state outputs, this approach generates no conflicts or errors in the LCMP, LSED, and LLINK programs.

Implementation Note

Note that STALLN is a performance-critical signal, so any external loading on STALLN affects the CW33000 internal nodes. Take care, therefore, to keep the external loading on STALLN at an absolute minimum to assure maximum performance; that is, buffer STALLN if used and place the buffer as close as possible to the CW33000 core at layout.

Reset Handling

When the CW33000 is reset, it asserts the BRESETP signal to allow the user to initialize the on-chip memory controller.

CPU Stall Handling

Although a load/store from/to on-chip memory does not cause a CPU stall, CPU stalls that affect the on-chip memory can occur. When a CPU stall occurs during an on-chip memory access, the on-chip memory system must recognize the stall and keep the load or store operation in step with the stalled CPU pipeline for the duration (one or more cycles) of the stall. The fundamental guideline of stall handling is that during a stall, the CPU and on-chip memory must re-do whatever they were doing prior to the stall; that is, they must re-drive all data and control signals. Note that the on-chip memory system is affected only by CPU stalls that occur during the ALU and MEM cycles, because the CPU issues the on-chip memory access control signals in phase 2 of the ALU cycle and concludes the access in phase 2 of the MEM cycle.

The following descriptions summarize how the on-chip memory system must handle stalls during both CPU loads and stores. Section 4.3, "DCDBI Timing," provides a more detailed description of stall handling.

Stall During a RD Cycle – The sequence of events for a stall that occurs during a RD cycle is as follows.

- During a normal (non-stalled) on-chip memory access, the ALU cycle for the access directly follows the RD cycle. During a RD-cycle stall, however, the CPU address and control signals that would normally occur in that ALU cycle are postponed and, instead, whoever was driving the bus during the RD cycle drives it again with the previous data.
- The on-chip memory access occurs during the first RUN cycle (ALU) after the stall.

Stall During an ALU Cycle – A stall during an ALU cycle can affect either a CPU store to the on-chip memory or a CPU load from the on-chip memory. When a stall occurs during the ALU cycle of a store, the following events must occur:

- STALLN disables the storing of new data into the on-chip memory.
- The CPU re-drives the address and control signals for this on-chip memory instruction during the stall.
- The on-chip memory latches the data during the first RUN cycle (that is, the MEM cycle) after the stall.

When a stall occurs during a CPU load from on-chip memory:

- The on-chip memory does not drive data onto the data bus, because the on-chip memory bus drivers are high impedance during the stall.
- The bus master who drove the data bus during the last RUN cycle re-drives the bus with the previous data throughout the stall.
- The CPU and on-chip memory re-issue the address and control signals for the on-chip memory instruction during the stall.
- The on-chip memory must drive the data onto the data bus during the first RUN cycle (the MEM cycle) after the stall.

Stall During a MEM Cycle – Stalls that occur during a MEM cycle can affect both a CPU store to on-chip memory and a CPU load from on-chip memory. For a MEM-cycle stall during a CPU store to on-chip memory:

- The store to the on-chip memory completes successfully during the initial MEM cycle, which is when the stall occurs, or at any time during the subsequent stall cycles.

- The CPU redrives the on-chip memory data from the ALU cycle during the stall, but it does not re-drive the on-chip memory address.

However, for a MEM-cycle stall during a CPU load from on-chip memory:

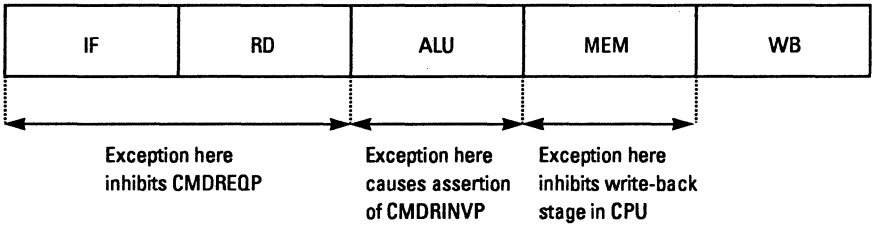
- The on-chip memory must drive valid on-chip memory data to the CPU during the MEM cycle in which the stall occurs. In addition, the on-chip memory must acknowledge the CPU stall and re-drive the data bus, parity bus, and appropriate control signals for the duration of the stall.

Exception Handling

The on-chip memory system must handle instruction cancellation due to an exception in a precise manner. Note that only an exception that occurs during the ALU cycle of a CPU store can cancel an on-chip memory access.

As summarized in Figure 4.2, if a CPU exception occurs prior to the ALU stage of an on-chip memory transaction instruction, the CPU inhibits the memory data request (CMDREQP) for the exceptional instruction, effectively cancelling the on-chip memory access because the on-chip memory never sees a valid request.

Figure 4.2 Effects of Exceptions During CPU Pipeline Stages



If a CPU exception occurs during the ALU stage of an on-chip memory transaction instruction, the CPU asserts the memory data request invalidate signal (CMDRINVP). If the exceptional instruction is for a CPU store to the on-chip memory, then the on-chip memory must disable the store at this point; it is not required, however, to disable a CPU load. If a stall follows the exception, the CPU asserts the CMDRINVP signal for the duration of the stall.

If an exception occurs during the MEM stage of a load from on-chip memory, the CPU inhibits the loading of the on-chip memory data to its general registers during the WB stage. Note that the on-chip memory must drive the data bus in this case to keep it from floating. An exception that occurs

during the MEM stage of a store to on-chip memory does not affect the store.

4.3 DCDBI Timing

This section discusses in detail the CW33000 DCDBI. It includes detailed timing diagrams and descriptions for CPU load and store functions to and from the on-chip memory; it also includes both stall and exception timing information.

The section illustrates these transactions:

- CPU Store Timing (Figure 4.3)
- CPU Load Timing (Figure 4.4)
- CPU Store, Stall During ALU Stage (Figure 4.5)
- CPU Load, Stall During ALU Stage (Figure 4.6)
- CPU Store, Stall During MEM Stage (Figure 4.7)
- CPU Load, Stall During MEM Stage (Figure 4.8)
- CPU Store with an Exception During the ALU Stage (Figure 4.9)
- CPU Store with an Exception (ALU Stage) and a Stall (MEM Stage) (Figure 4.10)

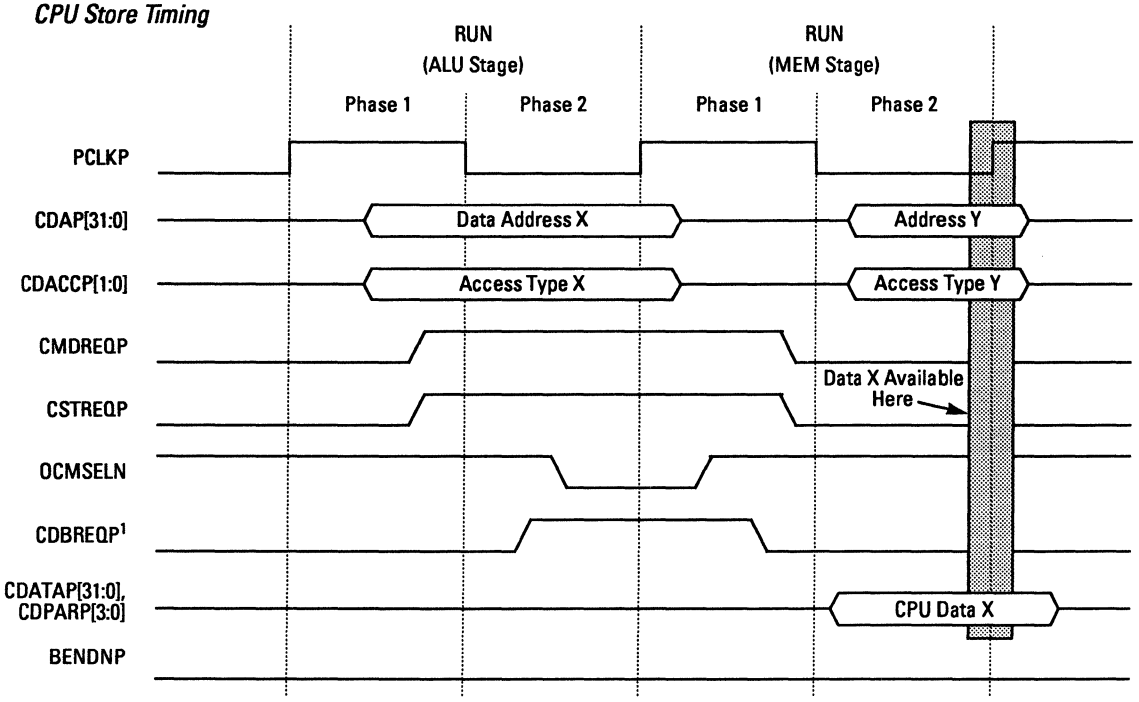
A detailed explanation of each transaction is followed by a figure that illustrates the transaction. Note that references in the figures to *OCMem* are to the customer-designed on-chip memory.

CPU Store

- The CPU places the address for the on-chip memory on CDAP[31:0], and sets the value of CDACCP[1:0] to specify to the on-chip memory the data size for the access.
- The CPU asserts both CMDREQP to validate the data address on CDAP[31:0] and CSTREQP to specify the memory request direction, in this case a store (HIGH). For a load request, the CPU drives CSTREQP LOW. Note that the CPU drives both CMDREQP and CSTREQP in the ALU stage to specify an on-chip memory transaction in the subsequent MEM stage.
- The on-chip memory must assert OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.

- The CPU asserts CDBREQP, an internal signal, to notify the BIU that the CPU intends to drive both the data bus, CDATAP[31:0], and the data parity bus, CDPARP[3:0], during the following MEM stage.
- The CPU drives data onto both the data bus and the data parity bus as phase-2-valid (2v) signals during the MEM stage.
- The on-chip memory latches the CPU data at the end of the MEM stage. Note that the state of the CW33000 BENDNP signal specifies the endianness of the data, which is determined during initialization and remains unchanged during operation.

Figure 4.3
CPU Store Timing

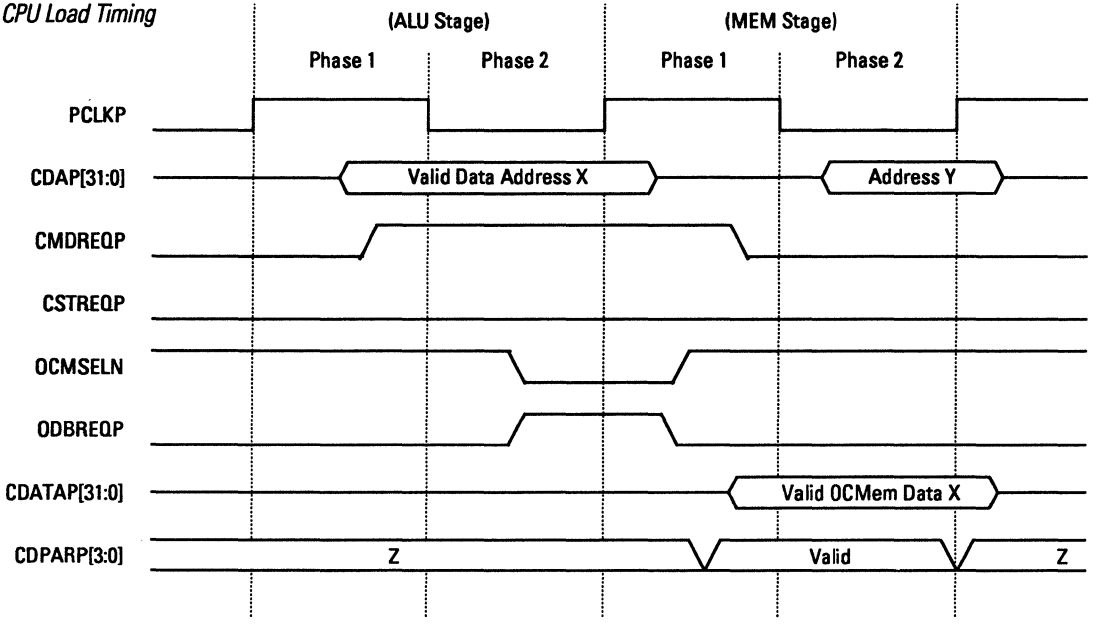


Note:
1. Internal signal shown for reference only.

CPU Load

- The CPU places the address for the on-chip memory on CDAP[31:0]. Note that CDACCP[1:0] (not shown) does not affect the on-chip memory during a CPU load because the on-chip memory must always supply a single 32-bit word during a CPU load.
- The CPU asserts CMDREQP to validate the memory data address on the CDA bus.
- The CPU drives CSTREQP LOW to specify the memory request direction, in this case, a load. Recall from the previous example that the CPU asserts CSTREQP (HIGH) to request a store.
- The on-chip memory must assert OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The on-chip memory must assert ODBREQP to notify the BIU that the on-chip memory intends to drive both CDATAP[31:0] and CDPARP[3:0] during the following MEM stage.
- The on-chip memory must drive data on both the data bus and the data parity bus as 2v signals during the MEM stage. Note that while the CPU does not check parity, the on-chip memory must drive CDPARP[3:0] to some value to keep them from floating to an unknown state.

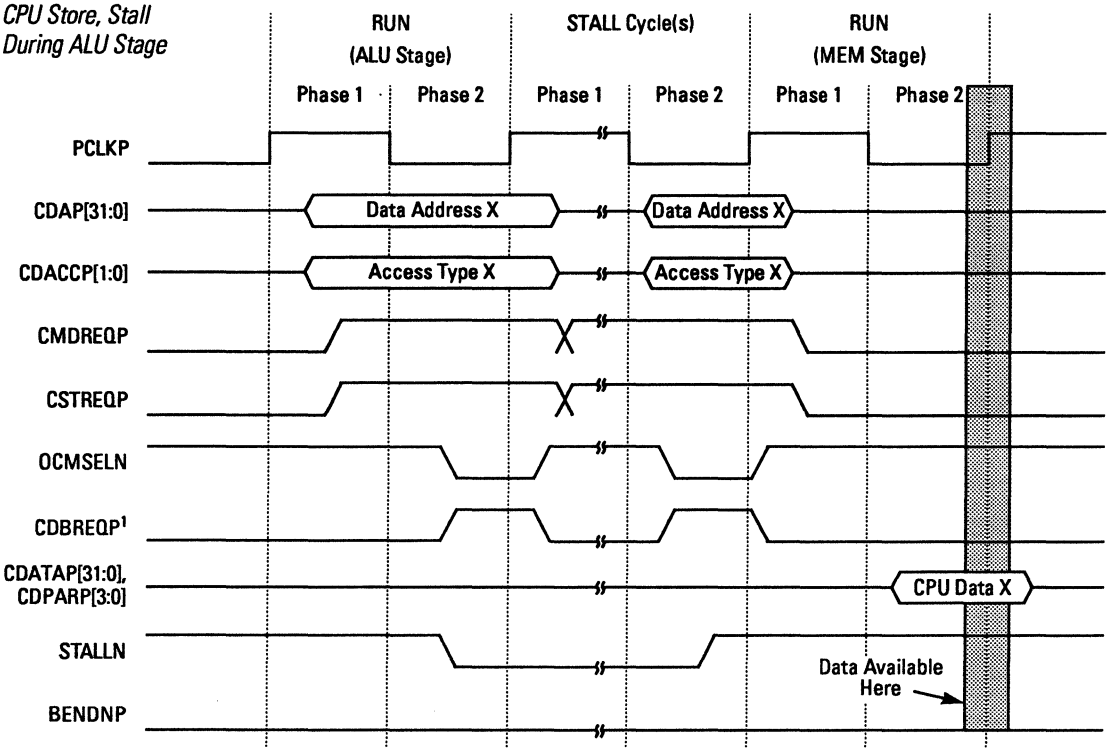
Figure 4.4
CPU Load Timing



CPU Store With a Stall During the ALU Stage

- The CPU places the address for the on-chip memory on CDAP[31:0] and sets CDACCP[1:0] to specify for the on-chip memory the data size for the access.
- The CPU asserts both CMDREQP to validate the data address on CDAP[31:0] and CSTREQP to specify the memory request direction, in this case, a store (HIGH).
- The on-chip memory must assert OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The CPU asserts CDBREQP to notify the BIU that the CPU intends to drive both CDATAP[31:0] and CDPARP[3:0] during the following MEM stage.
- A CPU stall occurs, so the next cycle is a STALL cycle instead of the MEM cycle. In this case:
  - The on-chip memory *must not* latch the data during the STALL cycle(s), because the available data are from a previous instruction.
  - The CPU does *not* drive the on-chip memory data onto the data bus during the STALL cycle(s); the previous bus owner continues to drive the bus.
  - All signals driven during the initial ALU cycle are *re-driven* for all STALL cycles.
  - STALL cycles *continue* until the CPU deasserts STALLN.
- The CPU deasserts STALLN. Then it drives the on-chip memory data onto the data bus and drives the data parity bus as 2v signals during the MEM stage.
- The on-chip memory latches the CPU data at the end of the MEM stage. Recall that the state of the CW33000 BENDNP signal specifies the endianness of the data, which is determined during initialization and remain unchanged during operation.

Figure 4.5
CPU Store, Stall During ALU Stage

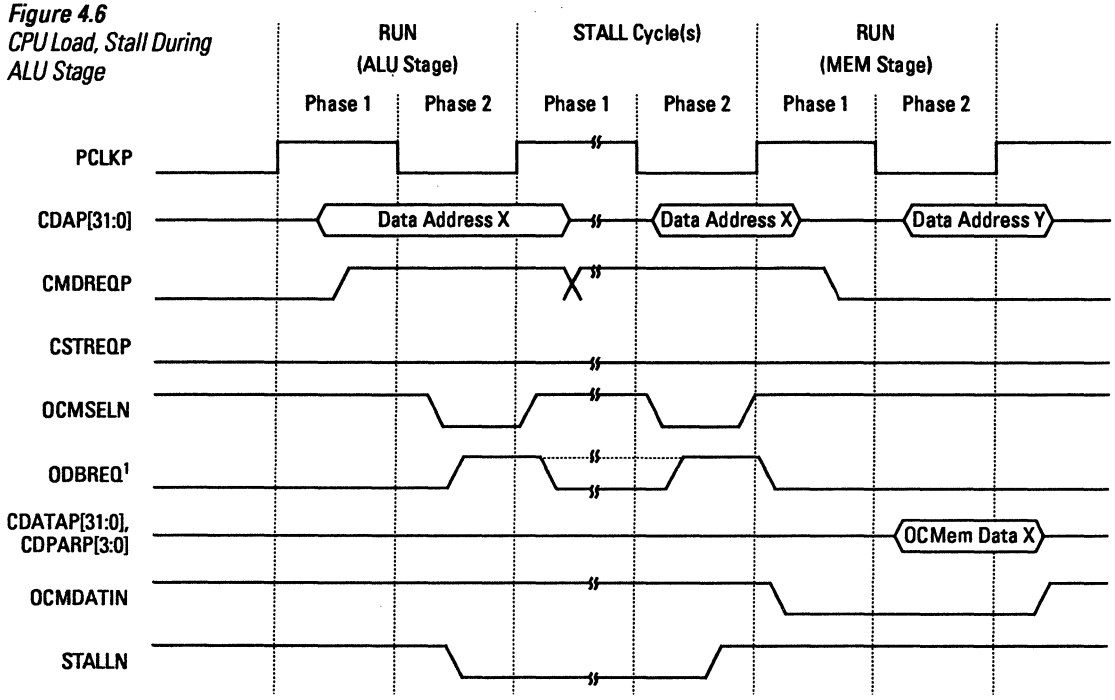


Note:
1. Internal signal shown for reference only.

CPU Load With a Stall During the ALU Stage

- The CPU places the on-chip memory address on CDAP[31:0].
- The CPU asserts CMDREQP to validate the memory data address on the CDA bus.
- The CPU specifies the memory request direction by driving CSTREQP LOW, requesting a load.
- The on-chip memory asserts OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The on-chip memory asserts ODBREQP to notify the BIU that the on-chip memory intends to drive both CDATAP[31:0] and CDPARP[3:0] during the following MEM stage.
- A CPU stall occurs, so the next cycle is a STALL cycle instead of the normal MEM cycle. In this case:
  - The CPU does *not* load the on-chip memory data during the STALL cycle(s).
  - The on-chip memory *must not* drive the data bus during the STALL cycle(s).
  - The CPU and on-chip memory *must re-drive* all signals driven during the initial ALU cycle for all STALL cycles.
  - STALL cycles *continue* until the CPU deasserts STALLN.
- The CPU deasserts STALLN.
- The on-chip memory drives data on both the data bus and the data parity bus as 2v signals during the first RUN cycle (that is, the MEM stage) after the STALL.

Figure 4.6
CPU Load, Stall During ALU Stage

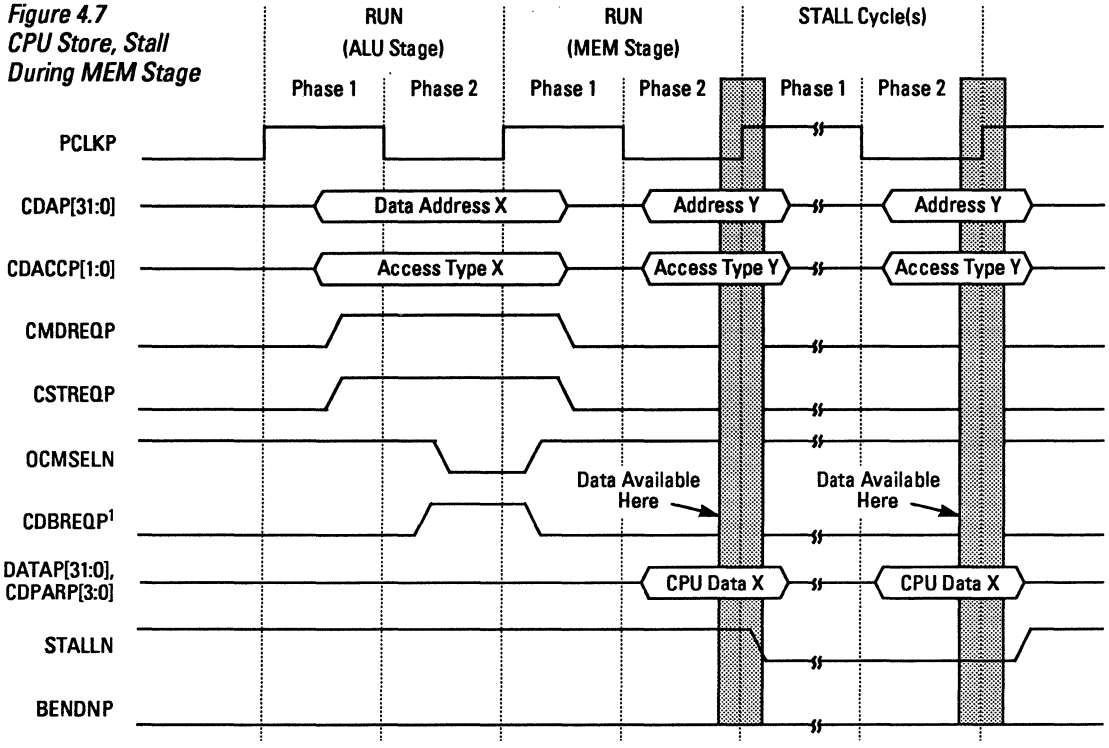


Note:
1. The signal shown with solid lines defines the minimum required functionality. The section indicated by the dotted lines shows optional functionality.

CPU Store With a Stall During the MEM Stage

- The CPU places the address for the on-chip memory on CDAP[31:0] and sets CDACCP[1:0] to specify the data size for the access.
- CMDREQP validates the memory data address on the CDA bus.
- The CPU specifies the memory request direction by driving CSTREQP HIGH, requesting a store.
- The on-chip memory asserts OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The CPU asserts CDBREQP to notify the BIU that the CPU intends to drive both CDATAP[31:0] and CDPARP[3:0] during the following MEM stage.
- The CPU drives data on both the data bus and the data parity bus as 2v signals during the MEM stage.
- A CPU stall occurs, so the next cycle is a STALL cycle instead of the normal WB cycle. In this case:
  - The on-chip memory *may latch* the data at end of phase 2 of the stalled MEM cycle or during any of the subsequent stall cycles.
  - All signals driven during the initial MEM cycle are *re-driven* for all STALL cycles.
  - STALL cycles *continue* until the CPU deasserts STALLN.
- The CPU deasserts STALLN and the CPU instruction pipeline proceeds to the WB stage.

Figure 4.7
CPU Store, Stall During MEM Stage

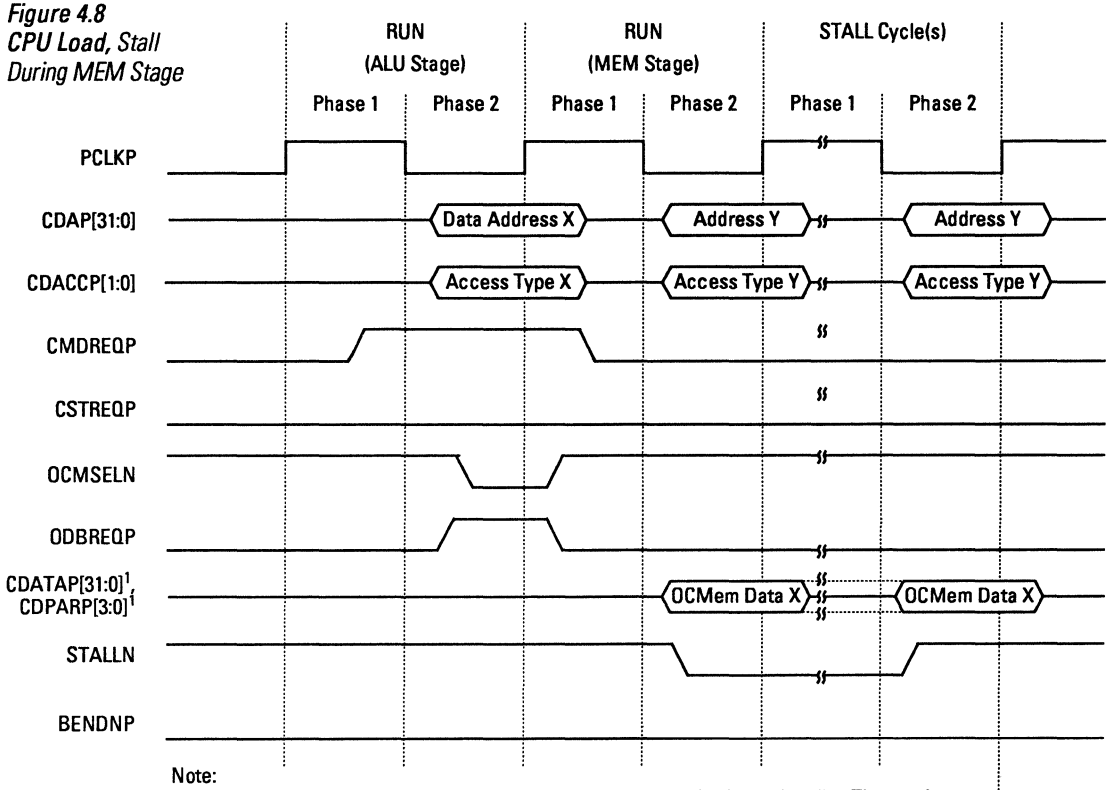


Note:
1. Internal signal shown for reference only.

CPU Load With a Stall During the MEM Stage

- The CPU places the address for the on-chip memory on CDAP[31:0].
- The CPU asserts CMDREQP to validate the memory data address on the CDA bus.
- The CPU specifies the memory request direction by driving CSTREQP LOW, requesting a load.
- The on-chip memory asserts OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The on-chip memory asserts ODBREQP to notify the BIU that the on-chip memory intends to drive both CDATAP[31:0] and CDPARP[3:0] during the following MEM stage.
- The on-chip memory drives data on both the data bus and the data parity bus as 2v signals during the MEM stage.
- A CPU stall occurs, so the next cycle is a STALL cycle instead of the normal WB cycle. In this case:
  - All signals driven during the initial MEM cycle are *re-driven* for all STALL cycles.
  - STALL cycles *continue* until the CPU deasserts STALLN.
- The CPU deasserts STALLN and the CPU pipeline proceeds to the WB stage.

Figure 4.8
CPU Load, Stall During MEM Stage

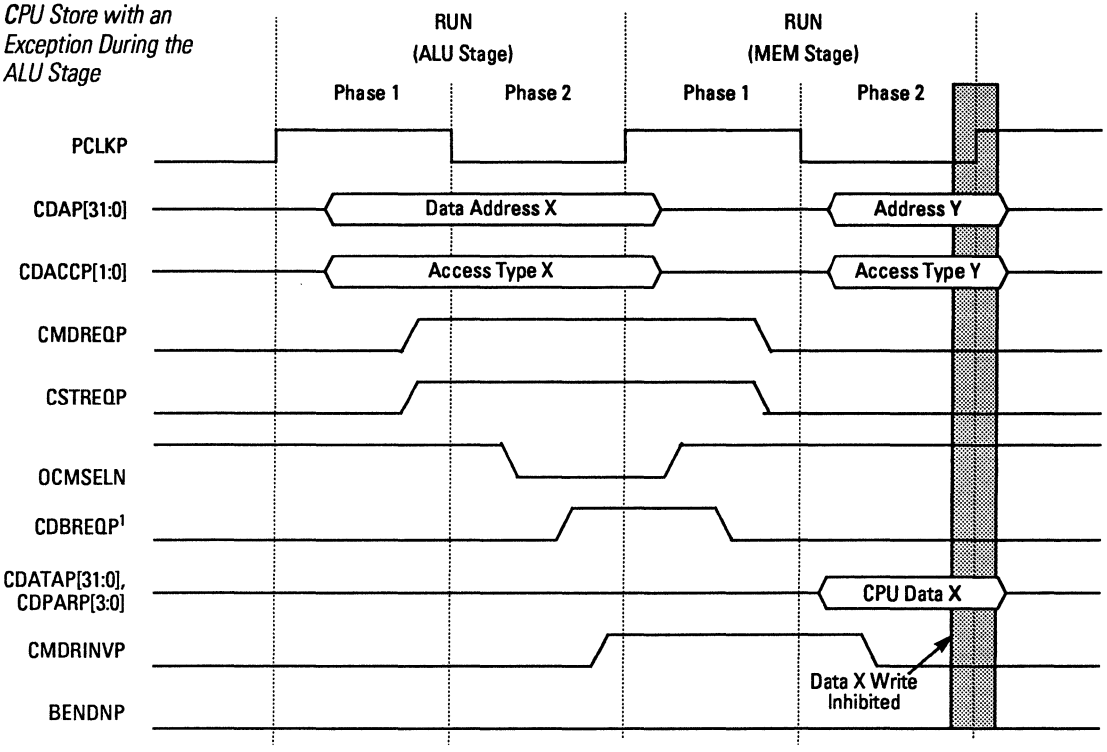


Note:
1. The signals shown with solid lines define the minimum required functionality. The section indicated by the dotted lines shows optional functionality.

CPU Store With a CPU Exception

- The CPU places the address for the on-chip memory on CDAP[31:0].
- The CPU sets CDACCP[1:0] to specify for the on-chip memory the data size for the store.
- The CPU asserts CMDREQP to validate the memory data address on the CDA bus.
- The CPU specifies the memory request direction by driving CSTREQP HIGH, requesting a store.
- The on-chip memory asserts OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The CPU asserts CDBREQP to notify the BIU that the CPU intends to drive both CDATAP[31:0] and CDPAR during the following MEM stage.
- An exception that occurs during the ALU stage causes the CPU to assert CMDRINVP as a 1s signal in the following MEM stage, notifying the on-chip memory to invalidate the memory data request.
  - *Regardless of the exception*, the CPU drives data on both the data bus and the data parity bus as 2v signals during the MEM stage.
  - The on-chip memory must *not* latch the presented CPU data.
- The CPU instruction pipeline continues execution.

Figure 4.9
CPU Store with an Exception During the ALU Stage

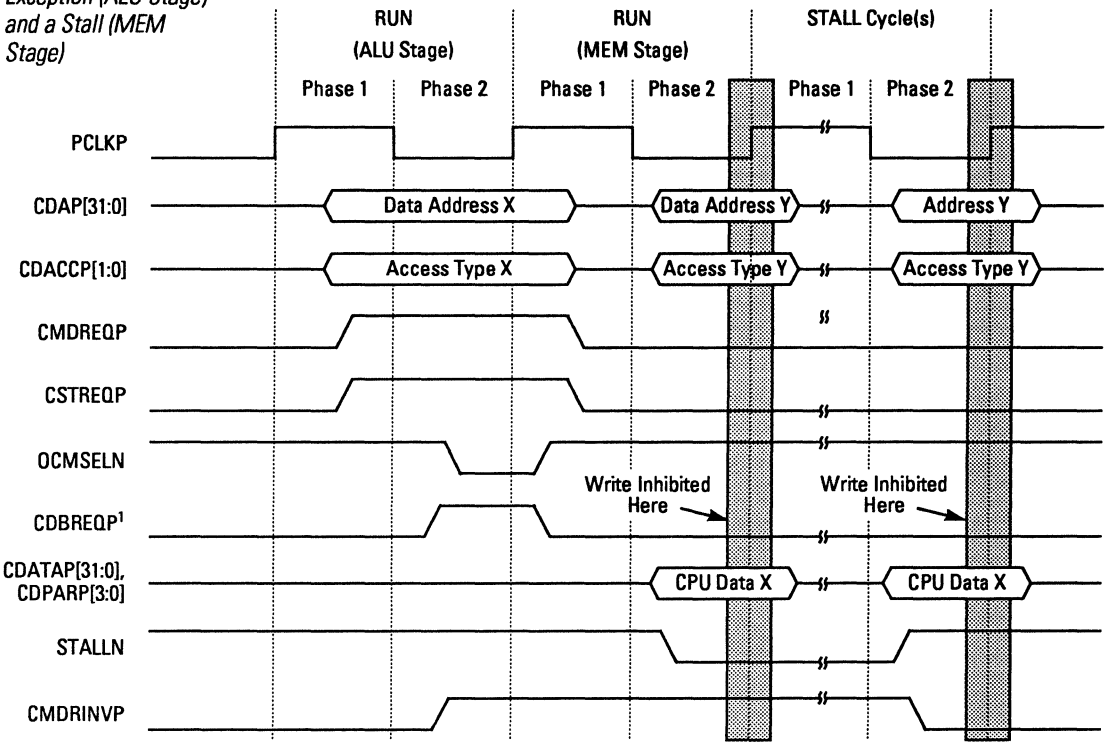


Note:
1. Internal signal shown for reference only.

CPU Store With an ALU-Stage Exception and a MEM-Stage Stall

- The CPU places the address for the on-chip memory on CDAP[31:0].
- The CPU sets CDACCP[1:0] to specify for the on-chip memory the data size for the store.
- CMDREQP validates the memory data address on the CDA bus.
- The CPU specifies the memory request direction by driving CSTREQP HIGH, requesting a store.
- The on-chip memory asserts OCMSELN to notify the BIU that the CPU memory request is for the on-chip memory, so the BIU should ignore the transaction.
- The CPU asserts CDBREQP to notify the BIU that the CPU intends to drive both CDATAP[31:0] and CDPARP[3:0] during the following MEM stage.
- An exception in the ALU stage causes the CPU to assert CMDRINVP as a 1s signal of the following MEM stage, notifying the on-chip memory to invalidate the memory data request.
- The CPU drives data on both the data bus and the data parity bus as 2v signals during the MEM stage. At the same time, a CPU stall occurs, so the next cycle is a STALL cycle instead of the normal WB cycle. In this case:
  - The on-chip memory must *not* latch the presented CPU data due to the assertion of CMDRINVP.
  - All signals driven during the initial MEM cycle, including the CPU data, are *re-driven* for all STALL cycles.
  - CMDRINVP must *continue to inhibit* the latching of CPU data by the on-chip memory for the duration of the STALL.
  - STALL cycles *continue* until STALLN is deasserted.
  - The CPU does *not* deassert CMDRINVP until after it deasserts STALLN.
- The CPU deasserts STALLN and the on-chip memory does not latch the CPU data because CMDRINVP is still asserted.
- The CPU deasserts CMDRINVP and the CPU pipeline proceeds to the WB stage.

Figure 4.10
CPU Store with an Exception (ALU Stage) and a Stall (MEM Stage)



Note:
1. Internal signal shown for reference only.

4.4 Data Formats and Addressing

This section discusses the signals that define the data format and byte order for data operands in the CW33000. For additional information on required data formats and addressing, refer to Section 2.1, "Data and Registers," of this document.

The signals that control data formats and addressing are:

- BENDNP
- CDACCP[1:0]
- CDAP[1:0]

The state of the BENDNP signal defines the endian byte ordering for the CW33000. External logic sets BENDNP during initialization and

BENDNP should not change during operation. If BENDNP equals one, the CW33000 utilizes a big-endian byte ordering where byte 0 (bits [31:24]) is the most significant byte of the word. Alternatively, if BENDNP equals zero, the CW33000 utilizes a little-endian byte ordering where byte 0 (bits [7:0]) is the least significant byte of the word.

The CW33000 CPU may load/store data items that consist of 32-bit words, 24-bit tribytes, 16-bit halfwords, and 8-bit bytes. Data transfer between the CPU and the on-chip memory, however, occurs in 32-bits words, although only part of each word may be valid. CDACCP[1:0], the data access type signals, specify the size of the valid data item for store operations only, according to Table 4.1.

Table 4.1
Data Size Encoding

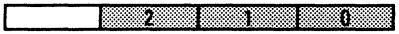
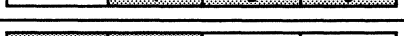
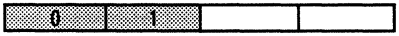
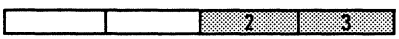
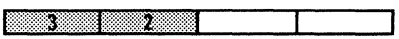
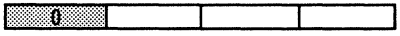
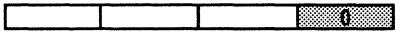
CDACCP[1:0]	*Data Size*
0 0	Byte
0 1	Halfword
1 0	Tribyte
1 1	Word

CDAP[1:0], the two least significant bits of the CPU Data Address bus, specify the location of the valid byte(s) within the 32-bit data word. In particular, these bits specify the valid byte that has the smallest byte address in the addressed field.

CDACCP[1:0] and CDAP[1:0] together encode which bytes within the addressed word are to be written during a CP Store Byte operation, as illustrated in Figure 4.11. These control signals should also generate the appropriate byte write enable signal while the entire word is addressed, that is when CDAP[1:0] = (0,0).

For CPU Load Byte operations, all 32 bits of the on-chip memory's addressed word should be presented to the CPU independent of these control signals, again when CDAP[1:0] = (0,0). The CPU handles byte selection according to the CPU load byte instruction being executed.

Figure 4.11
Byte Specifications for Loads/Stores

Access Type (CDACCP[1:0])	Low-Order Address Bits: CDA1 CDA0	Bytes Accessed	
		Big-Endian (BENDNP = 1) 31 — 0	Little-Endian (BENDNP = 0) 31 — 0
Word (1,1)	0 0		
Tribyte (1,0)	0 0		
	0 1		
Halfword (0,1)	0 0		
	1 0		
Byte (0,0)	0 0		
	0 1		
	1 0		
	1 1		

Chapter 5
I/O Interface Operation

This chapter contains waveforms that depict the CW33000's I/O functions as they appear at its interfaces. The waveforms illustrate both how the CW33000's signals interrelate and how operations initiated on one interface propagate to other interfaces.

The chapter is divided into the following sections that describe different types of memory transactions:

- Interpreting the Waveforms
- Word Read Transactions
- Block-Fetch Transactions
- Write Transactions
- Byte-Gathering Transactions
- Automatic Wait-State Generation
- Bus Errors

5.1 Interpreting the Waveforms

This chapter uses the following conventions for annotating illustrations and discussing waveforms in text:

- Addresses: Addresses are represented by the letter "A" and a number, where the number differentiates between addresses in the waveform. An X represents a *don't care* address value.
- Data and Instructions: The data being transferred in an operation are represented by the letter "D" and a number, where the number identifies the address with which the data are associated. An X represents a *don't care* data value.
- Data Parity: The parity associated with a data transfer is represented by the letter "P" and a number, where the number identifies the parity with the data with which it is associated.

For the sake of clarity, some CW33000 signals are not included in these illustrations. You may assume that these signals are deasserted for the duration of the operation.

5.2 Word Read Transactions

The most basic memory transaction is a word read, which is a transfer of data from an external memory system to the CW33000. A read transaction consists of four distinct states, as shown in Figure 5.1 and described below:

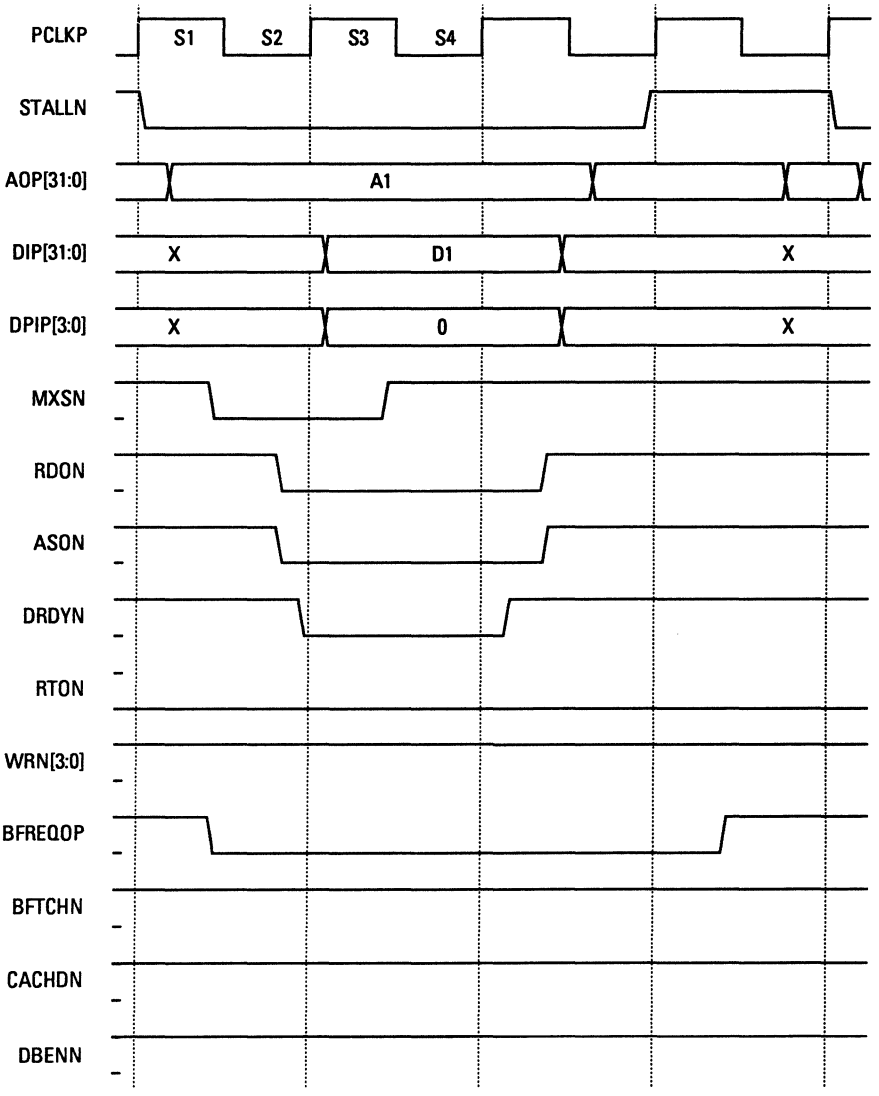
State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. The CW33000 also asserts BFREQOP, if appropriate.

State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid, and it asserts RDON to indicate that the memory system can put data on DIP[31:0].

State 3. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, CACHDN, and BFTCHN. If DRDYN and/or BERRN are asserted, the CW33000 progresses to State 4. If neither is asserted, the CW33000 makes the current cycle a wait cycle. The CW33000 continues to sample DRDYN and BERRN until the memory system asserts one or both.

State 4. At the rising edge of PCLKP, the CW33000 samples DIP[31:0] and DPIP[3:0]. The CW33000 drives the current instruction address onto AOP[31:0] and deasserts RDON, ASON, and BFREQOP.

Figure 5.1
Word Read
Transaction



5.3 Block-Fetch Transactions

A block-fetch transaction allows the CW33000 to fill its cache memories quickly. A block-fetch transaction is similar to a read transaction, but transfers a series of two to 16 words from memory to the CW33000. A 16-word block-fetch transaction is shown in Figure 5.2 and described below:

State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. The CW33000 also asserts BFREQOP to indicate that a block-fetch transaction is desired.

State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid, and it asserts RDON to indicate that the memory system can put data on DIP[31:0].

Wait. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, CACHDN, and BFTCHN. The memory system asserts BFTCHN to indicate that the current transaction is a block fetch. The assertion of BFTCHN causes the CW33000 to set the block address, AOP[5:2] (modulo the block size), to zero. BFTCHN must remain asserted for the duration of the block-fetch transaction.

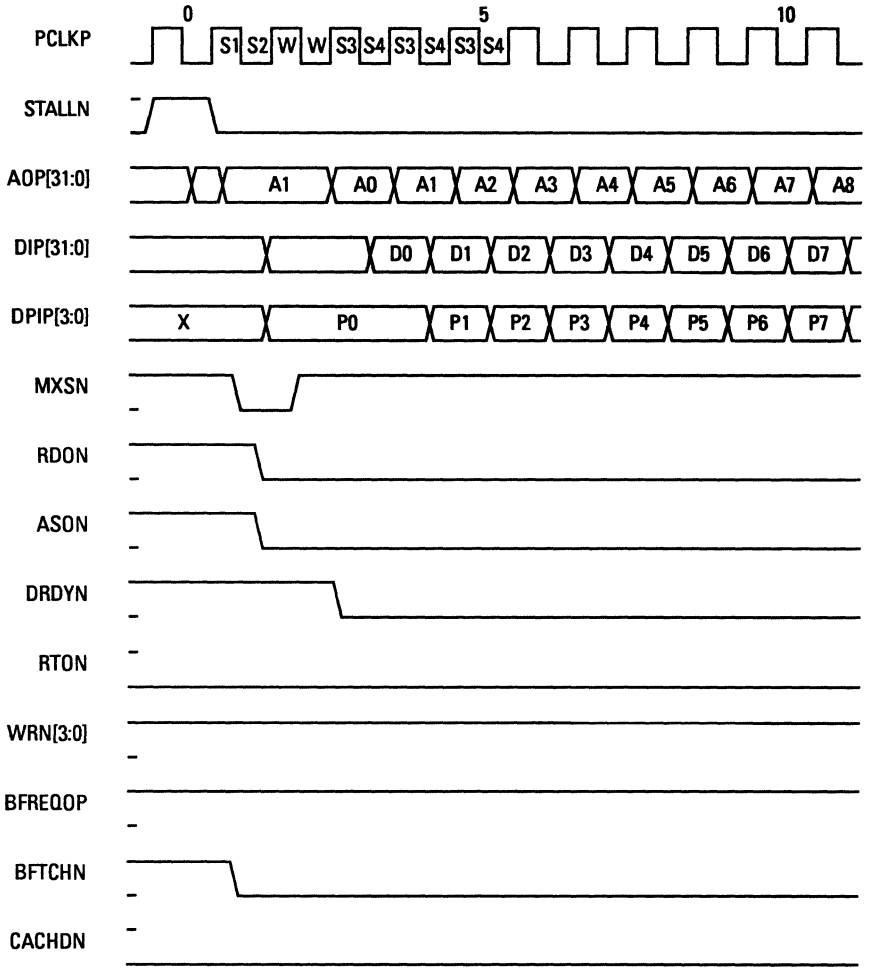
Because this transaction is a block fetch, the memory system must cause at least one wait state in this part of the transaction and so does *not* assert DRDYN. The CW33000 continues to sample DRDYN and BERRN until the memory system asserts one or both. When the memory system asserts DRDYN in the second half of the wait state, the CW33000 advances to State 3.

State 3. During the first transfer, the CW33000 samples CACHDN and PENN. CACHDN must be asserted; asserting PENN is optional. The memory system can insert a wait state for any cycle by deasserting DRDYN before the CW33000 samples it at the falling edge of PCLKP in State 3. If DRDYN is not asserted, the CW33000 does not increment the address in State 4.

State 4. At the rising edge of PCLKP, the CW33000 samples DIP[31:0] and DPIP[3:0] and increments the block address.

States 3 and 4 repeat until the block count reaches zero or a bus error occurs. When the block count reaches zero, the CW33000 ends the block transfer by deasserting ASON and RDON. The memory system deasserts DRDYN.

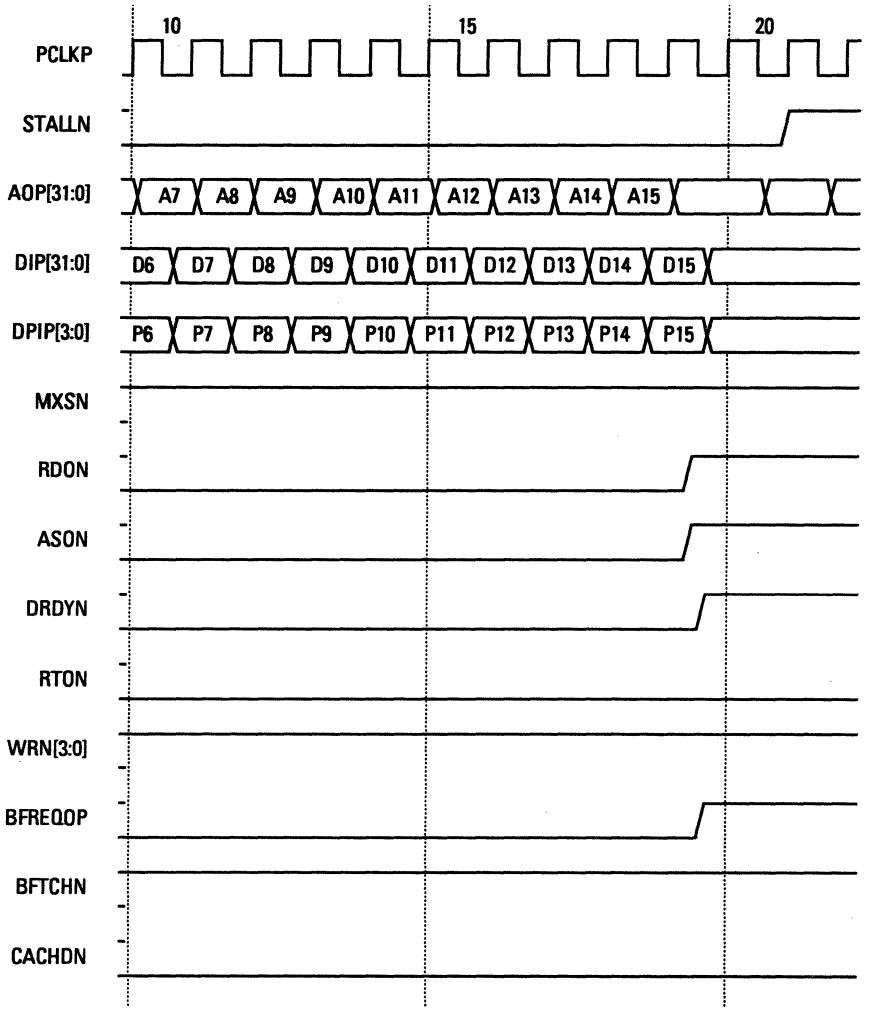
Figure 5.2
Block-Fetch
Transaction
(Sheet 1 of 2)



Note

The CW33000 automatically caches instructions or data (if the system includes a D-cache) fetched during a block fetch transaction regardless of the state of the CACHDN signal. Refer to Section 2.5, "Cache Operation," for more information about cache operation during block fetch transactions.

Figure 5.2 Block-Fetch Transaction (Sheet 2 of 2)



5.4 Write Transactions

A write transaction is a transfer of data from the CW33000 to an external memory system. Like a word read, a write consists of four distinct states, shown in Figure 5.3 and described below:

State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. The CW33000 also deasserts RTON to indicate that the current transaction is a write, and it deasserts RDON to indicate that the memory system *must not* drive data on DIP[31:0].

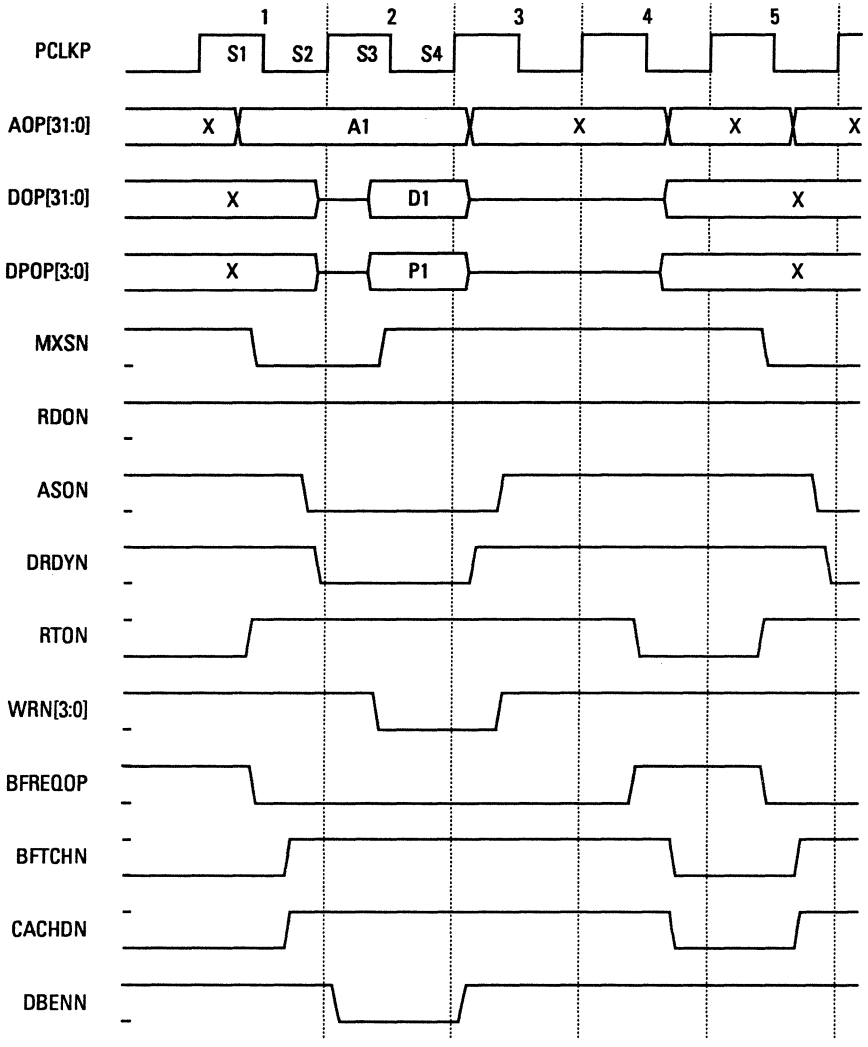
State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid.

State 3. The CW33000 drives the write data onto DOP[31:0] and DPOP[3:0]. It also asserts WRN[3:0] for each valid byte of data on DOP[31:0]. The CW33000 deasserts MXSN.

At the falling edge of PCLKP, the CW33000 samples DRDYN and BERRN. If DRDYN and/or BERRN is asserted, the CW33000 progresses to State 4. If neither is asserted, the CW33000 makes the current cycle a wait cycle. The CW33000 continues to sample DRDYN and BERRN until the memory system asserts one or both.

State 4. The memory system samples DOP[31:0] and DPOP[3:0]. At the end of this state, the CW33000 deasserts ASON, WRN[3:0], DOP[31:0], and DPOP[3:0].

Figure 5.3
Write Transaction



5.5 Byte-Gathering Transactions

To support byte-wide devices better, the CW33000 performs byte gathering, which appears on the memory interface as a series of four single-byte memory transactions. A read transaction consists of four distinct states as shown in Figure 5.4 and described below:

State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. The CW33000 also asserts BFREQOP, if appropriate.

State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid, and it asserts RDON to indicate that the memory system can put data on DIP[31:0].

Wait. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, CACHDN, BFTCHN, and BWIDEN. The assertion of BWIDEN causes the CW33000 to perform byte gathering, starting with the addressed byte.

Because the PROM in this example is too slow to provide single-cycle access, the memory system does *not* assert DRDYN and so causes a wait state. The CW33000 continues to sample DRDYN and BERRN until the memory system asserts one or both. When the memory system asserts DRDYN in the second half of the wait state, the CW33000 advances to State 3.

State 3. The CW33000 samples DRDYN, PENN, BERRN, CACHDN, and BFTCHN. When DRDYN and/or BERRN are asserted, the CW33000 progresses to State 4.

State 4. At the rising edge of PCLKP, the CW33000 samples DIP[31:0] and DPIP[3:0].

The four states described above are repeated four times, once for each byte, or until a bus error occurs. Note that the byte address increments by one for each transaction.

Note

External logic indicates a bus error during a byte-gathering transaction the same way it indicates a bus error during a word read transaction. If a bus error occurs during a byte-gathering transaction, external logic must assert BERRN during the first byte-gathering transaction only. It must not assert BERRN during the second, third, or fourth byte-gathering transactions.

Figure 5.4
Byte-Gathering
Transactions
(Sheet 1 of 2)

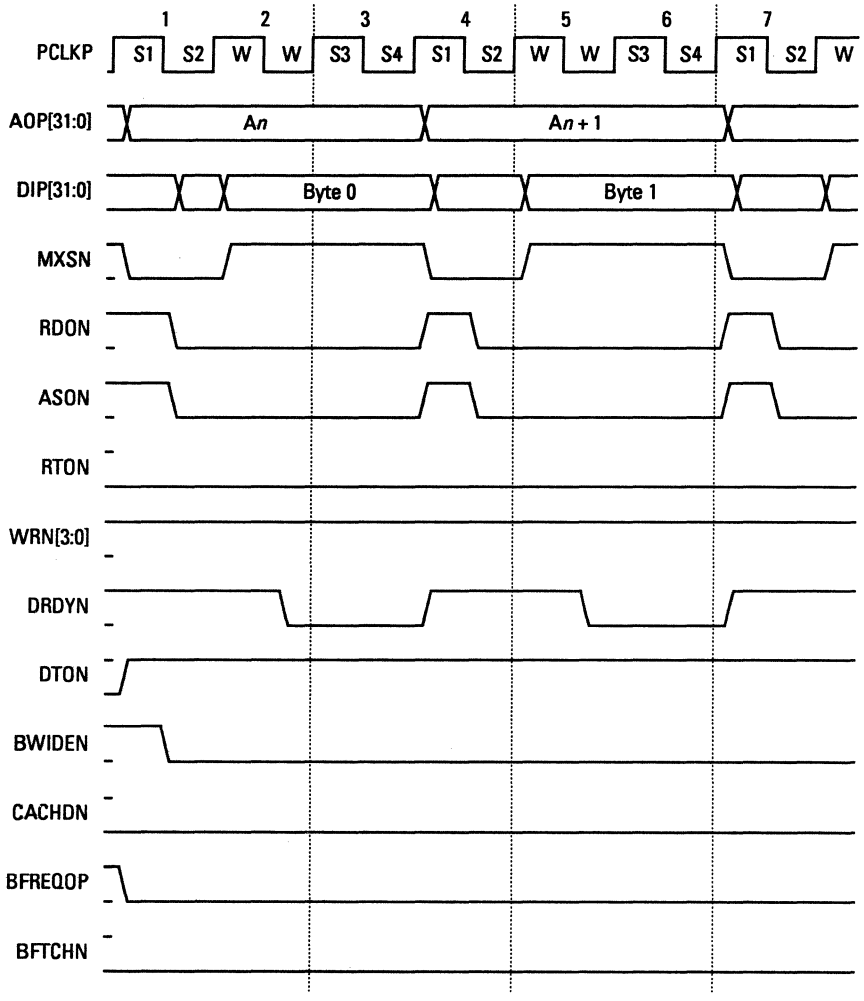
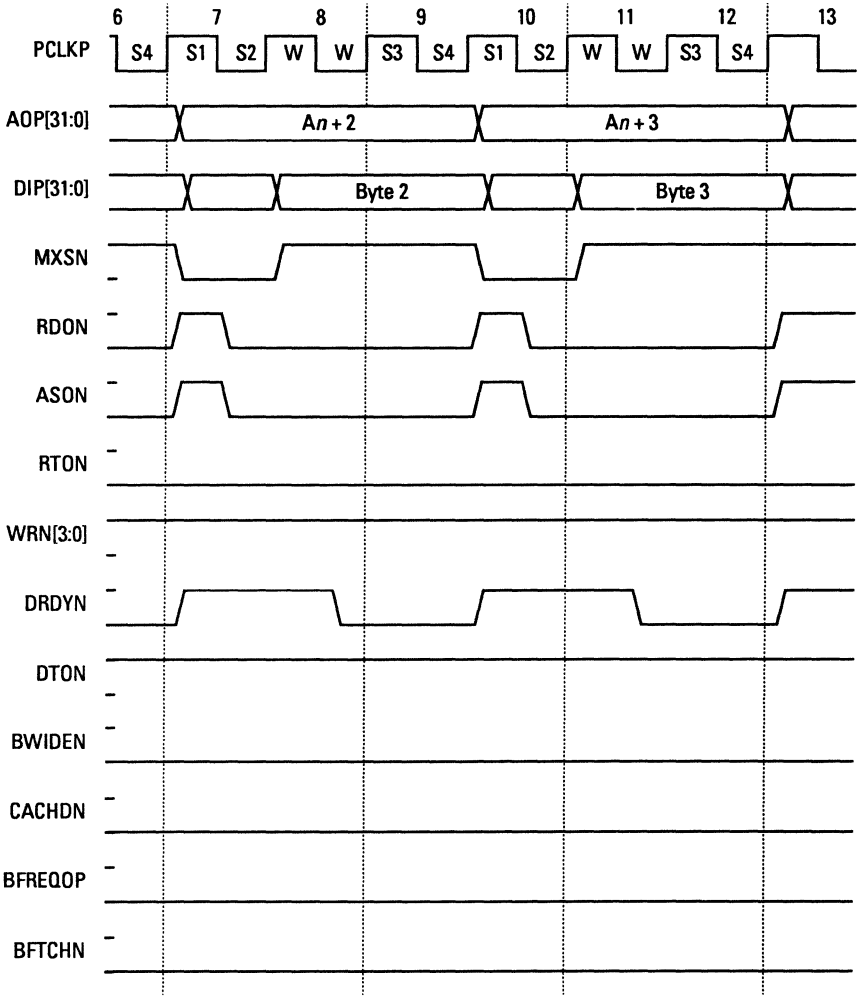


Figure 5.4
Byte Gathering
Transactions
(Sheet 2 of 2)



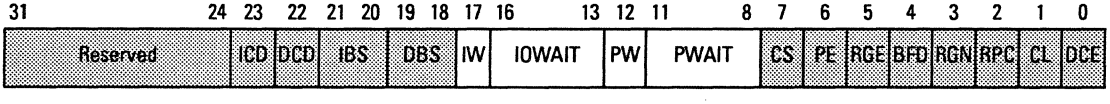
5.6 Automatic Wait-State Generation

To accommodate devices of varying access times, the BIU incorporates a wait-state generator for both the IOSELN and EPSELN address spaces. When either wait-state generator is enabled, the BIU latches the contents of the data bus a programmable number of clock cycles after the start of the memory transaction (as signaled by the assertion of MXSN). When the wait-state generators are enabled, the addressed device need not assert DRDYN to end the transaction. However, the addressed device may end the transaction before the wait-state count expires by asserting DRDYN. This feature allows devices of different speeds to be connected in the same address space; faster devices may end the transaction before the wait-state count expires. Note that the wait-state generators do not support block fetching.

Configuration

The wait-state generators are enabled and programmed via four fields in the Configuration Register, highlighted in the diagram below. Software can read and write the Configuration Register at address 0xFFFE.0020. These four fields have the following definitions:

Location: Memory — Cold Reset Initial Value: 0x0001.EF28
Address: 0xFFFE.0020 — Warm Reset Initial Value: Unchanged



IW **IOSELN Wait Disable** **17**

When set to one, IW disables the IOSELN Wait feature. When the IOSELN Wait feature is disabled, the CW33000 asserts IOSELN during IOSELN space accesses and waits until the accessed device asserts DRDYN before ending the memory transaction. When IW is set to zero, the CW33000 ends the memory cycle either after waiting the number of clock cycles defined by the IOWAIT field or when the external device asserts DRDYN.

IOWAIT **IOSELN Wait** **[16:13]**

The four-bit IOWAIT field defines the number of clock cycles that the CW33000 waits before ending a memory transaction in the IOSELN address space. The valid range for IOWAIT is from 0 to 15 cycles.

PW **EPSELN Wait Disable** **12**

When set to one, PW disables the EPSELN Wait feature. When the EPSELN Wait feature is disabled, the CW33000 asserts EPSELN

during EPSELN space accesses and waits until the PROM asserts DRDYN before ending the memory transaction. When PW is set to zero, the CW33000 ends the memory cycle either after waiting the number of clock cycles defined by the PWAIT field or when the external device asserts DRDYN.

PWAIT **EPSELN Wait** **[11:8]**
The four-bit PWAIT field defines the number of clock cycles that the CW33000 will wait before ending a memory transaction in the EPSELN address space. The valid range for PWAIT is from 0 to 15 cycles.

Operation

A read transaction with five wait states is shown in Figure 5.5 and described below:

State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. Block-fetch transactions are not supported in the IOSELN and EPSELN address spaces when the wait-state generator is enabled.

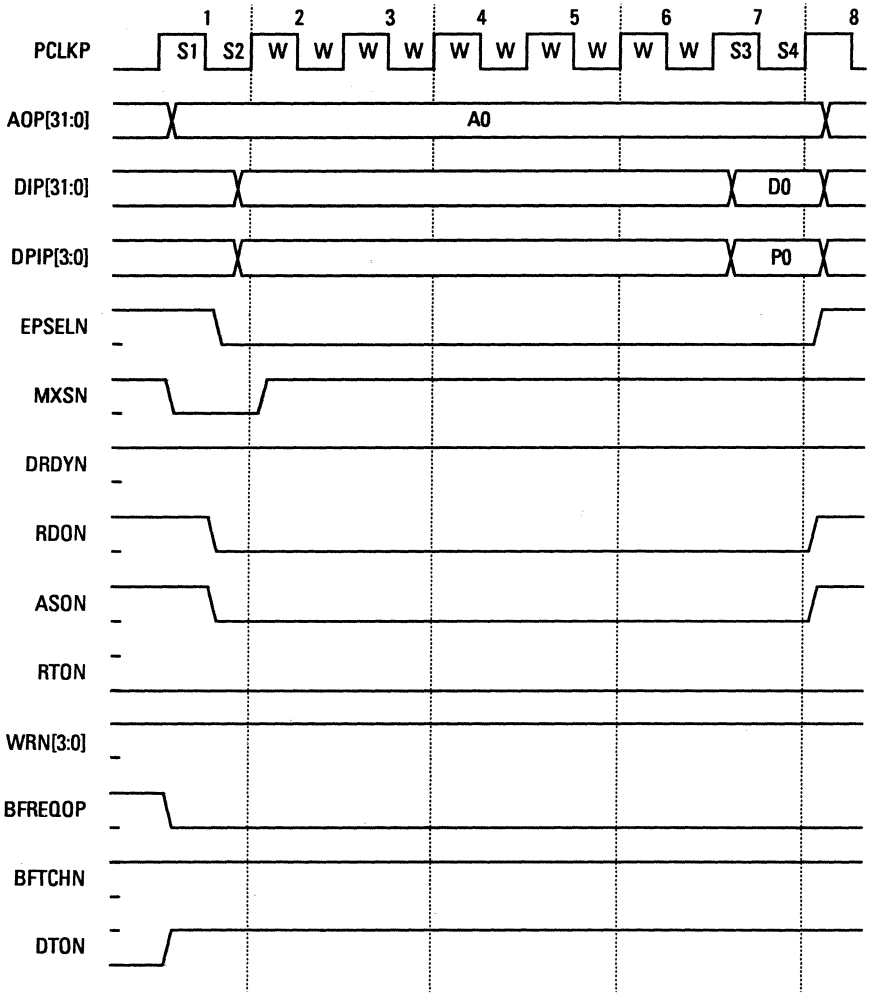
State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid, and it asserts RDON to indicate that the memory system can put data on DIP[31:0].

Wait. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, and CACHDN. Because the wait-state generator for the EPSELN address space is enabled, the CW33000 begins automatically generating wait states. When the wait-state count expires, the CW33000 advances to State 3. The memory system can assert DRDYN before the wait-state count expires and end the transaction early if desired.

State 3. The CW33000 samples PENN, BERRN, and CACHDN. The CW33000 progresses to State 4.

State 4. At the rising edge of PCLKP, the CW33000 samples DIP[31:0] and DPIP[3:0].

Figure 5.5
Automatic Wait-State Generation



5.7 Bus Errors

If the target of a memory access detects an error during the transaction, it signals the error by asserting BERRN. Asserting BERRN terminates the transaction and causes the CW33000 to execute a Bus Error Exception.

Read Transactions

In response to bus errors during read transactions, the CW33000 terminates transactions gracefully. If a bus error occurs during a byte-gathering operation, the CW33000 terminates the current memory transaction and any transactions that would be required to complete the byte-gathering operation. If a bus error occurs during a block-fetch transaction, the CW33000 terminates the current memory transaction and any transactions that would be required to complete the block fetch.

Figure 5.6 illustrates a bus error during a block-fetch transaction. The transaction is described below:

State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. The CW33000 also asserts BFREQOP to indicate that a block-fetch transaction is desired.

State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid, and it asserts RDON to indicate that the memory system can put data on DIP[31:0].

Wait. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, CACHDN, and BFTCHN. The memory system asserts BFTCHN to indicate that the current transaction is a block fetch. The assertion of BFTCHN causes the CW33000 to set the block address, AOP[5:3] modulo the block size, to zero.

Because this transaction is a block fetch, the memory system must cause a wait state in this cycle and so does *not* assert DRDYN. The CW33000 continues to sample DRDYN and BERRN until the memory system asserts one or both. When the memory system asserts DRDYN in the second half of the wait state, the CW33000 advances to State 3.

State 3. During the first transfer, the CW33000 samples CACHDN and PENN. CACHDN must be asserted, because the memory access is a block-fetch transaction; asserting PENN is optional. The memory system can insert a wait state for any cycle by

deasserting DRDYN before the CW33000 samples it at the falling edge of PCLKP in State 3. If DRDYN is not asserted, the CW33000 does not increment the address in State 4.

State 4. At the rising edge of PCLKP, the CW33000 samples DIP[31:0] and DPIP[3:0] and increments the block address.

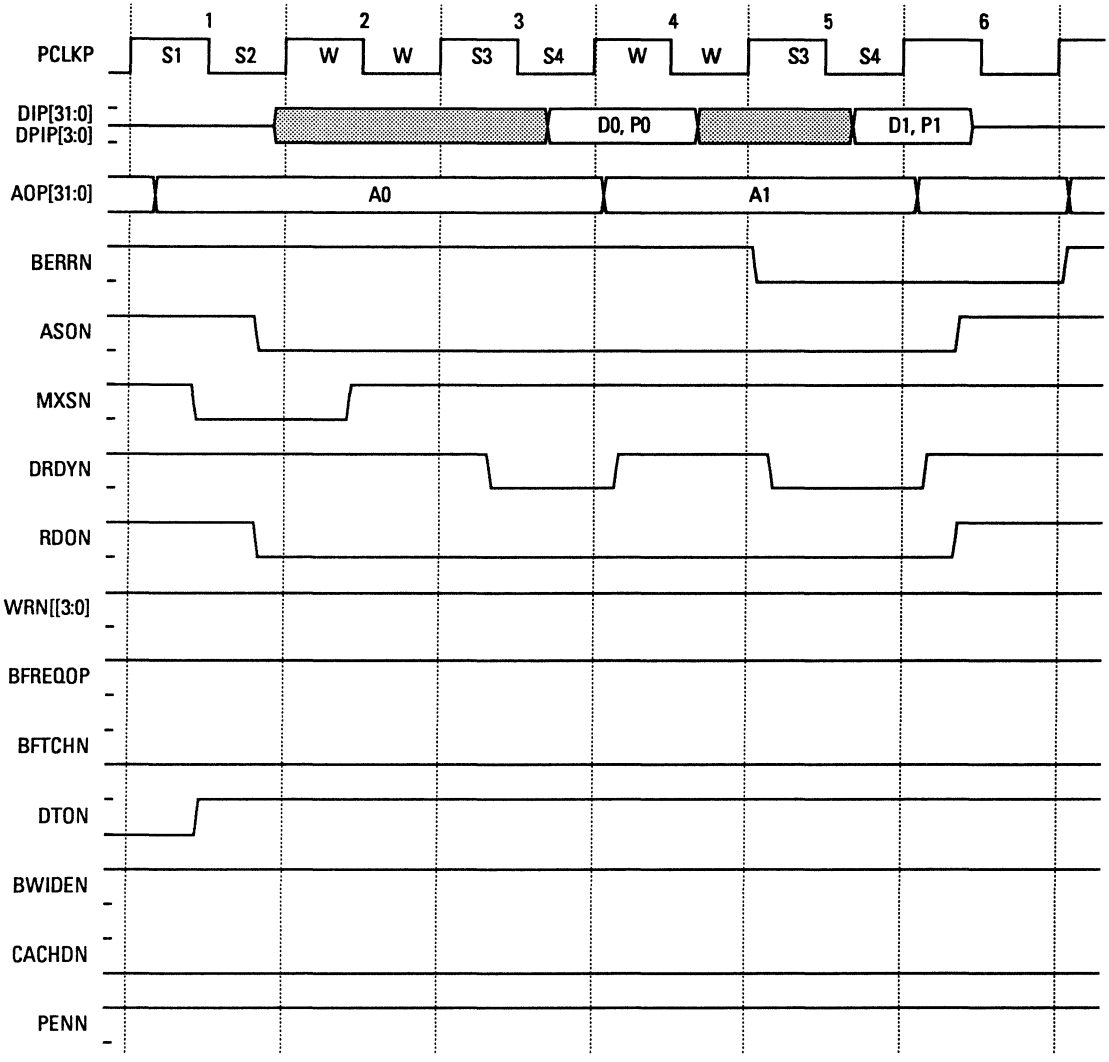
Wait. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, CACHDN, and BFTCHN. Because the DRAM in this example is too slow to provide single-cycle access, the memory system does *not* assert DRDYN and so causes a wait state. When the memory system asserts DRDYN in the second half of the wait state, the CW33000 advances to State 3.

State 3. The memory system asserts BERRN to indicate that it cannot complete the block-fetch transaction. At the falling edge of PCLKP, the CW33000 samples DRDYN, PENN, BERRN, CACHDN, and BFTCHN and detects the error.

State 4. The CW33000 deasserts ASON and RDON.

Note External logic must assert BERRN until after the CW33000 deasserts ASON.

Figure 5.6
Bus Error During a Block-Fetch Transaction



Write Transactions

Figure 5.3 illustrates a bus error during a write transaction. The transaction is described below:

State 1. At the rising edge of PCLKP, the CW33000 drives the address onto AOP[31:0] and asserts MXSN to indicate the start of a memory transaction. The CW33000 also asserts deasserts RTON to indicate that the current transaction is a write, and it deasserts RDON to indicate that the memory system *must not* drive data on DIP[31:0].

State 2. The CW33000 asserts ASON to indicate that the address on AOP[31:0] is valid.

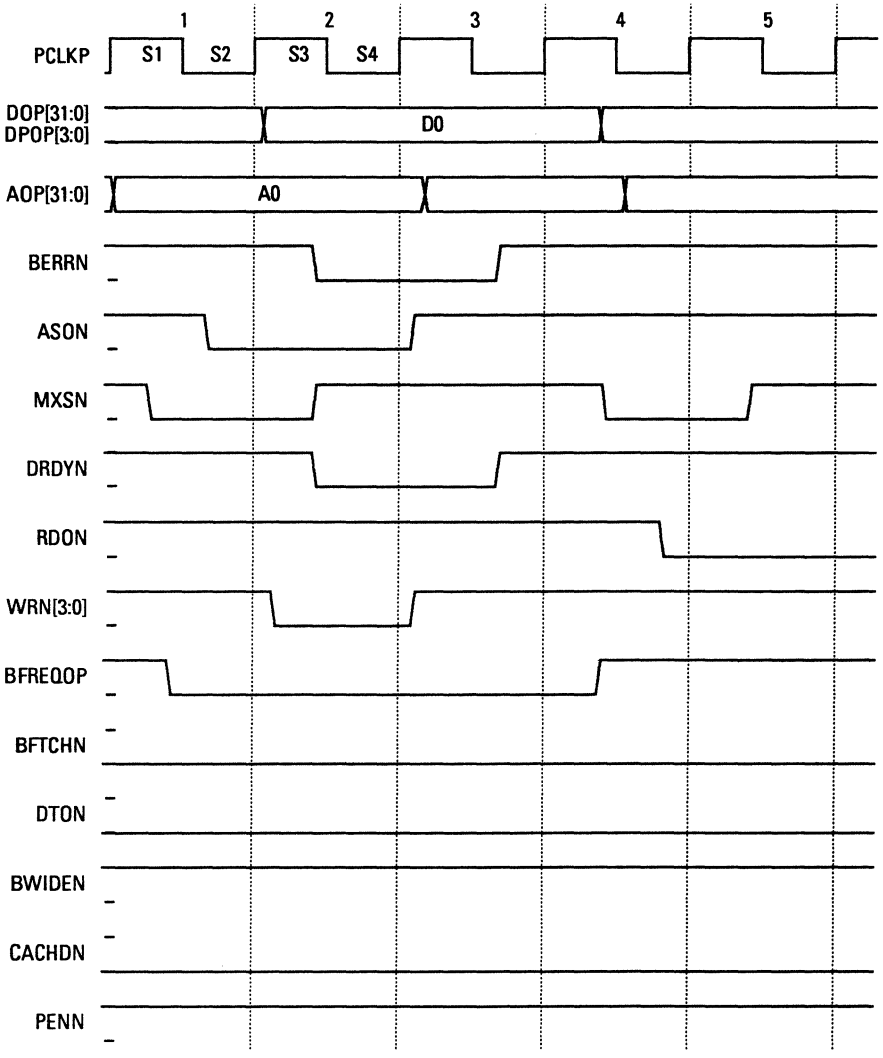
State 3. At the rising edge of PCLKP, the CW33000 drives the write data onto DOP[31:0] and DPOP[3:0]. It also asserts WRN[3:0] for each valid byte of data on DOP[31:0]. The CW33000 deasserts MXSN.

At the falling edge of PCLKP, the CW33000 samples DRDYN and BERRN. Because both DRDYN and BERRN are asserted, the CW33000 terminates the transaction and executes a Bus Error Exception.

State 4. At the rising edge of PCLKP, the CW33000 deasserts AOP[31:0], WRN[3:0], and ASON. The transaction ends.

If a bus error occurs during a write transaction, the memory system should not complete a transaction by latching data. Memory state should not be affected by write transactions that terminate with bus errors.

Figure 5.7
Bus Error During a Write Transaction



Chapter 6
AC Specifications

This chapter provides the AC specifications for the LSI Logic CW33000 MIPS Embedded Processor building block, including the AC timing as well as input and output loading and drive type information.

6.1 AC Timing

This section describes the AC timing characteristics of the CW33000's memory interface, including the Direct CPU Data Bus Interface to a user-designed on-chip memory. The timing relationships between PCLKP and various CW33000 signals that comprise its memory interface are depicted in Figures 6.1 through 6.22. The figures depict:

- Clock Timing (Figure 6.1)
- Cold Reset Timing (Figure 6.2)
- Warm Reset Timing (Figure 6.3)
- Three-State Enable Control Timing (Figure 6.4)
- Read Transaction Timing (Figure 6.5)
- Write Transaction Timing (Figure 6.6)
- Block-Fetch Transaction Timing (Figure 6.7)
- Synchronous Bus Arbitration Timing (Figure 6.8)
- Bus Snooping Timing (Figure 6.9)
- DRAM Read Timing (Figure 6.10)
- DRAM Write Timing (Figure 6.11)
- DRAM Block-Fetch Timing (Figure 6.12)
- DRAM Refresh Timing (Figure 6.13)
- DMA Access Timing Using BGNTP (Figure 6.14)
- DMA Access Timing Using DMARN (Figure 6.15)
- Timing for Miscellaneous Input and Output Signals (Figure 6.16)

- DCDBI Timing for a CPU Store (Figure 6.17)
- DCDBI Timing for a CPU Load (Figure 6.18)
- Data Cache Data Read Timing (Figure 6.19)
- Data Cache Tag and Valid Bit Read Timing (Figure 6.20)
- Data Cache Data Write Timing (Figure 6.21)
- Data Cache Tag and Valid Bit Write Timing (Figure 6.22)

The numbers in Figures 6.1 through 6.22 refer to the timing parameters listed in column 1 of Table 6.1. All of the timing parameters are valid for worst case commercial (WCCOM) die conditions as specified for the LSI Logic LCB007 process. Note that the AC timing is valid only when the customer-designed peripheral logic does not exceed the customer loading numbers provided and discussed in Section 6.2.

Figure 6.1
Clock Timing

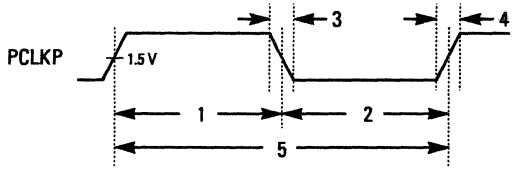


Figure 6.2
Cold Reset Timing

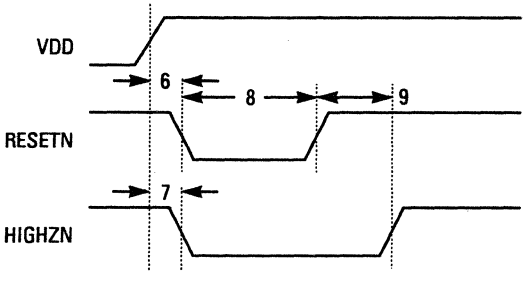
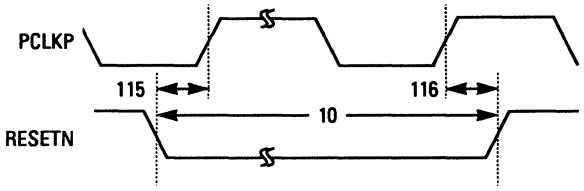
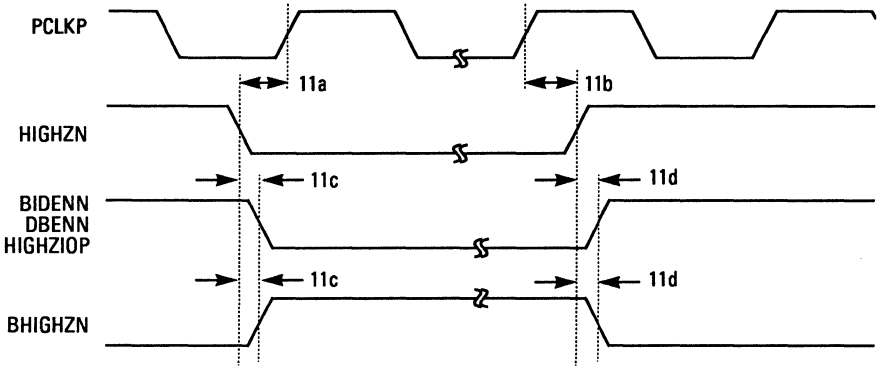


Figure 6.3
Warm Reset Timing



Note:
1. Parameters 115 and 116 are provided for designers who need to synchronize the CW33000's RESETN or HIGHZN state with other devices. These parameters can be ignored for asynchronous applications.

Figure 6.4
Three-State Enable
Control Timing



Note:
1. Parameters 11a and 11b are provided for designers who need to synchronize the CW33000's RESETN or HIGHZN state with other devices. These parameters can be ignored for asynchronous applications.

Figure 6.5
Read Transaction Timing

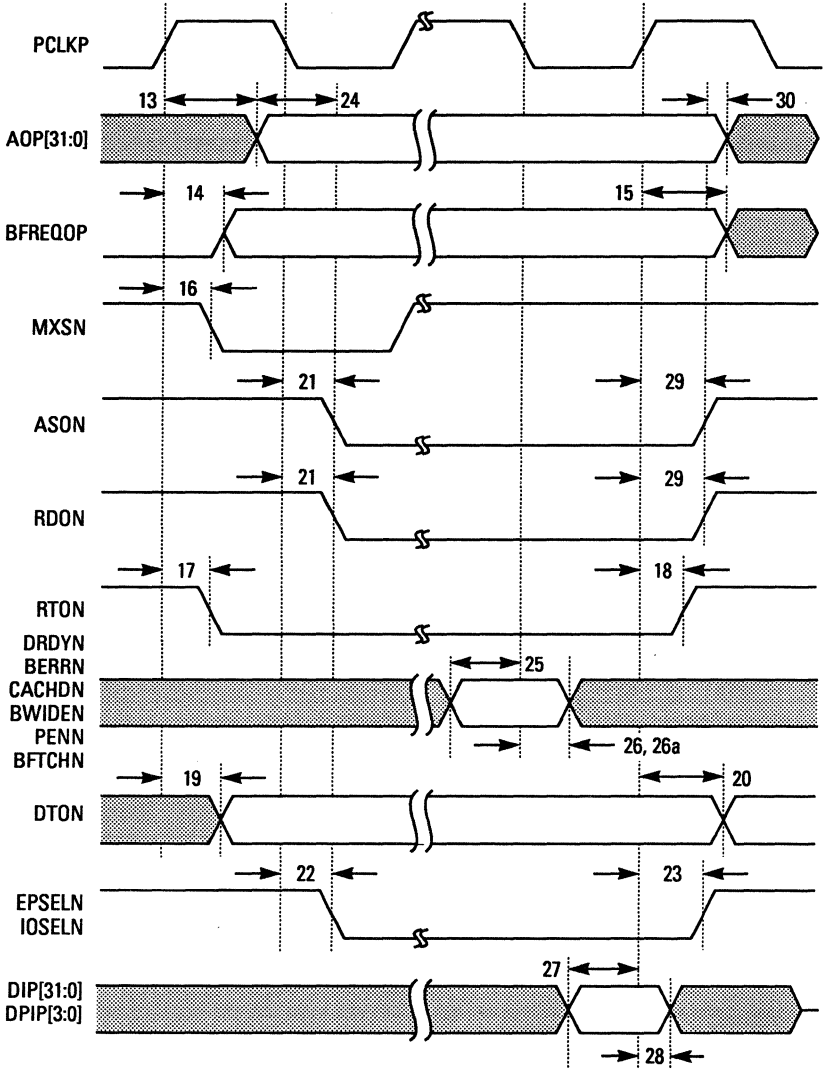


Figure 6.6
Write Transaction Timing

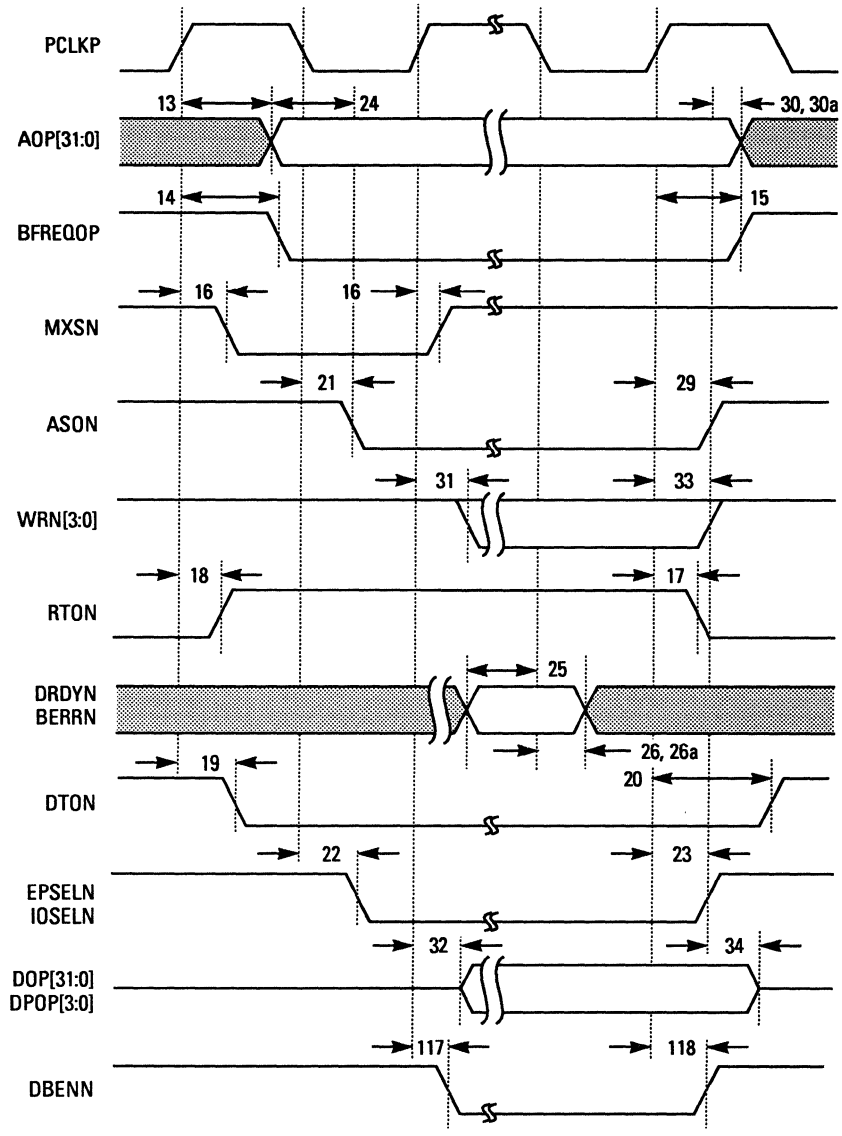
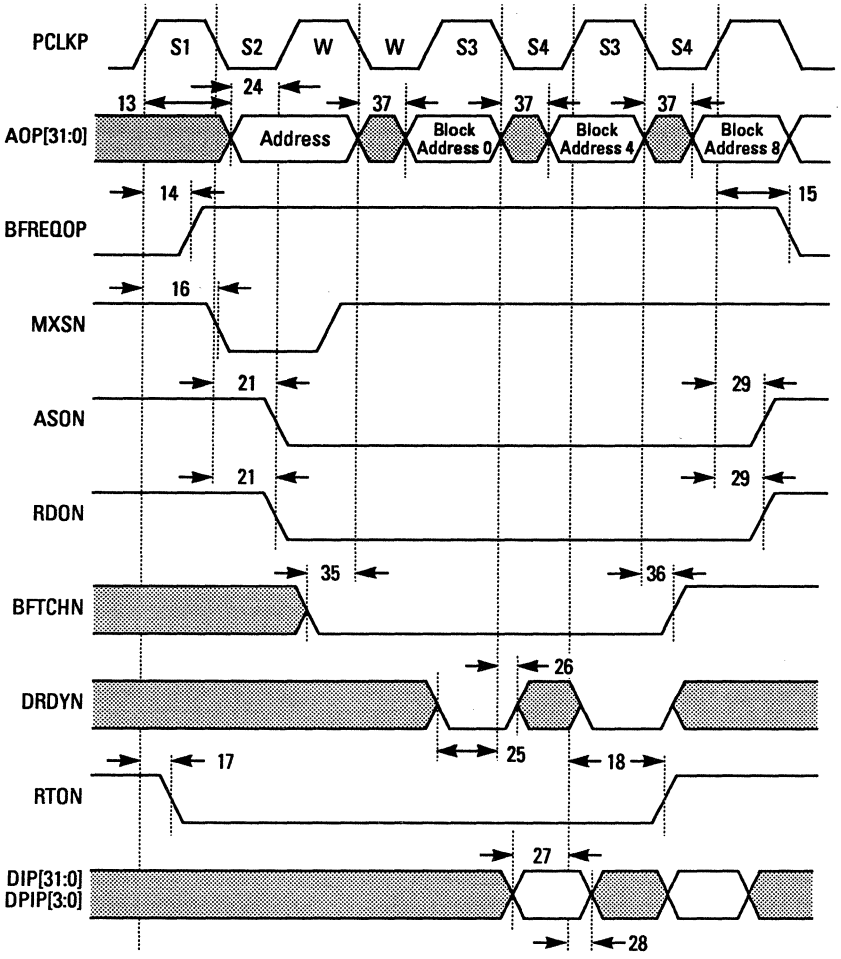


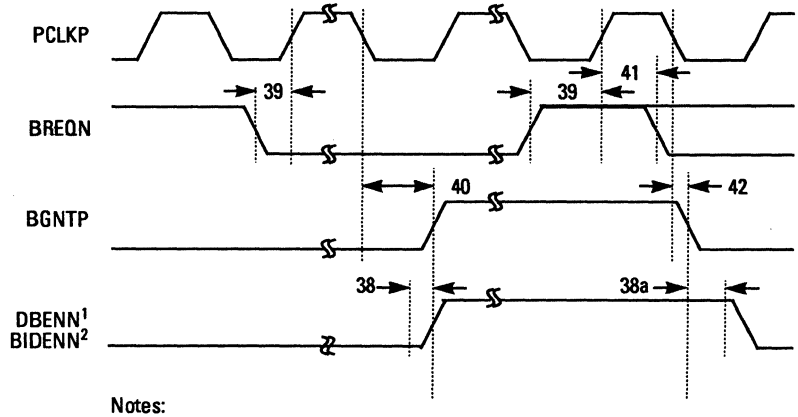
Figure 6.7
Block-Fetch
Transaction Timing



Note:

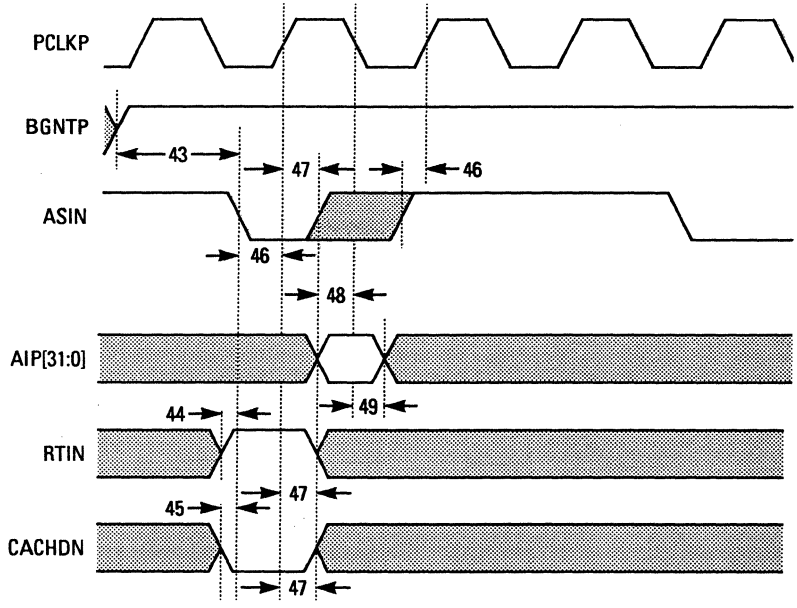
1. This waveform depicts a two-word block fetch. As shown in the waveform, the address increments three times for this two-word fetch. The memory system should ignore the final address in a block fetch of any length. For larger block fetches, States 3 and 4 repeat until the fetch is complete. The memory system may insert wait states in between States 4 and 3.

Figure 6.8
Synchronous Bus Arbitration Timing



Notes:
1. DBENN is the enable control signal for DOP[31:0] and DPDP[3:0].
2. BIDENN is the enable control signal for AOP[31:0], ASON, RDON, WRN[3:0], MXSN, RTON, BFREQOP, and DTON.

Figure 6.9
Bus Snooping Timing



Note:
1. There may be any number of cycles between assertion of BGNTP and assertion of ASIN for the purposes of bus snooping.

Figure 6.10
DRAM Read Timing

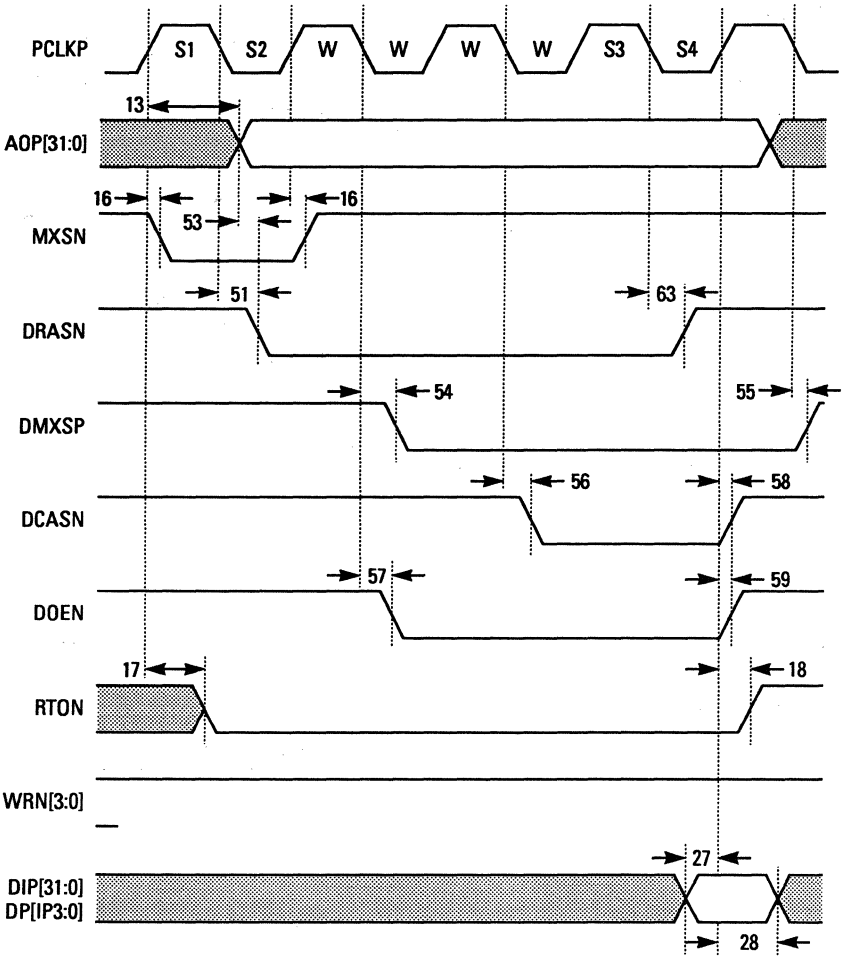


Figure 6.11
DRAM Write Timing

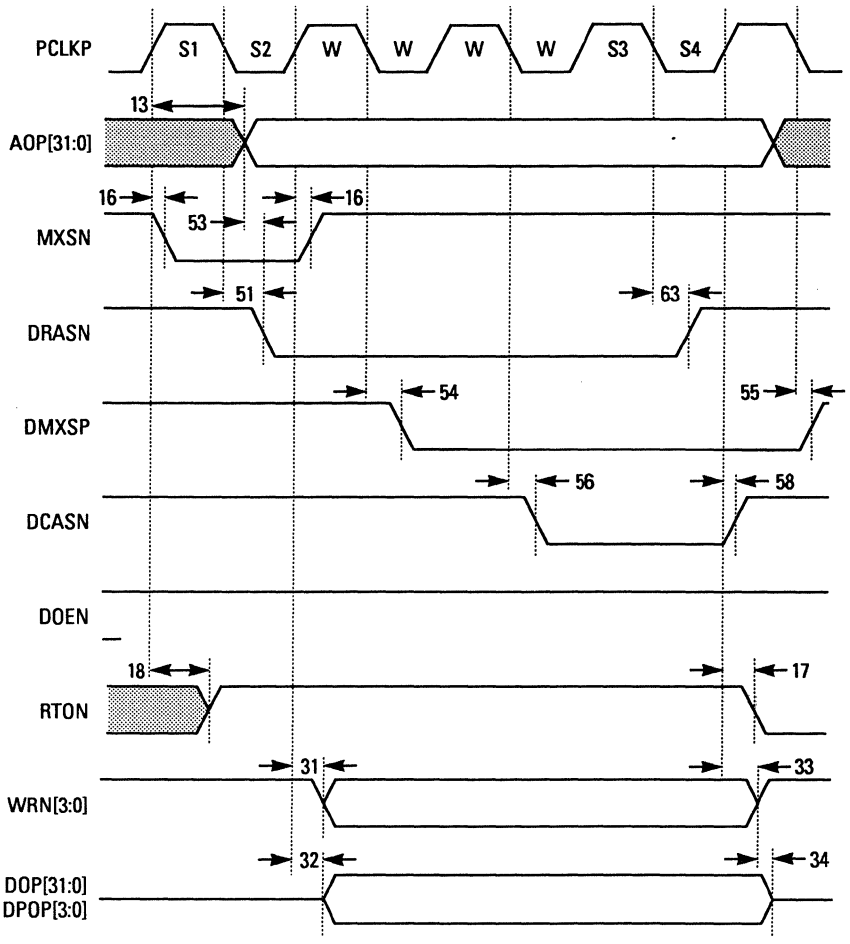


Figure 6.12
DRAM Block-Fetch Timing

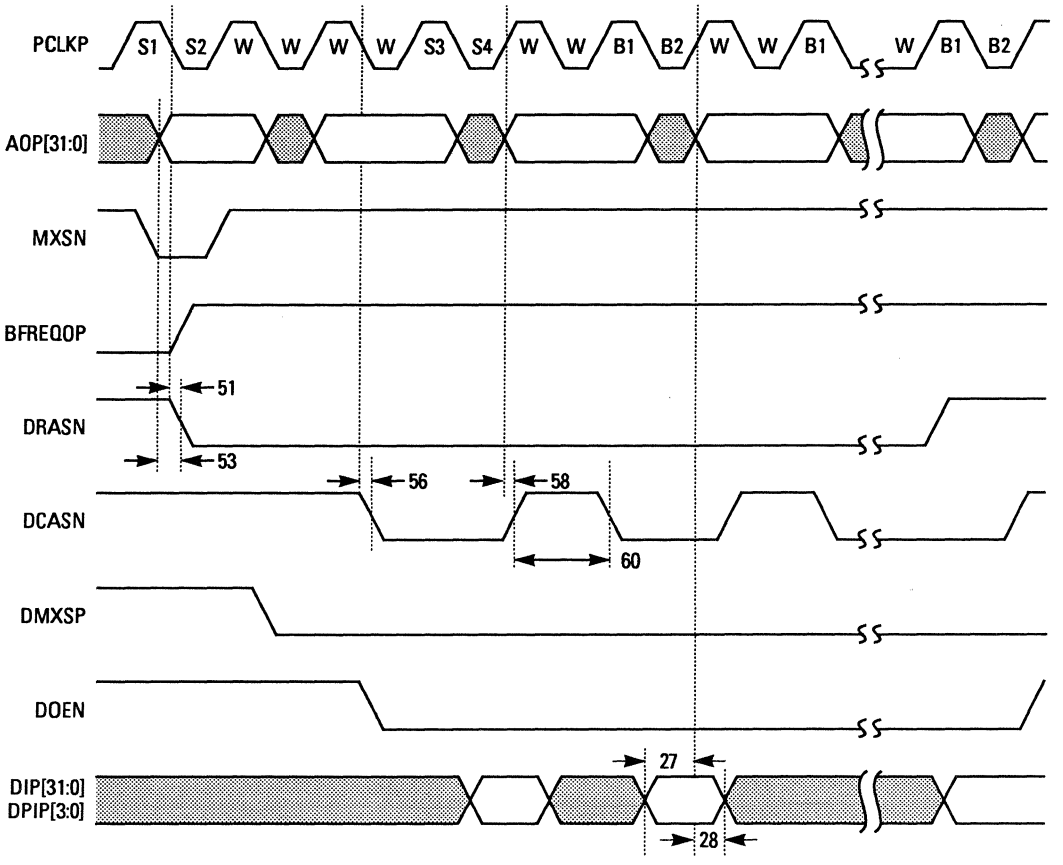


Figure 6.13
DRAM Refresh Timing

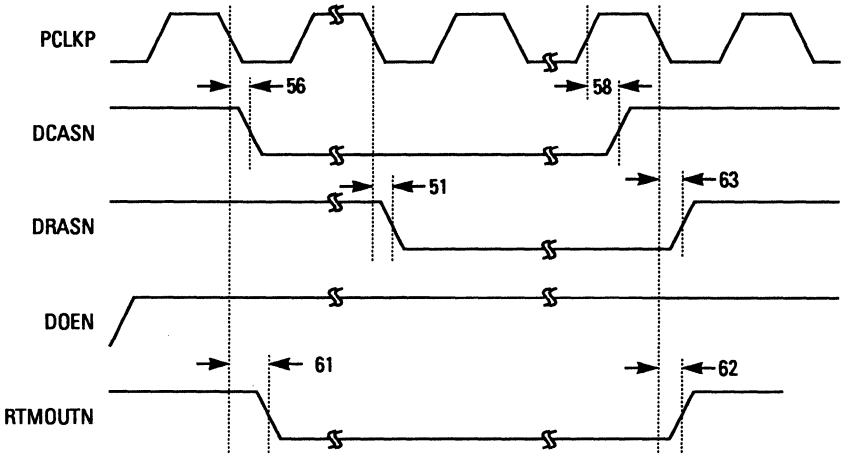
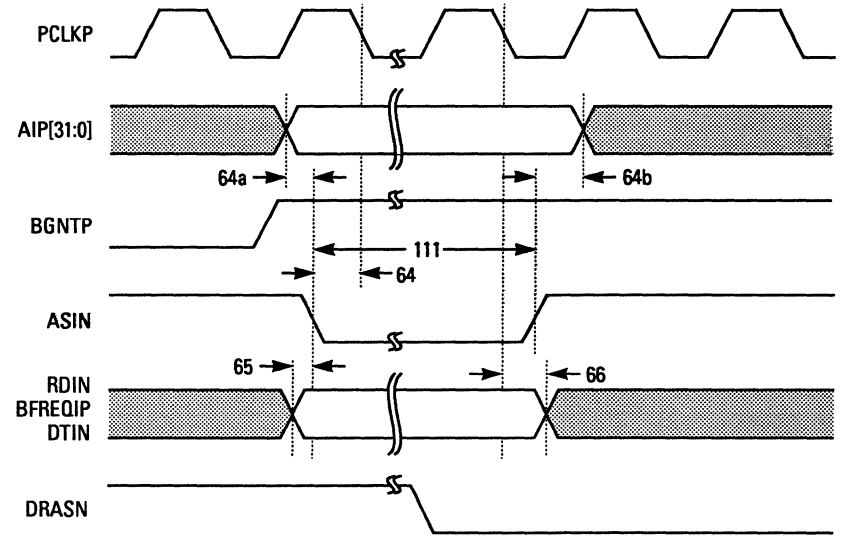
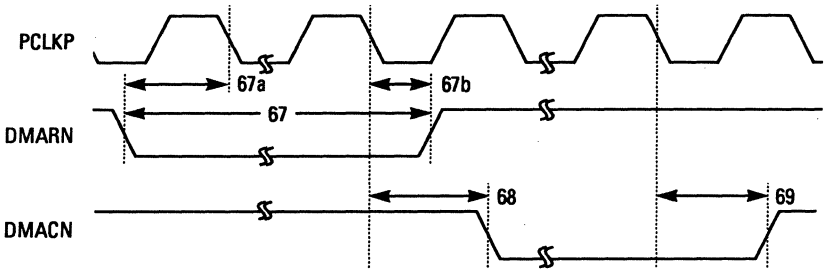


Figure 6.14
DMA Access Timing Using BGNTP



Notes:
1. The external master must assert ASIN until the Controller asserts DRASN (but not RTMOUTN).
2. When an *n*-word block fetch takes place, the DCASN that corresponds to the *n*th word (page-mode DRAM access) marks the end of the transaction.
3. There may be any number of cycles between the assertion of BGNTP and the assertion of ASIN for the purpose of starting the DRAM Controller's access.

Figure 6.15
DMA Access Timing Using DMARN



Note:
1. DMARN is an asynchronous signal. DMARN may be deasserted at any time, once the pulse width is satisfied. DMARN must be deasserted and then reasserted to initiate another DMACN sequence.

Figure 6.16
Timing for Miscellaneous Input and Output Signals

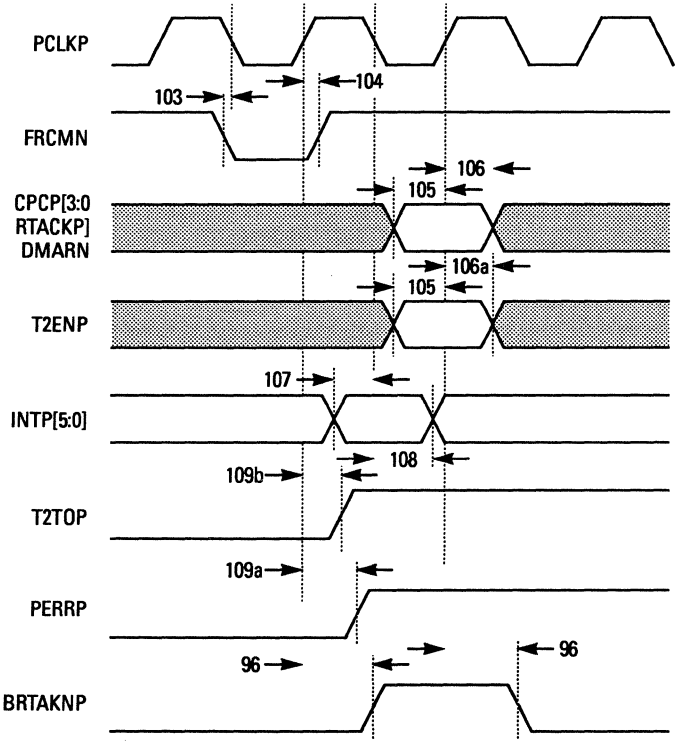
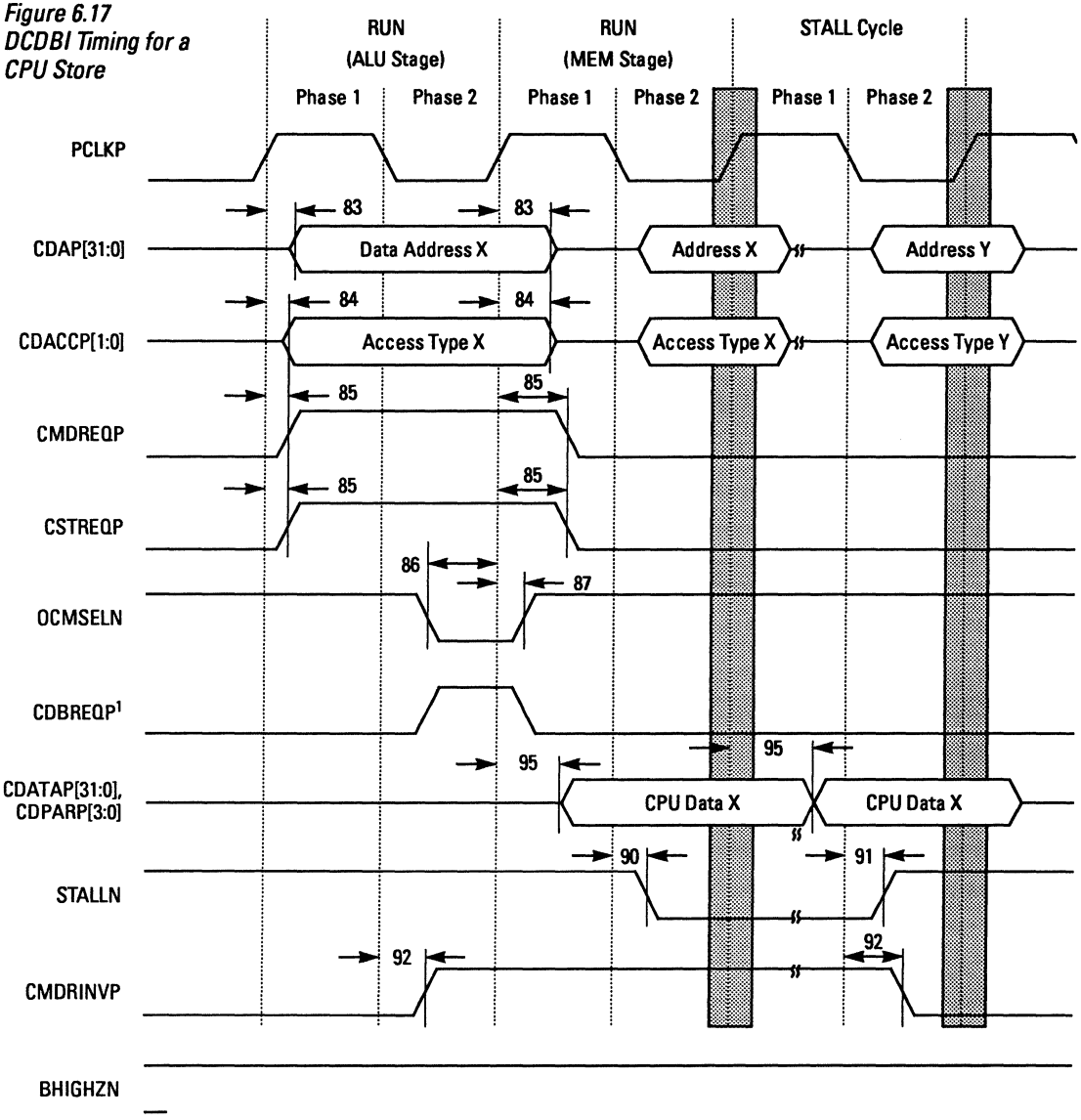


Figure 6.17
DCDBI Timing for a CPU Store



Note:
1. Internal signal shown for reference only.

Figure 6.18
DCDBI Timing for a CPU Load

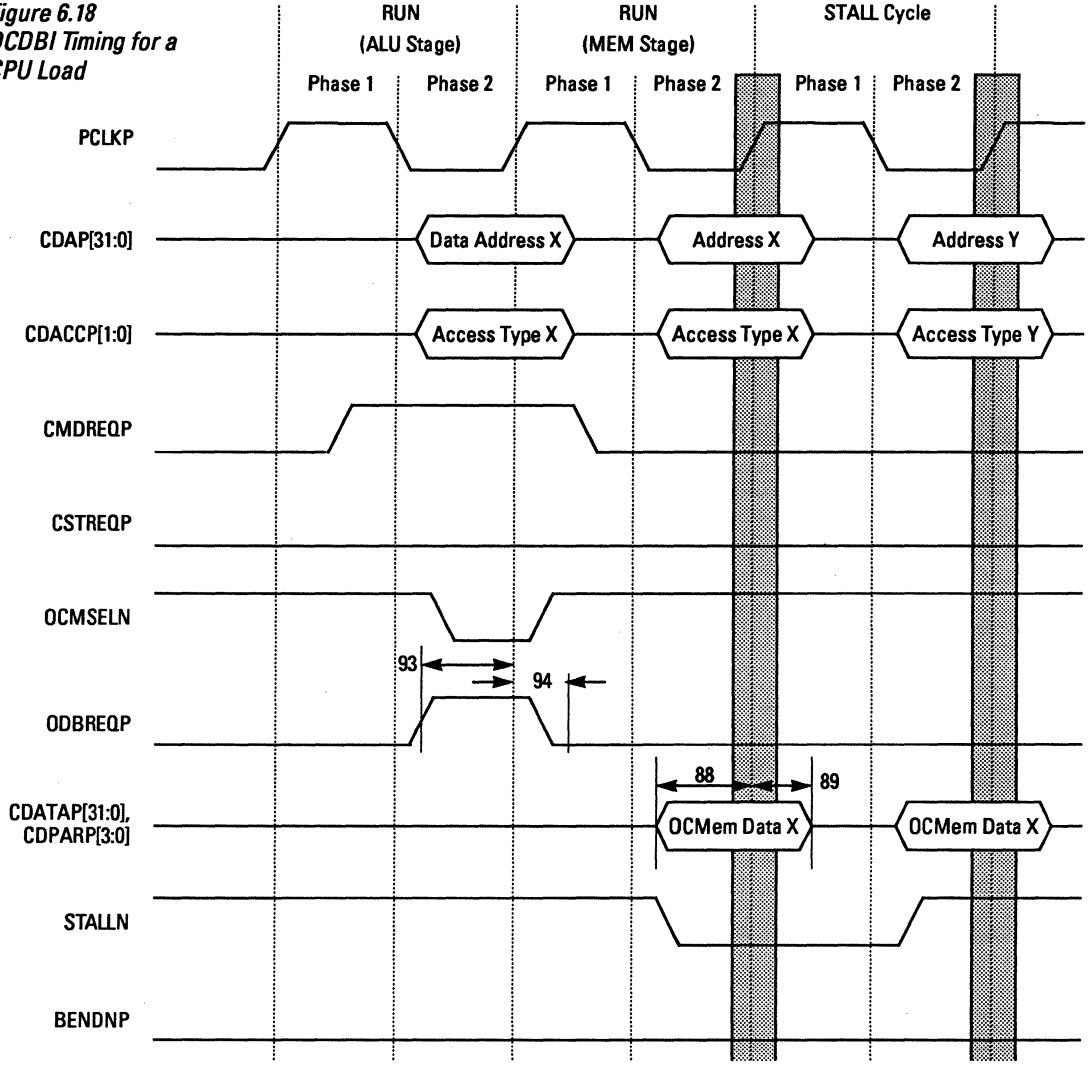


Figure 6.19
Data Cache Data Read Timing

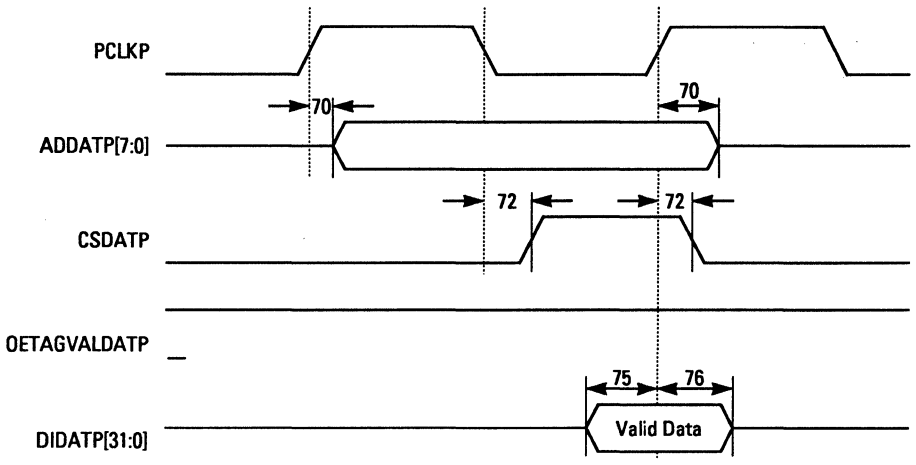


Figure 6.20
Data Cache Tag and Valid Bit Read Timing

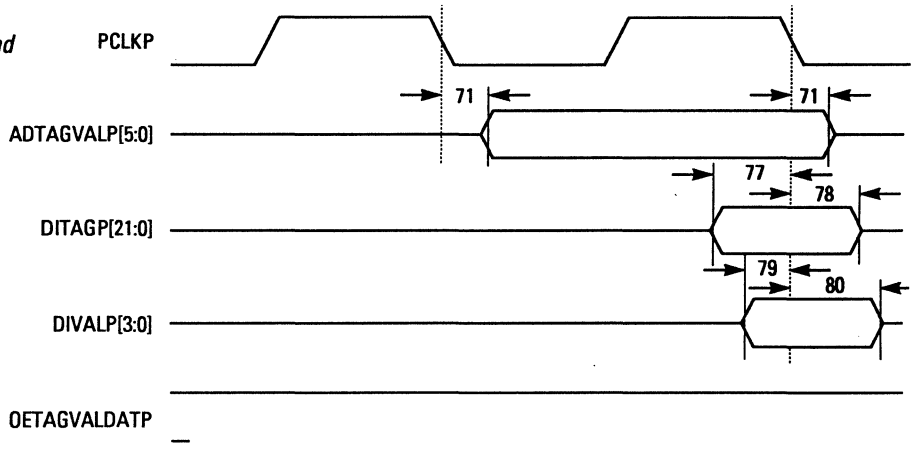


Figure 6.21
Data Cache Data Write Timing

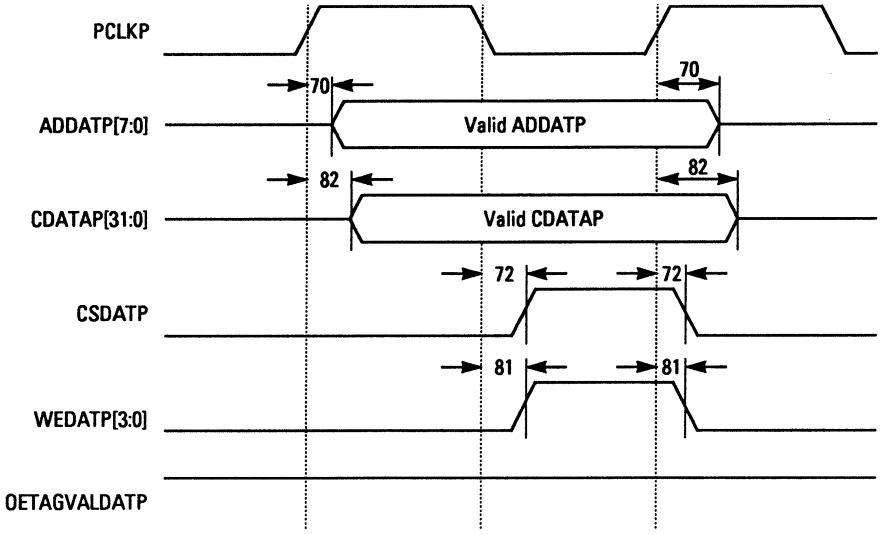


Figure 6.22
Data Cache Tag and Valid Bit Write Timing

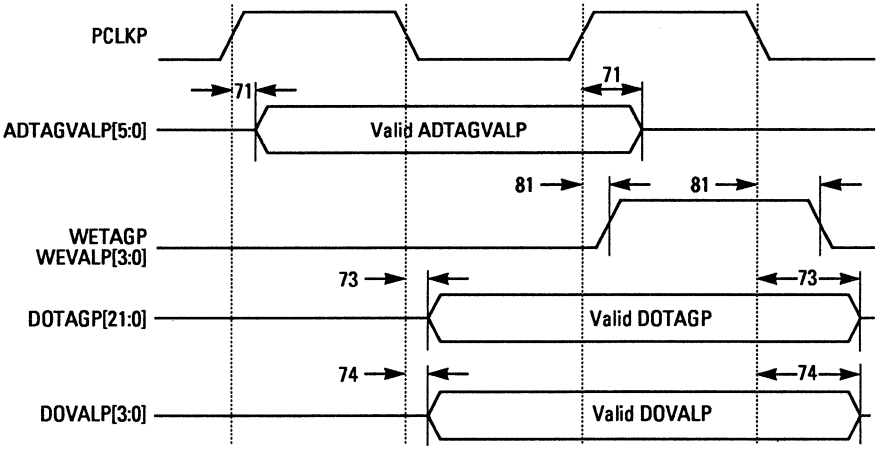


Table 6.1
AC Timing Values

Parameter	*Description*	*25 MHz* *Min*	*Max*	*Units*
1. t_{SH}	PCLKP High (>1.5 V)	18	–	ns
2. t_{SL}	PCLKP Low (< 1.5 V)	18	–	ns
3. t_{SF}	PCLKP Fall (transition)	–	2.5	ns
4. t_{SR}	PCLKP Rise (transition)	–	2.5	ns
5. t_{SCYC}	PCLKP cycle	40	–	ns
6. t_{VRL}	VDD (≥ 3.0 V) to RESETN Low	–	5	ns
7. t_{VHZL}	VDD (≥ 3.0 V) to HIGHZN Low	–	5	ns
8. t_{RLC}	RESETN Low (Cold)[1]	9	–	Cycles
9. t_{RHZH}	RESETN High to HIGHZN High (Cold)[1]	0	–	ns
10. t_{RLW}	RESETN Low (Warm)[1]	4	–	Cycles
11a. t_{HZSC}	HIGHZN Setup before PCLKP+	31	–	ns
11b. t_{HZHC}	HIGHZN Hold from PCLKP+	4	–	ns
11c. t_{HZNM}	HIGHZN- to BIDENN-, DBENN-, HIGHZIOP-, BHIGHZN+	–	8	ns
11d. t_{HZPM}	HIGHZN+ to BIDENN+, DBENN+, HIGHZ-IOP+, BHIGHZN-	–	8	ns
13. t_{SHAV}	PCLKP+ to Address Valid	–	17	ns
14. t_{SHBRV}	PCLKP+ to BFREQOP Valid[2]	–	8	ns
15. t_{SHBRI}	PCLKP+ to BFREQOP Invalid	–	10	ns
16. t_{SHML}	PCLKP+ to MXSN Low	–	7	ns
17. t_{SHRTL}	PCLKP+ to RTON Low	–	9	ns
18. t_{SHRTH}	PCLKP+ to RTON High	–	9	ns
19. t_{SHDTV}	PCLKP+ to DTON Valid	–	8	ns
20. t_{SHDTI}	PCLKP+ to DTON Invalid	–	8	ns
21. t_{SLAL}	PCLKP- to ASON, RDON Low	–	6	ns
22. t_{SLSL}	PCLKP- to IOSELN or EPSELN Low	–	8	ns
23. t_{SLSH}	PCLKP+ to IOSELN or EPSELN High	–	9	ns
24. t_{AVASL}	Address Valid to ASON, RDON, IOSELN, EPSELN Low	12	–	ns
25. t_{CSSL}	Control[3, 4] Setup to PCLKP-	9	–	ns
26. t_{CHSL}	Control[4, 5] Hold from PCLKP-	5	–	ns
26a. t_{BEHSL}	BERRN Hold from PCLKP-	5	–	ns
27. t_{DSSH}	Data and Parity Setup before PCLKP+	5	–	ns

(Sheet 1 of 4)

Table 6.1 (Continued)
AC Timing Values

Parameter	*Description*	*25 MHz Min*	*Max*	*Units*
28. t_{DHSH}	Data and Parity Hold from PCLKP+	5	–	ns
29. t_{SHASH}	PCLKP+ to ASON, RDON High	–	8	ns
30. t_{AHSH}	Address Hold from ASON, RDON, IOSELN, EPSELN High	1	–	ns
30a. t_{AHWH}	Address Hold from WRN[3:0] High	0	–	ns
31. t_{SHWRV}	PCLKP+ to WRN[3:0] Valid	–	6	ns
32. t_{SHDV}	PCLKP+ to Data and Parity Valid	–	17	ns
33. t_{SHWRH}	PCLKP+ to WRN[3:0] High	–	6	ns
34. t_{WRHDI}	WRN[3:0] High to Data and Parity Invalid	6	–	ns
35. t_{BFSSL}	BFTCHN Setup to PCLKP-[6]	9	–	ns
36. t_{BFHSL}	BFTCHN Hold from PCLKP-	0	–	ns
37. t_{SLBAV}	PCLKP- to Block Address[7]	–	14	ns
38. t_{BHZBGH}	Bus Signal Enables to BGNTP High		3	ns
38a. t_{BABGL}	Bus Signals Active after BGNTP Low	0	–	ns
39. t_{BSSH}	BREQN Setup before PCLKP+	5	–	ns
40. t_{SLBGH}	PCLKP- to BGNTP High	–	11	ns
41. t_{BHSH}	BREQN Hold after PCLKP+	5	–	ns
42. t_{SLBGL}	PCLKP- to BGNTP Low	–	11	ns
43. t_{BGHSL}	BGNTP High to ASIN Low	10	–	ns
44. t_{RTVASL}	RTIN Valid to ASIN Low[8]	1	–	ns
45. t_{CAVASL}	CACHDN Valid to ASIN Low[8]	1	–	ns
46. t_{ASSSH}	ASIN Setup before PCLKP+	5	–	ns
47. t_{ASHSH}	ASIN, RTIN, CACHDN Hold from PCLKP+	5	–	ns
48. t_{ASSL}	Address Setup to PCLKP-	5	–	ns
49. t_{AHSL}	Address Hold from PCLKP-	5	–	ns
51. t_{SLRL}	PCLKP- to DRASN Low	–	9	ns
53. t_{ASRL}	Address Setup to DRASN Low	12	–	ns
54. t_{SLMSL}	PCLKP- to DMXSP Low	–	7	ns
55. t_{SLMSH}	PCLKP- to DMXSP High	–	7	ns
56. t_{SLCAL}	PCLKP+/- to DCASN Low	–	8	ns
57. t_{SLOEL}	PCLKP- to DOEN Low	–	8	ns
58. t_{SHCAH}	PCLKP+ to DCASN High	–	10	ns
59. t_{SHOEH}	PCLKP+ to DOEN High	–	20	ns

(Sheet 2 of 4)

Table 6.1 (Continued)
AC Timing Values

Parameter	*Description*	*25 MHz* *Min*	*Max*	*Units*
60. t_{CAPW}	DCASN Precharge Time	20	–	ns
61. t_{SLRTOV}	PCLKP- to RTMOUTN Valid	–	10	ns
62. t_{SLRTOI}	PCLKP- to RTMOUTN Invalid	–	10	ns
63. t_{SLRAH}	PCLKP- to DRASN High	–	10	ns
64. t_{ASVSL}	ASIN Setup to PCLKP-	5	–	ns
64a. t_{ASASL}	Address Setup to ASIN Low	5	–	ns
64b. t_{AHASH}	Address Hold from ASIN High	5	–	ns
65. t_{CSAL}	RDIN, BFREQP, and DTIN Setup to ASIN Low	0	–	ns
66. t_{CHSH}	RDIN, BFREQP, and DTIN Hold from PCLKP-	5	–	ns
67. t_{DMAPW}	DMARN Pulse Width	1	–	Cycle
67a. t_{DMASC}	DMARN Setup to PCLKP-	5	–	ns
67b. t_{DMAHC}	DMARN Hold from PCLKP-	5	–	ns
68. t_{SLDMCV}	PCLKP- to DMACN Valid	–	8	ns
69. t_{SLDMCI}	PCLKP- to DMACN Invalid	–	8	ns
70. t_{CADDT}	PCLKP+ to ADDATP[7:0] Changing	–	5	ns
71. t_{CADTV}	PCLKP- to ADTAGVALP[5:0]	–	9	ns
72. t_{CCSD}	PCLKP- to CSDATP Changing	–	4	ns
73. t_{CDOT}	PCLKP- to DOTAGP[21:0]	–	8	ns
74. t_{CDOV}	PCLKP- to DOVALP[3:0]	–	16	ns
75. t_{DIDSC}	DIDATP[31:0] Setup to PCLKP-	3	–	ns
76. t_{DIDHC}	DIDATP[31:0] Hold from PCLKP-	5	–	ns
77. t_{DITSC}	DITAGP[21:0] Setup to PCLKP+	6	–	ns
78. t_{DITHC}	DITAGP[21:0] Hold from PCLKP+	6	–	ns
79. t_{DIVSC}	DIVALP[3:0] Setup to PCLKP+	3	–	ns
80. t_{DIVHC}	DIVALP[3:0] Hold from PCLKP+	9	–	ns
81. t_{CWDTV}	PCLKP+ to WEDATP[3:0], WETAGP, WEVALP[3:0] High and PCLK- to WEDATP[3:0], WETAGP, WEVALP[3:0] Low	–	4	ns
82. t_{CDOD}	PCLKP+ to CDATAP[31:0]	–	12	ns
83. t_{CCDA}	PCLKP+ to CDAP[31:0] Active or Inactive	–	16	ns
84. t_{CCDACC}	PCLKP+ to CDACCP[1:0] Active or Inactive	–	13	ns

(Sheet 3 of 4)

Table 6.1 (Continued)
AC Timing Values

Parameter	Description	25 MHz Min	25 MHz Max	Units
85. t_{CCREQ}	PCLKP+ to CMDREQP, CSTREQP High or Low	–	13	ns
86. t_{OCMSC}	OCMSELN Setup to PCLKP+	5	–	ns
87. t_{OCMHC}	OCMSELN Hold from PCLKP+	1	–	ns
88. t_{CDSC}	CDATAP[31:0], CDPARP[3:0] Setup to PCLKP+	18	–	ns
89. t_{CDAC}	CDATAP[31:0], CDPARP[3:0] Hold from PCLKP+	0	–	ns
90. t_{CCSTL}	PCLKP- to STALLN Low	–	15	ns
91. t_{CCSTH}	PCLKP- to STALLN High	–	15	ns
92. t_{CCMD}	PCLKP- to CMDRINVP High or Low	–	13	ns
93. t_{ODBSC}	ODBREQP Setup to PCLKP+	5	–	ns
94. t_{ODPHC}	ODBREQP Hold from PCLKP+	2	–	ns
95. t_{CPDP}	PCLKP+ to CDATAP[31:0], CDPARP[3:0]	–	23	ns
96. t_{CNBV}	PCLKP- to BRTAKNP Valid	–	16	ns
103. t_{FSSL}	FRCMN Setup to PCLKP-	16	–	ns
104. t_{FHSH}	FRCMN Hold from PCLKP+	9	–	ns
105. t_{RSSH}	Miscellaneous Synchronous[9] Setup to PCLKP+	5	–	ns
106. t_{RHSH}	Miscellaneous Synchronous[10] Hold from PCLKP+	4	–	ns
106a. t_{THSH}	T2EN Hold from PCLKP+	6	–	ns
107. t_{FSSL}	INTP[5:0] Setup to PCLKP-	5	–	ns
108. t_{FHSH}	INTP[5:0] Hold from PCLKP-	4	–	ns
109a. t_{CPPV}	PCLKP+ to PERRP Valid	–	12	ns
109b. t_{CPTV}	PCLKP+ to T2TOP Valid	–	9	ns
111. t_{ASPW}	ASIN Pulse Width	1	–	Cycle
115. t_{RSCP}	RESETN Setup to PCLKP+	5	–	ns
116. t_{RSCN}	RESETN Hold from PCLKP+	5	–	ns
117. t_{SHDBV}	PCLKP+ to DBENN Valid	–	6	ns
118. t_{SHDBI}	PCLKP+ to DBENN Invalid	–	8	ns

(Sheet 4 of 4)

1. *PCLKP must be active for RESETN to take effect.*
2. *Block-fetch transactions must have one wait state in the first word of a block fetch. A block-fetch transaction is requested on any read access to cacheable memory after a cache miss.*

3. *These control signals include DRDYN, BERRN, CACHDN, BWIDEN, and PENN. To prevent affecting the next transaction, these signals should be deasserted within 1.5 system clock cycles of PCLKP-.*
4. *Once sampled at PCLKP-, BWIDEN is ignored until four bytes have been fetched or a bus error has been detected. PENN is ignored if BWIDEN is sampled at PCLKP-.*
5. *These control signals include DRDYN, CACHDN, BWIDEN, and PENN.*
6. *To prevent affecting the next transaction, BFTCHN should be deasserted within 1.5 system clock cycles of PCLKP-.*
7. *Block address 0 is placed on the address bus in response to BFTCHN LOW. Each subsequent address is issued in response to DRDYN LOW. ASON and RDON are deasserted after all data in a block have been fetched or a bus error occurs.*
8. *Cache invalidation occurs when RTIN is deasserted and CACHDN is asserted.*
9. *Miscellaneous synchronous signals include CPCP[3:0], RTACKP, and T2ENP.*
10. *Miscellaneous synchronous signals include CPCP[3:0] and RTACKP.*

6.2 Loading Information

This section provides specific loading information for the CW33000 inputs and loading and drive type information for the outputs. It also provides loading information for the customer-designed peripheral logic. The section is organized this way:

- Input Loading
- Output Loading and Driver Type
- Customer Output Loading

Input Loading

Table 6.2 lists the total number of equivalent standard loads and total capacitive load presented by the CW33000 inputs to the customer-designed peripheral logic. Note that the number of equivalent standard loads represents contributions from both the fan-in and the wire in the CW33000. Also, capacitive load numbers assume 0.0784 pF per standard load (for the LSI Logic LCB007 process), and both the number of standard loads and the total capacitive load values are rounded off.

Table 6.2
CW33000 Input Loading

Signal Name	*Number of Standard Loads*	*Total Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Total Capacitive Load (pF)*
AIP.31	4.1	0.33	CDATAP.30	50.5	3.96
AIP.30	3.2	0.25	CDATAP.29	54.7	4.29
AIP.29	3.1	0.25	CDATAP.28	49.9	3.92
AIP.28	3.7	0.29	CDATAP.27	59.7	4.69
AIP.27	2.5	0.20	CDATAP.26	64.3	5.05
AIP.26	2.0	0.16	CDATAP.25	64.2	5.04
AIP.25	1.7	0.14	CDATAP.24	62.4	4.90
AIP.24	4.0	0.32	CDATAP.23	78.8	6.18
AIP.23	1.5	0.12	CDATAP.22	68.1	5.34
AIP.22	1.8	0.15	CDATAP.21	70.1	5.50
AIP.21	2.8	0.22	CDATAP.20	70.7	5.55
AIP.20	2.6	0.21	CDATAP.19	70.9	6.05
AIP.19	1.9	0.15	CDATAP.18	76.2	5.56
AIP.18	4.2	0.33	CDATAP.17	64.9	5.09
AIP.17	2.1	0.17	CDATAP.16	70.6	5.54
AIP.16	3.9	0.31	CDATAP.15	73.4	5.76
AIP.15	1.6	0.13	CDATAP.14	70.5	5.53
AIP.14	2.9	0.23	CDATAP.13	71.0	5.57
AIP.13	1.9	0.15	CDATAP.12	71.5	5.61
AIP.12	2.5	0.20	CDATAP.11	67.3	5.28
AIP.11	2.1	0.17	CDATAP.10	63.9	5.01
AIP.10	2.4	0.19	CDATAP.9	71.0	5.57
AIP.9	2.2	0.18	CDATAP.8	73.9	5.80
AIP.8	6.6	0.52	CDATAP.7	73.8	5.79
AIP.7	7.0	0.55	CDATAP.6	76.9	6.03
AIP.6	1.8	0.15	CDATAP.5	68.5	5.37
AIP.5	2.8	0.22	CDATAP.4	70.0	5.49
AIP.4	2.7	0.22	CDATAP.3	68.0	5.34
AIP.3	3.1	0.25	CDATAP.2	77.1	6.05
AIP.2	2.9	0.23	CDATAP.1	74.5	5.84
AIP.1	3.3	0.26	CDATAP.0	73.1	5.74
AIP.0	3.1	0.25	CDPARP.3	28.9	2.27
ASIN	6.9	0.54	CDPARP.2	27.5	2.16
BENDNP	18.6	1.46	CDPARP.1	38.8	3.05
BERRN	35.2	2.76	CDPARP.0	27.8	2.18
BFREQIP	2.9	0.23	CPCP.3	5.2	0.41
BFTCHN	3.3	0.26	CPCP.2	4.8	0.38
BREQN	5.0	0.40	CPCP.1	4.8	0.38
BWIDEN	17.7	1.39	CPCP.0	6.2	0.49
CACHDN	22.3	1.75	DCTSTP	6.0	0.47
CDATAP.31	51.6	4.05	DIDATP.31	2.4	0.19

(Sheet 1 of 3)

Table 6.2 (Continued)
CW33000 Input Loading

Signal Name	*Number of Standard Loads*	*Total Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Total Capacitive Load (pF)*
DIDATP.30	1.4	0.11	DIP.21	4.3	0.34
DIDATP.29	1.4	0.11	DIP.20	4.7	0.37
DIDATP.28	1.9	0.15	DIP.19	4.7	0.37
DIDATP.27	1.3	0.11	DIP.18	4.6	0.36
DIDATP.26	1.4	0.11	DIP.17	4.3	0.34
DIDATP.25	1.9	0.15	DIP.16	4.9	0.39
DIDATP.24	1.6	0.13	DIP.15	4.3	0.34
DIDATP.23	2.2	0.18	DIP.14	4.3	0.34
DIDATP.22	1.9	0.15	DIP.13	4.0	0.31
DIDATP.21	1.7	0.14	DIP.12	4.7	0.37
DIDATP.20	2.0	0.16	DIP.11	4.3	0.34
DIDATP.19	2.1	0.17	DIP.10	4.6	0.36
DIDATP.18	2.2	0.18	DIP.9	4.7	0.37
DIDATP.17	1.9	0.15	DIP.8	4.7	0.37
DIDATP.16	1.5	0.12	DIP.7	20.7	1.63
DIDATP.15	1.8	0.15	DIP.6	30.9	2.43
DIDATP.14	1.9	0.14	DIP.5	26.2	2.06
DIDATP.13	2.3	0.18	DIP.4	23.5	1.84
DIDATP.12	1.6	0.13	DIP.3	22.2	1.75
DIDATP.11	1.8	0.15	DIP.2	23.3	1.83
DIDATP.10	1.6	0.13	DIP.1	22.4	1.76
DIDATP.9	2.0	0.16	DIP.0	20.5	1.61
DIDATP.8	2.0	0.16	DITAGP.21	3.4	0.27
DIDATP.7	1.4	0.11	DITAGP.20	3.2	0.25
DIDATP.6	2.0	0.16	DITAGP.19	3.2	0.25
DIDATP.5	2.5	0.20	DITAGP.18	3.6	0.29
DIDATP.4	1.8	0.15	DITAGP.17	3.3	0.26
DIDATP.3	3.0	0.24	DITAGP.16	3.0	0.24
DIDATP.2	2.3	0.18	DITAGP.15	3.0	0.24
DIDATP.1	2.0	0.16	DITAGP.14	3.2	0.25
DIDATP.0	1.5	0.12	DITAGP.13	3.2	0.25
DIP.31	4.3	0.34	DITAGP.12	3.6	0.29
DIP.30	4.3	0.34	DITAGP.11	3.4	0.27
DIP.29	4.3	0.34	DITAGP.10	3.1	0.25
DIP.28	4.7	0.37	DITAGP.9	3.0	0.24
DIP.27	4.3	0.34	DITAGP.8	3.1	0.25
DIP.26	4.7	0.37	DITAGP.7	3.6	0.29
DIP.25	4.0	0.32	DITAGP.6	3.1	0.25
DIP.24	4.4	0.35	DITAGP.5	3.0	0.24
DIP.23	4.7	0.37	DITAGP.4	3.0	0.24
DIP.22	4.3	0.34	DITAGP.3	2.9	0.23

(Sheet 2 of 3)

Table 6.2 (Continued)
CW33000 Input Loading

Signal Name	*Number of Standard Loads*	*Total Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Total Capacitive Load (pF)*
DITAGP.2	3.2	0.25	ICTSTP	5.9	0.47
DITAGP.1	3.6	0.29	INTP.5	6.6	0.52
DITAGP.0	8.5	0.67	INTP.4	4.3	0.34
DIVALP.3	2.4	0.19	INTP.3	5.6	0.44
DIVALP.2	2.4	0.19	INTP.2	4.6	0.36
DIVALP.1	2.3	0.18	INTP.1	4.7	0.37
DIVALP.0	2.2	0.18	INTP.0	6.4	0.51
DMARN	2.2	0.18	OCMSELN	4.7	0.37
DPIP.3	4.5	0.36	ODBREQP	11.7	0.92
DPIP.2	4.5	0.36	PCLKP	706.9	55.43
DPIP.1	5.1	0.40	PENN	3.6	0.29
DPIP.0	7.1	0.56	RDIN	1.1	0.09
DRDYN	18.2	1.43	RESETN	4.3	0.34
DTIN	5.5	0.44	RTACKP	1.7	0.14
FRCMN	48.2	3.78	RTIN	6.0	0.47
HIGHZN	9.1	0.72	T2ENP	3.5	0.28

(Sheet 3 of 3)

Output Loading and Driver Type

Table 6.3 provides the output loading values for the CW33000. These values represent the load presented looking back into the CW33000 outputs. Note that the number of equivalent standard loads represents contributions from both the fan-out and the wire in the CW33000. Also, capacitive load numbers assume 0.0784 pF per standard load (for the LSI Logic LCB007 process), and both the number of standard loads and the total capacitive load values are rounded off. Table 6.4 lists the driver type for each of the outputs.

Table 6.3
CW33000 Output Loading

Signal Name	*Number of Standard Loads*	*Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Capacitive Load (pF)*
ADDATP.7	1.4	0.11	ADDATP.0	1.5	0.12
ADDATP.6	1.4	0.11	ADTAGVALP.5	10.7	0.84
ADDATP.5	1.5	0.12	ADTAGVALP.4	9.9	0.78
ADDATP.4	1.4	0.11	ADTAGVALP.3	13.2	1.04
ADDATP.3	1.4	0.11	ADTAGVALP.2	9.6	0.76
ADDATP.2	1.8	0.15	ADTAGVALP.1	8.7	0.69
ADDATP.1	2.2	0.18	ADTAGVALP.0	11.1	0.87

(Sheet 1 of 4)

Table 6.3 (Continued)
CW33000 Output Loading

Signal Name	*Number of Standard Loads*	*Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Capacitive Load (pF)*
AOP.31	22.5	1.77	CDAP.31	41.8	3.28
AOP.30	29.6	2.32	CDAP.30	43.2	3.39
AOP.29	25.8	2.03	CDAP.29	42.3	3.32
AOP.28	29.9	2.35	CDAP.28	43.2	3.39
AOP.27	22.9	1.80	CDAP.27	42.5	3.34
AOP.26	25.4	2.00	CDAP.26	41.8	3.28
AOP.25	21.8	1.71	CDAP.25	44.4	3.49
AOP.24	28.9	2.27	CDAP.24	40.5	3.18
AOP.23	24.9	1.96	CDAP.23	47.0	3.69
AOP.22	23.8	1.87	CDAP.22	42.0	3.30
AOP.21	19.8	1.56	CDAP.21	39.4	3.09
AOP.20	26.2	2.06	CDAP.20	38.0	3.27
AOP.19	27.5	2.16	CDAP.19	37.9	3.09
AOP.18	22.1	1.74	CDAP.18	39.7	3.12
AOP.17	21.0	1.65	CDAP.17	39.3	3.09
AOP.16	19.8	1.56	CDAP.16	41.7	3.27
AOP.15	17.4	1.37	CDAP.15	41.9	3.29
AOP.14	25.8	2.03	CDAP.14	43.0	3.38
AOP.13	19.2	1.51	CDAP.13	41.8	3.28
AOP.12	26.2	2.06	CDAP.12	43.8	3.44
AOP.11	25.6	2.01	CDAP.11	38.8	3.05
AOP.10	24.3	1.91	CDAP.10	49.3	3.87
AOP.9	27.7	2.18	CDAP.9	48.9	3.84
AOP.8	20.8	1.63	CDAP.8	47.6	3.74
AOP.7	26.0	2.04	CDAP.7	48.9	3.84
AOP.6	28.9	2.27	CDAP.6	49.4	3.88
AOP.5	10.4	0.82	CDAP.5	56.7	4.45
AOP.4	9.9	0.78	CDAP.4	56.6	4.44
AOP.3	9.3	0.73	CDAP.3	55.7	4.37
AOP.2	10.0	0.79	CDAP.2	54.2	4.25
AOP.1	10.1	0.80	CDAP.1	53.9	4.23
AOP.0	10.1	0.80	CDAP.0	56.1	4.40
ASON	4.7	0.37	CDATAP.31	51.6	4.05
BFREQOP	5.0	0.39	CDATAP.30	50.5	3.96
BGNTP	21.5	1.69	CDATAP.29	54.7	4.29
BHIGHZN	56.8	4.46	CDATAP.28	49.9	3.92
BIDENN	6.5	0.51	CDATAP.27	59.7	4.68
BRESETP	93.9	7.37	CDATAP.26	64.3	5.05
BRTAKNP	24.7	1.94	CDATAP.25	64.2	5.04
CDACCP.1	37.6	2.95	CDATAP.24	62.4	4.90
CDACCP.0	39.3	3.09	CDATAP.23	78.8	6.18

(Sheet 2 of 4)

Table 6.3 (Continued)
CW33000 Output Loading

Signal Name	*Number of Standard Loads*	*Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Capacitive Load (pF)*
CDATAP.22	68.1	5.34	DOP.22	37.2	2.92
CDATAP.21	70.1	5.50	DOP.21	35.0	2.75
CDATAP.20	70.7	5.55	DOP.20	35.7	2.80
CDATAP.19	70.9	5.56	DOP.19	35.8	2.81
CDATAP.18	76.2	5.98	DOP.18	39.4	3.09
CDATAP.17	64.9	5.09	DOP.17	33.6	2.64
CDATAP.16	70.6	5.54	DOP.16	36.8	2.89
CDATAP.15	73.4	5.76	DOP.15	34.6	2.72
CDATAP.14	70.5	5.53	DOP.14	38.9	3.05
CDATAP.13	71.0	5.57	DOP.13	36.8	2.89
CDATAP.12	71.5	5.61	DOP.12	42.1	3.30
CDATAP.11	67.3	5.28	DOP.11	38.7	3.04
CDATAP.10	63.9	5.01	DOP.10	33.8	2.65
CDATAP.9	71.0	5.57	DOP.9	36.0	2.83
CDATAP.8	73.9	5.80	DOP.8	36.1	2.83
CDATAP.7	73.8	5.79	DOP.7	32.9	2.58
CDATAP.6	76.9	6.03	DOP.6	33.5	2.63
CDATAP.5	68.5	5.37	DOP.5	36.0	2.83
CDATAP.4	70.0	5.49	DOP.4	38.5	3.02
CDATAP.3	68.0	5.34	DOP.3	32.9	2.58
CDATAP.2	77.1	6.05	DOP.2	36.9	2.90
CDATAP.1	74.5	5.84	DOP.1	28.7	2.25
CDATAP.0	73.1	5.74	DOP.0	35.0	2.75
CMDREQP	32.1	2.52	DOTAGP.21	3.0	0.24
CMDRINVP	34.1	2.68	DOTAGP.20	3.6	0.29
CSDATP	3.3	0.26	DOTAGP.19	4.9	0.39
CSTREQP	62.7	4.92	DOTAGP.18	8.9	0.70
DBENN	7.2	0.57	DOTAGP.17	4.7	0.37
DCASN	4.6	0.36	DOTAGP.16	6.9	0.54
DMACN	11.7	0.92	DOTAGP.15	7.7	0.61
DMXSP	2.3	0.18	DOTAGP.14	5.7	0.45
DOEN	1.2	0.10	DOTAGP.13	4.3	0.38
DOP.31	40.5	3.18	DOTAGP.12	3.4	0.27
DOP.30	38.4	3.01	DOTAGP.11	8.3	0.65
DOP.29	35.9	2.82	DOTAGP.10	7.6	0.60
DOP.28	34.2	2.69	DOTAGP.9	2.7	0.22
DOP.27	34.9	2.74	DOTAGP.8	3.1	0.25
DOP.26	28.0	2.20	DOTAGP.7	6.5	0.51
DOP.25	30.8	2.42	DOTAGP.6	6.1	0.48
DOP.24	34.4	2.70	DOTAGP.5	2.8	0.22
DOP.23	34.1	2.68	DOTAGP.4	2.5	0.20

(Sheet 3 of 4)

Table 6.3 (Continued)
CW33000 Output Loading

Signal Name	*Number of Standard Loads*	*Capacitive Load (pF)*	*Signal Name*	*Number of Standard Loads*	*Capacitive Load (pF)*
DOTAGP.3	6.4	0.51	RDON	5.3	0.42
DOTAGP.2	6.3	0.50	RTMOUTN	2.8	0.22
DOTAGP.1	2.3	0.18	RTON	53.6	4.21
DOTAGP.0	10.4	0.82	STALLN	112.4	8.82
DOVALP.3	3.4	0.27	TDONEP	17.7	1.39
DOVALP.2	2.7	0.22	THITP	5.3	0.42
DOVALP.1	2.7	0.22	T2TOP	5.3	0.42
DOVALP.0	3.6	0.29	WEDATP.3	3.9	0.31
DPOP.3	2.3	0.18	WEDATP.2	2.8	0.22
DPOP.2	1.5	0.12	WEDATP.1	2.8	0.22
DPOP.1	2.2	0.18	WEDATP.0	3.2	0.25
DPOP.0	2.2	0.18	WETAGP	2.6	0.21
DRASN	2.0	0.16	WEVALP.3	2.8	0.22
DTON	40.8	3.20	WEVALP.2	2.9	0.23
EPSELN	6.3	0.50	WEVALP.1	2.7	0.22
HIGHZIOP	5.5	0.44	WEVALP.0	3.1	0.25
IOSELN	5.1	0.40	WRN.3	7.4	0.58
MXSN	6.0	0.47	WRN.2	5.2	0.41
OETAGVALDATP	2.0	0.16	WRN.1	5.3	0.42
PERRP	10.4	0.82	WRN.0	6.1	0.48

(Sheet 4 of 4)

Table 6.4
CW33000 Output Driver Type

Signal Name	*Driver Type*	*Signal Name*	*Driver Type*
ADDATP[7:0]	2 x C	DOP[31:0]	C
ADTAGVALP[5:2]	2 x C	DOTAGP[21:0]	2 x C
ADTAGVALP[1:0]	C	DOVALP[3:0]	C
AOP[31:0]	C	DPOP[3:0]	C
ASON	2 x C	DRASN	2 x C
BFREQOP	C	DTON	2 x C
BGNTP	C	EPSELN	2 x C
BHIGHZN	C	HIGHZIOP	C
BIDENN	3 x C	IOSELN	2 x C
BRESETP	C	MXSN	2 x C
BRTAKNP	B	OETAGVALDATP	A
CDACCP[1:0]	C	PERRP	B
CDAP[31:0]	D	RDON	2 x C
CDATAP[31:0]	C	RTMOUTN	C
CMDRINVP	C	RTON	2 x C
CMDREQP	C	STALLN	2 x C
CSDATP	C	T2TOP	C
CSTREQP	C	TDONEP	C
DBENN	3 x C	THITP	C
DCASN	2 x C	WEDATP[3:0]	B
DMACN	2 x C	WETAGP	C
DMXSP	2 x C	WEVALP[3:0]	C
DOEN	2 x C	WRN[3:0]	C

Customer Output Loading

Table 6.5 lists the assumed customer loading on the CW33000 outputs. To ensure the CW33000 timing given earlier in this chapter, the customer design must not present loading in excess of the values provided in the table.

The customer loading numbers make the following assumptions. Note that the values for wire and fan-out capacitance are process-specific, so the calculations must be repeated if a different process is used.

1. Process: For the LSI Logic LCB007 process, the unit wire capacitance is 0.1702 pF/ and one standard load is 0.0784 pF.
2. Data Cache: The data cache, if used, must be positioned such that the interconnect length does not exceed 7 mm, and the data cache signals support no other interconnect.
   - Wire contribution per signal = 7 mm x (0.1702 pF/mm) = 1.2 pF
   - Fan-out contribution = 0.3 pF for RAM
   - Total load for each data cache signal = 1.2 + 0.3 = 1.5 pF
3. CPU Control Signals: For each CPU control signal, the customer peripheral logic presents a maximum of 15 mm of wire and a fan-out of 5.
   - Wire contribution per signal = 15 mm x (0.1702 pF/mm) = 2.6 pF
   - Fan-out contribution = 5 std. lds. x 0.0784 pF/std. ld. = 0.3 pF
   - Total load for each CPU control signal = 2.9 pF = 3 pF (rounded)
4. Control Signals: For each control signal, the customer peripheral logic presents a maximum of 20 mm of wire and a fan-out of 5.
   - Wire contribution per signal = 20 mm x (0.1702 pF/mm) = 3.4 pF
   - Fan-out contribution = 5 std. lds. x 0.0784 pF/std. ld. = 0.4 pF
   - Total load for each control signal = 3.8 pF
5. CDATAP Bus: Both the data cache and on-chip memory, if used, contribute to the loading on this bus.
   - Data cache wire contribution per signal (for 7 mm) = 1.2 pF
   - D-cache data input contribution per signal (1 std. ld.) = 0.08 pF
   - On-chip memory wire contribution per signal (for 7 mm) = 1.2 pF
   - On-chip memory input contribution per signal (1 std. ld.) = 0.08 pF

- Total load for each CDATAP signal = 2.56 pF = 3 pF (rounded)

6. CDAP Bus: Assume 7 mm of wire and a fan-out of 4
   - Wire contribution per signal = 7 mm x (0.1702 pF/mm) = 1.2 pF
   - Fan-out contribution = 4 std. lds. x 0.0784 pF/std. ld. = 0.3 pF
   - Total load for each CDAP signal = 1.5 pF

Table 6.5 Customer Output Loading

Signal Name	*Customer Load presented to the CW33000 (pF)*	*Signal Name*	*Customer Load presented to the CW33000 (pF)*
ADDATP[7:0]	1.5	DOTAGP[21:0]	1.3
ADTAGVALP[5:0]	1.5	DOVALP[3:0]	1.3
AOP[31:0]	3.8	DPOP[3:0]	3.8
ASON	4.5	DRASN	3.5
BFREQOP	3.5	DTON	3.5
BGNTP	3.5	EPSELN	3.5
BHIGHZN	3.8	HIGHZIOP	3.8
BIDENN	3.8	IOSELN	3.5
BRESETP	4.5	MXSN	3.8
BRTAKNP	3.5	OETAGVALDATP	5.0
CDACCP[1:0]	3.8	PERRP	3.5
CDAP[31:0]	1.5	RDON	3.8
CDATAP[31:0]	3.0	RTMOUTN	3.5
CMDRINVP	3.0	RTON	3.5
CMDREQP	3.0	STALLN[1]	1.5
CSDATP	2.0	T2TOP	3.5
CSTREQP	3.0	TDONEP	3.5
DBENN	3.8	THITP	3.5
DCASN	3.5	WEDATP[3:0]	2.2
DMACN	3.5	WETAGP	1.4
DMXSP	3.5	WEVALP[3:0]	1.4
DOEN	3.5	WRN[3:0]	3.8
DOP[31:0]	3.8		

1. STALLN is a performance-critical signal; keep the external loading on STALLN as low as possible.

Appendix A
Customer Feedback

We would appreciate your feedback on this document. Please copy the following page, add your comments, and fax it to us at:

LSI Logic Corporation
Microprocessor Publications
M/S G-812
Fax 408.433.8989

If appropriate, please also fax copies of any marked-up pages from this document.

IMPORTANT: Please include your name, phone number, FAX number, and company address so that we may contact you directly for clarification or additional information. Thank you for your help in improving the quality of our documents.

Reader's Comments

FAX your comments to:

LSI Logic Corporation
Microprocessor Publications
M/S G-812
Fax 408.433.8989

Does the publication meet your needs? Yes__ No__

Is the material:

- Complete? Yes__ No__
- Easy to use? Yes__ No__
- Written for the appropriate technical level? Yes__ No__

What could we do to improve this document?

_____

_____

_____

_____

If you found errors in this document, please specify the error and page number. If appropriate, please fax a marked-up copy of the page(s).

_____

_____

_____

Please complete the information below.

Name _____ Date _____

Telephone _____ FAX _____

Title _____

Department _____ Mail Stop _____

Company Name _____

Street _____

City _____

State _____ Zip _____

Your feedback is important to us. Thank you for your comments.

U.S. Distributors by State

CAlabama
Huntsville
Hamilton Avnet
205.837.9177

Arizona
Phoenix
■ Hamilton Avnet
602.961.6401

Wyle Laboratories
602.437.2088

California
Calabassas
■ Wyle Laboratories
818.880.9000

Chatsworth
■ Hamilton Avnet
818.594.8200

Costa Mesa
■ Hamilton Avnet
714.641.4100

Gardena
Hamilton Avnet
213.217.6700

Irvine
■ Wyle Laboratories
714.863.9953

Sacramento
Hamilton Avnet
916.648.3264

San Diego
Hamilton Avnet
619.571.7500

Wyle Laboratories
619.565.9171

Santa Clara
■ Wyle Laboratories
408.727.2500

Sunnyvale
■ Hamilton Avnet
408.743.3300

Colorado
Denver
Hamilton Avnet
303.799.0663

■ Wyle Laboratories
303.457.9953

Connecticut
Danbury
Hamilton Avnet
203.797.1100

Florida
Miami/Ft Lauderdale
Hamilton Avnet
305.979.2802

Orlando
Hamilton Avnet
407.628.3888

St Petersburg
Hamilton Avnet
813.573.3930

Georgia
Atlanta
Hamilton Avnet
404.447.7500

Illinois
Chicago
■ Hamilton Avnet
708.860.7780

Indiana
Indianapolis
Hamilton Avnet
317.844.9333

Iowa
Cedar Rapids
Hamilton Avnet
319.362.4757

Kansas
Kansas City
Hamilton Avnet
913.888.0155

Kentucky
Lexington
Hamilton Avnet
606.259.1475

Maryland
Baltimore
■ Hamilton Avnet
301.995.3580

Massachusetts
Boston
■ Wyle Laboratories
617.272.7300

■ Hamilton Avnet
508.532.3701

Michigan
Detroit
Hamilton Avnet
313.347.4270

Grand Rapids
Hamilton Avnet
616.243.8805

Minnesota
Minneapolis
Hamilton Avnet
612.932.0678

Missouri
Kansas City
Hamilton Avnet
913.888.0155

St. Louis
Hamilton Avnet
314.537.4250

New Jersey
North Jersey
Hamilton Avnet
201.575.3490

South Jersey
Hamilton Avnet
609.424.0110

New Mexico
Alburquerque
Hamilton Avnet
505.765.1500

New York
Long Island
Hamilton Avnet
516.231.9800

Rochester
Hamilton Avnet
716.475.9140

Syracuse
Hamilton Avnet
315.437.2641

North Carolina
Raleigh
Hamilton Avnet
919.872.3604

Ohio
Cleveland
Hamilton Avnet
216.349.4910

Columbus
Hamilton Avnet
614.882.7389

Dayton
Hamilton Avnet
513.439.6700

Oregon
Portland
Hamilton Avnet
503.526.6200

Wyle Laboratories
503.643.7900

Oklahoma
Tulsa
Hamilton Avnet
918.252.7297

Pennsylvania
Pittsburgh
Hamilton Avnet
412.281.4150

Texas
Austin
Hamilton Avnet
512.832.4306

Wyle Laboratories
512.345.8853

Dallas
Hamilton Avnet
214.404.9906

Wyle Laboratories
214.235.9953

Houston
Hamilton Avnet
713.240.7898

Wyle Laboratories
713.879.9953

Utah
Salt Lake City
Hamilton Avnet
801.972.4300

Wyle Laboratories
801.974.9953

Washington
Seattle
Hamilton Avnet
206.881.6697

Wyle Laboratories
206.881.1150

Wisconsin
Milwaukee
Hamilton Avnet
414.784.4510

■ Distributors with Design Resource Centers

Sales Offices and Design Resource Centers

LSI Logic Corporation
U.S. Headquarters
■ **1551 McCarthy Blvd**
Milpitas CA 95035
Tel: 408.433.8000
Telex: 171092
Fax: 408.434.6457

Alabama
■ 600 Boulevard South
Suite 104J
Huntsville AL 35802
Tel: 205.883.3527
Fax: 205.883.3529

Arizona
8283 N Hayden Road
Suite 270
Scottsdale AZ 85258
Tel: 602.951.4560
Fax: 602.951.4580

California
■ 2540 N First Street
Suite 201
San Jose CA 95131
Tel: 408.954.1561
Fax: 408.954.1565

■ Two Park Plaza
Suite 1000
Irvine CA 92714
Tel: 714.553.5600
Fax: 714.474.8101

10731 Treena Street
Suite 209
San Diego CA 92131
Tel: 619.689.7140
Fax: 619.689.7145

■ 15821 Ventura Blvd
Suite 275
Encino CA 91436
Tel: 818.379.2400
Fax: 818.783.5548

Colorado
3333 Quebec Street
Suite 6014
Denver CO 80207
Tel: 303.399.1112
Fax: 303.399.6558

Florida
1333 Gateway Drive
Suite 1019C
Melbourne FL 32901
Tel: 407.728.9481
Fax: 407.728.9587

■ 1900 Glades Road
Suite 201
Boca Raton FL 33431
Tel: 407.395.6200
Fax: 407.394.2865

Illinois
■ 1450 American Lane
Suite 1610
Schaumburg IL 60173
Tel: 708.995.1600
Fax: 708.995.1622

Maryland
■ 6903 Rockledge Dr
Suite 230
Bethesda MD 20817
Tel: 410.897.5800
Fax: 410.897.8389

10480 Little Patuxent Pkwy
Suite 500
Columbia MD 21044-3502
Tel: 301.740.5664
Fax: 301.740.2048

Massachusetts
■ 9 Hillside Avenue
Prospect Place
Waltham MA 02154
Tel: 617.890.0180
Fax: 617.890.6158

Minnesota
■ 8300 Norman Center Drive
Suite 730
Minneapolis MN 55437
Tel: 612.921.8300
Fax: 612.921.8399

New Jersey
■ 379 Thornall Street
Edison NJ 08837
Tel: 908.549.4500
Fax: 908.549.4802

New York
1065 Route 82
Hopewell Junction
New York NY 12533
Tel: 914.226.1620
Fax: 914.226.1315

Eastview Park
7979 Victor-Pittsford Road
Victor NY 14564
Tel: 716.223.8820
Fax: 716.223.8822

North Carolina
■ 4601 Six Forks Road
Phase 2, Suite 528
Raleigh NC 27609
Tel: 919.783.8833
Fax: 919.783.8909

Ohio
6784 Loop Road
Centerville OH 45459
Tel: 513.438.2821
Fax: 513.438.2317

Oregon
■ 15455 NW Greenbrier Pkwy
Suite 210A
Beaverton OR 97006
Tel: 503.645.9882
Fax: 503.645.6612

Pennsylvania
2300 Computer Ave
Building G
Willow Grove PA 19090
Tel: 215.830.1404
Fax: 215.657.4920

Texas
1120 Capital of Texas
Highway South
Building 2
Suite 300
Austin TX 78746
Tel: 512.329.1044
Fax: 512.329.6331

■ 5080 Spectrum Drive
Suite 1010 West
Dallas TX 75248
Tel: 214.788.2966
Fax: 214.233.9234

Washington
■ 3015 112th Avenue NE
Suite 205
Bellevue WA 98004
Tel: 206.822.4384
Fax: 206.827.2884

LSI Logic Corporation
of Canada Inc
Headquarters
■ Petro-Canada Centre
Suite 3410
150 6th Avenue SW
Calgary AB T2P 3Y7
Tel: 403.262.9292
Fax: 403.262.9494

■ 6400 Roberts Street
Suite 492
Burnaby BC V5G 4C9
Tel: 604.294.8444
Fax: 604.294.8443

■ P.O. Box 8249
Station F
Edmonton AB T6H 4PI
Tel: 403.450.4400
Fax: 403.450.4411

■ 260 Hearst Way
Suite 400
Kanata ON K2L 3H1
Tel: 613.592.1263
Telex: 053.3849
Fax: 613.592.3253

■ 755 St Jean Boulevard
Suite 600
Pointe Claire PQ H9R 5M9
Tel: 514.694.2417
Fax: 514.694.2699

■ 401 The West Mall
Suite 1110
Etobicoke ON M9C 5J5
Tel: 416.620.7400
Fax: 416.620.5005

Denmark
EV Johanssen Electronic AS
■ Titangade 15
DK-2200 Copenhagen
Tel: 45.31.839022
Fax: 45.31.839222

Finland
Oy Fintronic AB
Heikkilantie 2 A
00210 Helsinki
Tel: 358.0.6926022
Fax: 358.0.6821251

France
LSI Logic SA
■ Tour Chenonceaux
204 Rond-Point
du Pont de Sevres
F-92516 Boulogne-Billancourt
Paris
Tel: 33.146.206600
Telex: 631475
Fax: 33.146.203138

Microel SA
Immeuble "Micro"
Avenue de la Baltique
ZA de Courtaboeuf BP 3
F-91941 Les Ulis Cedex
Tel: 33.1.69070824
Fax: 33.1.69071723

Germany
LSI Logic GmbH
European Headquarters
Kronstadter Strasse 9
W-8000 Munich 80
Tel: 49.89.99313100
Fax: 49.89.936806

■ Arabellastrasse 33
W-8000 Munich 81
Tel: 49.89.9269030
Fax: 49.89.917096

■ Am Seestern
Niederkasseler Lohweg 8
W-4000 Dusseldorf 11
Tel: 49.211.5961066
Fax: 49.211.592130

■ Mittlerer Pfad 4
W-7000 Stuttgart 31
Tel: 49.711.139690
Fax: 49.711.8661428

AE Advanced Electronics GmbH
■ Otto-Hahn-Strasse
W-6277 Bad Camberg
Tel: 49.6434.5041
Fax: 49.6434.4277

AL Advanced Logic GmbH
Strassfurther Strasse 1
0-3014 Magdeburg
Tel: 37.91.48112
Fax: 37.91.48112

AE Advanced Electronics GmbH
■ Stefan-George-Ring 19
W-8000 Munich 81
Tel: 49.89.9300980
Fax: 49.89.93009866

AE Advanced Electronics GmbH
Mittlerer Pfad 4
W-7000 Stuttgart 31
Tel: 49.711.881118
Fax: 49.711.8661359

Israel
LSI Logic Limited
■ 40 Sokolov St
Ramat Hasharon 47235
P.O.B. 1331
Tel: 972.3.5403741
Telex: 371662
Fax: 972.3.5403747

Italy
LSI Logic SPA
■ Centro Direnzionale Colleoni-
Palazzo Orione Ingresso 1
20041 Agrate Brianza
Milano
Tel: 39.39.6056881
Fax: 39.39.6057867

Sales Offices and Design Resource Centers (Continued)

Japan
LSI Logic KK
Headquarters
■ Kokusai-Shin Akasaka Bldg.
West Wing, 6-1-20 Akasaka
Minato-ku Tokyo 107
Tel: 81.33.589.2711
Fax: 81.33.589.2740

■ 2-10-1 Kashuga
Tsukuba-shi 305
Tel: 81.298.52.8371
Fax: 81.298.52.8376

■ Chrystal Tower 14F
1-2-27 Shiromi
Chuo-ku Osaka 540
Tel: 81.6.947.5281
Fax: 81.6.947.5287

■ Tama-Plaza Daisen Bldg.
2-19-2 Utsukushigaoka
Midori-ku Yokohama 227
Tel: 81.45.902.4111
Fax: 81.45.902.4533

Tsukuba Mitsui Building
1-6-1 Takezono
Tsukuba-shi 305
Tel: 81.298.56.5011
Fax: 81.298.56.5030

LSI Logic Corporation of Korea Limited
■ 7th Floor Namseoul Bldg.
1304-3 Seocho-Dong
Seocho-ku Seoul
Tel: 82.2.561.2921
Fax: 82.2.554.9327

Netherlands
Arcobel BV
■ Griekenweg 25
Postbox 344
NL-5340 AH Oss
Tel: 31.4120.30335
TWX: 37489
Fax: 31.4120.30635

Norway
Art Chip AS
Ole Deviks vei 44
N-0668 Oslo 6
Tel: 47.2.720740
Fax: 47.2.656960

Spain
LSI Logic SA
Orense 68
28020 Madrid
Tel: 34.1.5705600
Fax: 34.1.5702807

Sweden
LSI Logic Limited
■ Torhamnsgatan 39
S-16440 KISTA
Tel: 46.8.703.4680
Fax: 46.8.7506647

Switzerland
LSI Logic Sulzer AG
■ Erlenstrasse 36
CH-2555 Bruegg/Biel
Tel: 41.32.536363
Fax: 41.32.536367

Taiwan
LSI Logic Corporation
■ 3F 92 Tun-Hua S. Rd
Sec 2 Taipei ROC
Tel: 886.2.755.3433
Fax: 886.2.755.5176

United Kingdom
LSI Logic Limited
■ Grenville Place, The Ring
Bracknell Berkshire
England RG12 1BP
Tel: 44.344.426544
Telex: 848679
Fax: 44.344.481039

■ Lomond House
Beveridge Square
Livingstone Scotland
EH 54 6QF
Tel: 44.506.416767
Fax: 44.506.414836

Manhattan Skyline Limited
Manhattan House, Unit 1
The Switchback, Gardner Rd
Maidenhead Berkshire
England SL6 7RJ
Tel: 44.628.75851
Fax: 44.628.782812

■ Sales Offices with Design Resource Centers

Printed on Recycled Paper

Printed in USA
062.7111.5536.101.2K.TP.J/SCL

LSI LOGIC

LSI Logic Corporation
U.S. Headquarters
Milpitas CA
408.433.8000
Fax: 408.434.6457

LSI Logic GmbH
European Headquarters
Munich Germany
49.89.99313100
Fax: 49.89.936806

LSI Logic K.K.
Headquarters
Tokyo Japan
81.33.589.2711
Fax: 81.33.589.2740

LSI Logic Corporation
of Canada Inc.
Headquarters
Calgary
403.262.9292
Fax: 403.262.9494

Order No. C14002.A

Please see inside back pages for information on the Sales Office or Distributor nearest you.