EK-369AA-OD-007B

TURBOchannel

Hardware Specification

On-line version.

Previous versions of this document are obsolete and should be discarded. This document supersedes all previous versions.

January 1993

TURBOchannel Specifications are controlled by the TURBOchannel Industry Group. For information about how to order Specifications call

Digital Equipment Corporation

Telephone: 508.493.7044
FAX: 508.493.3692

TURBOchannel Technology Transfer Agreement

Grant of Right to Use TURBOchannel Technology

In exchange for your agreeing to the warranty disclaimer and liability limitation stipulated in this Technology Transfer Agreement, Digital Equipment Corporation (Digital) grants at no cost to you a royalty-free nonexclusive license to use TURBOchannel technology (as specified in the *TURBOchannel Specifications*) to design and develop any kind of option board, computer system, or application-specific integrated circuit (ASIC). This Agreement does not grant you any other rights in Digital's patents, copyrights, trade secrets, trademarks, or licenses to TURBOchannel technology. The purchase cost of the TURBOchannel kit is basically the cost to reproduce the materials.

Warranty Disclaimer

The TURBOchannel technology is transferred "as is." Digital expressly disclaims all implied warranties including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Digital does not warrant, guarantee, or make any representations regarding the use of or the results of the use of the specification and related documents in terms of correctness, accuracy, or reliability. Digital believes the documentation is accurate; however, you must assume the risk as to the results and performance of any product you develop that is based on the TURBOchannel technology.

Limits of Liability

You agree that Digital shall not be liable to you under this Agreement for any damages, including without limitation any lost profits or lost savings, or any consequential, incidental, or punitive damages arising out of the use or inability to use the TURBOchannel Hardware Specification and related documents, or for any claim by another party. Your exclusive remedy under this Agreement shall be the furnishing by Digital of the technical support provided herein. You agree to hold Digital harmless for all claims and damages arising from any third party as a result of their use of or inability to use any product you develop based on TURBOchannel technology.

Important Changes to the TURBOchannel Hardware Specification

This version of the *TURBOchannel Hardware Specification* includes all changes made since the original specification was released. Revision bars mark technical changes added since Version 2C of the *TURBOchannel Specifications*.

Changes in Version 3

- **Protocol**: Clarification of protocol behavior.
- **Block I/O Write**: Allows system initiated transfers of entire data blocks to an option.
- **Parity**: Specifies optional parity implementation.
- **I/O Addressing**: This section has been enhanced to include byte masking for I/O read transactions. Figure 1-3, "Interpretation of the *ad* signal during an I/O read", has been replaced by Figure 1-4. Figure 1-4 has been renamed, "Interpretation of the *ad* signal during an I/O transaction".
- **Timing**: This section has been expanded to specify how the timing of signals are to be measured.
- **Implementation Notes**: This section has been expanded to clarify the TURBOchannel *ad* floating bus requirement.
- **Mechanical drawing changes.**
- **Removal of system parameters.**

Changes in Version 2C

- **I/O Read of an Option by the System**: This section has been added and contains a detailed explanation of "I/O read of an option by the system", Figure 1-1.
- **I/O Write of an Option by the System**: This section has been added and contains a detailed explanation of "I/O write of an option by the system", Figure 1-2.
- **DMA**: An explanation of DMA burst has been added to this section.

  The explanation of the use of *~err* has been expanded.
- **DMA Read of the System by an Option**: This section has been added and contains a detailed explanation of Figure 1-6, "One-word DMA read of the system by an option", and Figure 1-7, "Two-word DMA read of the system by an option", and
- **DMA Write to the System by an Option**: This section has been added and contains a detailed explanation of Figure 1-8, "One-word DMA write to the system by an option", and Figure 1-9, "Two-word DMA write to the system by an option".
- **Electrical**: Metric units have been added.
- **Timing**: This section has been expanded.
- **Implementation Notes**: Fahrenheit temperatures have been added.
- **Connector**: An explanation of pin assignment has been added.
- **Metric units:** Notation that units are in inches (and centimeters) have been added.

Changes in Version 2B

- **Table 1-3, Timing Requirements for TURBOchannel Signals**: Minimum times for *~rdy*, *~conflict*, *~int*, *~rReq*, *~wReq* have been revised to one nanosecond.
- **Implementation Notes**: This section has been updated to clarify that an *~err* signal is valid for an option only when the option has asserted the *~ack* signal.

Changes in Version 2A

- **DMA**: This section has been revised to show that deassertion of *~rReq* and *~wReq* may not occur until *~ack* is asserted by the system.
- **Interrupts**: This section now explains that when the *~int* signal has been asserted, the option may not deassert the signal until software dismisses the interrupt condition.
- **Implementation Notes**: This section has been changed to explain signal deassertion/assertion during reset and when the option is not selected.

Changes in Version 2.0

- **General Description**: A section has been added to explain that systems can time-multiplex multiple option slots onto a single memory port, or dedicate a memory port to each option slot. Note that systems dedicating a memory port to each slot may have clock frequency, clock phase, and slot address space discontinuities between slots.
- **Table 1-1, TURBOchannel Signals**: This table has been revised to show that the *~err* signal is used for all forms of DMA error. The previous description did not list parity errors.
- **I/O Transactions**: This section has been revised to specify a timeout period of at least 10 microseconds. The 5- to 10-microsecond range must be eliminated for compatibility with early TURBOchannel option designs.
- **I/O Transactions**: This section now shows that there may be intervening I/O transactions before a conflicting I/O transaction is reissued. This may occur in systems that do not preserve ordering of processor read and write transactions. For example, if an I/O write from a write buffer is conflicting, a higher priority I/O read, posted after the write, may be issued before the write is reissued. A similar situation is possible when I/O writes are issued before conflicting I/O reads are reissued.
- **I/O Addressing**: This section has been corrected to show 27 address bits.
- **I/O Addressing**: This section now explains that ordering of physical slots in the address space is system-specific, and that there may be address-space gaps between slots. This is necessary for implementations with TURBOchannel slots connected through intermediate adapter logic that may itself require some address space.
- **Figure 1-6, One-word DMA Read of an Option by the System**: The *~ack* signal cannot reassert until t6 to provide at least one deassertion cycle after data burst.
- **DMA**: This section now explains that systems with parity checking enabled assert *~err* if incorrect parity is detected on *ad*.
- **Parity**: A section has been deleted to correct an editorial error.

- **Interrupts**: This section was added to explain that interrupts are level-sensitive and that slot priority is system-specific.
- **Clock**: A section has been added to explain that clock phase and frequency variations may exist between different option slots. This may be the case for systems that connect TURBOchannel slots to an asynchronous memory subsystem through repeater logic, or to a memory subsystem through a crossbar interconnect.
- **Power**: This section now shows the metric values for the airflow specification and that airflow obstructions must be uniformly distributed to avoid creation of airflow dead zones.
- **ROM**: ROM information has been reduced in this specification, and a reference to the *TURBOchannel Firmware Specification* has been added.
- **Mechanical Overview**: This section now explains that physical slot numbering is system-specific.
- **Airflow**: Airflow information has been changed to show that it may be from either left-to-right or right-to-left. The bulkhead specification has been changed to show that its dimensions are the maximum usable connector area.

Conventions Used in This Specification

The Hardware Specification uses these conventions:

- *Terms in italic type like this* show TURBOchannel signals.
- Square brackets ([ ]) surround address ranges (expressed in bits).
- A tilde (~) precedes signals that are active-low.
- Signals not preceded by a tilde are active-high.

Contents

1 TURBOchannel Hardware Specification

Figures

Tables

1

TURBOchannel Hardware Specification

General Description

The TURBOchannel is a synchronous, asymmetrical I/O channel. The beginning of a TURBOchannel cycle is defined by the rising edge of the channel clock signal (*clk*); all signals are specified with respect to that clock edge. A TURBOchannel can be operated at any fixed frequency in the range of 12.5 to 25 MHz.

The TURBOchannel is asymmetrical: One system module and some number of option modules connect to the TURBOchannel. Typically, the system module contains the main memory system and the processor, and the option modules contain controllers for peripheral devices.

Two kinds of transactions are permitted on the TURBOchannel:

- The system module can read or write to an option module; this is an an I/O transaction.
- An option module can read or write to the system module; this is a DMA transaction.

An option module cannot address another option module on the TURBOchannel.

Systems can time-multiplex multiple option slots onto a single memory port to share its bandwidth, or dedicate a memory port to each option slot.

Signals

Table 1-1 lists the source and function of each signal on the TURBOchannel. Signals preceded by a tilde (~) are active-low. Signals not preceded by a tilde are active-high.

The meaning of the *ad* lines depends on the type of transaction and the transaction phase (either address or data).

Table 1-1. TURBOchannel Signals

Signal Name	Source	Function
ad[P, 31..0]	bussed	Address/data bus
~sel	system	I/O read/write select
~write	system	I/O read/write specifier
~ack	system	DMA read/write acknowledge
~err	system	DMA error
~reset	system	System reset
clk	system	Channel clock
~rdy	option	I/O read/write ready
~conflict	option	I/O read/write conflict
~rReq	option	DMA read request
~wReq	option	DMA write request
~int	option	I/O interrupt

Transactions

The minimum length of a TURBOchannel transaction is two clock cycles. Some system implementations may be able to achieve back-to-back transactions. Back-to-back I/O transactions to the same option must have an inter-transaction cycle between them, guaranteeing the deassertion of *~sel* to that option. Other implementations may insert extra idle cycles between transactions.

Inter-transaction Cycle

An inter-transaction cycle is defined as a cycle when the system and the option complete the last cycle of an I/O or DMA transaction.

The system

- May drive the *ad* lines to valid voltage levels following a DMA read, or an I/O write.
- Must not drive the *ad* lines following a DMA write or an I/O read transaction.
- Deasserts *~sel* following an I/O transaction.
- May assert *~sel* following a DMA read transaction.
- Deasserts *~ack* at the end of a DMA transaction.
- May assert *~ack* at the end of an I/O transaction.

The option

- Stops driving the *ad* lines.
- Deasserts *~conflict*.
- Deasserts *~rdy*.

Idle Cycles

An idle cycle is defined as a cycle when

The system

- Drives the *ad* lines to valid voltage levels.
- Maintains deassertion *~sel*.
- Maintains deassertion *~ack*.

I/O Transactions

Processor load/store instructions to the I/O slot address range generate I/O read/write transactions. As the system drives the address onto the TURBOchannel, the system decodes the address and asserts a select signal (*~sel*) for the specified option slot.

For read transactions, the system waits for the slot to drive data onto the TURBOchannel and assert the ready signal (*~rdy*). Figure 1-1 (page 5) shows signal activity when a system performs an I/O read of an option. The section "I/O Read of an Option by the System" (page 6) contains a detailed explanation of Figure 1-1.

For write transactions, the system drives data onto the TURBOchannel until the option asserts the ready signal. Figure 1-2 (page 8) shows signal activity when a system performs an I/O write to an option. The section "I/O Write of an Option by the System" (page 9) contains a detailed explanation of Figure 1-2.

Options should minimize ready assertion latency to maximize channel utilization. If the option does not respond within the system-specified timeout period, the system aborts the transaction. The timeout period must be listed in the system guide and must be at least 10 microseconds.

The system could select an option for an I/O transaction when the option is already committed to a DMA transaction. The option can

- Assert the *~conflict* signal.

  Because the *~conflict* signal affects system performance and interrupt service latency, this signal should be used only when there is no other means of breaking a deadlock. The *~conflict* signal must be asserted with *~rdy* and is used only when the option is selected.

- Not respond and let the system time out. The system could retry the I/O transaction at a future time. Other I/O transactions can occur before a conflicting I/O transaction is reissued.

A variant of I/O write, called "Block I/O Write", is an optional part of the TURBOchannel architecture. Up to sixty-four 32-bit words may be transferred in one Block I/O Write transaction. The minimum block size for a Block I/O Write transaction is two 32-bit words.

To implement Block I/O Writes, a *system* must:

- Specify in its documentation whether Block I/O Writes are implemented.
- Specify the maximum block size supported (up to the architecture limit of 64 words).
- Provide a mechanism for system firmware or software to enable a Block I/O Write on a per-slot basis.
- Provide a mechanism for system firmware or software to limit a system's Block I/O Write block size to that handled by an option.
- Disable the Block I/O Write function on power up. System firmware will decide if Block I/O Write capability is to be enabled.

The "TURBOchannel Firmware Specification" contains mechanisms for controlling Block I/O Writes.

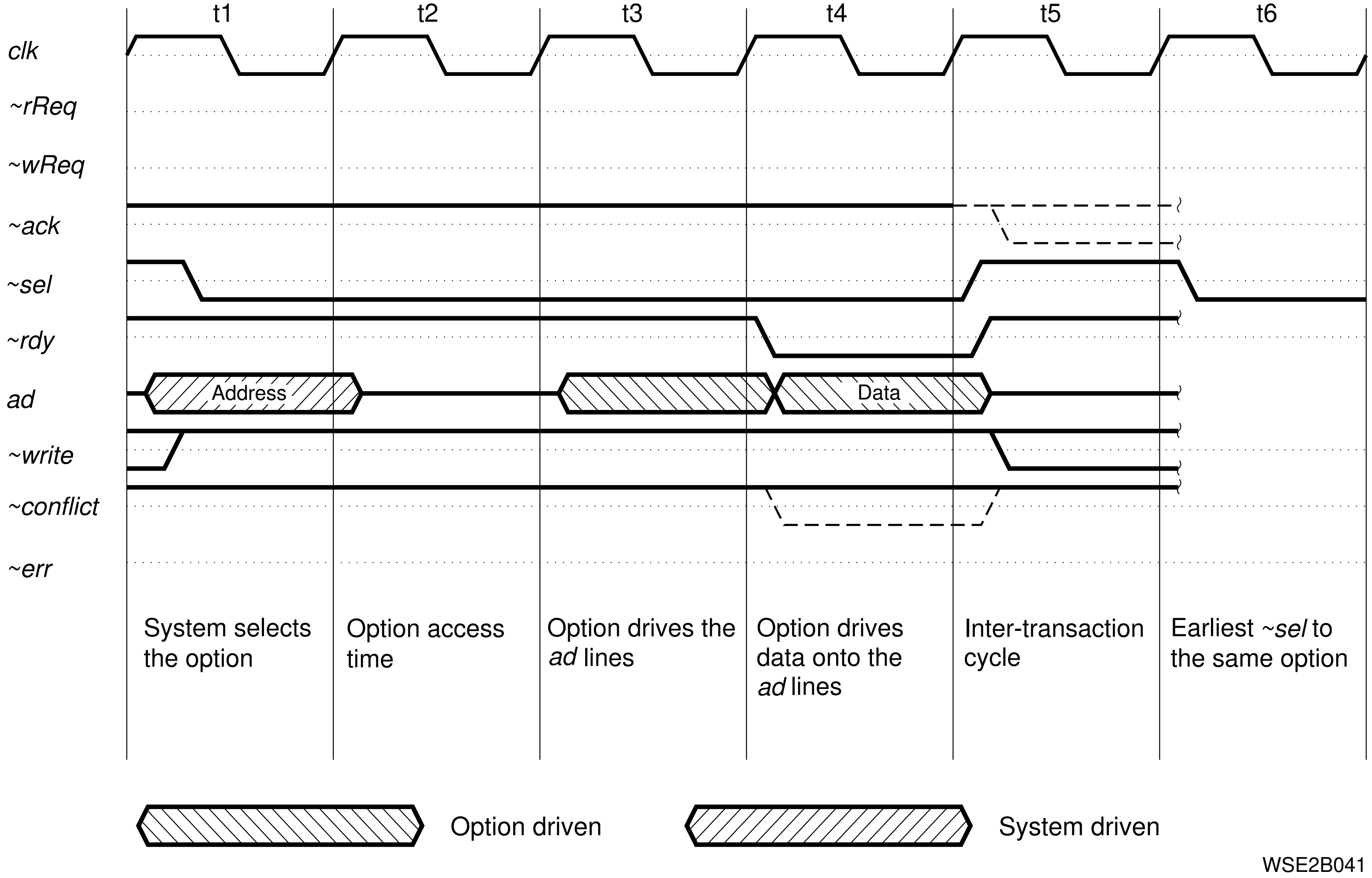
To implement Block I/O Writes, an *option* must:

- Specify in its documentation whether Block I/O Writes are implemented.
- Specify the maximum block size supported (up to the architectural limit of 64 words).
- Operate in systems that do not implement Block I/O Writes.
- Operate with any combination of I/O writes and Block I/O Write transactions.
- Receive data transferred in a Block I/O Write in the order sent by the system.
- Disable the Block I/O Write function on power up. System firmware will decide if Block I/O Write capability is to be enabled.

Systems must be designed such that software applications do not have to behave differently in order to take advantage of Block I/O Writes. A software driver executing on a system that does not implement Block I/O Writes should be identical to a software driver executing on a system that does implement Block I/O Writes.

The operating system or firmware must determine what option slots support Block I/O Writes and set system register bits accordingly.

Figure 1-1. I/O read of an option by the system



I/O Read of an Option by the System

This section contains a detailed explanation of Figure 1-1.

- **Cycle t1**

  The system selects the option. The system

  – Drives an address onto the *ad* lines. Bits 1, 2, 3, and 4 of the address form a byte mask; each bit indicates expected data validity in the corresponding byte lanes of the 32-bit data word to follow. If a byte mask bit is 1, the corresponding byte lane will **not** be used from the addressed word.

  – Asserts *~sel* to the selected option.

  – Deasserts *~write* to signal a read function to the option.

- **Cycle t2**

  Option access time. The option

  – Stores the system specified address from the *ad* lines.

  – Begins retrieving data at the address that was specified.

- **Cycle t3**

  This cycle does not exist if the option is fast enough to retrieve and drive valid data onto the *ad* lines immediately following cycle t2.

  If the cycle exists, the option must drive the *ad* lines to valid voltage levels.

  If the cycle exists, it repeats until one of the following conditions occurs:

  – The option is ready with data.

  – A system-defined timeout takes place and the system aborts the transaction (t3 cycles repeat until a system timeout occurs).

    The particular cycle in which a system-defined timeout occurs becomes the inter-transaction cycle, t5.

- **Cycle t4**

  Valid data is available during this cycle. The option

  – Asserts *~rdy*.

  – Drives valid data onto the *ad* lines.

  If the option is unable to perform the I/O read function, the option must

  – Assert *~conflict*.

  – Assert *~rdy*.

  – Drive the *ad* lines to valid voltage levels.

    EXAMPLE: The option is committed to a DMA transaction.

- **Cycle t5**

  This is an inter-transaction cycle. The system

  – Recognizes assertion of the *~rdy* signal.

  – Stores the option data from the *ad* lines.

- Deasserts ~*sel*.
- May assert ~*ack* for a pending DMA request by this option.

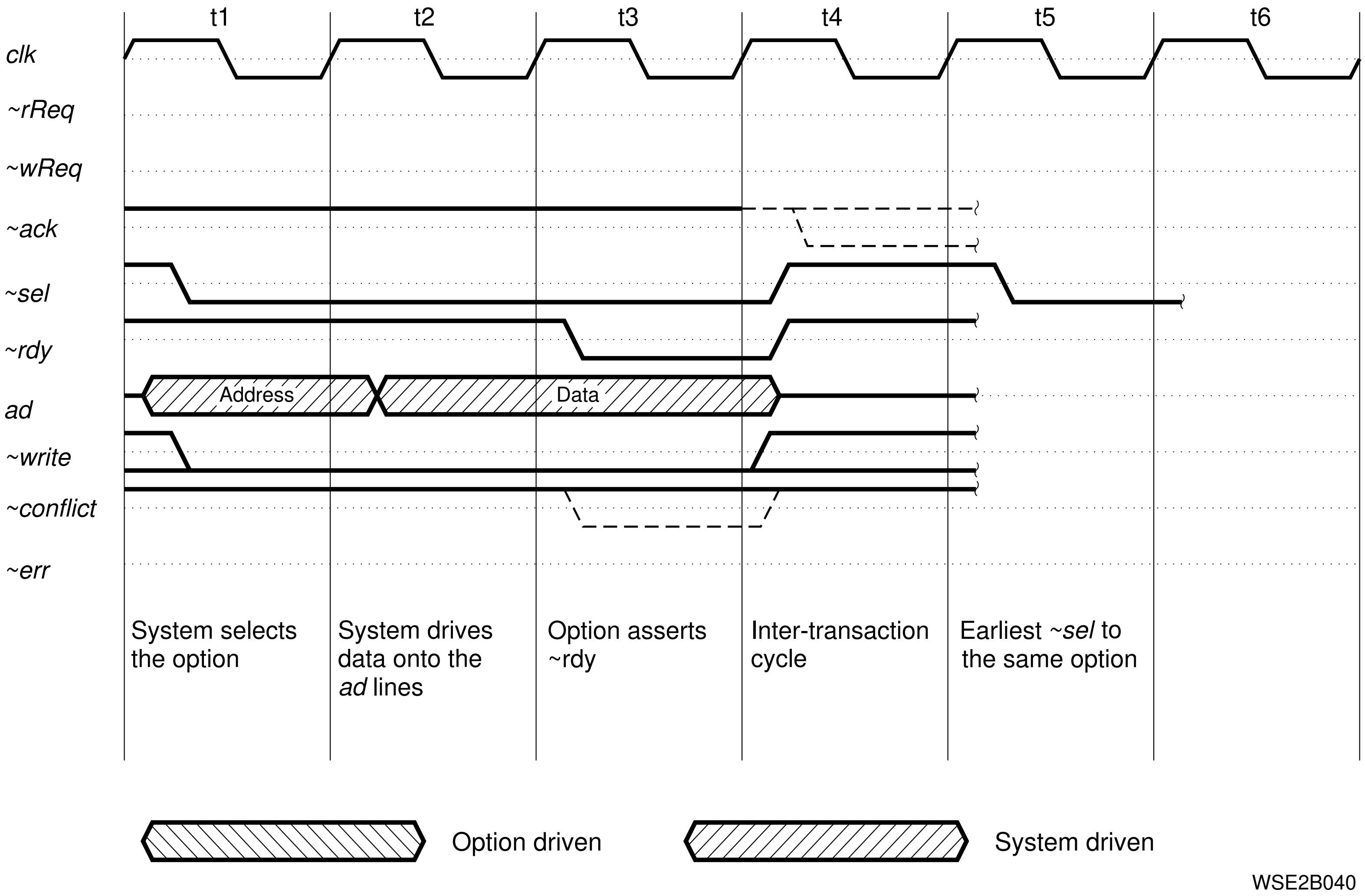
The option

- Stops driving the *ad* lines.
- Deasserts ~*rdy*.
- Deasserts ~*conflict* (if it was asserted in cycle t4).

- **Cycle t6**

  This is the earliest time ~*sel* can be asserted to the same option.

Figure 1-2. I/O write of an option by the system



I/O Write of an Option by the System

This section contains a detailed explanation of Figure 1-2.

- **Cycle t1**

  The system selects the option. The system

  - Drives the address onto the *ad* lines. Bits 1, 2, 3, and 4 of the address form a byte mask; each bit indicates data validity in the corresponding byte lanes of the 32 bit data word to follow. If a byte mask bit is 1, the corresponding byte lane is **not** stored in the addressed word.
  - Asserts ~*sel* to the selected option.
  - Asserts ~*write* to signal a write function to the option.

- **Cycle t2**

  This cycle does not exist if the option is fast enough to accept system data from the *ad* lines immediately following cycle t1.

  If the cycle exists

  - The option must store the system supplied address in the cycle following cycle t1.
  - The system drives valid data on the *ad* lines.

  If the cycle exists, it repeats until one of the following conditions occurs:

  - The option is ready to receive system data.
  - A system defined timeout takes place and the system aborts the transaction (t2 cycles repeat until a system timeout occurs).

    The particular cycle in which a system defined timeout occurs becomes the inter-transaction cycle, t4.

- **Cycle t3**

  The system drives valid data on the *ad* lines.

  The option

  - Stores the system specified address from the *ad* lines **if** this cycle immediately follows cycle t1.
  - Asserts ~*rdy*.

  If the option is unable to perform the I/O write function, the option must

  - Assert ~*conflict*.
  - Assert ~*rdy*.

    EXAMPLE: The option is committed to a DMA transaction.

- **Cycle t4**

  This is an inter-transaction cycle. Having recognized the assertion of the ~*rdy* signal, the system

  - Deasserts ~*sel*.
  - Deasserts ~*write*.

- May assert *~ack* for a pending DMA request by this option.
- Stops driving the *ad* lines if *~ack* is asserted.

The option

- Stores the system data from the *ad* lines.
- Deasserts *~rdy*.
- Deasserts *~conflict* (if it was asserted in cycle t3).

■ **Cycle t5**

This is the earliest time *~sel* can be asserted to the same option.

Figure 1-3. Six Word Block I/O Write Example

Block I/O Write

A Block I/O Write makes use of a different behavior in the *~sel* and *~write* signals to indicate that a block of I/O write data is being sent to the option, instead of just one single 32-bit word. In a mechanism similar to **I/O Write to an Option by the System**, a single address is presented at the beginning of the transaction, then a "burst" of data cycles is sent with each subsequent data word implying an incremented address.

Block I/O Write to an Option by the System

This section describes the detailed signaling of the Block I/O Write transaction. Refer to the example in Figure 1-3.

- **Cycle t1**

  The system selects the option. The system

  - Drives the address onto the *ad* lines. Bits 1, 2, 3, and 4 of the address form a byte mask; each bit indicates data validity in the corresponding byte lanes of the 32 bit data word to follow. If a byte mask bit is 1, the corresponding byte lane is not stored in the addressed word. The single byte mask applies to all words of a Block I/O Write transaction.
  - Asserts *~sel* to the selected option.
  - Asserts *~write* to signal a write function to the option.

- **Cycle t2**

  This cycle does not exist if the option is fast enough to accept system data from the *ad* lines immediately following cycle t1.

  If the cycle exists

  - The option must store the system supplied address in the cycle immediately following cycle t1.
  - The system drives word 1 of the Block I/O Write burst onto the *ad* lines.

    The cycle repeats until one of the following conditions occurs:

    > The option is ready to receive system data.

    > A system defined timeout takes place and the system aborts the transaction (t2 cycles repeat until a system timeout occurs).

    The particular cycle in which a system defined timeout occurs becomes the inter-transaction cycle, t6.

- **Cycle t3**

  The system

  - Drives word 1 of the Block I/O Write burst onto the *ad* lines.
  - Maintains assertion of *~sel*.
  - Maintains assertion of *~write*.

The option

- Stores the system specified address from the *ad* lines **if** this cycle immediately follows cycle t1.
- Asserts *~rdy*.
- Asserts *~conflict* if the option is unable to complete the Block I/O Write function.

  EXAMPLE: *~conflict* must be asserted if

  > The option is committed to a DMA transaction.

  > The option is unable to accept a data transfer of the maximum Block I/O Write burst.

~conflict and *~rdy* are not allowed to be asserted during later cycles of a Block I/O Write transaction.

- **Cycle t4**

  This is a mandatory cycle. Having recognized the assertion of the *~rdy* signal, the system

  - Maintains assertion of *~sel*.
  - Maintains assertion of *~write*.
  - Drives the *ad* lines with legal voltage values. Data contained on the *ad* lines is unpredictable in this cycle.

  The option

  - Stores the system data (word 1 of the Block I/O Write burst) from the *ad* lines.
  - Deasserts *~rdy*.
  - Deasserts *~conflict* (if it was asserted in cycle t3).

    If the option asserted *~conflict* in cycle t3, this cycle will become the inter-transaction cycle as defined in cycle t6, thus terminating the Block I/O Write transaction.

- **Cycle t5**

  This cycle repeats until the system drives all of the remaining data words of the Block I/O Write burst.

  The system

  - Maintains assertion of *~sel*.
  - Maintains assertion of *~write*.
  - Drives the remaining data words of the Block I/O Write burst, one word per cycle, on the *ad* lines.

  The option detects that *~sel* and *~write* are not deasserted in this cycle and determines that this is a Block I/O Write transaction. The option

  - Maintains deassertion of *~rdy*.
  - Stores the system data from the *ad* lines if this cycle does not immediately follow cycle t4.

A Block I/O Write transaction burst size is indicated by the number of cycles that *~sel* and *~write* are asserted after *~rdy* is deasserted.

- **Cycle t6**

  This is an inter-transaction cycle. Having recognized that all data words of the Block I/O Write transaction have been sent, the system

  – Deasserts *~sel*.

  – Deasserts *~write*.

  – May assert *~ack* for a pending DMA request by this option.

  – Stops driving the *ad* lines if *~ack* is asserted.

  The option

  – Stores the system data, (the last word of the Block I/O Write burst), from the *ad* lines.

  – Deasserts *~rdy* (**if** *~conflict* was asserted in cycle t3).

- **Cycle t7**

  This is the earliest time that *~sel* can be asserted to the same option.

I/O Addressing

Each slot has a 4- to 512-megabyte address range for I/O transactions. The size of the slot address space is system-specific in power-of-two multiples. For systems with less than 512-megabyte spaces, the high-order address bits are undefined and must be treated as *don't care* values. Options should use the minimal number of low-order address bits to decode internal logic. Addresses are 27-bit word addresses; therefore, the least significant two address bits of the byte address are implicitly 0.

Figure 1-4 shows the interpretation of the *ad* signals during an I/O transaction. Bits *ad*[31..5] specify the word address within the slot space. Bits *ad*[4..1] specify the byte masks for the *ad* signals during the subsequent data cycle of the transaction. If a byte mask bit is one for an I/O write transaction, the corresponding byte lane is not stored in the addressed option word. If a byte mask bit is one for an I/O read transaction, the corresponding masked byte lane may contain invalid data. The option is only required to return valid data for unmasked byte lanes. However, the option must still drive the masked *ad* lines to logic one or zero; the *ad* lines corresponding to masked byte lanes must not be left tristated during the data cycle.

The number of slots, size of the slot space, and base address of the slots is system-specific. The order of physical option slots in the address space is also system-specific. The address space between adjacent physical option slots may have system-specific gaps. The address space must be documented in the system guide.

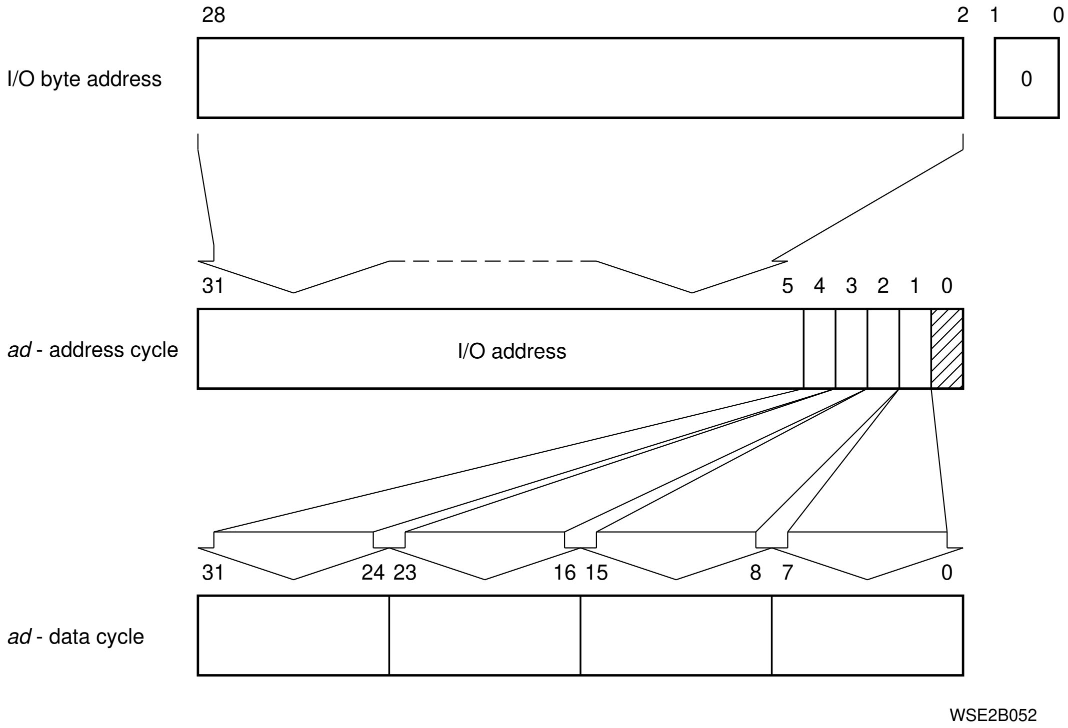


Figure 1-4. Interpretation of the *ad* signal during an I/O transaction

The address for a Block I/O Write is the address of the first word supplied in the write. Successive words are written to monotonically increasing (32 bit) word addresses. The address has the same format as normal I/O reads and writes, which means that the byte mask is applied to every word written.

DMA

All words in TURBOchannel DMA transactions are 32 bits wide. A DMA transaction can be any length up to an implementation-defined limit, which must be at least 64 words and a power-of-two multiple. The flow of two or more data words, one after the other, is called a DMA burst.

To obtain the best possible performance, data words are transmitted one per cycle. When the first DMA read data word is available, subsequent data words of the block are available (and must be accepted) in every cycle. DMA write data words are accepted (and must be supplied) in every cycle.

Request arbitration for DMA transactions is implementation-specific. Fixed-priority schemes will allow a specified slot to achieve full bandwidth; other schemes (such as fair service) are also allowed. The system guide must describe the arbitration scheme used.

An option requests a DMA transaction by asserting either the *~rReq* signal or the *~wReq* signal; these signals must not be asserted simultaneously. When the *~rReq* or *~wReq* signal has been asserted, the option may not deassert the signal until the system grants service by asserting the *~ack* signal. The option indicates the block length by continuing to assert *~rReq* or *~wReq* for as many cycles after *~ack* as the number of data words.

DMA transaction addresses have a 16-Gigabyte address space (34 address bits). Bits *ad*[4..0] are the upper five address bits. Bits *ad*[31..5] contain the rest of the word address. The two low-order address bits are implicitly 0. A system does not always support the entire address space; the consequences of addressing nonexistent memory in a DMA transaction is implementation-dependent. Figure 1-5 (page 17) shows a DMA transaction *ad* signal.

During the data cycle of a DMA transaction TURBOchannel transmits a word of data in every channel cycle; this high level of performance may require some implementations to use page-mode accesses to the system memory dynamic RAMs. DMA transactions cannot cross 2048-byte address boundaries.

The system asserts the *~err* signal when the system cannot successfully complete the requested DMA transaction. For example

- When an uncorrectable memory error occurs during a DMA read transaction; the system also asserts the uncorrectable data.
- When a memory error occurs during a DMA write transaction.
- When an option requests more than the maximum block size for a DMA read or write transaction
- When parity checking is enabled and the system detects incorrect parity on the *ad* signals.

The option must terminate its DMA transaction on the cycle immediately after the *~err* signal is asserted.

The *~ack* signal can remain asserted for several cycles when DMA read errors occur (for example, when memory is pipelined). Option logic must be designed so that an *~ack* signal from an aborted transfer cannot be interpreted as an *~ack* signal for a subsequent DMA request.

The signals *~ack* and *~sel* must never be asserted to the same option at the same time.

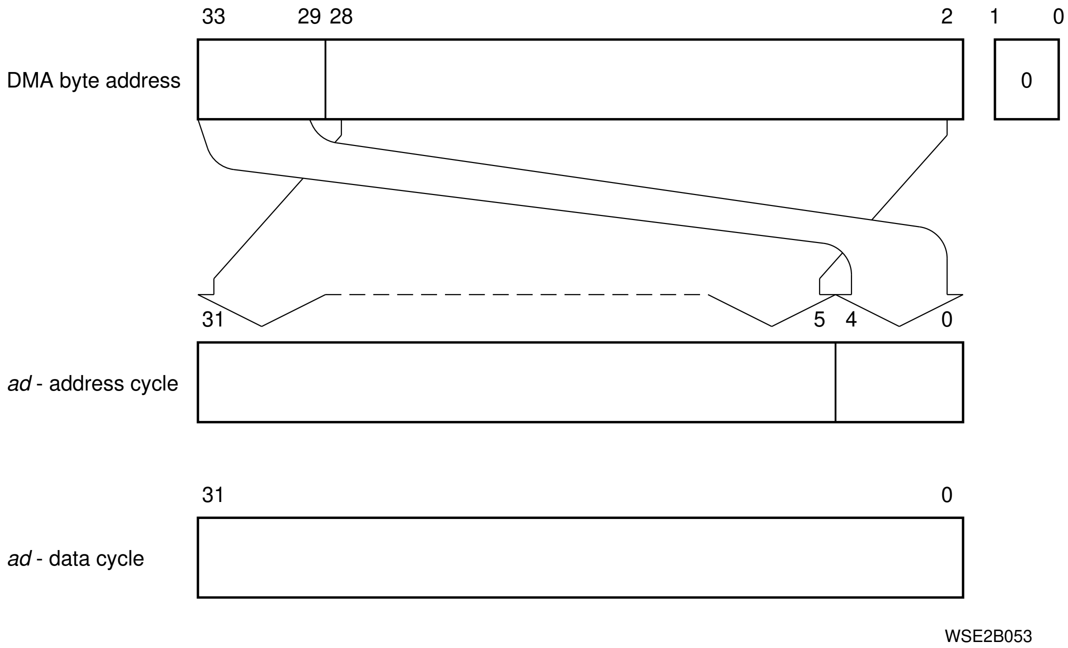


Figure 1-5. Interpretation of the *ad* signal during a DMA transaction

Figures 1-6 and 1-7 (pages 19 and 20) show signal activity when an option performs a one- or two-word DMA read of the system. The section "DMA Read of the System by an Option" (page 23) contains a detailed explanation of Figures 1-6 and 1-7. Figures 1-8 and 1-9 (pages 21 and 22) show signal activity when an option performs a one- or two-word DMA write to the system. The section "DMA Write to the System by an Option" (page 24) contains a detailed explanation of Figures 1-8 and 1-9.

Figure 1-6. One-word DMA read of the system by an option

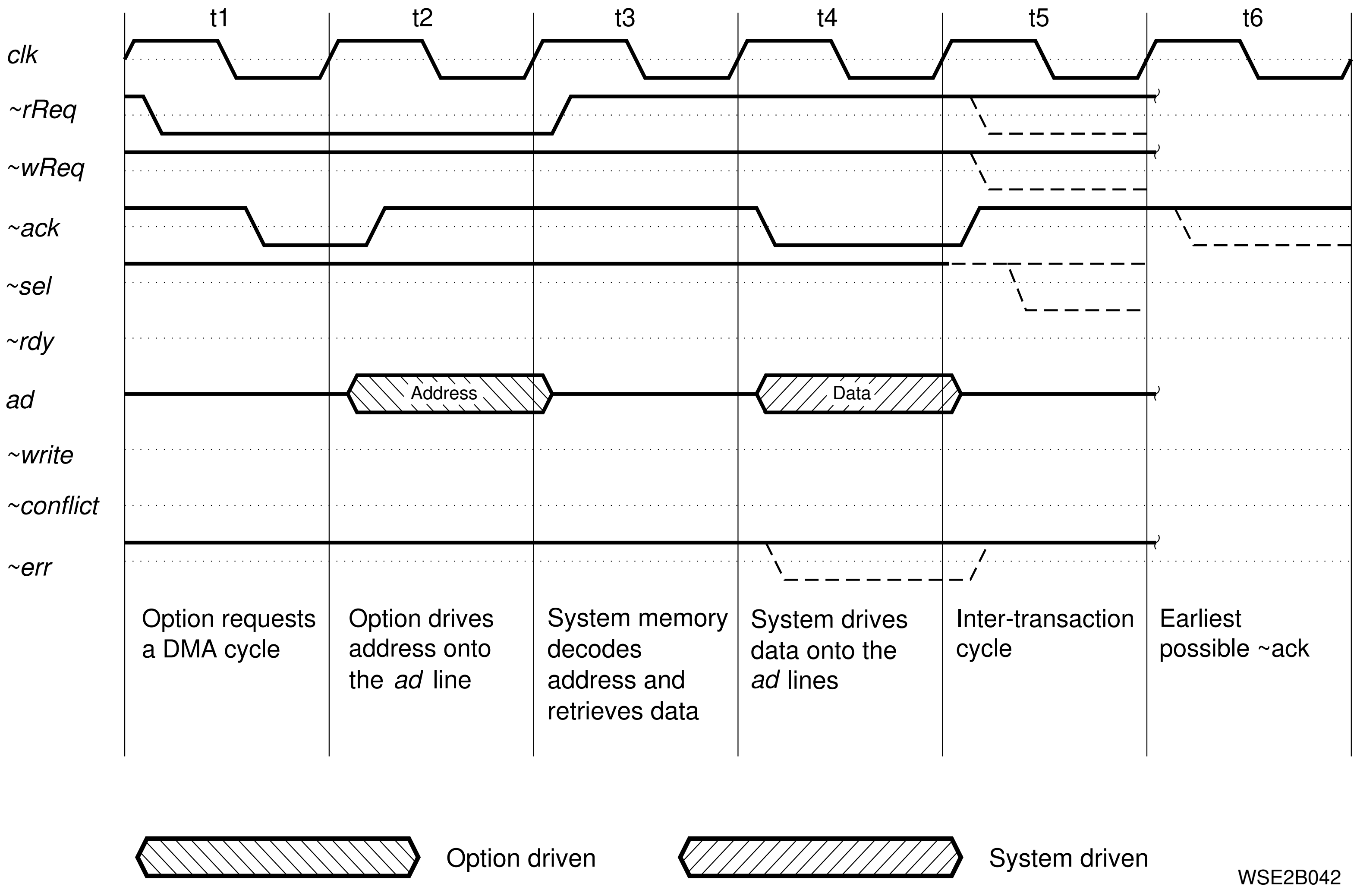


Figure 1-7. Two-word DMA read of the system by an option

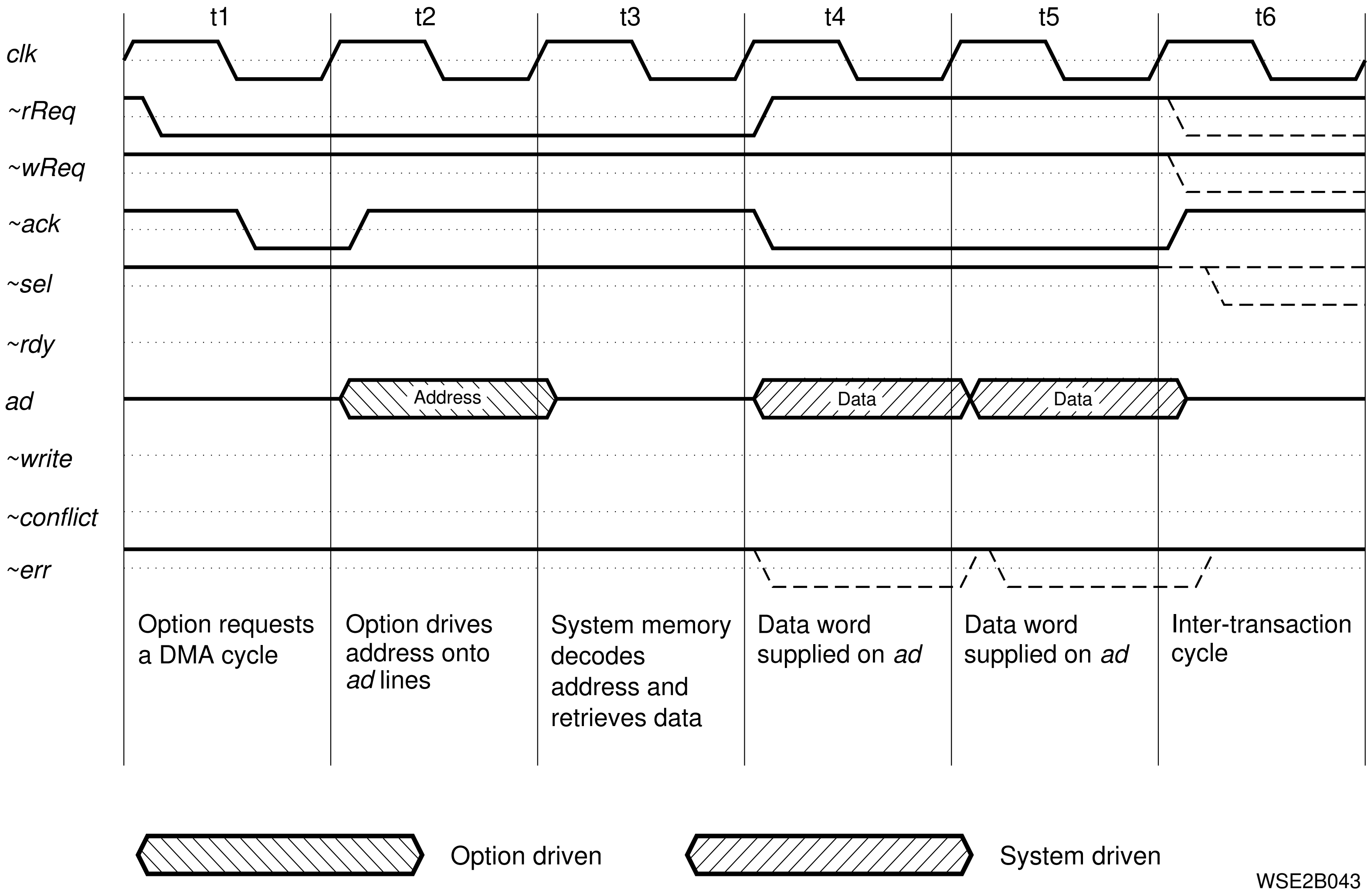


Figure 1-8. One-word DMA write to the system by an option

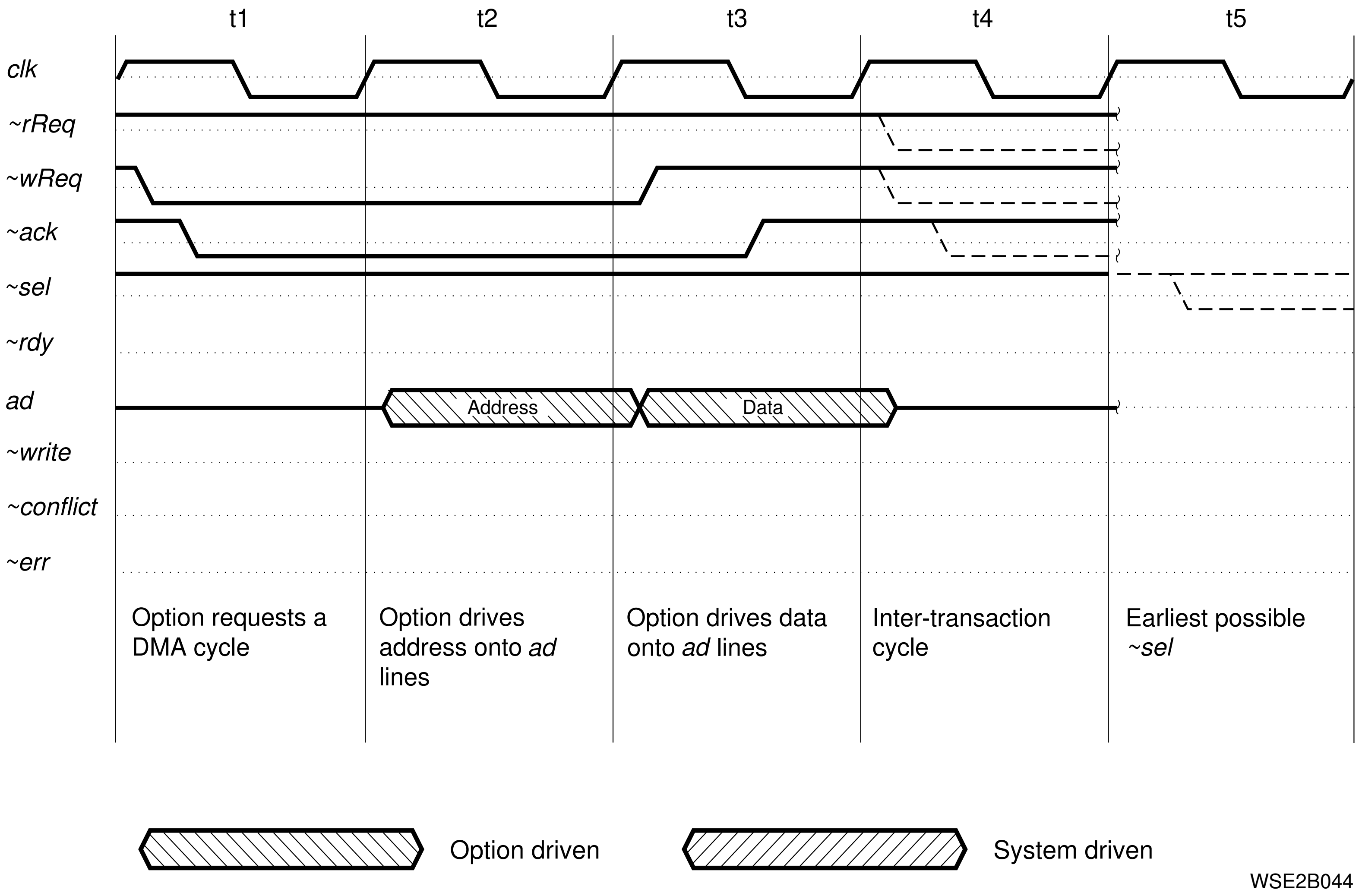
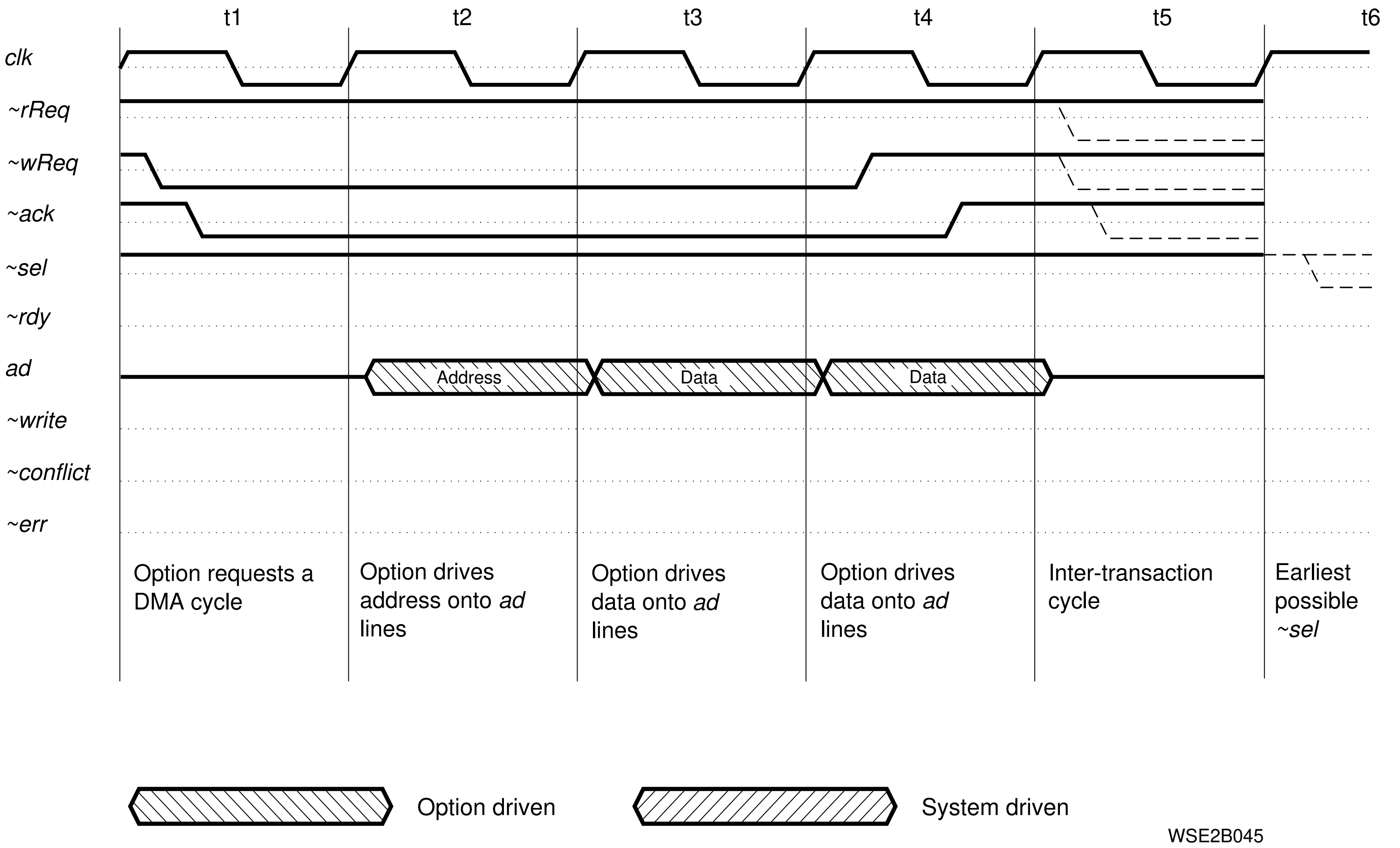


Figure 1-9. Two-word DMA write to the system by an option



DMA Read of the System by an Option

This section contains a detailed explanation of Figures 1-6 and 1-7.

- **Cycle t1**
  - The option asserts *~rReq* and waits for *~ack* from the system.
  - The system senses *~rReq* and could react quickly enough to assert *~ack* within the same cycle, or could choose to wait many cycles before asserting *~ack*.

  Once an option asserts a DMA request, either *~rReq* or *~wReq*, the signal must remain asserted until the system responds. An option must never assert *~rReq* and *~wReq* simultaneously.
- **Cycle t2**
  - The option drives the address onto the *~ad* lines in response to *~ack* from the system.
  - The system deasserts *~ack*.
  - The option continues to assert *~rReq* for as many cycles after *~ack* was sensed as the number of words of data that are being requested.
- **Cycle t3**
  - System memory decodes the address and retrieves the data. The number of cycles required for this decode and retrieval of data depends upon the speed of system memory, also know as memory access latency (this may be many t3 cycles).
  - If only one word of data is being requested, the option deasserts *~rReq*.
- **Cycle t4**

  The system
  - Drives the first data word onto the *ad* lines.
  - Reasserts *~ack*.
  - Asserts *~err* if an uncorrectable error occurs.
- **Cycle t5**

  The option
  - Continues to sense *~ack* and note the bits on the *ad* lines.
  - Stores a snapshot of the data to registers in response to the assertion of *~ack*.

  If a single word of data was requested, this is an inter-transaction cycle.
  - The system deasserts *~ack*.
  - This is the earliest time that the option could assert *~rReq* or *~wReq*.
  - This is the earliest possible time that the system could assert *~sel*.

  If two or more words of data were requested, the system
  - Drives them onto the *ad* lines one word per cycle and continues to assert *~ack* for every cycle of data, one 32-bit word per cycle.
  - Asserts *~err* if an uncorrectable error occurs.

The system must be able to place data words onto the bus, and the option must be able to accept those words every cycle in succession.

- **Cycle t6**

  If a single word of data was requested, this is the earliest time that the system can assert *~ack*.

  If two words of data were requested, this is an inter-transaction cycle.

  – The system deasserts *~ack*.

  – This is the earliest time that the option could assert *~rReq* or *~wReq*.

  – This is the earliest possible time that the system could assert *~sel*.

DMA Write to the System by an Option

This section contains a detailed explanation of Figures 1-8 and 1-9.

- **Cycle t1**

  – The option asserts *~wReq* and starts the wait for *~ack* from the system.

  – The system could assert *~ack* within the same cycle, or could choose to wait many cycles before asserting *~ack*.

  Once an option asserts a DMA request, either *~rReq* or *~wReq*, the signal must remain asserted until the system responds. An option must never assert *~rReq* and *~wReq* simultaneously.

- **Cycle t2**

  The option

  – Drives the address onto the *~ad* lines in response to *~ack* from the system.

  – Continues to assert *~wReq* for as many cycles after *~ack* was sensed as the number of words of data that are being requested.

- **Cycle t3**

  – System memory decodes the address and prepares to accept data.

  – The option drives the first data word onto the *ad* lines.

  If only one word of data is to be written to memory,

  – The option deasserts *~wReq*.

  – The system senses *~wReq*; if *~wReq* is deasserted, the system deasserts *~ack*.

  If more than one word of data is to be written to memory (a DMA burst),

  – The option continues to assert *~wReq*.

  – The system continues to assert *~ack*.

- **Cycle t4**

  The system reads the data on the *ad* lines.

  If the write transaction is a two-word write,

  – The option deasserts *~wReq*.

- The system senses *~wReq*; if *~wReq* is deasserted, the system deasserts *~ack*.

If the write transaction is a single word or if this is the final write cycle of a DMA burst, this is an inter-transaction cycle:

- The option can assert *~rReq* or *~wReq* and start a new transaction.
- The system can assert *~sel* or *~ack* in response to the *~rReq* or *~wReq* from the option.

■ **Cycle t5**

If the write transaction is a two-word write, this is an inter-transaction cycle:

- The option can assert *~rReq* or *~wReq* and start a new transact ion.
- The system can assert *~sel* or *~ack* in response to the *~rReq* or *~wReq* from the option.

Parity

Parity verifies the integrity of data transfer between options and systems. Parity is an optional feature that is implemented when determined as necessary by the system or option designer.

1. Systems and Options that support parity must supply valid parity when they drive the *ad* bus, once parity *checking* has been enabled by option firmware. Parity is checked only when valid data words or addresses are driven onto the *ad* bus; parity is not checked when the bus is driven for the sole purpose of preventing a floating bus condition. All 32 *ad* lines are used for parity generation and checking; parity on I/O operations is computed independently of the byte mask value. Odd-word parity is used; the number of ones in *ad*[P,31..0] must be an odd number.
2. When *~reset* is asserted, systems and options must disable parity checking of the *ad* bus.
3. Option firmware may enable parity generation and checking on the system side and option side of the TURBOchannel.
4. System and Option hardware will power-on to a parity checking disabled state.
5. System hardware must implement enabling/disabling of I/O slot interrupts *separate from* the enabling/disabling of parity error interrupts and *separate from* parity checking enable. The required hardware implementation is shown in Figure 1-10.

   The following sections discuss parity error response for the various transaction types, assuming that both the system and option have *ad* bus parity checking enabled.

Figure 1-10. Hardware parity implementation

I/O Reads. There is no mechanism for an option to return an explicit error acknowledgment if it detects a parity error in the slot address/mask. Hence, the option should generate a parity error interrupt and let the I/O read transaction time out. The option TURBOchannel interface should not issue the read to its internal logic.

If the system detects a parity error in the I/O read data, it should generate a parity error interrupt and signal a bus error to the processor.

I/O Writes. There is no mechanism for an option to return an explicit error acknowledgment if it detects a parity error in the slot address/mask or data. Hence, the option should generate a parity error interrupt. It is also recommended that the option TURBOchannel interface ignore the write data and not issue the write to its internal logic.

DMA Reads. There is no mechanism for the system to explicitly signal a parity error of the DMA address. Hence, the system should assert *~ack* and *~err* for a period of one cycle following the address transfer. The system TURBOchannel interface should generate a parity error interrupt. It is recommended that the system TURBOchannel interface not issue the read to the memory system. The option must terminate its DMA transaction on the cycle immediately after the *~err* signal is asserted. The option TURBOchannel interface should generate a DMA error interrupt. The option TURBOchannel interface should indicate an error to its internal logic.

If an option TURBOchannel interface detects a parity data error, it should generate an interrupt, abort the transaction, and indicate an error to its internal logic.

DMA Writes. There is no mechanism for the system to explicitly signal a parity error of the DMA address or data. Hence, the system TURBOchannel interface should generate a parity error interrupt. The system TURBOchannel interface should not write the affected word(s) to the memory system.

Hardware Error Registers. It is recommended that parity options and systems capture the transaction type (I/O versus DMA), the transaction direction (read versus write), the transaction phase (address versus data), and the *ad* bus value when they detect a parity error.

Interrupts

Interrupts are level-sensitive. Slot priority is determined by system hardware and software. When the option has asserted *~int*, the signal may not be deasserted until software dismisses the interrupt condition.

Electrical

For bussed signals (*ad*[P, 31..0]) on a TURBOchannel the system module and option modules must present a total capacitive load of no more than 180 pF. The total trace length for a bussed signal on the system module cannot exceed 16 inches (40.64 centimeters). An option module must present a capacitive load of no more than 20 pF for any bussed signal. The total trace length for a bussed signal on an option module cannot exceed 2 inches (5.08 centimeters). Capacitance measurements must include all connectors, components, lands, and traces.

The load imposed by an option module for system-generated control signals cannot exceed 50 pF, except for the clock signal, which is 100 pF. This must include the connector, all components, lands, and traces.

For option-generated control signals, the load imposed by the system module cannot exceed 65 pF including the connector, all components, lands, and traces.

An option that is more than one slot wide must connect all signals to one slot. However, the option can draw power from all connectors. This slot is identified in the *TURBOchannel Mechanical Drawings*.

Table 1-2 lists the DC parameters for TURBOchannel signals.

Table 1-2. DC Parameters for TURBOchannel Signals

Value	*ad* (P,31..0)	System-generated	Option-generated
Voh	2.4 V (minimum)	same	same
Vih	2.0 V (minimum)	same	same
Vil	0.8 V (maximum)	same	same
Vol	0.5 V (maximum)	same	same
Iih	+70 uA (maximum)	+0.5 mA (maximum)	+0.5 mA (maximum)
Iil	-70 uA (maximum)	-1.5 mA (maximum)	-1.5 mA (maximum)
Ioh	-1.0 mA (minimum)	same	same
Iol	+2.0 mA (minimum)	same	same

Timing

Setup and hold signal timing specifications for system-generated signals must be met by each option in the system. Signal timing specifications for option-driven signals are measured with full capacitive loading on all signals. All signal timing specifications are measured from the rising edge of the *clk* signal at the option.

All signal time measurements are made with respect to an input voltage threshold, Vti, and an output voltage threshold, Vto.

- Vti = 1.4v
- Vto = 1.4v

Table 1-3 lists timing requirements for each TURBOchannel signal. All times are in nanoseconds.

Table 1-3. Timing Requirements for TURBOchannel Signals

Signal	Source	Minimum	Maximum	Setup	Hold
ad	system			5	2
ad to tristate	system	3	22		
~sel, ~write, ~ack, ~err, ~reset	system			13	2
ad	option	3	34		
ad to tristate	option	3	22		
~rdy, ~conflict, ~int	option	1	12		
~rReq, ~wReq	option	1	7		

System-driven signal voltages are expected to be stable and valid from the setup through the hold time shown in Table 1-3. For system-driven signals, setup is the time before the rising edge of *clk*, hold is the time after the same rising edge of *clk*.

Option-driven signal voltages are expected to be stable and valid within the timing limits shown in Table 1-3. Minimum and maximum times are measured from the rising edge of *clk*.

Clock

The TURBOchannel uses a free-running clock (*clk*) at any fixed frequency from 12.5 to 25 MHz. Clock signal rise and fall times must not exceed 5 nanoseconds measured at the *clk* signal pin on the option connector. The clock signal must be high for at least 15 nanoseconds and low for at least 15 nanoseconds.

Figure 1-11 shows timing requirements for each phase of the *clk* signal.

Clock skew between the system module and option modules must be controlled; the system module must function correctly for option modules that meet timing specifications.

Systems can have option slots with different clock phase and frequency variations.

For an option module's *clk* signal, the module must have a diode terminator connected to ground and a diode terminator connected to the +5 volt supply.

Power

A TURBOchannel system module provides two supply voltages to each TURBOchannel option: +5 volt and +12 volt. Sequencing of the +5 volt and +12 volt supplies is not guaranteed.

Table 1-4 describes the voltage and current available to a TURBOchannel option.

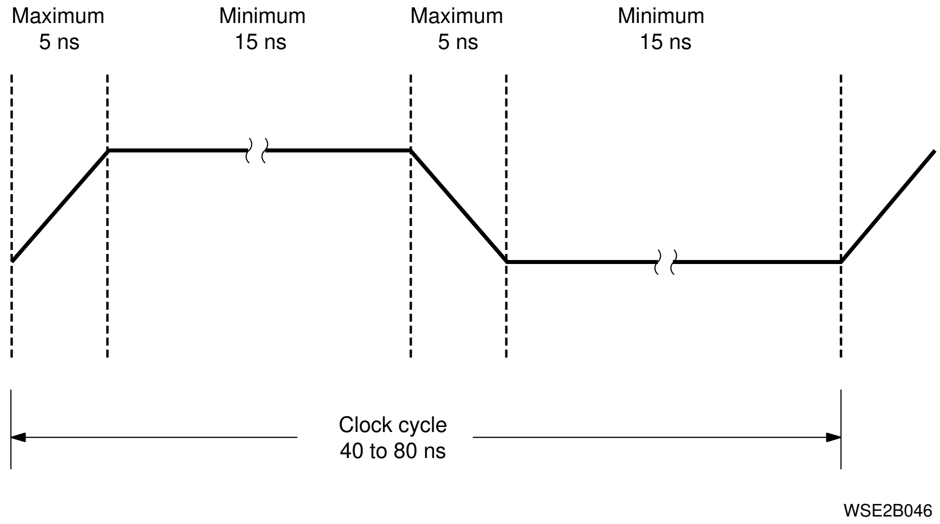


Figure 1-11. Timing requirements for the *clk* signal

Table 1-4. Power Available to TURBOchannel Options

	Maximum Current		
Voltage	**Single-Width Option**	**Double-Width Option**	**Triple-Width Option**
+5 V ±5%	4.0 A	8.0 A	12.0 A
+12 V ±5%	0.5 A	1.0 A	1.5 A

The *~reset* signal is asserted for at least 250 milliseconds after power is switched on and the +5 volt supply has become stable. The *~reset* signal is reasserted at least 500 microseconds before the +5 volt supply drops.

The system will maintain an airflow of at least 50 LFM (linear feet per minute) (25 cm/s (centimeters per second)) below an option module. The system will maintain an airflow of at least 150 LFM (76 cm/s) above an option module. A module's components or daughter cards must not obstruct more than 50 percent of the side-to-side cross-sectional area above the module. Module obstructions must be uniformly distributed so there are no downstream airflow dead zones.

Implementation Notes

During reset, an option must

- Deassert the *~rdy*, *~conflict*, *~rReq*, *~wReq*, and *~int* signals.
- Tristate the *ad* signals.

When an option is not selected, the option must not assert the *~rdy* or *~conflict* signals. Thus, a system need not qualify these signals with the option select line.

Table 1-5 shows which features an option must implement to connect to the corresponding signals:

Table 1-5. Option Implementation Required for Signal Connection

Features	Signal
parity	*ad*[P]
I/O conflicts	*~conflict*
DMA	*~wReq*, *~rReq*, *~ack*, *~err*
interrupts	*~int*

Options that do not have 32-bit internal data paths must still drive all 32 *ad* signals when supplying I/O read data.

Options that cannot address the entire 16-Gigabyte DMA address space must drive 0 onto high-order address signals.

Logic families selected to implement the system and option interfaces should use the slowest logic that meets the timing requirements of the TURBOchannel. Logic families should also use internal clamping to control signal reflections. Sufficient capacitance should be used to decouple the supply voltages on system and option modules.

Options should be designed

- To meet all specifications while operating in office environments at 10° to 40°C (50° to 104°F) ambient temperature with a 10°C (18°F) internal rise and 10% to 90% relative humidity.
- To meet appropriate international Class A electromagnetic interference regulations and international safety regulations.
- With appropriate electrostatic discharge protection for their application and for routine shipping and installation handling.

Systems that time-multiplex option slots onto a single memory port may distribute common *~write* and *~err* signals to multiple slots. Options can observe transitions of these signals during transactions to other option slots. The *~write* signal is only valid for an option when the option's *~sel* signal is asserted. The *~err* signal is only valid for an option when the option's *~ack* signal is asserted.

The *ad* bus should not be left floating for more than one TURBOchannel clock period. For example, when the TURBOchannel is idle, the system must drive the *ad*[P,31..0] lines. After an I/O read address has been sent and the bus has been turned around, the option must drive the *ad* lines until the valid data is ready. Referring to the I/O read timing diagram, Figure 1-1, the option must drive the *ad* bus during the t3 cycle(s). The *ad* bus will float during the t2 cycle to avoid bus contention.

After a DMA read address has been sent by the option and the bus has been turned around, the system must drive the *ad* lines until the DMA read data is ready.

Referring to the one-word DMA read timing diagram, 1-7, the system will drive the *ad* bus if additional t3 cycles are required due to memory access latency. The *ad* bus will float during the first t3 cycle to avoid bus contention.

TURBOchannel options and systems must drive the parity line to a valid logic state when driving the *ad* lines so that the parity line will not float. For systems and options that do not implement parity, the logic value of this parity line need not be the correct parity value.

Connector

The TURBOchannel uses a 96-pin DIN connector (DIN 41612). The system module uses female connectors; option modules use male connectors. Suggested connectors are shown in Table 1-6.

Table 1-6. Suggested TURBOchannel connectors

AMP	535043-4	Female	0.656" stack height
AMP	650640-4	Female	0.867" stack height
AMP	536142-4	Female	1.706" stack height
AMP	53253-1	Male	

Options must not connect to the four NC pins. Table 1-7 lists pin assignments for a TURBOchannel connector.

Note that row numbering for TURBOchannel connector pins does not correspond to the row numbering for the suggested connector. Column assignments do correlate. Pins A1 through A32, B1 through B32, and C1 through C32 for the suggested connector correspond to TURBOchannel pins A32 through A1, B32 through B1, and C32 through C1, respectively.

Table 1-7. TURBOchannel Connector Pin Assignments

Row	Column A	Column B	Column C
1	*ad*[31]	GND	*ad*[30]
2	*ad*[29]	GND	*ad*[28]
3	*ad*[27]	GND	*ad*[26]
4	*ad*[25]	GND	*ad*[24]
5	+5 V	GND	+5 V
6	*ad*[23]	GND	*ad*[22]
7	*ad*[21]	GND	*ad*[20]
8	*ad*[19]	GND	*ad*[18]
9	*ad*[17]	GND	*ad*[16]
10	+5 V	GND	+5 V
11	*ad*[15]	GND	*ad*[14]
12	*ad*[13]	GND	*ad*[12]
13	*ad*[11]	GND	*ad*[10]
14	*ad*[9]	GND	*ad*[8]
15	+5 V	GND	+5 V
16	*ad*[7]	GND	*ad*[6]
17	*ad*[5]	GND	*ad*[4]
18	*ad*[3]	GND	*ad*[2]
19	*ad*[1]	GND	*ad*[0]
20	+5 V	GND	+5 V
21	NC	GND	NC
22	NC	GND	NC
23	*~conflict*	GND	*ad*[P]
24	*~ack*	GND	*~err*
25	+5 V	GND	+12 V
26	*~rReq*	GND	*~wReq*
27	†	GND	*~write*
28	*~sel*	GND	*~rdy*
29	*~int*	GND	*~reset*
30	GND	GND	GND
31	GND	*clk*	GND
32	GND	GND	GND

†Row 27 column A is assigned to the *~iaCE* signal in prototype implementations. This signal is similar to *~sel* but is only asserted for the first cycle of the I/O transaction. Option modules should not use this signal.

ROM

A TURBOchannel option module ROM must contain information about itself and the identity of the option. The ROM may also contain additional option firmware. ROM requirements are defined in the *TURBOchannel Firmware Specification*.