PACEWRAP™ PR3100A WRITE/READ AND PARITY BUFFER

PRELIMINARY

FEATURES:

- 8-word write buffer with byte-gathering capability
- Read buffer with programmable depth of up to 32 words
- Intelligent read buffer controller significantly minimizes the cache miss penalty for block refills from simple non-interleaved memory systems
- Full PR3000A block refill and streaming support
- Supports data writes while streaming
- Bus snooper to assist in maintaining cache coherency in multiprocessor systems
- Parity generation on reads from main memory to allow the use of memory systems without parity
- Static column DRAM support
- Available in 160-pin PQFP package

DESCRIPTION

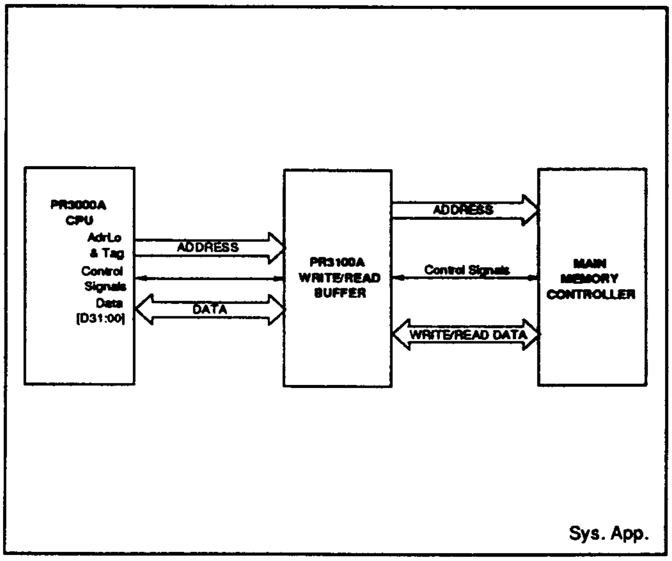
The PaceMips PR3100A is a single-chip write/read buffer designed to support the R2000A and R3000A RISC microprocessors. The PR3100A provides a high performance memory system interface for the R2000A/R3000A computer systems while minimizing parts count and power dissipation.

The use of a write buffer in a R2000A/R3000A computer system permits the microprocessor to perform writes to the memory system at cache speeds, thereby maintaining main memory coherency without costly processor stall cycles. The PR3100A buffers the full 32-bit address, 32 bits of data, 4 data parity bits and the 3-bit access type bus on each memory write operation, unlike other implementations which require multiple chips to support the full address and data bus widths. The write buffer depth of 8 address/data pairs provides enough storage space to buffer most store operations without stalling the processor. Only a large group of sequential stores will fill the buffer and force a stall.

The PR3100A also contains a read buffer with a programmable depth of up to 32 words. This feature is fully compatible with the R3000A block refill and streaming modes, improving both instruction and data cache miss performance.

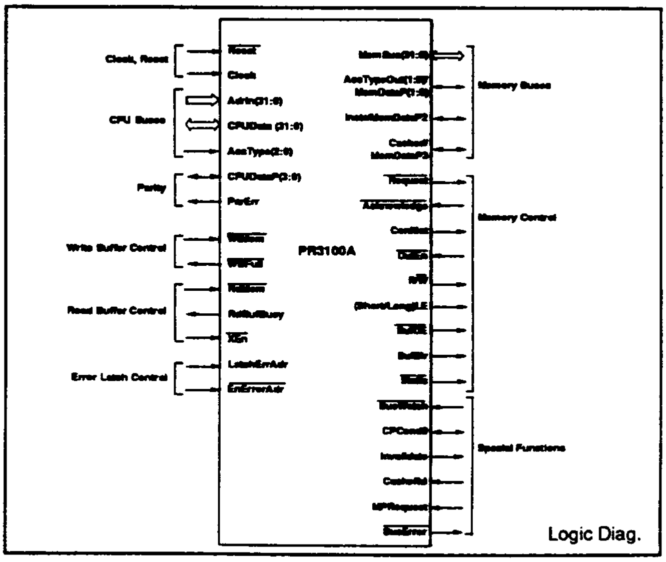
In addition to the primary write and read buffer features, the PR3100A provides several functions to improve system performance and reduce part count. For one, it is optimized to reduce stall cycles. An on-chip row address comparator is included for static column DRAM support. This allows fast access of DRAM memory systems which significantly improves performance. A bus snoop circuit is also included to assist in maintaining cache coherency in multiprocessor systems.

SYSTEM APPLICATION



Sys. App.

LOGIC DIAGRAM



Logic Diag.

PERFORMANCE
SEMICONDUCTOR CORPORATION

Means Quality, Service and Speed

4/1/92

1.0 WRAP OPERATION

WRITE BUFFER OPERATION

The PR3100A write buffer is an 8-deep FIFO buffering all writes to main memory to allow the R3000A to operate at cache speed without stalling during writes. For each CPU write, the write buffer captures the data word including parity, and the address including the AccType bits and holds them until they can be written out to the main memory.

The R3000A CPU indicates to the PR3100A that a write is being performed by asserting MemWr, which is connected to WtMem on the PR3100A. The write buffer then latches in the address/data pair off of the cache bus and begins the write cycle out to memory using a multiplexed address/data bus and a set of control signals including Request, Acknowledge, LE, OutEn and R/W. When the current CPU write fills the 8 word FIFO, PR3100A asserts the WBFull signal which is used by the CPU to cause a stall on the next write attempt. Once there is an empty location in the buffer, WBFull will be deasserted and CPU writes may continue.

When the CPU performs a read (during Burst reads) of an address currently contained in the write buffer, the PR3100A will assert the Conflict signal. The PR3100A will also assert RdBufBusy to stall the CPU until the write buffer can be emptied to the main memory, after which the read will continue.

Long LE / Short LE

The PR3100A allows programmable duration of LE signal. The selection of Long LE or Short LE can be set at reset time. For systems that use an external buffer for higher drive between the multiplexed bus of PR3100A and the multiplexed bus of the rest of the system, the designer can eliminate the timing problems by choosing the Long LE signal that lasts for one complete clock cycle. The address then will be driven for two full clock cycles. For details refer to Figures 5.12 and 5.13. In systems that would exploit the full speed of PR3100A, the Short LE signal that is one-half clock cycle long can be chosen. The address will be driven for one full clock cycle when Short LE is selected. Short LE is shown in most of the timing diagrams.

Byte Gathering

The write buffer may perform byte gathering, if enabled, to reduce the number of buffer locations required to buffer partial word stores and to reduce the number of main memory write cycles required. When multiple partial word stores occur to the same word address, the PR3100A merges them into one buffer location. If the GatherAll mode is enabled at initialization, the PR3100A will perform the gathering function on all buffer locations. If GatherAll is not enabled, only sequential writes to the same word address will be gathered. The only buffer location not accessible to the gathering function is the location currently being transferred to the memory.

If byte gathering results in an undefined combination, such as byte 0 and byte 3, the method that the PR3100A uses to empty the buffer location will vary with the ByteEnc mode configuration. If the write buffer is programmed to operate in the encoded mode, the AccTypeOut bits, AdrOut(1:0) and the endian programming will encode the byte reference, as they do from the CPU. Since only bytes, half-words, tri-bytes and words may be encoded in this manner, undefined byte combinations will require multiple memory accesses to empty the buffer. If the encoded mode is disabled, AccTypeOut(1:0) and AdrOut(1:0) become individual byte enables, allowing any combination of bytes to be written to memory in one cycle. A high on a particular byte enable signal enables the corresponding byte. Write buffer locations are emptied in the order received except for gathered data.

Error Address Latch

The PR3100A incorporates an internal latch that is capable of storing the current write or read addresses and subsequently writing it out to the memory bus. This feature can be used by error handling routines to read an address back from the write buffer for error analysis and recovery. When LatchErrAdr signal is asserted, the address currently available on the memory bus or the address of the current cycle will be latched. This address may be later read by asserting EnErrorAdr, which places the latched address onto the memory bus as data. If LatchErrAdr is asserted during reads, then BusError is asserted when data is transferred to the CPU. Refer to Figure 5.15 for timing details.

READ BUFFER OPERATION

The read buffer is a FIFO with a programmable depth of up to 32 words, designed to efficiently support the block refill and streaming modes of the R3000A CPU. When block refills are not being performed the read buffer becomes a single word register. When block transfers are requested, the buffer depth can be configured to 4, 8, 16 or 32 words. Though there is a single buffer serving both instructions and data, separate buffer depths may be configured by the IBlkSize and DBlkSize bits during initialization. The buffer depths for instructions and data accesses must be the same as the programmed block sizes in the CPU.

During cached reads, the CpCond0 signal determines whether a block read or a single read is performed. CpCond0 may be an output from the PR3100A or an input to the PR3100A as determined by the configuration of CpCond0In at reset. If it is configured as an output, the PR3100A uses the configuration of the Block bit at reset to determine whether block refills are enabled. If block transfers are enabled, the PR3100A will generate CpCond0 to the PR3000A with the correct timing to perform a block transfer of the number of words configured by IBlkSize or DBlkSize. If not, a single word transfer will be performed.

When configured as an output, during run cycles, CPCond(0) can be used to determine if the write buffer is empty. The

CPU's CpCond(0) is driven during run cycles by an internal signal that is asserted HIGH whenever there are pending writes in the write buffer; that is, CpCond(0) is driven HIGH whenever the write buffer is not empty. Software could poll the CpCond(0) input using coprocessor 0 branch condition instructions to check if the write buffer is empty. CpCond(0) will remain HIGH until Acknowledge is asserted for the last word in the write buffer. CpCond(0) is driven HIGH two cycles after a store to an empty buffer.

If CpCond0 is configured as an input, it will be sampled by both the PR3100A and the PR3000A, and used to determine whether a block transfer is to be performed. In this case, the Block bit configuration programmed at reset is not used. All uncached reads, as indicated by the AccType2 signal, are always single word transfers.

When the CPU initiates a block read, the PR3100A will start a memory block read and begin filling the buffer. The RdBufBusy signal, which the PR3100A leaves asserted at all times except when read data is actually being transferred to the CPU, will force the CPU into a read busy stall while the buffer is being filled. The buffer will be filled with the number of words required to transfer the entire block to the CPU without causing any single word retry stalls. The number of words required in the buffer is calculated by the PR3100A based on the memory access time and block size programmed at reset. The buffer is filled to the point that the CPU will be able to access the full block size without stalling while the remainder of the block is still being brought in from the memory. The last word enters the buffer just prior to when it is needed by the CPU.

When block transfer mode is activated, the PR3100A may access the memory in either a normal read mode or a burst read mode, depending on the configuration of the BurstRead bit at reset. When the burst mode is enabled, the PR3100A generates only the first address of the block to the memory and then all of the data words in the block are read sequentially. If the burst mode is disabled, the PR3100A will generate an address for each word of the transfer.

Operation During Read Conflicts

All conflicts force the entire write buffer to empty prior to completing the read that caused the conflict. If the CPU is performing a block read, data will be loaded into the read buffer from the memory until the conflict occurs.The write buffer will then be flushed. If the first conflict occurs after data transfer from the read buffer to the CPU has begun, the CPU is forced into a read stall and data transfer to the CPU continues after the conflict is resolved. If the conflict is detected before data transfer to the CPU has started, the CPU remains stalled until the conflict is resolved. Conflicts are resolved transparentlyto the external system.

If Burst mode is enabled, conflicts are handled differently since the memory block read cannot be interrupted in the Burst mode. In this situation, if a conflict occurs during the block read, the entire block is still read from memory, the write buffer is flushed and the entire block is then re-read from memory.This operation is transparent to the memory system. For conflict signal timing, see Figure 5.16.

Byte-Assembly (Dynamic Bus Sizing)

To support dynamic bus sizing, PR3100A uses the byte-assembly operation to directly perform 32-bit instruction or data reads from 8-bit I/0 devices without any intervening glue logic. To save board space, this feature permits the use of a single byte-wide boot PROM instead of four devices. The PR3100A does four byte reads from the PROM and assembles the bytes internally to form a 32-bit word.

The ByteAssemble input must be asserted if a memory read cycle is to be byte-assembled. Both cached and uncached word reads may be byte-assembled. Partial word (half-word and tribyte) reads can also be byte assembled. ByteAssemble is sampled only on the first rising edge of the clock after the PR3100A terminates the address phase of a word read as shown in Figure 5.14.

ByteAssemble is not sampled for partial word reads. If ByteAssemble is asserted, then two cycles later the first byte address (based on the Endian) is driven on the MemBus (31:0) with another LE. The memory controller must not respond to the first LE and an Acknowledge should be asserted only for the second LE as shown in Figure 5.14. The data sample on MemBus (7:0) on the falling clock edge of Acknowledge. Consecutive byte addresses are driven on the bus until the four bytes of the word are read and internally assembled into a word as shown in Figure 1.1.

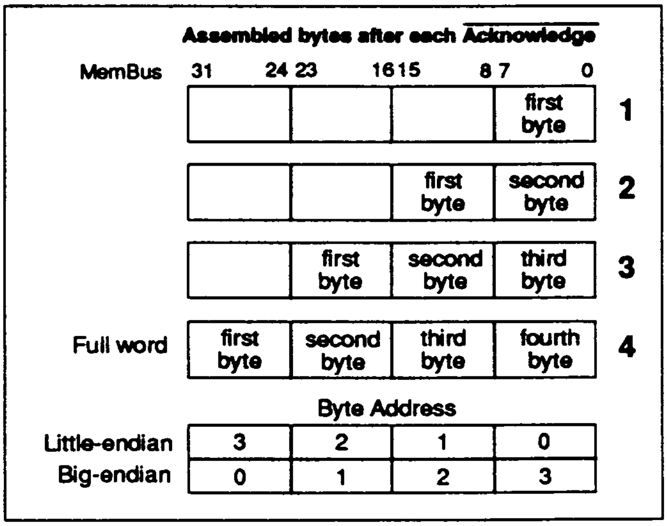


Figure 1.1 Byte-Assembly Operation

Byte assembly may be used during block refills both in Block and Burst mode. However, ByteAssemble is sampled only once on the first rising clock edge after the address phase is terminated. Therefore, if ByteAssemble is asserted then the whole block is read in byte-assembly mode. As shown in Figure 5.14 after the first word is byte-assembled, the first byte address of the second word will be driven on the bus until the whole block is read in Byte-assembly mode. This is true even for Burst mode, that is, the byte address of every word in the block will be driven onto the bus by PR3100A identical to Block mode (Burst mode is disabled during byte-assembly).

If ByteAssemble is asserted and the PR3100A is configured to operate with individual byte enables (ByteEnc is low), then this mode is disabled and the byte encoding mode is enforced for the access so that the latched Membus (1:0) can be directly connected to an 8-bit I/O device. During byte-assembly, the PR3100A defaults to the byte encoding mode and the MemBus (3:2) indicates byte access (00 = byte access) during the address phase. The byte address is driven on MemBus (1:0), which increments 0,1,2,3 in Big-Endian mode or 3,2,1,0 in Little-Endian mode, so that the first byte read is assembled on bits (31:24) of the final word as shown in Figure 1.1.

PARITY SUPPORT

The PR3100A supports parity in two modes, based on the programming of the GenParity bit at reset. If GenParity is asserted, the PR3100A will generate the byte parity required for the R3000A caches as data is read in from the main memory. This allows parity in the main memory to be optional, which can in turn reduce system part count. If GenParity is disabled, the parity bits will be passed directly through the PR3100A from the memory to the caches and vice versa. Thus, if the error protection afforded by parity is desired for main memory, it may be implemented.

The PR3100A also checks parity during writes from the CPU. If a parity error is detected in the data word as it is received from the CPU, the PR3100A will assert the ParErr signal which may be used by the memory controller to handle or flag the error. The data and parity, with the error, will still be passed to the main memory.

STATIC COLUMN DRAM SUPPORT

The PR3100A provides support for DRAM static column operation, improving memory access performance particularly during write sequences.The DRAM configuration is programmed by the DRAMSize(1:0) bits at reset and is used by an on-chip row and bank address comparator. On every memory access, the row and bank addresses are compared with those of the previous access. If the addresses are equal, the Static signal will be asserted for use by the DRAM controller in generating the RAS and CAS signals to the memories. If PR3100A output enable goes high between memory accesses, the previous address is disregarded and Static is not asserted. The Static signal becomes high-impedance when the PR3100A does not have control of the memory bus. The following diagram illustrates the DRAM configurations supported by the PR3100A.

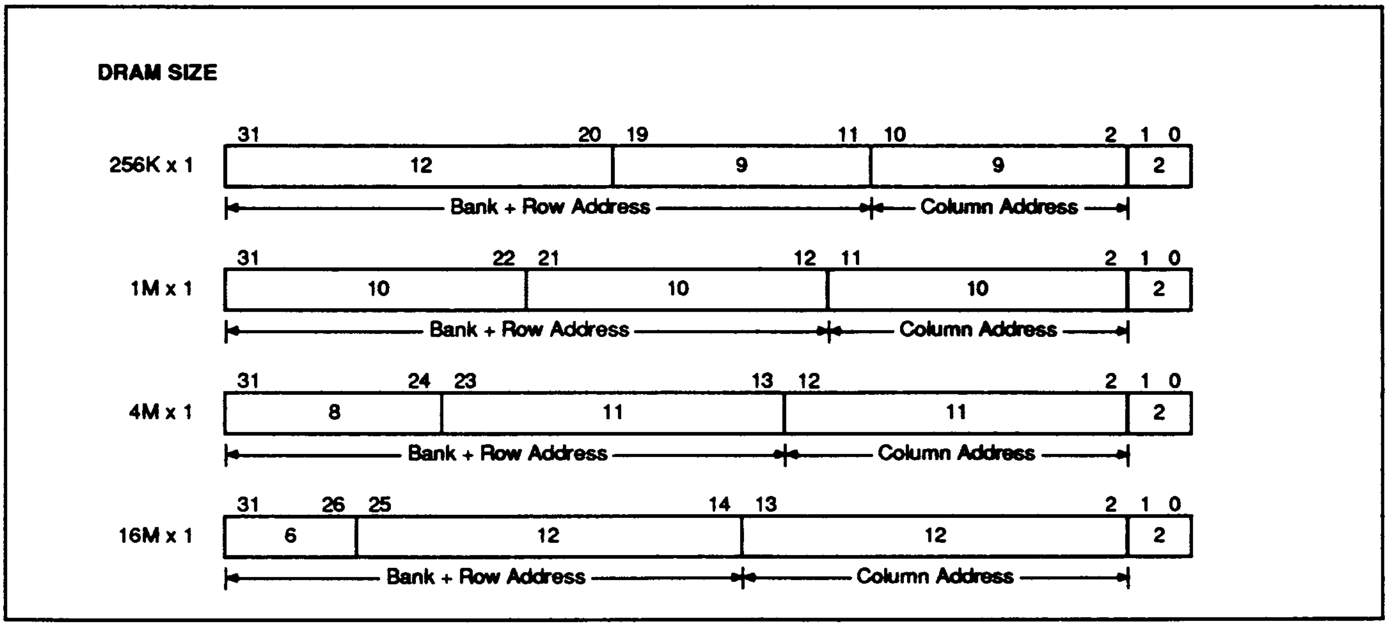


Figure 1.2 Address Configuration

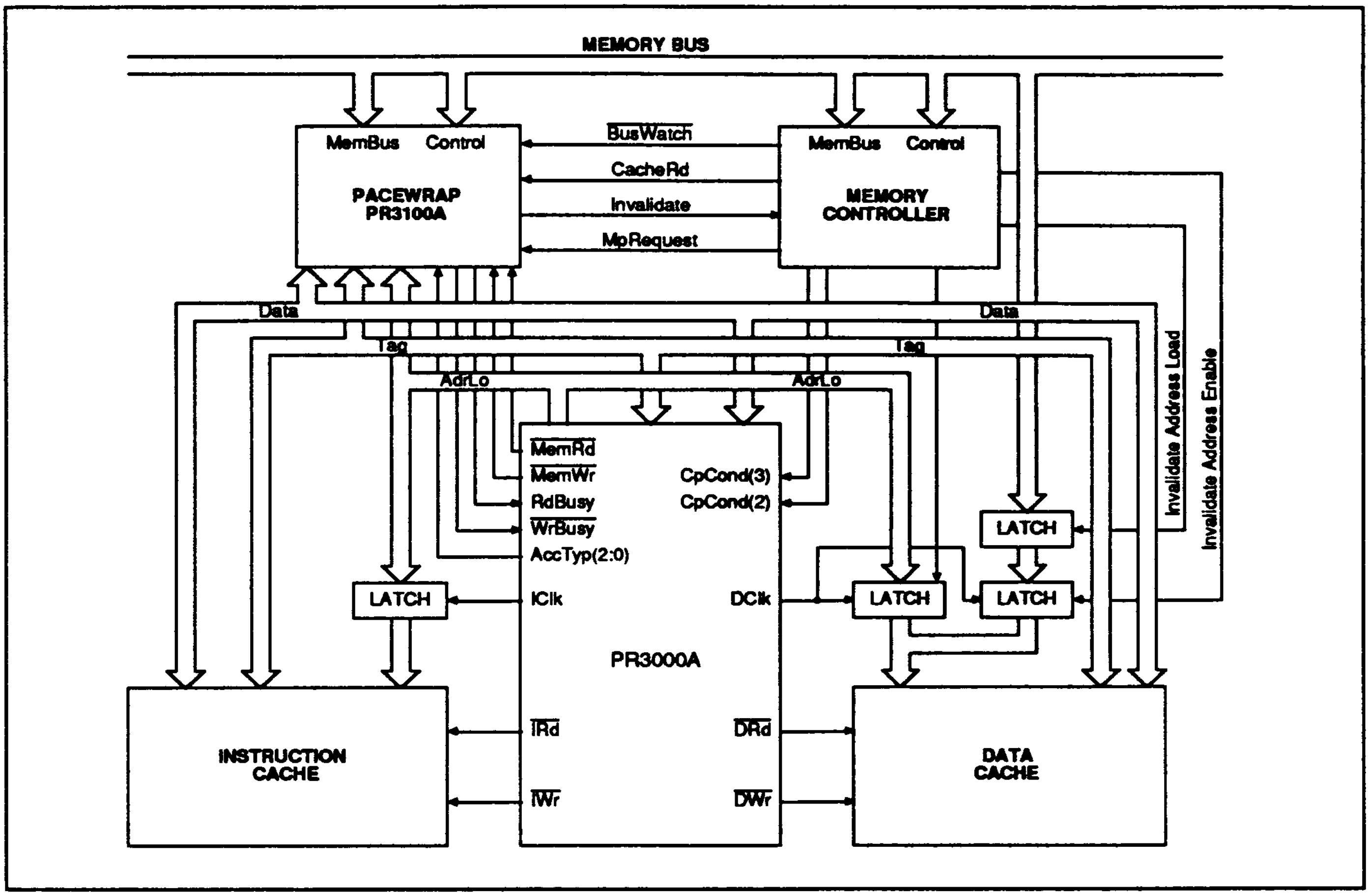


Figure 1.3 The PACEWRAP PR3100A in a Multiprocessor System

MULTIPROCESSOR SUPPORT

The PR3100A provides the capability to perform the PR3000A cache invalidation sequence with minimum external logic. When the MP invalidate sequence is implemented in a system, cache coherency can be maintained for multiple CPUs and DMA devices.

If the memory system controller detects a write or read attempt into a watched address (ie. cacheable address), it should issue the MpRequest (CpCond(3)) signal to the processor and the PR3100A. This will force the R3000A into an MP stall cycle, where it may begin the invalidate sequence. The memory controller should also assert the $\overline{\text{BusWatch}}$ signal to the PR3100A to cause it to latch the address off of the memory bus. The controller should also cause the AdrLo portion of the memory address to be latched into an external register for use later in the MP sequence. After the first MP stall has occurred, the CPU will perform a cache read which should use the AdrLo latched previously by the memory controller in the external (invalidate) latch.

The PR3100A will then latch the tag corresponding with the cache read using the CacheRd signal generated by the memory controller. If the latched cache tag corresponds to the address latched off of the memory bus previously, the PR3100A will generate the Invalidate signal which should be used to generate CpCond(2) to be used to invalidate the cache line.

This method of maintaining cache coherency is effective, but can significantly compromise performance if misused. It is best to minimize the number of watched addresses by minimizing the amount of shared, cached memory. When this is done, the time consuming MP invalidate sequence will not be forced to occur so much that CPU performance is significantly affected. The Figure 1.3 illustrates the use of the PR3100A in a multiprocessor system.

BUS OPERATION

The PR3100A features an easy to use multiplexed address and data bus with a simple control protocol for ease of design. The bus is designed to interface directly with the system memory controller with a simple handshake.

Basic Single Word Read Cycle

The processor performs a read operation by asserting the RdMem signal. The PR3100A samples this signal on the falling edge to the clock. The CPU will be stalled by an active level on the ReadBufBusy signal, which should be connected to the RdBusy input of the CPU. The RdBusy signal will normally be active, unless the PR3100A is currently sending data to the CPU. On the rising edge of the clock after sampling the RdMem signal active, the PR3100A will assert its bus request (Request) to obtain control of the bus. The PR3100A will remain in this state until the OutEn signal, sampled active by a falling clock edge, indicates that the bus is granted to the PR3100A. The address will now become active on the MemBus and LE will pulse high for one half clock to latch the address. The LE signal is designed to be connected directly to the latch enable input of a '373 type latch to facilitate simple demultiplexing of the bus. If the bus is already granted to the PR3100A (OutEn low), as the Request signal becomes active , the address will become valid 1 1/2 clocks after the falling clock edge which sampled RdMem low. This is the minimum bus acquisition cycle. The signal BufOE is asserted whenever the PR3100A is the bus master. BufOE is high when OutEn is high or no read/write are pending. The signal BufDir indicates the direction of data transfer. BufDir is high when address/data is driven by PR3100A. BufDir is low when data is read from memory.

The address is valid for one clock (Short LE). The MemBus will then become an input for sampling the read data from the memory. The PR3100A will remain in this state indefinitely, allowing the memory controller to the add wait states, until sampling the Acknowledge signal active (low) on the falling edge of the clock. The PR3100A will then, on the same falling clock, place the data from the MemBus into the read buffer to begin transfer to the CPU. The RdBufBusy signal will then become inactive (low) to cause the CPU to complete the stall and enter the fixup cycle. When the CPU activates XEn, the PR3100A will transfer the data from the buffer to the CPU, ending the cycle.

If there are no writes pending in the write buffer, the PR3100A will release the system bus by removing its request. This will occur on the same rising clock edge that ends the address phase of the bus cycle. The OutEn signal may be deasserted at the same time that Acknowledge is generated or any time after. OutEn is sampled on the falling edge of the clock and an inactive state will cause the bus to become high impedance on the next rising clock edge. Refer to Figure 5.1 for details.

Basic Single Word Write Cycle

The single word write cycle begins very similar to the single word read cycle. The PR3100A will sample the WtMem signal active on the falling edge of the clock. On the next rising clock edge, the PR3100A will request the system bus by activating its Request output. The PR3100A will wait in this mode until sampling the OutEn signal active on a falling clock edge. On the next rising clock edge, the address will be generated for one clock and the LE signal will become active for half a clock to sample the address. If there are no other words pending in the write buffers, and if the CPU is not requesting a read, the Request signal will be deasserted on the same rising clock that the data will be generated. The data will remain valid until the Acknowledge signal is sampled low on a falling clock edge. OutEn may be deasserted with the Acknowledge signal or any time thereafter. When OutEn is sampled inactive on the falling edge of the clock, the bus becomes high impedance on the next rising edge of the clock. Back to back writes require a minimum of 2 clocks per word. Refer to Figure 5.9 for details on multiple operations.

If a processor write causes the buffer to become full, the WBFull signal will be asserted. This signal should be connected to the WrBusy signal on the PR3000A processor to cause the processor to stall on the next write attempt. As soon as an Acknowledge is received from the memory system indicating that a data word has been written, the PR3100A will deassert the WBFull signal indicating that there is now space in the buffer for another word. Figure 5.5 illustrates WBFull timing.

Block Transfers

Block transfers are possible whenever the CPU performs cached read operations, either instruction or data fetches. If CpCond0 is programmed as an input at reset time, block transfers are initiated by the memory system controller by generating this signal. In this mode, the PR3100A will initially assume that the transfer is a block and place the block address on the bus. If the memory controller requires a single word transfer, it may change it from a block to a word by generating the CpCond0 input in the proper state. This must occur before the first acknowledge is received by the PR3100A. When this happens, the PR3100A will place the word address on the bus and issue another LE pulse to latch it. The memory system must ignore the first address and respond only to the second. In this mode of operation, Figure 5.6 depicts CpCond0 timing. Please note that the CpCond0 timing for the PR3100A is not identical to the timing required for the R3000A CPU. However, if the CpCond0 signal is made valid at the proper time for the PR3100A, it may again be made valid (or simply held valid) during the time when sampled by the CPU. Thus, the same signal may be connected to both the CPU and the PR3100A as long as the timing requirements for both are met.

Block transfers may also be performed with no intervention from the memory controller.

If the CpCond0 signal is configured as an output, and block transfers are enabled at reset by the block bit, then all cached reads will automatically become block transfers and the PR3100A will issue CpCond0 to the CPU at the proper time to cause the block transfer to occur. During run cycles, CPCond(0) can be used to determine if the write buffer is empty. The CPU's CpCond(0) is driven during run cycles by an internal signal that is asserted HIGH whenever there are pending writes in the write buffer; that is, CpCond(0) is driven HIGH whenever the write buffer is not empty. Software could poll the CpCond(0) input using coprocessor 0 branch condition instructions to check if the write buffer is empty. CpCond(0) will remain HIGH until Acknowledge is asserted for the last word in the write buffer. CpCond(0) is driven HIGH two cycles after a store to an empty buffer (Figure 5.7) See Figure 5.17 for detailed Block Read timing.

Burst Mode

When configured at reset to enable burst mode, the PR3100A will perform all block transfers as bursts. A burst operation is similar to the normal single word read cycle with the exception that all words subsequent to the first will occur without an address phase. This allows the transfer of up to 1 word per clock, or 160Mbytes/second at 40MHz. In the burst mode, the memory system controller must count the number of words and send the proper number of acknowledges to the PR3100A. When a burst is complete, if there are no writes pending in the write buffer, the PR3100A will deassert its Request signal on the rising clock edge beginning the last data cycle. Figure 5.4 illustrates the burst mode timing.

Intermingled Writes and Reads

In normal operation, both writes and reads are occurring frequently. The PR3100A handles both with maximum parallelism. Reads always take precedence over writes since a read inherently stalls the CPU. Thus, if there are words pending in the write buffer and a read occurs, the read will be performed as soon as the current write is complete. Exceptions to this rule occur when a write buffer flush must be performed due to read conflicts, and when an illegal byte combination has been gathered in the write buffer and requires multiple writes cycles due to the encoded mode being configured. In both instances, the required writes are performed prior to servicing any read requests. When writes and reads immediately follow each other, the PR3100A Request line remains active. When the OutEn signal is detected active, the R/W signal will become valid with the new address on the rising edge clock. Refer to Figure 5.9 (Part 1 and Part 2) for details.

Writes While Streaming

The PR3100A will support writes while the CPU is streaming provided the write buffer does not fill during this time. The diagram in Figure 5.8 depicts the timing associated with this mode of operation.

Interrupting a Bus Cycle

A read or write bus cycle may be interrupted any time before the Acknowledge signal has been received if required by the memory system arbiter. This can be useful for resolution of deadlock conditions or other time-sensitive arbitration issues.

When the bus must be taken away from the PR3100A, the system controller must simply deassert the OutEn signal. The PR3100A will respond by placing all of its bus signals in a high-impedance condition on the next rising clock. This allows the memory system controller to grant the bus to another user. When the arbitration conflict is resolved, the controller may return bus mastership to the PR3100A by asserting its OutEn signal again. The PR3100A will respond by returning its outputs to the same state (except for Static signal which will be deasserted) that they were in prior to the grant being taken away. This may occur even in the middle of a block or burst transfer, though the memory controller must remember where in the block the interruption took place since the PR3100A does not retransmit the address. It should also be noted that if the PR3100A has begun transferring words to the CPU in either block refill or streaming modes, the CPU will be forced to stall when the block is interrupted.

Bus Errors

Bus errors are handled by the PR3100A during read operations. If the memory system controller detects a bus error, it signals the PR3100A with the LachtErrAdr signal. The PR3100A responds by latching the current address in the error address latch and asserting the BusError signal to the CPU. If the PR3000A CPU is currently in a read busy stall, the bus error will terminate the stall and the PR3100A will supply the appropriate timing of the read busy signal and the bus error signal to cause the word to be invalidated. If the error was received in the middle of a block transfer, the word which caused the error and all subsequent words will be invalidated. In this situation, the memory system must continue to supply acknowledges to the PR3100A to complete the block transfer. See Figure 5.15 for timing details.

2.0 SIGNAL DESCRIPTIONS

Clock	I	Supplies PR3100A operating clock. Should be connected to SysOut of PR3000A.
Reset	I	Resets PR3100A and configures mode of operation according to the states of the signals LatchErrAdr, EnErrorAdr, CacheRd, BusWatch and MpRequest during last four clocks before the rising edge of Reset. Connected to Reset on the CPU.
AdrIn(31:0)	I	Memory Address Input from the CPU. A combination of AdrLo[0:15] and Tag[16:31] buses from the CPU.
CpuData(31:0)	I/O	CPU Data Bus. Input to the PR3100A during CPU writes to the write buffer. Output from the PR3100A when the CPU reads the read buffer, when XEn is low.
AccType(2:0)	I	Access Type Bus from the CPU indicating the type of bus transfer occurring (byte, halfword, tri-byte, word, cached, uncached, instruction, data). Used by the PR3100A for byte gathering and for defining main memory access type.
CpuDataP(3:0)	I/O	Byte parity bits over the CpuData bus. During CPU write operations, parity is checked by the PR3100A and ParErr is asserted if an error is detected. The parity bits are passed to the memory along with the data. During reads, the parity bits are output to the CPU along with the data to be written into the cache when XEn is low. The parity bits may be generated by the PR3100A during reads from memory which does not support parity, or passed directly from the memory if parity is implemented.
ParErr	O	Parity Error. Generated by the PR3100A when a parity error is detected during a write from the CPU to the write buffer.
WtMem	I	Write Memory. Connected to the CPU MemWr signal which is asserted whenever the CPU performs a write (store) operation.
WBFull	O	Write Buffer Full. Connected to the CPU WrBusy signal to indicate that the 8-word buffer is currently full. This causes the CPU to stall on the next attempted write operation. WBFull will be deasserted as soon as the buffer has an available space.
RdBufBusy	O	Read Buffer Busy. Connected to the CPU RdBusy signal to indicate that CPU reads will be forced to stall at this time. The CPU will enter a read busy stall if a read is attempted while RdBufBusy is active. RdBufBusy will be deasserted as soon as the buffer can support a transfer to the CPU.
RdMem	I	Read Memory. Connected to CPU MemRd signal which indicates when the CPU is performing a read of the main memory. Used by PR3100A to begin a read cycle.
XEn	I	Read Enable. Connected to the CPU XEn signal to enable data from the read buffer onto the CPU data bus.
LatchErrAdr	I	Latch Error Address. Will cause the address associated with the current MemBus cycle to be latched into the internal address error register.
EnErrorAdr	I	Enable Error Address. Enables the contents of the internal address error register onto the MemBus. Overrides OutEn.
MemBus(31:0)	I/O	Memory Bus. A multiplexed address & data bus used to transfer databetween memory system and PR3100A. Tri-state when OutEn and EnErrorAdr are high.
AccTypeOut(1:0) / MemDataP(1:0)	I/O	Multiplexed Bus. During address time (LE high) on the MemBus, AccTypeOut(1:0) define the type of data transaction to follow (byte, half-word, tri-byte, word) if the encoded mode is configured. If the non-encoded mode is configured at reset, AccTypeOut(1:0) become individual byte enables during address transactions. During data time (LE low), the signals become the parity bits for the two least significant bytes of the data word. These bits will be outputs during writes and inputs during reads. Tri-state when OutEn is high.
MpRequest	I	Multiprocessor request. When high, indicates an MP stall is beginning. Used by read buffer to indicate where an MP stall has started so that it may resume a block transfer atcorrect point after the stall iftransfer was in process when stall was entered.
BusError	O	Bus error indication connected to BusError input on the CPU.

3.0 SIGNAL DESCRIPTIONS (Continued)

Instr / MemDataP2	I/O	Multiplexed signal; for diagnostic purposes. During address time (LEhigh), Instr is an active-high output that indicates that the current MemBus transaction is a cached instruction read. When low, Instr indicates a cached data read. Instr is also low during writes. During data time (LE low), MemDataP2 is the parity bit corresponding to the second most significant byte on the MemBus (MemBus(23:16). MemDataP2 is an input during read cycles & an output during write cycles. Tri-state when $\overline{\text{OutEn}}$ is high.
Cached / MemDataP3	I/O	Multiplexed signal; for diagnostic purposes. During address time (LE high), a high on Cached indicates current MemBus read is a cached access. When low, read is a non-cached access. Cached is low during writes. During data time (LE low), MemDataP3 is the parity bit corresponding to most significant MemBus byte (MemBus(31:24). Tri-state when $\overline{\text{OutEn}}$ is high.
Short LE	O	Short address latch enable. Lasts 1/2 clock cycle.When high, indicates an address is available on MemBus for one clock cycle. Used by external latch to latch address off of MemBus.
Long LE	O	Long address latch enable. Lasts for one full clock cycle. When high, indicates that an address is currently available on the MemBus for two clock cycles. It should be used by an external latch to latch the address off of the MemBus.
$\overline{\text{Request}}$	O	Memory bus request. When low, indicates PR3100A requires access to MemBus either to transfer data from write buffer to memory or from memory to read buffer.
$\overline{\text{Acknowledge}}$	I	Memory bus acknowledge. When low, indicates completion of a memory cycle. If cycle is a read, indicates to the PR3100A that data is valid on the MemBus. If cycle is a write, indicates that the memory system has latched the data off of the MemBus.
$\overline{\text{OutEn}}$	I	Memory bus output enable. When low, indicates to the PR3100A that MemBus access has been granted (in response to $\overline{\text{Request}}$) and that it may begin driving the MemBus and its controls. $\overline{\text{OutEn}}$ should remain low during the entire memory transaction, including burst transfers, except during bus interruption (see text).
R/$\overline{\text{W}}$	O	Memory bus read / write control. When high, indicates to the memory that the current MemBus transaction is a read. When low, indicates that the current MemBus transaction is a write. Tri-state when $\overline{\text{OutEn}}$ is high.
Conflict	O	Signal asserted high when address of data pending in the Write Buffer matches an address being read by PR31000A from memory. This may occur only during Burst Reads.
$\overline{\text{Static}}$	O	Static column address. When low, indicates that successive memory accesses have the same row address. May be used by an external DRAM controller to support fast static column DRAMs. Tri-state when $\overline{\text{OutEn}}$ is high.
$\overline{\text{BusWatch}}$	I	Invalidate address latch control. Used by the PR3100A to latch the current address off of the MemBus to be used later in an invalidate compare.
CacheRd	I	Cache read. Used by the PR3100A to latch the information on the AdrIn and CPU Data buses for use in the invalidate comparison.
Invalidate	O	Invalidate control. When high, indicates tags latched by $\overline{\text{BusWatch}}$ and CacheRd signals were equal. Used to generate CpCond2 toCPU to invalidate the cache line.
CpCond0	I/O	When configured as an input at reset, a high indicates to the PR3100A that a block transfer is requested. When configured as an output, CpCond0 indicates block mode read or single read during read stalls and write buffer status during run cycles.
$\overline{\text{ByteAssemble}}$	I	In this mode PR3100A assembles each CPU word (by SHIFT) from 4 bytes latched from MemBus (7:0) in four successive read cycles, see Figure 5.14.
$\overline{\text{BufOE}}$	O	When LOW, indicates MemBus is active. Used to enable external buffer between MemBus of PR3100A and the rest of the memory system.
BufDir	O	Indicates direction of data on MemBus. May be connected to an external buffer. When HIGH, direction is from the PR3100A to memory; when LOW from memory to PR3100A.

3.0 PR3100A INITIALIZATION AND CONFIGURATION

The PR3100A mode of operation is configured during the last four clocks of the reset sequence, similar to the R3000/R3000A processor. The four clock cycles preceding the rising edge of the $\overline{\text{Reset}}$ signal are called the W cycle, X cycle, Y cycle and Z cycle, respectively. During these four cycles, the states of the five Configuration pins: LatchErrAdr, $\overline{\text{EnErrorAdr}}$, $\overline{\text{BusWatch}}$, CacheRd and MpRequest, are used to define the PR3100A operation modes. The table below defines the configuration programming (see Figure 5.8a).

Input Pin	W Cycle	X Cycle	Y Cycle	Z Cycle
LatchErrAdr	DBlkSize0	DBlkSize1	MpNormal	$\overline{\text{BigEndian}}$
$\overline{\text{EnErrorAdr}}$	IBlkSize0	IBlkSize1	BurstRead	Long LE
$\overline{\text{BusWatch}}$	DRAMSize0	DRAMSize1	ByteEnc	GatherAll
CacheRd	$\overline{\text{Gather}}$	$\overline{\text{Reserved}}$	Block	CpCond0In
MpRequest	MemAcc0	MemAcc1	MemAcc2	GenParity

DBlkSize (1:0)

Defines the block size used by the PR3100A during data cache refills. Identical in function to the R3000/R3000A block size configured at reset. Should be programmed identically to the R3000A DBlkSize.

DBlkSize(1:0)	Block (words)
00	32
01	16
10	8
11	4

IBlkSize (1:0)

Defines the block size used by the PR3100A during instruction cache refills. Identical in function to the R3000/R3000A block size configured at reset. Should be programmed identically to the R3000A IBlkSize.

IBlkSize(1:0)	Block (words)
00	32
01	16
10	8
11	4

$\overline{\text{BigEndian}}$

Defines the byte ordering. If high, the Little Endian byte ordering is selected, otherwise the Big Endian byte ordering is selected. This is identical to the Little and Big Endian selection of the R3000/R3000A. Should be programmed identically to the R3000A $\overline{\text{BigEndian}}$.

$\overline{\text{Gather}}$

Defines whether or not byte gathering is enabled. When low, sequential writes to the same word address will result in data merger. GatherAll is ignored if $\overline{\text{Gather}}$ is disabled.

Long LE

Selects Long LE or Short LE. When HIGH selects Long LE. When LOW selects Short LE.

DRAMSize(1:0)

Defines the size of the DRAMs used in the main memory system. Used by the PR3100A row address comparitor to support the static column mode. Refer to Fig. 1.2.

DRAMSize(1:0)	DRAM Size	Row Address
00	16M x 1	18
01	4M x 1	19
10	1M x 1	20
11	256K x 1	21

ByteEnc

When high, AccTypeOut(1:0) and AdrOut(1:0) encode the byte selection on the MemBus. When ByteEnc is low, the AccTypeOut and AdrOut(1:0) (MemBus(1:0)) signals become individual byte enables. Byte Enables are active high.

When Byte Enc is low:

Individual Byte Enables	MemBus Bits	MemDataP
AdrOut0	(7:0)	0
AdrOut1	(15:8)	1
AccTypeOut0	(23:16)	2
AccTypeOut1	(31:24)	3

When Byte Enc is high:

Table 4.1 Byte Specifications for Load/Stores

Access Type	Low-Order Address Bits	Bytes Accessed							
		Big-Endian				Little-Endian			
1 0	1 0	31			0	31			0
1 1 (word)	0 0	0	1	2	3	3	2	1	0
1 0 (triple-byte)	0 0	0	1	2			2	1	0
	0 1		1	2	3	3	2	1	
0 1 (halfword)	0 0	0	1					1	0
	1 0			2	3	3	2		
0 0 (byte)	0 0	0							0
	0 1		1					1	
	1 0			2			2		
	1 1				3	3			

CpCond0In

When CpCond0In is high, CpCond0 is configured as an input to indicate a block read (see Figure 7.6 for timing details) or a single word read during block refill. When CpCond0In is low, CpCond0 is configured as an output and informs the CPU about block/single read during cache miss cycles or write buffer status during run cycles. Low on CpCond0 indicates empty write buffer and high indicates that write buffer is not empty.

Reserved

Reserved for future implementation. This pin should be driven LOW.

GenParity

When high, enables the PR3100A parity generation mode during memory reads. In this mode, the PR3100A generates the parity bits to supply to the R3000A when parity is not implemented in the main memory. When low, parity generation is disabled.

MpNormal

When low, the clock edge causing the assertion of the invalidate signal will be followed by a memory bus cycle transferring the cache data and address information latched by the CacheRd signal as part of the MP sequence. When high the Cache Data is not latched.

Block

When Block is high, a block transfer will be performed. When low a simple word read will be performed. If CpCond0 is an input, Block is ignored.

BurstRead

When high, the PR3100A burst read mode will be enabled during block refills according to the DBlkSize and IBlkSize configured. When low Cache word address will be put on the memory bus during a block refill.

GatherAll

When high, enables byte gathering in all write buffer locations. Gather must also be enabled to enable GatherAll.Byte gathering will never occur in the next word to be written to the memory (ie. at the top of the buffer).

MemAcc Delay (2:0)

Defines the memory access time per word between acknowledges in clock cycles during block reads. If static column DRAMs are used, the clock cycles for static operation should be programmed. Used by the PR3100A to determine how many words must be present in the read buffer before it can begin transferring data to the R3000A/R3010A without interruption.

MemAcc Delay (2:0)	Clock Cycles
000	8
001	7
010	6
011	5
100	4
101	3
110	2
111	1

4.0 ELECTRICAL SPECIFICATIONS, COMMERCIAL TEMPERATURE RANGE[1, 2 & 3] (T=0°C TO 70°C, V=5V±5%)

4.1 MAXIMUM RATINGS [3]

Symbol	Parameter	Conditions	Min.	Max.	Units
V_{CC}	Supply Voltage		-0.5	+7.0	V
V_{IN}	Input Voltage(1)		-0.5	+7.0	V

Notes:
1. V_{IN} Min. = -3.0V for pulse width less than 15ns.
2. $V_{in} \leq V_{CC} + 0.5$
3. Not more than one output should be shorted at a time. Duration of the short should not exceed 30 seconds.

4.2 RECOMMENDED OPERATING CONDITIONS[1,2&3]

Grade	Ambient Temperature	GND	V_{CC}
Commercial	0°C to +70°C	0V	5.0V ± 5%

Notes:
1. The case temperature must be limited by using adequate air flow and/or an appropriate heat sink or other thermal management design.
2. The maximum operating junction temperature should be limited to 125°C.
3. For optimum performance and improved reliability, it is recommended that the operating junction temperature should be kept below 85°C.

4.3 CAPACITIVE LOAD DERATING FACTOR

Symbol	Parameter	Conditions	25MHz		33MHz		40MHz		Units
			Min.	Max.	Min.	Max.	Min.	Max.	
C_{LD}	Load Derate		0.5	1	0.5	1	0.5	1	ns/25pF

4.4 DC ELECTRICAL CHARACTERISTICS

Symbol	Parameter	Conditions	25MHz		33MHz		40MHz		Units
			Min.	Max.	Min.	Max.	Min.	Max.	
V_{OH}	Output HIGH Voltage	V_{CC} = Min. I_{OH} = -4mA	2.4		2.4		2.4		V
V_{OL}	Output LOW Voltage	V_{CC} = Min. I_{OL} = 4mA		0.4		0.4		0.4	V
V_{IH}	Input HIGH Voltage		2	V_{CC}+0.5	2	V_{CC}+0.5	2	V_{CC}+0.5	V
V_{IL}	Input LOW Voltage [1]		-0.5	0.8	-0.5	0.8	-0.5	0.8	V
V_{IHS}	Input HIGH Voltage [2]		3.5	V_{CC}+0.5	3.5	V_{CC}+0.5	3.5	V_{CC}+0.5	V
V_{ILS}	Input LOW Voltage [2]		-0.5	0.6	-0.5	0.6	-0.5	0.6	V
C_{IN}	Input Capacitance			10		10		10	pF
C_{OUT}	Output Capacitance			7		7		7	pF
I_{CC}	Operating Current	V_{CC} = 5.25V		150		200		240	mA
C_{Ld}	Load Capacitance			50		50		50	pF
I_{IL}	Input LOW Current	V_{IN} = GND V_{CC} = Max.	-10		-30		-50		µA
I_{IH}	Input HIGH Current	V_{IN} = V_{CC} V_{CC} = Max.		10		30		50	µA
I_{OZL}	Output 3 State Current LOW	V_{OUT} = 0.5V V_{CC} = Max.	-40		-60		-100		µA
I_{OZH}	Output 3 State Current HIGH	V_{OUT} = 2.4V V_{CC} = Max.		40		60		100	µA

Notes:
1. Transient inputs with V_L and I_L not more negative than –3.0V and –100mA, respectively are permissible for pulse widths up to 15ns.
2. V_{IHS} and V_{ILS} apply to Clock.

4.5 AC ELECTRICAL CHARACTERISTICS – (PART I)
COMMERCIAL TEMPERATURE RANGE (T = 0° to 70°C, V = 5V ±5%)

Symbol	Parameter Description	25MHz		33MHz		40MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
t_1	AdrIn (15:0) (AdrLo) to Clock falling setup (read)	3		2		2		ns
t_2	AdrIn (15:0) (AdrLo) to Clock falling hold (read)	3		2		2		ns
t_3	AdrIn (15:0) (AdrLo) to Clock rising setup (write)	3		2		2		ns
t_4	AdrIn (15:0) (AdrLo) to Clock rising hold (write)	3		2		2		ns
t_5	AdrIn (31:16) (Tag) to Clock falling setup	3		2		2		ns
t_6	AdrIn (31:16) (Tag) to Clock falling hold	3		2		2		ns
t_7	AccType (2:0) to Clock falling setup	3		2		2		ns
t_8	AccType (2:0) to Clock falling hold	3		2		2		ns
t_9	CPUData (31:0)/CPUDataP (3:0) to Clock falling setup (write)	3		2		2		ns
t_{10}	CPUData (31:0)/CPUDataP (3:0) to Clock falling hold (write)	3		2		2		ns
t_{11}	CPUData (31:0)/CPUDataP (3:0) valid from $\overline{\text{XEn}}$ falling (read)		10		8		6.5	ns
t_{12}	CPUData (31:0)/CPUDataP (3:0) tri-state from $\overline{\text{XEn}}$ rising (read)		8		6		5	ns
t_{13}	CPUData (31:0)/CPUDataP (3:0) valid from Clock falling (instruction read)		15		12		10	ns
t_{14}	$\overline{\text{WtMem}}$ to Clock falling setup	3		2		2		ns
t_{15}	$\overline{\text{WtMem}}$ to Clock falling hold	3		2		2		ns
t_{16}	$\overline{\text{WBFull}}$ valid from Clock rising		15		11		9	ns
t_{17}	$\overline{\text{RdMem}}$ to Clock falling setup	7		5		4		ns
t_{18}	$\overline{\text{RdMem}}$ to Clock falling hold	3		2		2		ns
t_{19}	RdBufBusy valid from Clock falling		18		14		11	ns
t_{20}	ParErr valid from Clock rising		14		11		9	ns
t_{21}	CpCond0 valid from Clock rising		18		14		11	ns
t_{23}	CpCond0 to Clock rising setup	3		2		2		ns
t_{24}	CpCond0 to Clock rising hold	3		2		2		ns
t_{25}	Conflict valid from Clock rising		24		21		17	ns
t_{26}	$\overline{\text{Request}}$ valid from Clock rising		16		13		12	ns
t_{27}	R/$\overline{\text{W}}$ valid from Clock rising		14		12		10	ns
t_{28}	$\overline{\text{OutEn}}$ to Clock falling setup	3		2		2		ns
t_{29}	$\overline{\text{OutEn}}$ to Clock falling hold	3		2		2		ns
t_{30}	MemBus (31:2) MemDataP (3:2) valid from Clock rising		19		16		14	ns
t_{30p}	MemBus (1:0) MemDataP (1:0) Valid from Clock rising		20		18		16	ns
t_{30a}	MemBus (31:0) driven (Low Z) from Clock rising	3.5		3.5		3.5		ns
t_{31}	MemBus (31:2) tri-state from Clock rising		16		12		10	ns
t_{32}	MemBus (31:0) to Clock falling setup	3		2		2		ns
t_{33}	MemBus (31:0) to Clock falling hold	3		2		2		ns
t_{34}	$\overline{\text{Acknowledge}}$ to Clock falling setup	3		2		2		ns
t_{35}	$\overline{\text{Acknowledge}}$ to Clock falling hold	3		2		2		ns
t_{36}	Short LE valid from Clock rising		12		10		9	ns

4.5 AC ELECTRICAL CHARACTERISTICS – (PART II) COMMERCIAL TEMPERATURE RANGE

Symbol	Parameter Description	25MHz		33MHz		40MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
t_{36L}	Long LE valid from Clock edge		12		10		9	ns
t_{37}	Short LE valid from Clock falling		12		10		9	ns
t_{38}	R/$\overline{\text{W}}$ tri-state from Clock rising		16		12		10	ns
t_{39}	$\overline{\text{Static}}$ valid from Clock rising		18		16		14	ns
t_{40}	$\overline{\text{Static}}$ tri-state from Clock rising		16		12		10	ns
t_{41}	LatchErrAdr to Clock falling setup	6		4		3		ns
t_{42}	LatchErrAdr to Clock falling hold	3		2		2		ns
t_{43}	$\overline{\text{EnErrorAdr}}$ to Clock falling setup	3		2		2		ns
t_{44}	$\overline{\text{EnErrorAdr}}$ to Clock falling hold	3		2		2		ns
t_{45}	$\overline{\text{BusWatch}}$ to Clock falling setup	3		2		2		ns
t_{46}	$\overline{\text{BusWatch}}$ to Clock falling hold	3		2		2		ns
t_{47}	MpRequest to Clock rising setup	3		2		2		ns
t_{48}	MpRequest to Clock rising hold	3		2		2		ns
t_{49}	CacheRd to Clock falling setup	3		2		2		ns
t_{50}	CacheRd to Clock falling hold	3		2		2		ns
t_{51}	Invalidate from Clock rising		14		12		10	ns
t_{52}	$\overline{\text{Reset}}$ to Clock falling setup	5		4		3		ns
t_{53}	$\overline{\text{Reset}}$ to Clock falling hold	3		2		2		ns
t_{54}	Configuration[1] to Clock falling setup	3		2		2		ns
t_{55}	Configuration[1] to Clock falling hold	3		2		2		ns
t_{56}	Short LE pulse width	18		13		11		ns
t_{56L}	Long LE pulse width	37		28		23		ns
t_{57}	MemBus Address to Short LE Falling Setup	7		5		4		ns
t_{57L}	MemBus Address to Long LE Falling Setup	25		19		15		ns
t_{58}	$\overline{\text{BusError}}$ from Clock falling		18		14		11	ns
t_{59}	Address hold from Short LE falling	13		10		8		ns
t_{59L}	Address hold from Long LE falling	33		25		20		ns
t_{60}	$\overline{\text{BufOE}}$ low from clock edge		13		11		9	ns
t_{61}	$\overline{\text{BufOE}}$ high from clock edge		13		11		9	ns
t_{62}	BufDir low from clock edge		14		11		9	ns
t_{63}	BufDir high from clock edge		14		11		9	ns
t_{65}	BufDir tri-state from clock edge		11		9		7	ns
t_{65a}	BufDir high from tri-state		16		14		11	ns
t_{66}	Byte Assemble setup to clock rising	3		2		2		ns
t_{67}	Byte Assemble hold to clock rising	3		2		2		ns

Note:
1. Configuration inputs consist of: LatchErrAdr, $\overline{\text{EnErrorAdr}}$, $\overline{\text{BusWatch}}$, CacheRd, & MpRequest

5.0 TIMING DIAGRAMS

Figure 5.1 – Read Timing with Bus Acquisition – Data Timing

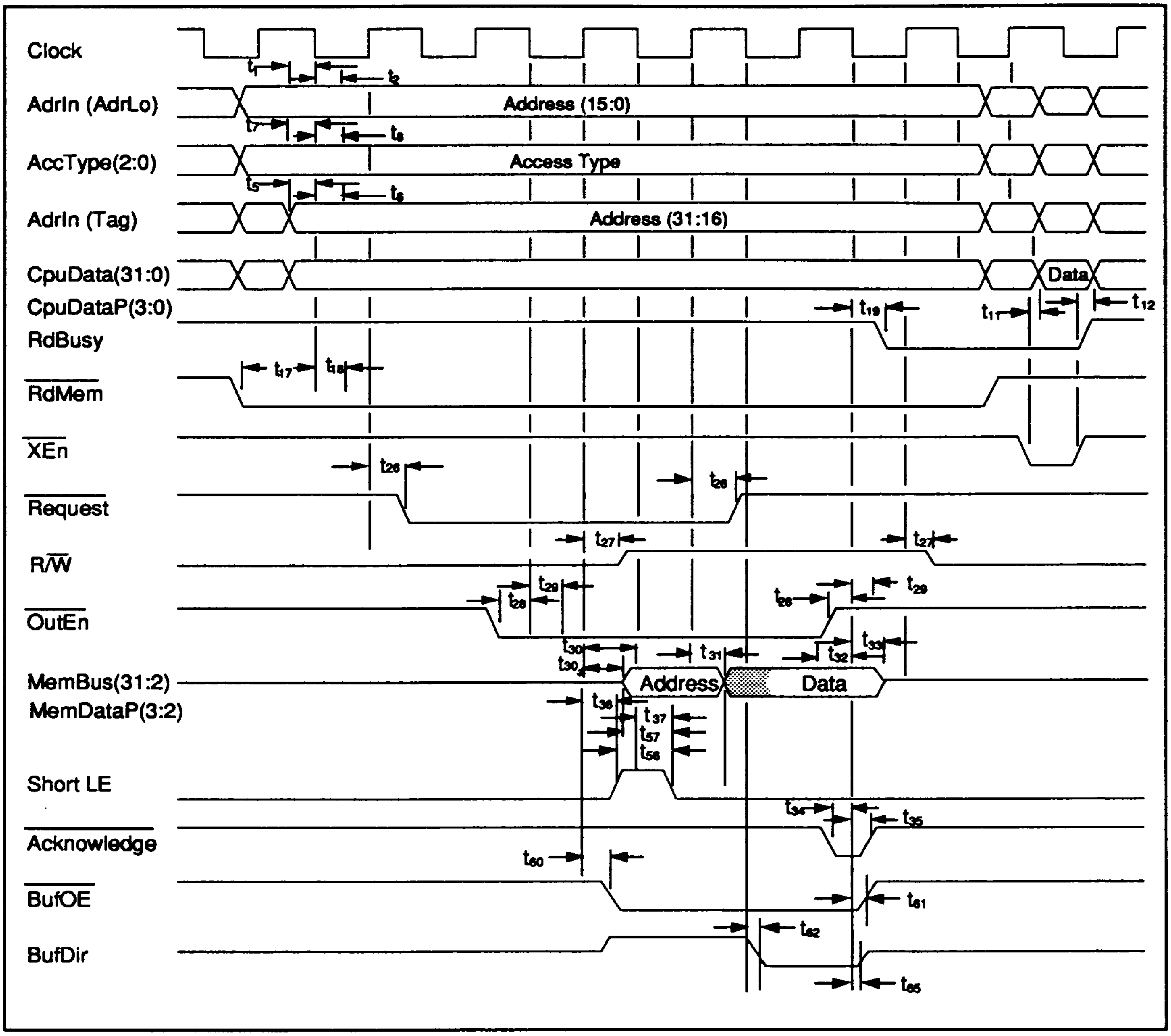


Figure 5.1a – Read Timing with Bus Acquisition – Instruction Timing

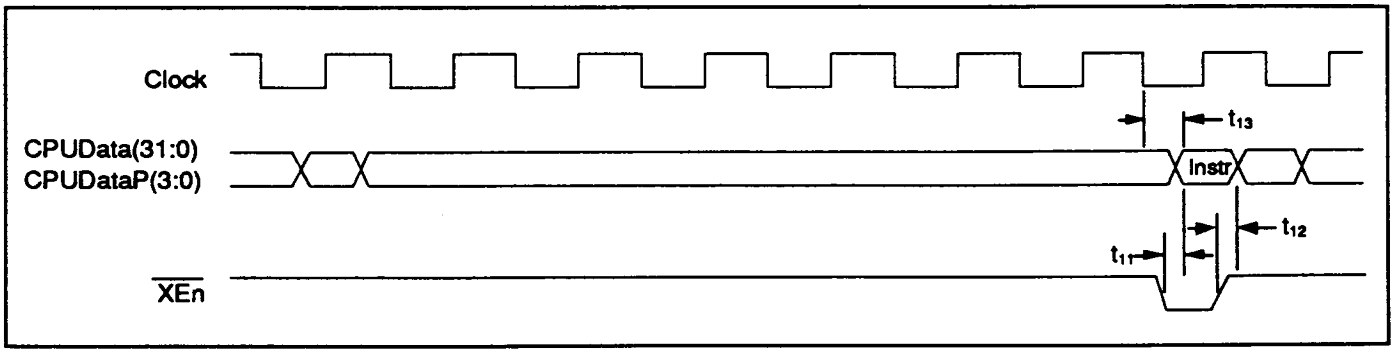


Figure5.2 – Write Timing with Bus Acquisition

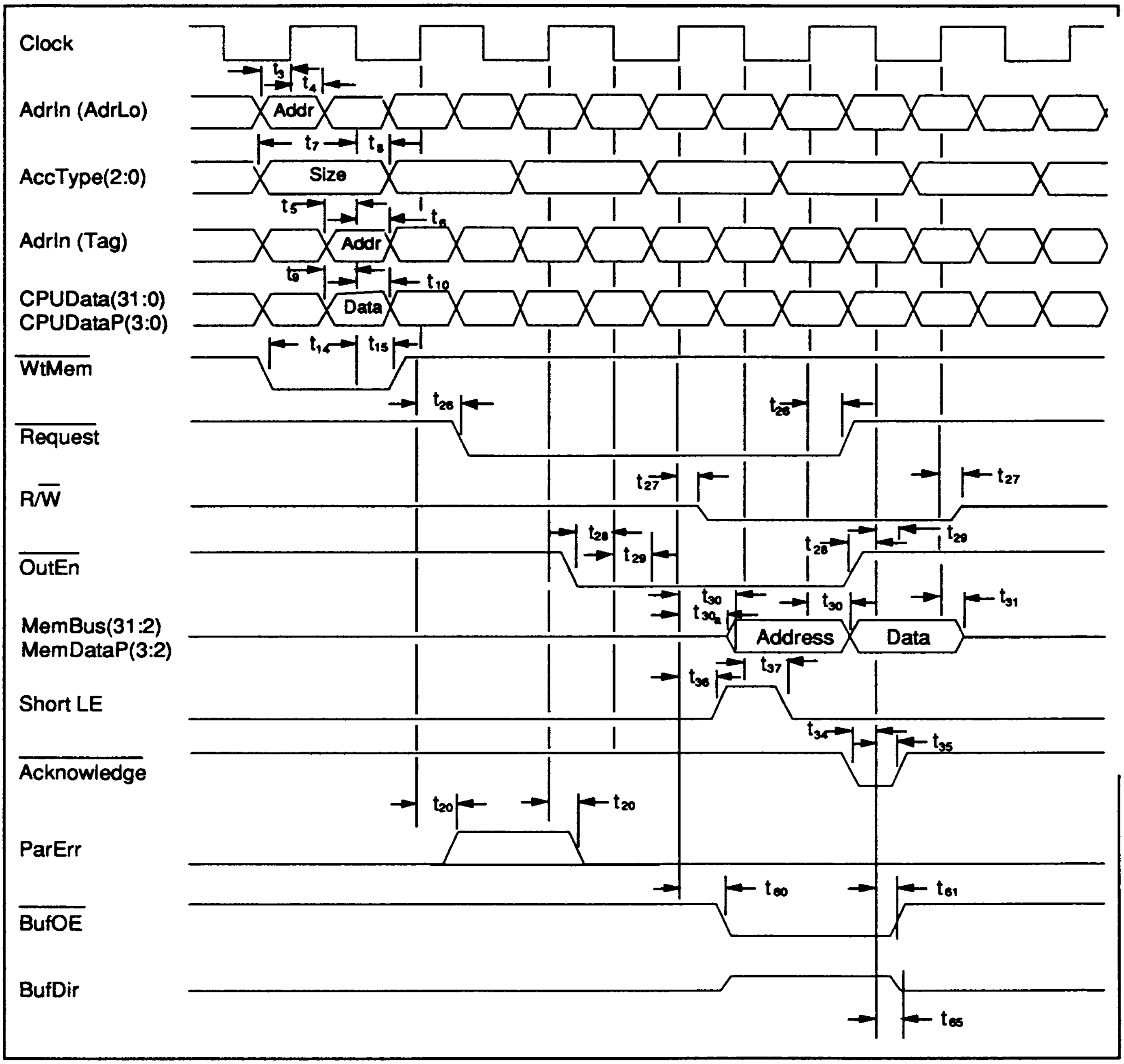


Figure 5.3 – Read Timing Without Bus Acquisition

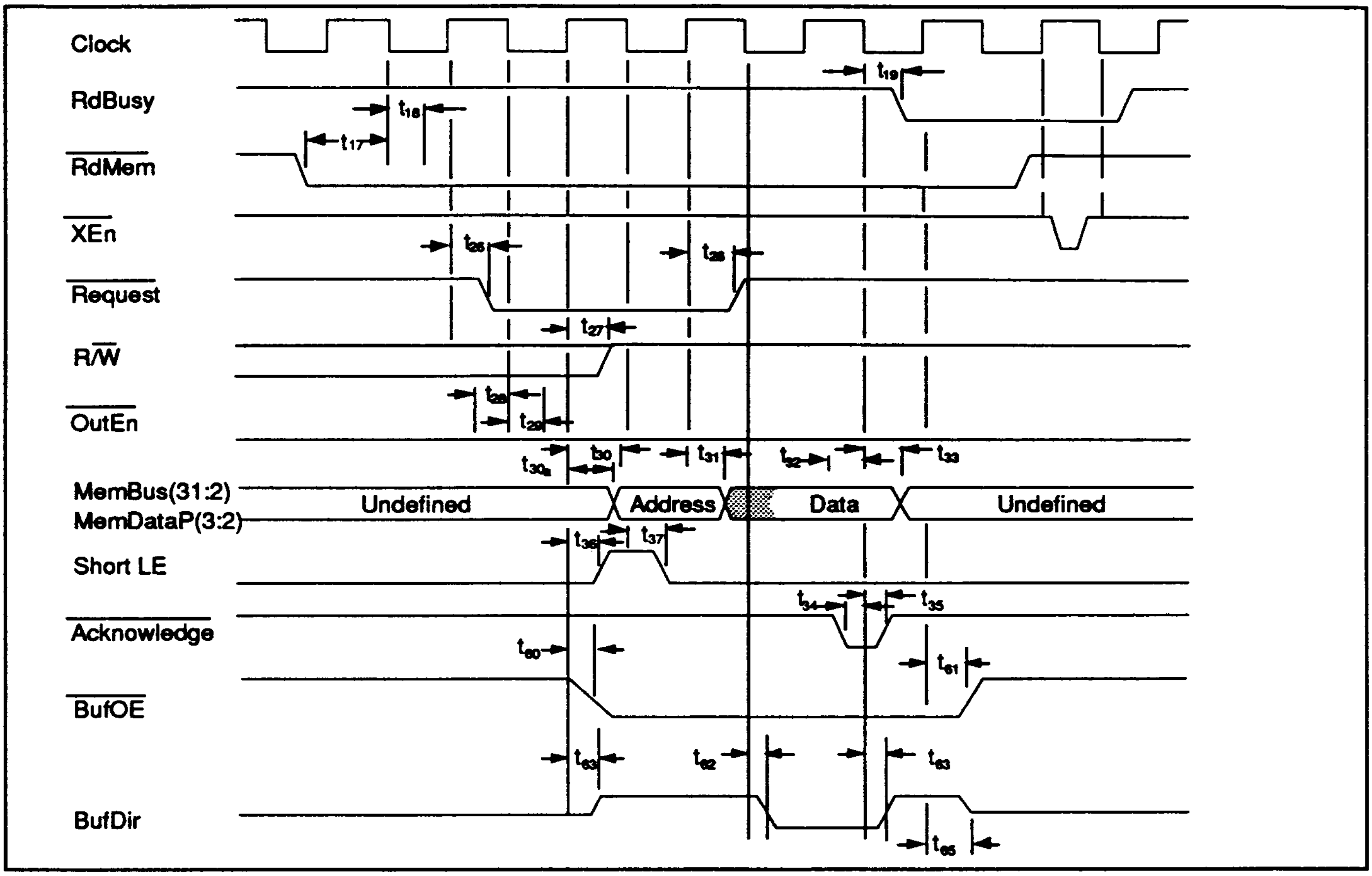


Figure 5.4 – Burst Read Timing

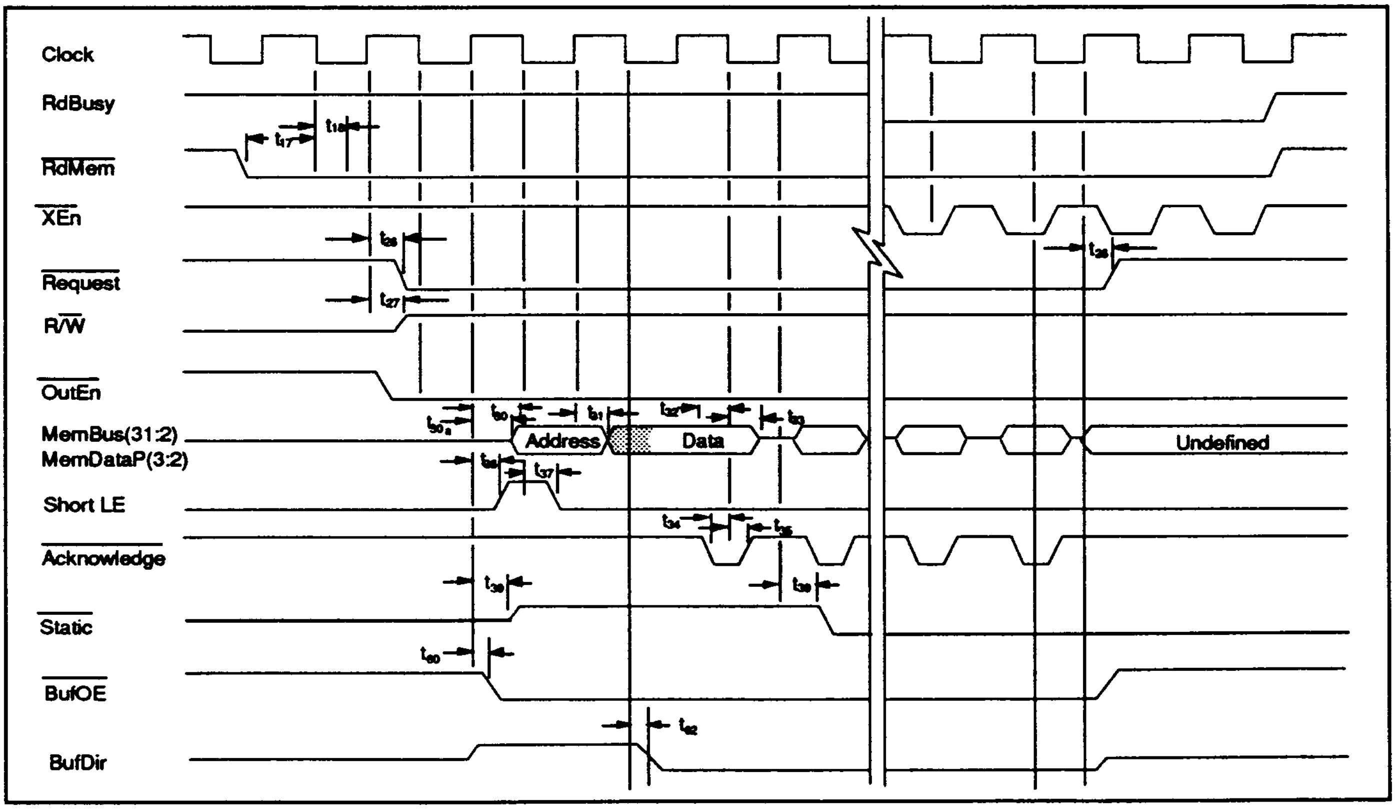
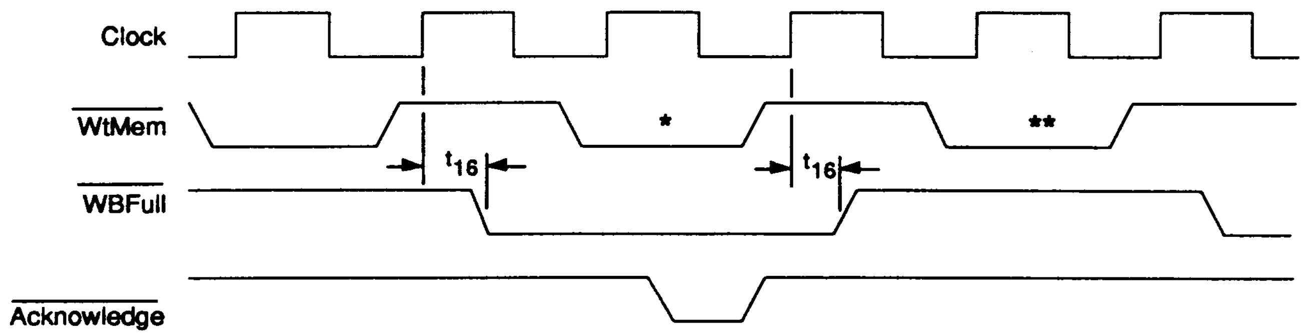
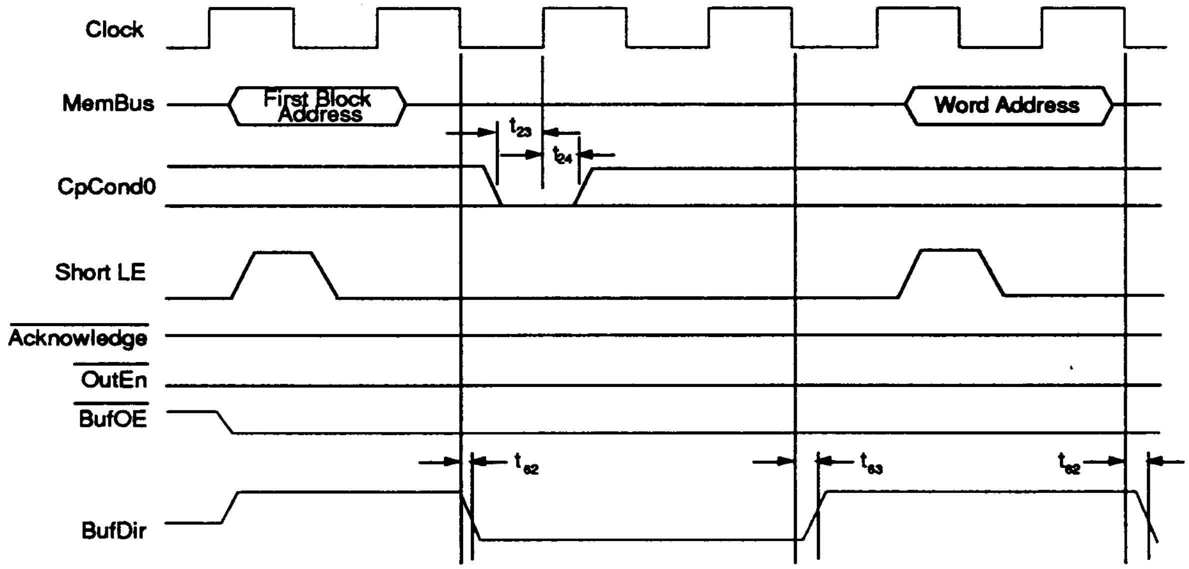


Figure5.5 – WBFull Timing



* This write is ignored since the buffer is full.
** This write is a repeat of the previous one (fixup cycle).

Figure5.6 – CpCond0 as an Input



Note: There is no Acknowledge after the first Short LE

Figure 5.7 – CpCond0 as an Output

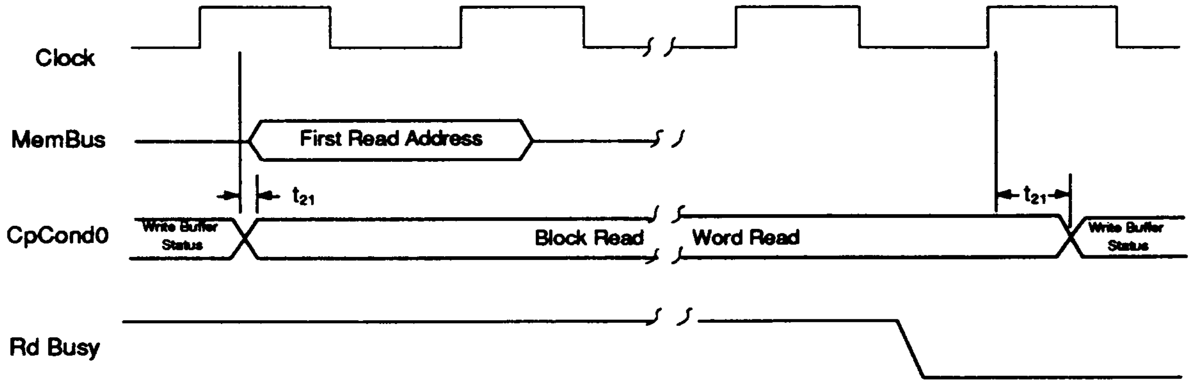


Figure 5.8 – Write While Streaming

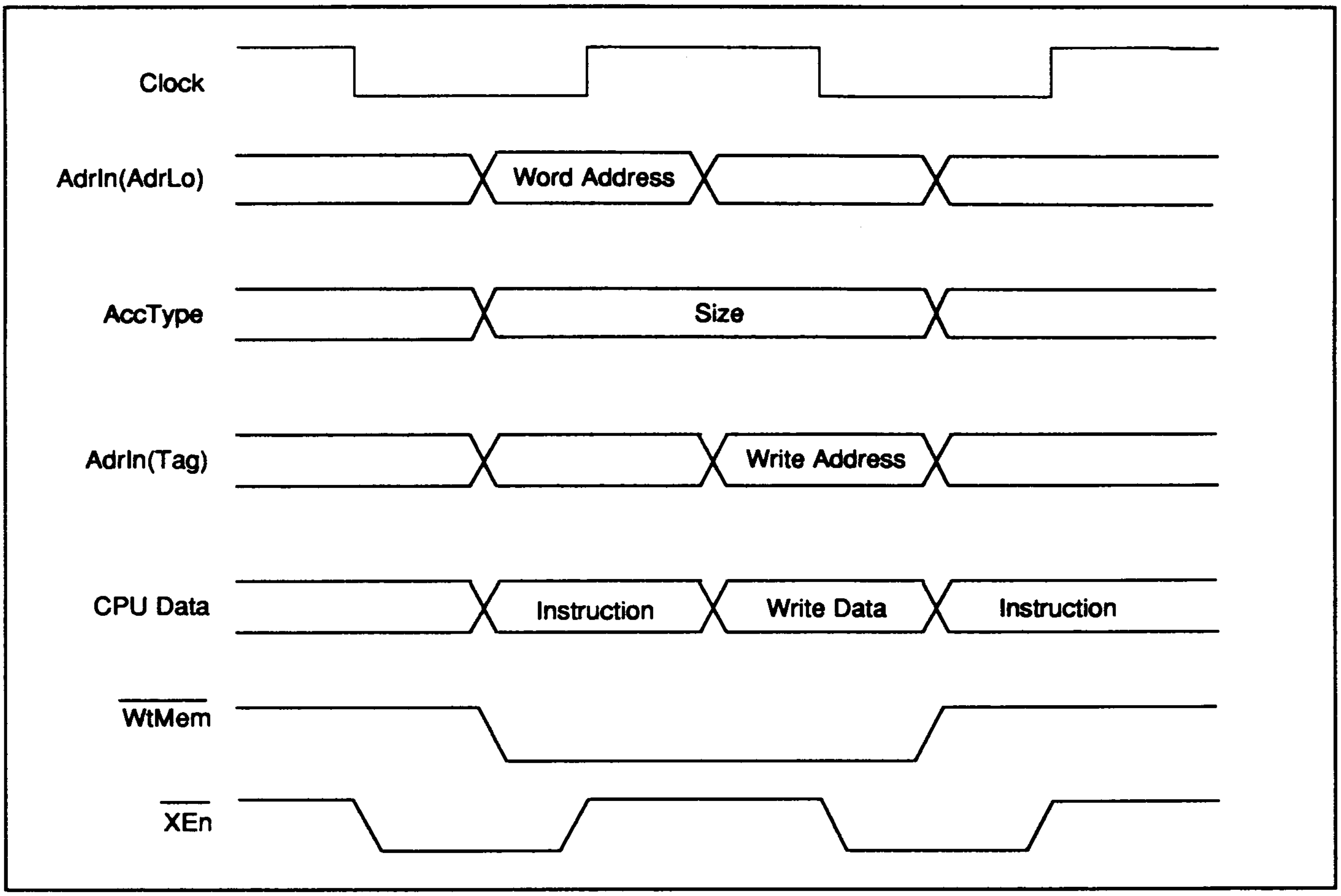
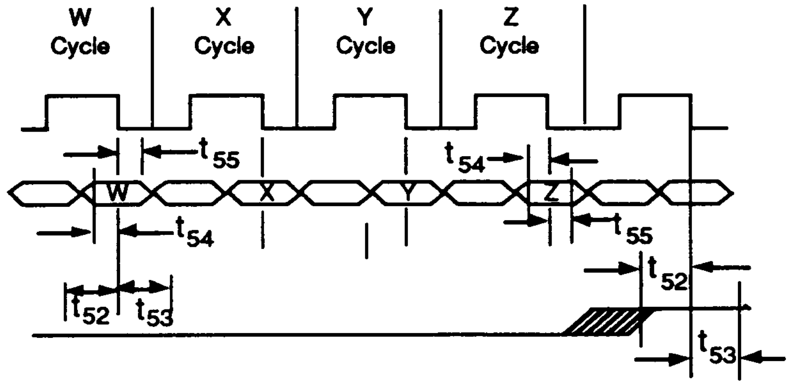


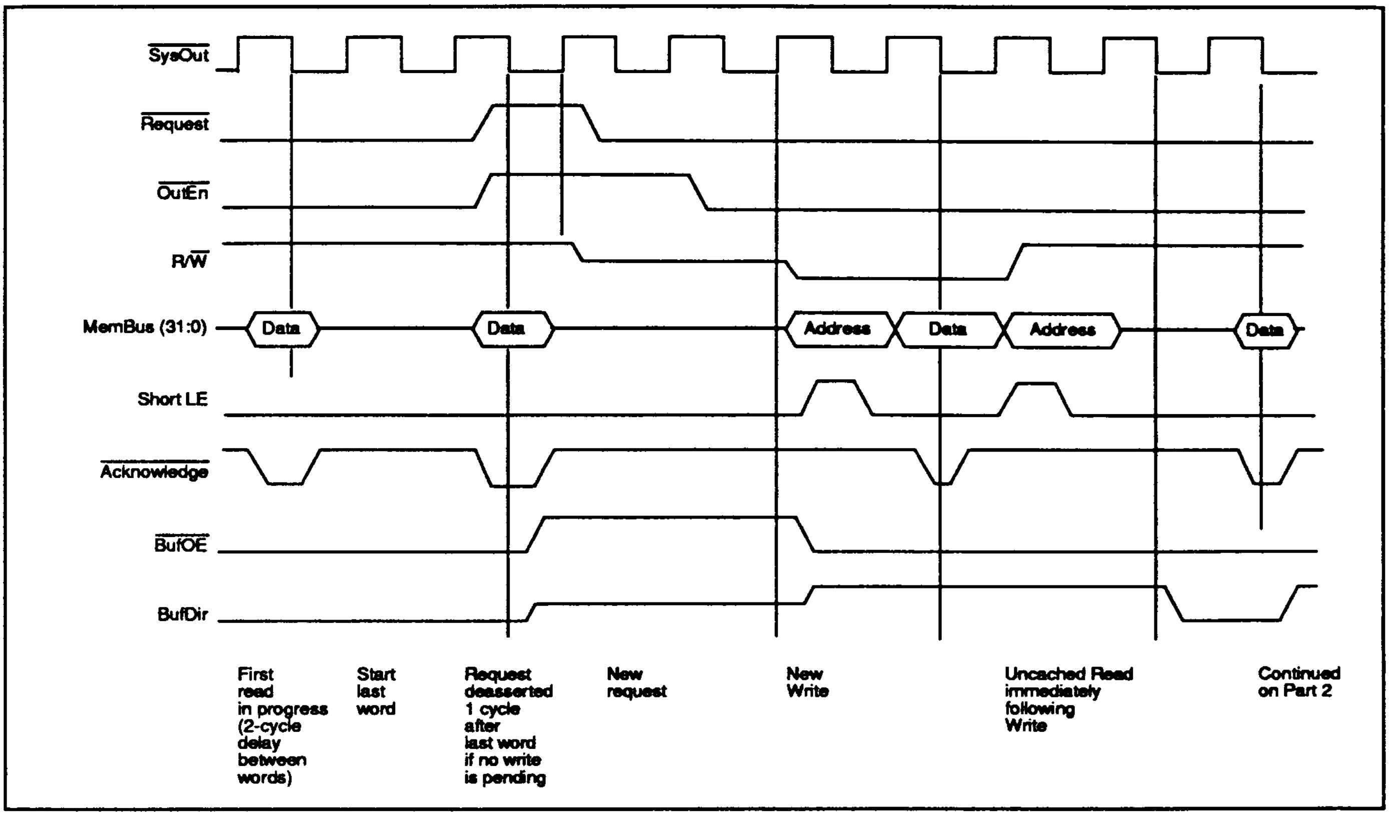
Figure 5.8 a– Configuration Inputs and Reset Timing



Note 1:

Configuration Pins are — LatchErrAdr, EnErrorAdr, BusWatch, CacheRd, & MpRequest

Figure 5.9 – Multi-Operation Timing (Part 1)



First read in progress (2-cycle delay between words)

Start last word

Request deasserted 1 cycle after last word if no write is pending

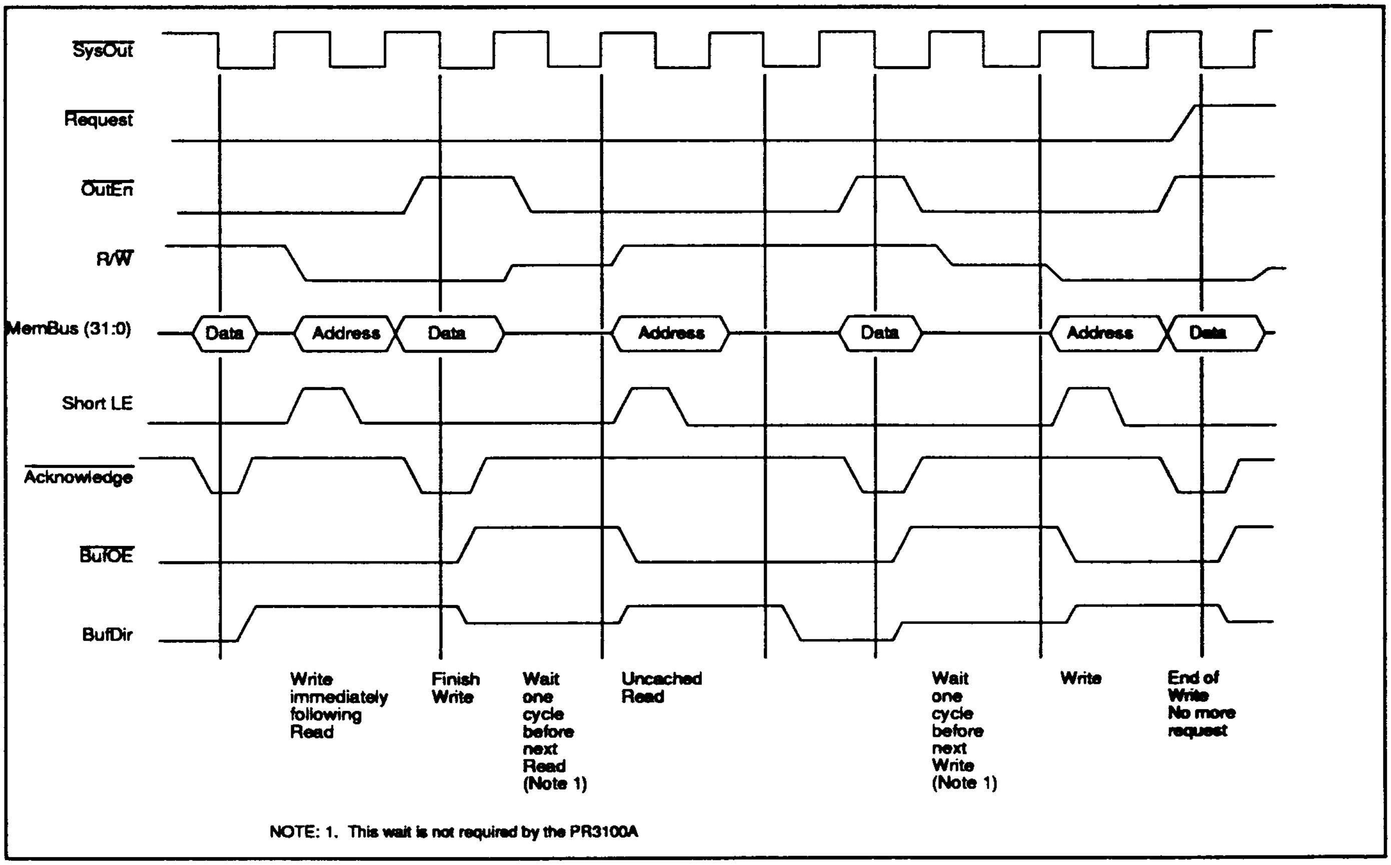
New request

New Write

Uncached Read immediately following Write

Continued on Part 2

Figure5.9 – Multi-Operation Timing (Part 2)



Write immediately following Read

Finish Write

Wait one cycle before next Read (Note 1)

Uncached Read

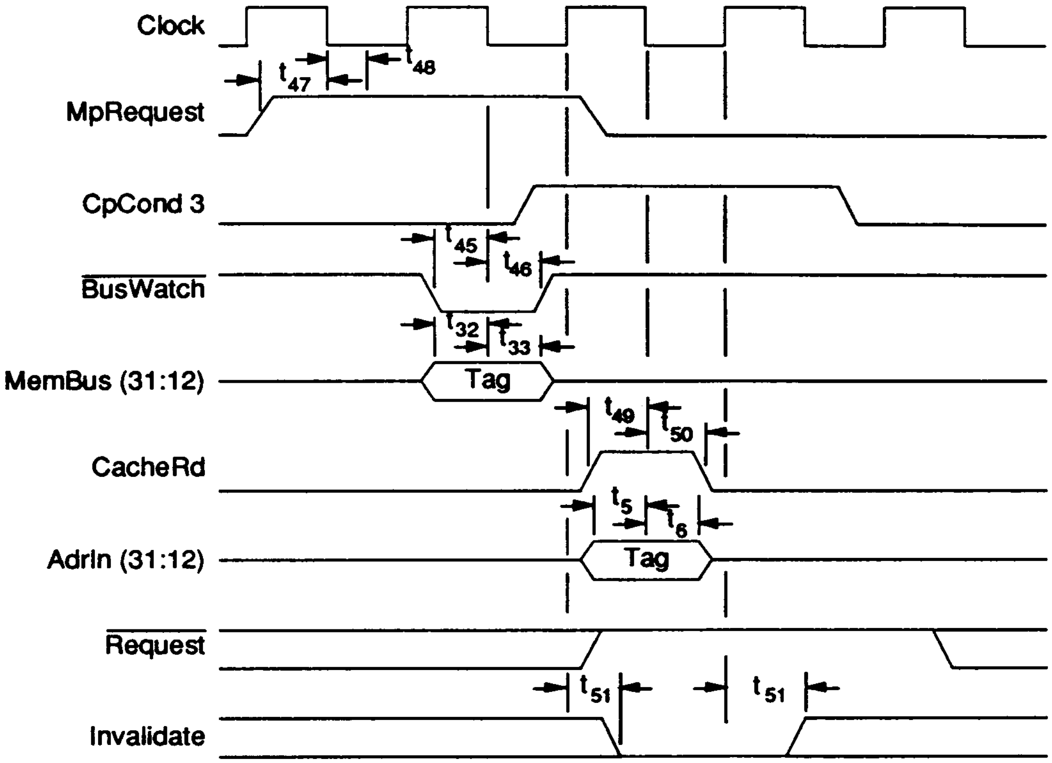
Wait one cycle before next Write (Note 1)

Write

End of Write No more request

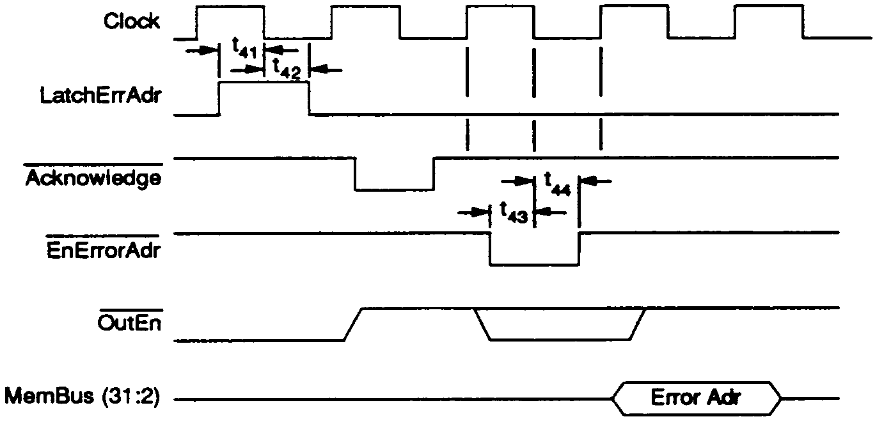
NOTE: 1. This wait is not required by the PR3100A

Figure 5.10 – MP Mode Timing



CpCond 3 shown for clarity only.
CpCond 3 assertion and de-assertion must follow MpRequest by 1 1/2 cycles.
BusWatch can be asserted with or after MpRequest.
MemBus (31:12) will be sampled when BusWatch is asserted.
Request and Invalidate will be de-asserted 1/2 cycle following the assertion of BusWatch.
CacheRd can be asserted any cycle following the assertion of BusWatch.
AdrIn (31:12) will be sampled when CacheRd is asserted.
Invalidate will be asserted 1/2 cycle after the assertion of CacheRd if MemBus (31:12) equals AdrIn (31:12).
Request will be valid 1 1/2 cycles after the assertion of CacheRd.

Figure 5.11 – Timing Error Register



LatchErrAdr can be asserted before or concurrent with Acknowledge.
LatchErrAdr must be asserted before or with deassertion of Acknowledge.
EnErrorAdr can be asserted with or after LatErrAdr.
EnErrorAdr can be asserted for one or more cycles.
Only MemBus (31:2) are latched. MemBus (1:0) are always 0.

Figure 5.12 – Long LE Timing for Read

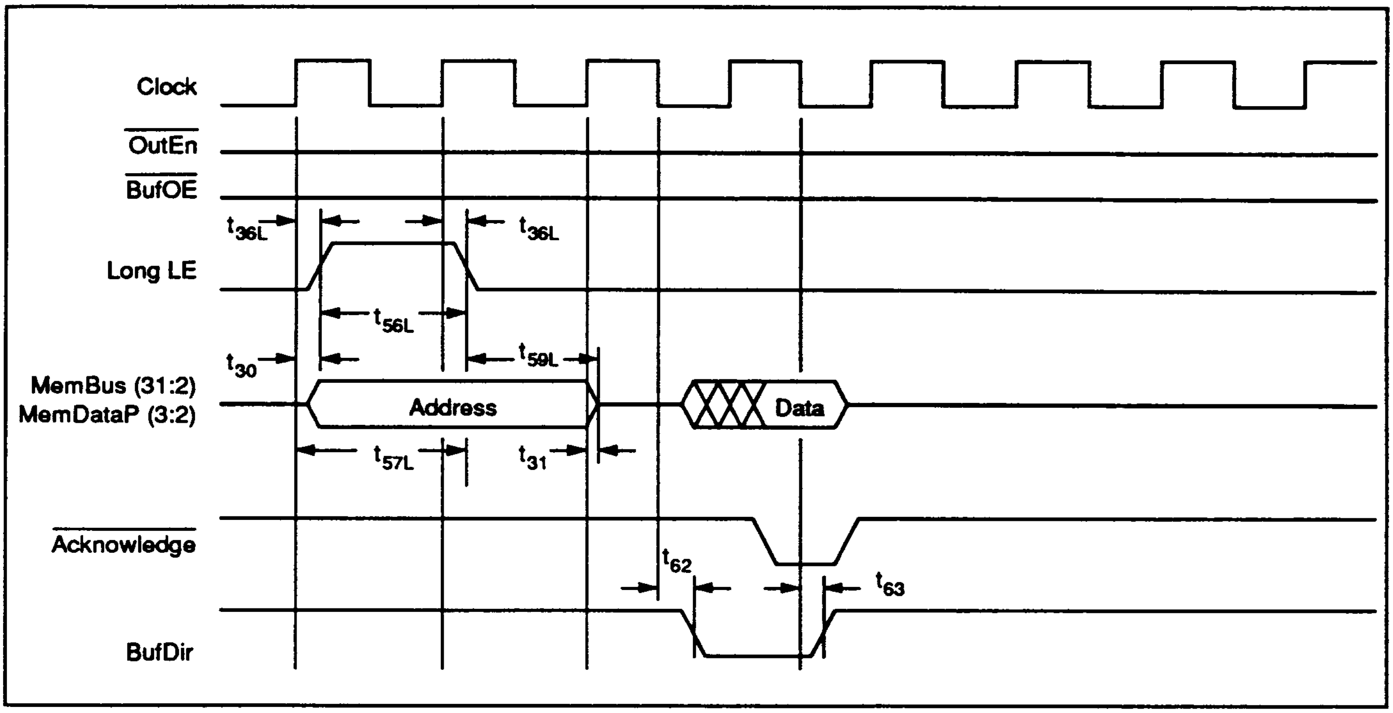


Figure 5.13 – Long LE Timing for Write

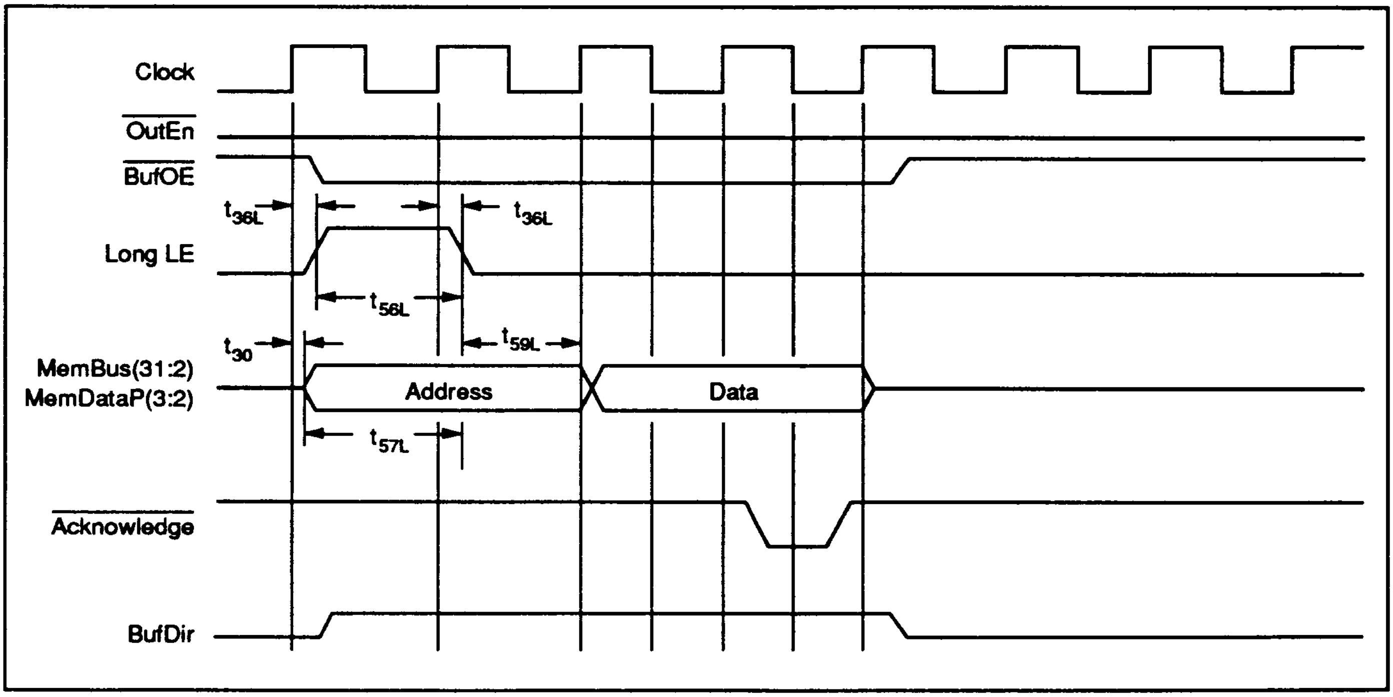
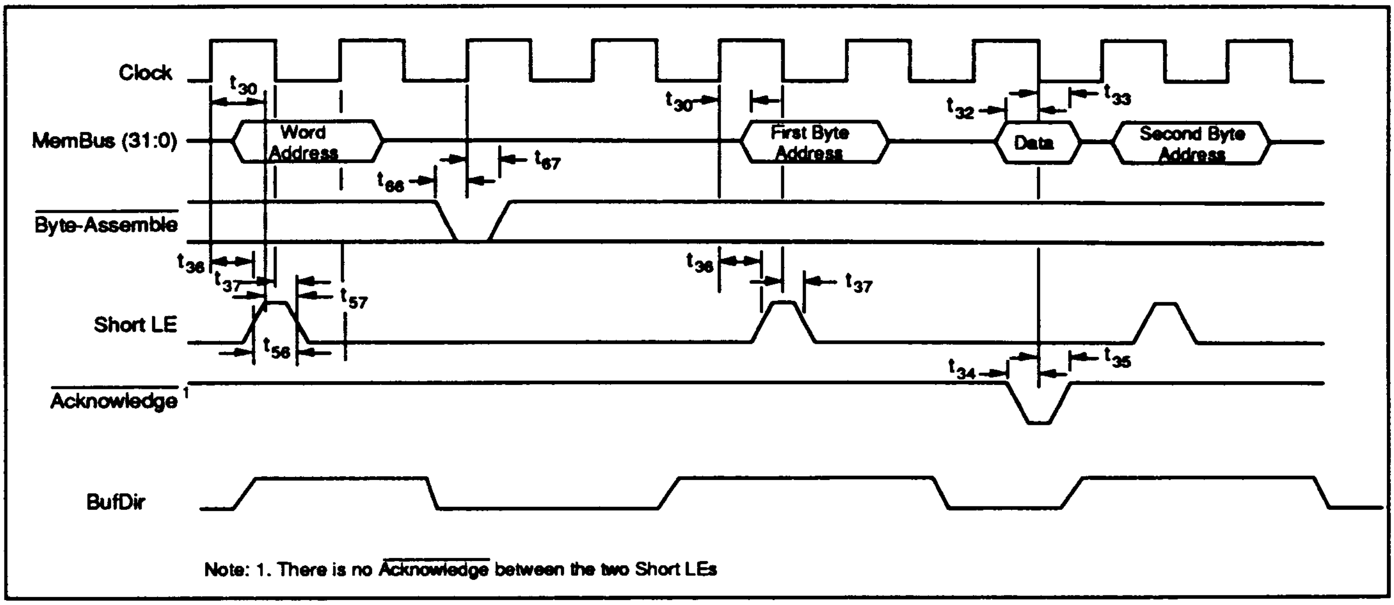


Figure 5.14 – Byte Assembly Timing



Note: 1. There is no Acknowledge between the two Short LEs

Figure 5.15 – BusError Timing

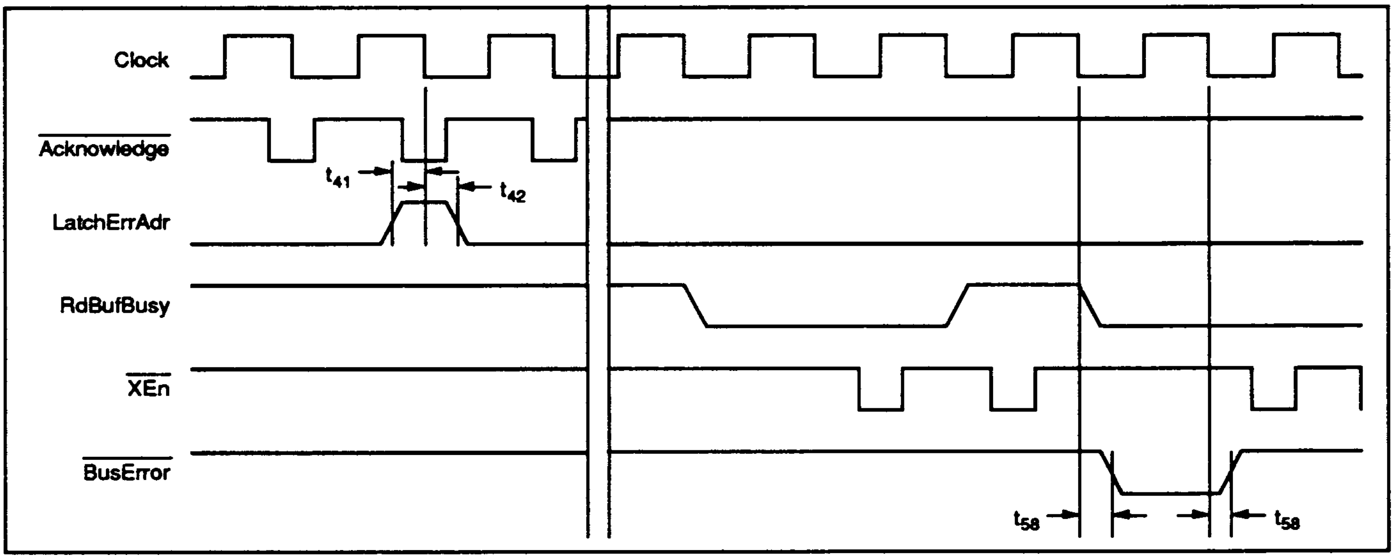


Figure 5.16 – Conflict Timing

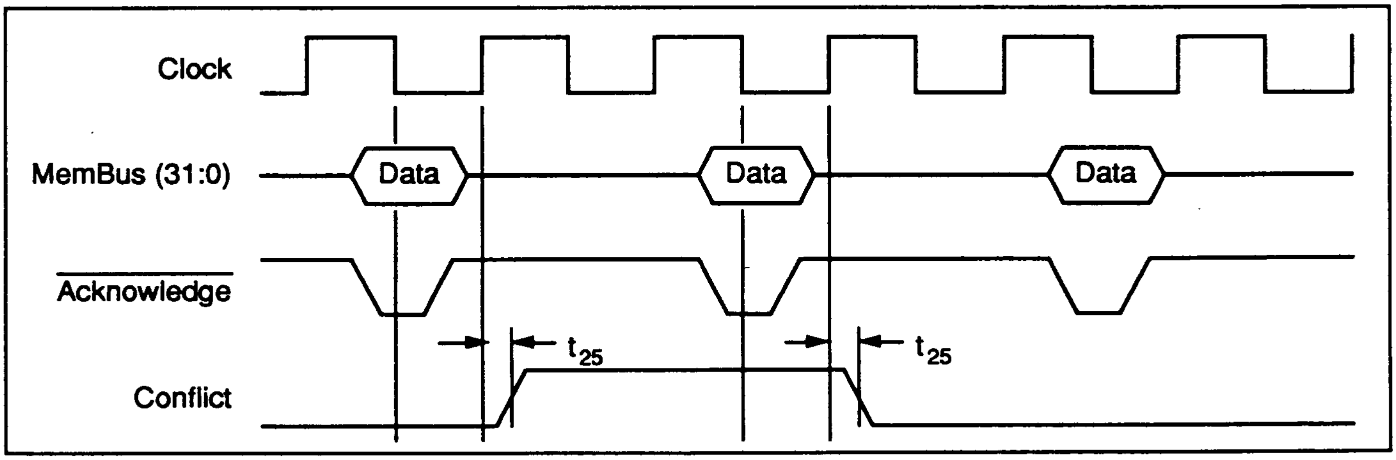
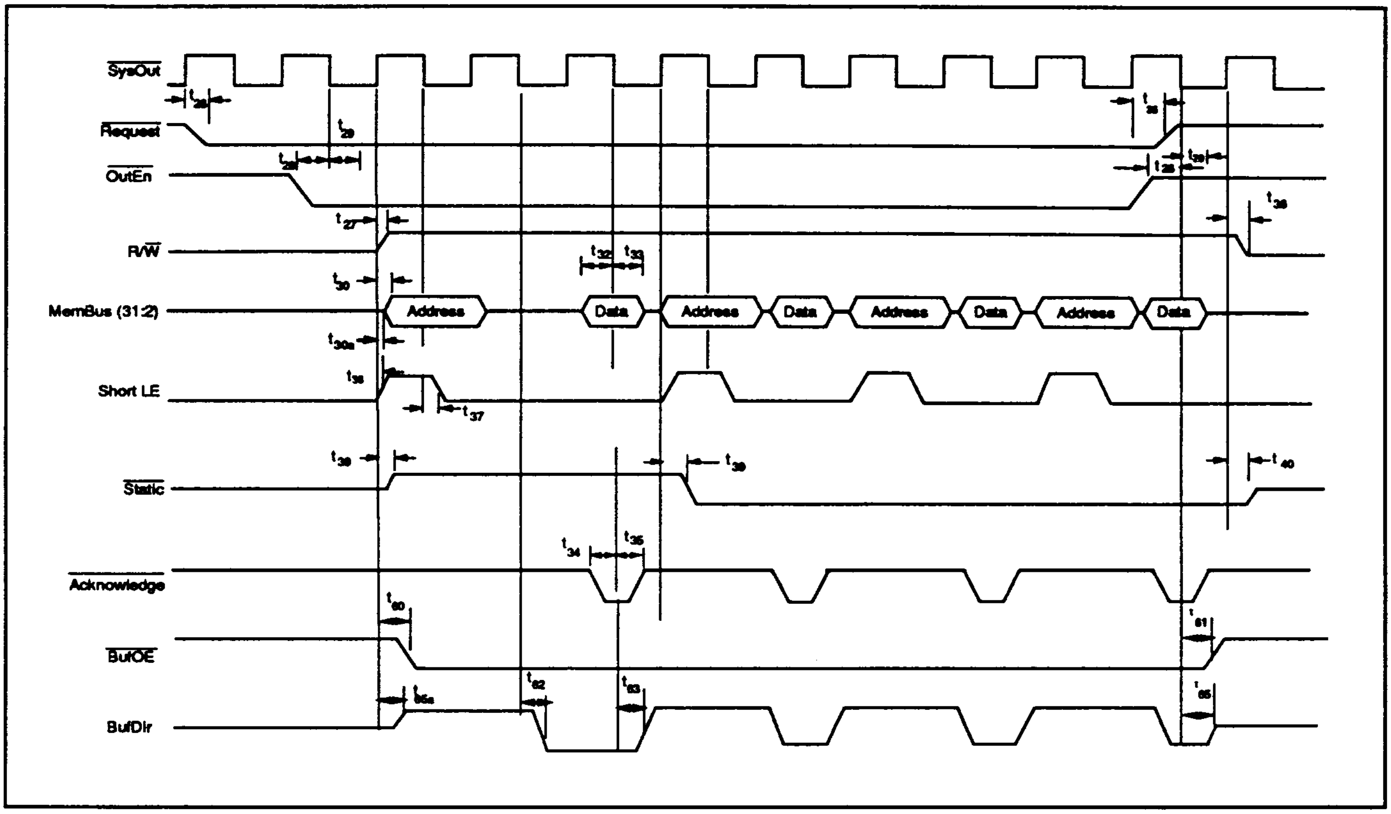


Figure 5.17 – 4-Word Block Read Timing



6.0 MECHANICAL DATA

TABLE 6.1 PR3100A PINOUT – 160 LEAD PLASTIC QUAD FLAT PACK

PIN NAME	PIN NUMBER	PIN NAME	PIN NUMBER	PIN NAME	PIN NUMBER
CPUData(0)	128	ParErr	127	MemBus(9)	110
CPUData(1)	129	XEn	150	MemBus(10)	109
CPUData(2)	130	AdrIn(0)	40	MemBus(11)	107
CPUData(3)	131	AdrIn(1)	43	MemBus(12)	106
CPUData(4)	132	AdrIn(2)	44	MemBus(13)	105
CPUData(5)	134	AdrIn(3)	45	MemBus(14)	103
CPUData(6)	136	AdrIn(4)	46	MemBus(15)	102
CPUData(7)	137	AdrIn(5)	47	MemBus(16)	100
CPUData(8)	138	AdrIn(6)	48	MemBus(17)	99
CPUData(9)	139	AdrIn(7)	49	MemBus(18)	97
CPUData(10)	140	AdrIn(8)	51	MemBus(19)	96
CPUData(11)	142	AdrIn(9)	52	MemBus(20)	95
CPUData(12)	143	AdrIn(10)	53	MemBus(21)	94
CPUData(13)	144	AdrIn(11)	54	MemBus(22)	92
CPUData(14)	145	AdrIn(12)	55	MemBus(23)	91
CPUData(15)	146	AdrIn(13)	56	MemBus(24)	89
CPUData(16)	147	AdrIn(14)	57	MemBus(25)	88
CPUData(17)	148	AdrIn(15)	58	MemBus(26)	87
CPUData(18)	152	AdrIn(16)	59	MemBus(27)	86
CPUData(19)	153	AdrIn(17)	61	MemBus(28)	85
CPUData(20)	154	AdrIn(18)	62	MemBus(29)	84
CPUData(21)	155	AdrIn(19)	63	MemBus(30)	83
CPUData(22)	156	AdrIn(20)	64	MemBus(31)	82
CPUData(23)	157	AdrIn(21)	65	VCC0	1
CPUData(24)	159	AdrIn(22)	67	VCC1	9
CPUData(25)	160	AdrIn(23)	68	VCC2	20
CPUData(26)	2	AdrIn(24)	70	VCC3	21
CPUData(27)	3	AdrIn(25)	71	BufOE	24
CPUData(28)	4	AdrIn(26)	72	VCC5	36
CPUData(29)	5	AdrIn(27)	73	ByteAssemble	66
CPUData(30)	6	AdrIn(28)	74	VCC7	80
CPUData(31)	7	AdrIn(29)	75	VCC8	93
CPUDataP(0)	10	AdrIn(30)	76	VCC9	108
CPUDataP(1)	11	AdrIn(31)	77	VCC10	121
CPUDataP(2)	12	Request	25	VCC11	135
CPUDataP(3)	13	Acknowledge	16	VCC12	151
AccType(0)	29	LE	34	Gnd0	8
AccType(1)	28	OutEn	17	Gnd1	22
AccType(2)	26	R/W	33	Gnd2	27
Reset	14	AccTypeOut(0)	123	Gnd3	41
Clock	23	AccTypeOut(1)	124	BufDir	50
RdMem	30	Instr	125	Gnd5	69
WtMem	18	Cached	126	Gnd6	81
RdBufBusy	15	BusError	60	Gnd7	90
WbFull	31	MemBus(0)	120	Gnd8	98
LatchErrAdr	78	MemBus(1)	119	NC	101
EnErrorAdr	79	MemBus(2)	118	NC	104
CpCond0	35	MemBus(3)	117	Gnd11	112
MpRequest	32	MemBus(4)	116	Gnd12	122
Invalidate	37	MemBus(5)	115	Gnd13	133
CacheRd	39	MemBus(6)	114	Gnd14	141
BusWatch	38	MemBus(7)	113	Gnd15	149
Static	19	MemBus(8)	111	Gnd16	158
Conflict	42				

FIGURE 6.1 PR3100A PIN DIAGRAM – 160 LEAD PLASTIC QUAD FLAT PACK

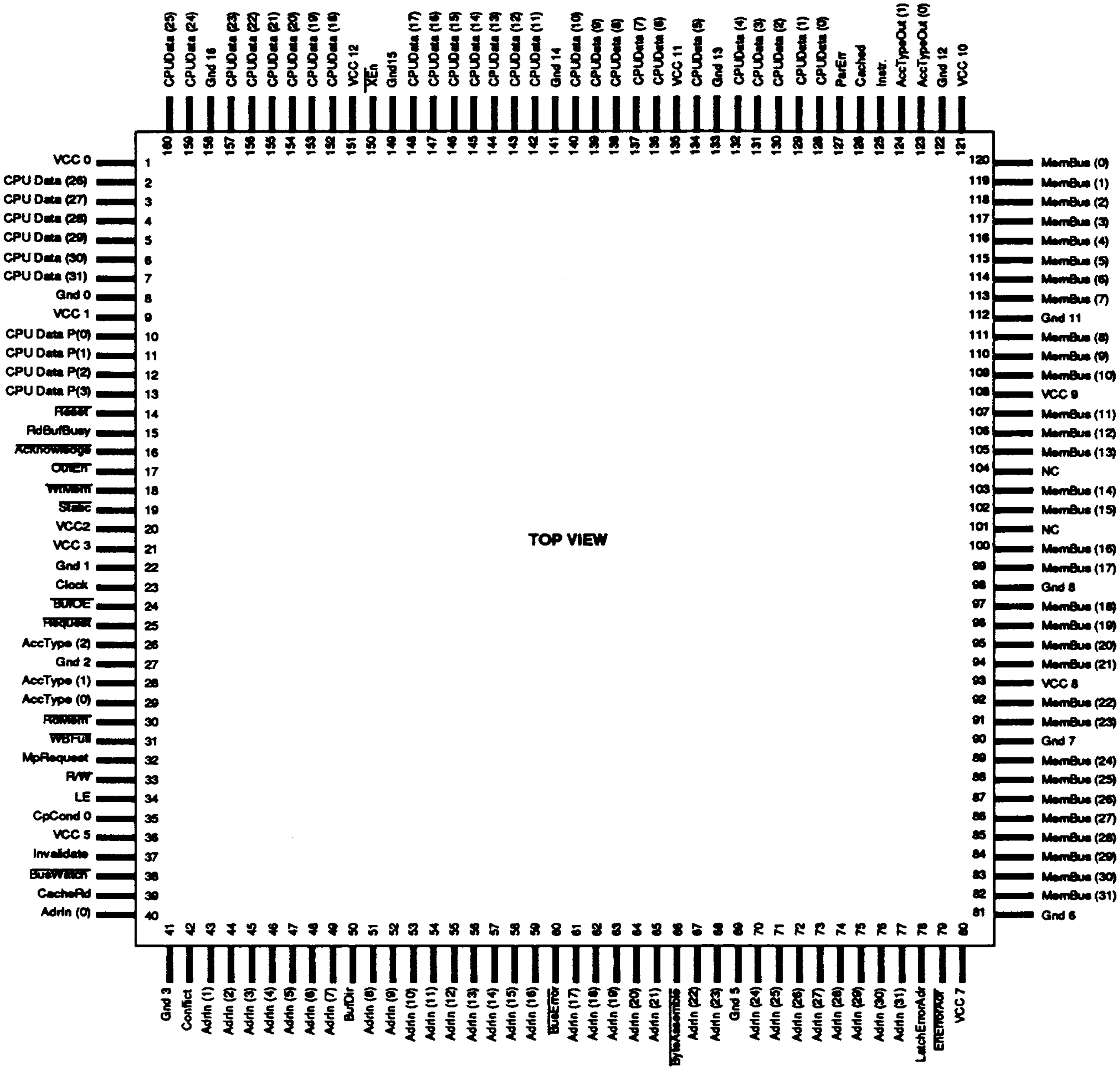
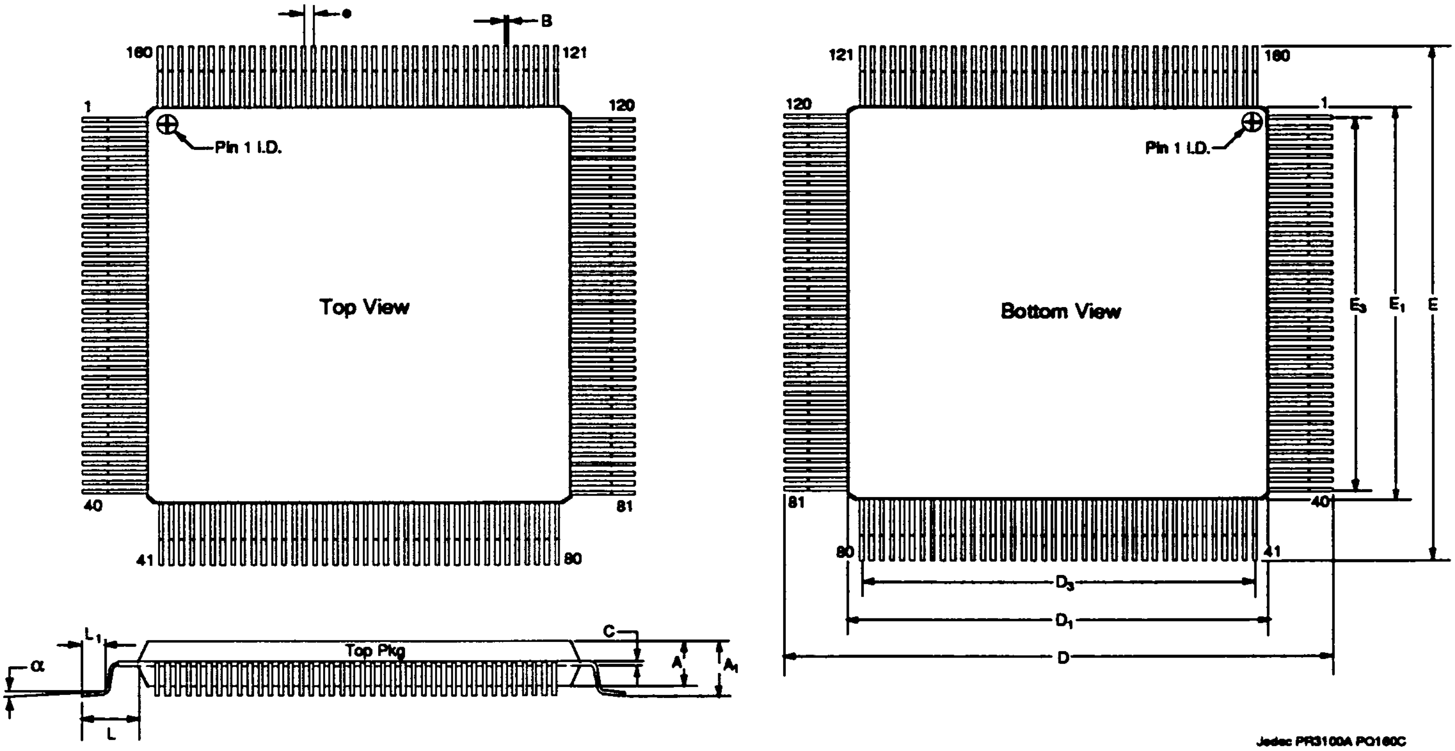


FIGURE6.2 PR3100A PACKAGE DIMENSIONS – 160 LEAD PLASTIC QUAD FLAT PACK, CAVITY UP



Jedec PR3100A PQ160C

Symbol	Min.		Max.	
	in.	mm.	in.	mm.
A	0.135	3,50	0.145	3,70
A_1	0.140	3,62	0.160	4,00
B	0.010	0,29	0.012	0,31
C	0.004	0,10	0.008	0,20
D/E	1.240	31,50	1,260	32,0
D_1/E_1	1.085	27,55	1.100	27,75
D_3/E_3	1.000	25,4	1.000	25,4
e	0.025 BSC	0,64 BSC	0.25 BSC	0,64 BSC
L	0.040	1,00	0.080	2,00
L_1	0.025	0,65	0.040	1,00
α	0°–5°	0°–5°	0°–5°	0°–5°

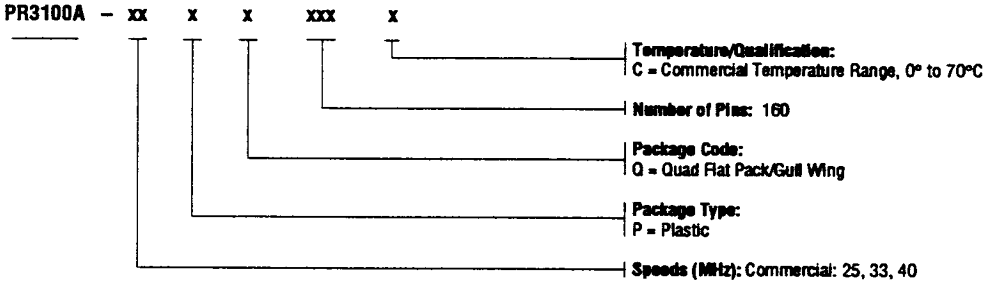
BSC = Basic Spacing between Centers

7.0 MOUNTING

A variety of sockets allow low insertion force or zero insertion force mountings, and a choice of terminals such as soldertail, surface mounts or wire warp. Several sockets are available from the following sample list of socket manufactures. Contact the manufacturer directly for the latest socket specifications.

- AMP Incorporated
  P.O. Box 3608
  Harrisburg, PA 17105-3608
  (800) 522-6752
- Burndy Corporation
  Richards Avenue
  Norwalk, CT 06856
  (203) 838-4444
- Yamaichi Electronics Inc.
  1425 Koll Circle, Suite 106
  San Jose, CA 95112
  (408) 452-0797
- Textool/3M Test
  and Interconnect Products Department
  3M Austin Center
  P.O. Box 2963
  Austin, TX 78769-2963
  (800) 225-5373

ORDERING INFORMATION



PR3100A – xx x x xxx x

Temperature/Qualification:
C = Commercial Temperature Range, 0° to 70°C

Number of Pins: 160

Package Code:
Q = Quad Flat Pack/Gull Wing

Package Type:
P = Plastic

Speeds (MHz): Commercial: 25, 33, 40