DIGITAL'S TRI/ADD PROGRAM

A FULL SPECTRUM OF HARDWARE SOLUTIONS

TURBOchannel™
TECHNICAL OVERVIEW

Specifications for this open I/0 interconnect are offered without license or royalties

Version 3.0

digital™ Digital Equipment Corporation
100 Hamilton Avenue ; Palo Alto. CA U.S.A. 94301 ; [1] 415.853.6531 ; FAX [1] 415.853.0155 ; triadd@decwrl.dec.com

Australia	France	Germany	Italy	Japan	U.K.	U.S.A./Canada
0014.800.125.388	05.90.287-4	0130.81.197-4	167-8.19087-	0031.12.2363	0800.89.2610	1.800.678.OPEN

Order number: EK-TURBO-OW-001

1. Introduction

TURBOchannel is a high-performance, open I/O interconnect for desktop computers and servers designed by Digital Equipment Corporation. It is open in that Digital encourages other vendors to develop TURBOchannel-based systems and peripherals, and actively offers support to developers through its TRI/ADD Program. TURBOchannel is one of the two I/O interconnect options in the Advanced RISC Computer (ARC) specification of the Advanced Computing Environment initiative (ACE).

The TURBOchannel interconnect's high performance is the consequence of its innovative architecture, which is specialized and optimized for the I/O function. A TURBOchannel interconnect provides asymmetric communication between a single system module consisting of processors and memory, and several option modules, each typically a controller for some I/0 device or network interface. The TURBOchannel architecture departs from the pure bus topology that has heretofore been the norm for small computer interconnections. Instead, the TURBOchannel control signals have a radial point-to-point topology; a TURBOchannel-based system provides separate control lines for each of several peripheral option slots. The benefits of this architectural innovation are

- simple, efficient protocol
- low signal count
- low-latency transactions
- high bandwidth DMA block transfer
- simplified option module design

The TURBOchannel specification allows for variation in system implementations, providing a broad range of implementation cost and performance characteristics. Similarly, the specification allows for a wide range of cost and performance characteristics in the design of option modules as well. Of course, any option conforming to the specification will be compatible with any TURBOchannel system implementation.

The TURBOchannel specification includes a convention for storing optionspecific information in ROM on the option to support automatic configuration and initialization.

TURBOchannel's performance is high enough to support industry standard interconnects, such as VME, SCSI and FDDI, through appropriate adapter option modules.

This overview briefly describes the TURBOchannel specification, shows how certain benefits follow from the design innovations, and reviews the current state of development of TURBOchannel system implementations and options

2. TURBOchannel Description

The TURBOchannel interconnect is a synchronous asymmetrical 32-bit I/O channel. It connects a single system to a number of peripheral options. The system generally includes a processor and system memory. The options are usually peripheral device controllers, network interfaces, or adapters to other interconnects.

One of the TURBOchannel signals, clk, a system-generated clock, provides the synchronization. The clock may run at any fixed frequency between 12.5 and 25 MHz. To support DMA at 25 MHz, the system memory would need to have an interleaved architecture. The 12.5 MHZ rate would be appropriate for a system with a single bank of memory. Section 3 contains further discussion on the possible architectures of TURBOchannel implementations.

TURBOchannel is asymmetrical in that the system has a special role. Communication is always between the system and one of the options, never directly between two options.

The TURBOchannel control topology is radial point-to-point, as indicated in Figure 1. The system provides separate sets of control signals for each option. This topological feature distinguishes TURBOchannel from a general-purpose bus, in which control signals, data lines, and address lines are all party lines. The radial control topology obviates the need for much of the handshaking, address decoding, and contention resolution that are needed in a conventional bus protocol. Consequently, the TURBOchannel protocol has many fewer overhead cycles and so enjoys higher data throughput than a traditional symmetric bus based on the same technology. Two further consequences of the radial control topology are smaller signal counts per option and much simpler option logic; both of these contribute to reducing an option's cost.

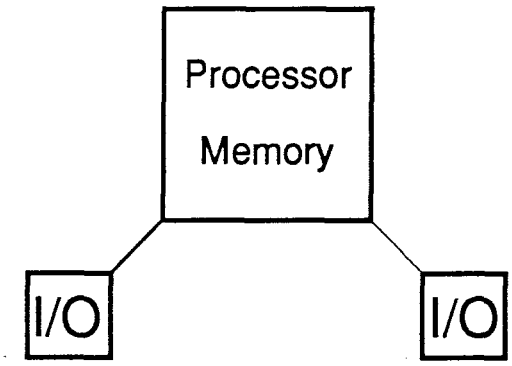

Figure 1. TURBOchannel radial topology

The TURBOchannel address/data path is 32 bits wide. These lines may be implemented either as a bus interconnect or as a point-to-point interconnect, separate for each option. In the bus case, the address/data path may be embedded in the primary system bus that connects the processor with memory, or it may be completely separate from the system bus. Implementation variations are discussed further in Section 3

2.1 TURBOchannel Transaction Categories

TURBOchannel transactions fall into two categories, each including both read and write transactions:

- I/O transactions: I/O read and I/O write
- DMA transactions: DMA read and DMA write

I/O transactions are driven by the system and transfer one 32-bit data word per transaction. In an I/O read transaction, the system processor reads from option registers or buffers; in an 1/0 write transaction, the system processor writes to the option registers or buffers. From the software point of view, I/0 transactions are initiated by processor load/store instructions to the I/0 portion of the address space. System hardware responds to such instructions by generating the control signal sequences appropriate for the transaction. The addressing is discussed in Section 2.3. Details of the I/O transaction signal activity are given in Section 2.4.

DMA transactions are driven by the option and generally transfer blocks of data between option buffers and system memory. In a DMA read transaction, the option reads from system memory; in a DMA write transaction, the option writes to system memory. The size of the data block in each DMA transaction is determined by the option, up to a system-defined limit.

2.2 TURBOchannel Signals

TURBOchannel uses just 44 signal pins for each option. These are shown in Table 1. In Table 1, the prefix ~ indicates signals that are active when low. ad[P, 31,.. 0] are the 32-bit address and data lines, together with an optional parity signal. TURBOchannel conventionally uses odd word parity. Parity is optional for both system and option, and, of course, can be effective only when both system and option implement it. The ad channel is multiplexed in the sense that it carries addresses and data at different parts of the transaction cycle

The clock signal is a trapezoid wave form at some fixed frequency between 12.5 and 25 MHz determined by the system implementation. The c1k signal is generated by the system and runs to every option module. ~reset is a system reset signal, which also runs to every option

The remaining control signals are option-specific. That is, they are dedicated lines running to each option slot. In an I/O transaction the system asserts –sel for the option it wishes to communicate with. For an I/O write, the system also asserts ~write; for an I/O read, ~write is not asserted. The option uses ~rdy to shake hands with the system on I/O transactions and uses ~conflict to avert an I/O transaction when a DMA transaction is pending. Details are given in Section 2.4.1

Table 1. TURBOchannel Signals

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 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

The option uses *~rReq* and *~wReq* to initiate DMA reads and writes, respectively. The system uses ~ack for handshaking on DMA transactions. Details are given in Section 2.4.2

The system uses *~err* to indicate non-correctable memory errors and parity errors when parity is enabled.

The option uses *~int* to interrupt the system processor. Options typically use interrupts to inform the system of the completion of high-latency I/O operations, such as reading a block from disk. The details of interrupt enabling, vectoring, and priority levels are system-dependent.

2.3 Addressing

A system implementation has a fixed number of option slots and assigns a fixed range of addresses of its physical address space to each slot. The slot address space may be between 4 and 512 Mbytes.

On the first clock cycle of an I/O transaction, the system drives a 27-bit address onto *ad[31..5]* This is understood as a word address, or as a 29-bit byte address with the two least significant bits implicitly zero. This address in *ad[31..5]* is actually an offset from the base of the option slot's address space. If the slot address space is assigned less than 512 Mbytes, the most significant bits of this address must be zero.

Byte addressability for I/O write transactions is provided through a byte mask in the *ad[4..1*]. If any of these four bits is one, then the corresponding byte of the addressed word is not written in the I/O write transaction. *ad[0]* is not used in I/O transaction addressing.

On the first cycle of a DMA transaction the option drives a 32-bit address onto the *ad*[31. *0*]; the five most significant bits of the address are in *ad[4.. 0]*, the remaining bits in *ad[31.. 5]*. This 32-bit address is interpreted as a word address in a 16-Gbyte physical address space. It should lie within the physical addresses assigned to system memory. The consequences of addressing nonexistent memory in a DMA transaction is implementation-dependent.

2.4 Transaction Details

In order to illustrate the simplicity of the TURBOchannel protocol, this section presents the TURBOchannel signal activity for both I/O and DMA transactions.

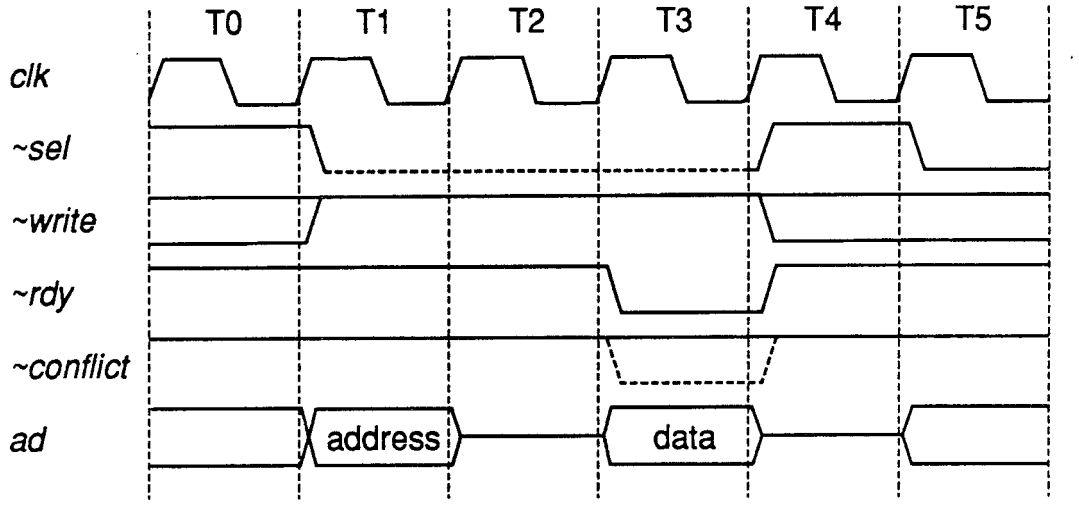

Figure 2. I/O Read Transaction

2.4.1 I/0 Transactions

Figure 2 shows the timing for an I/O read, and Figure 3 for an I/O write. Cycle 0 represents the fact that there must be at least one cycle of non-asserted *~sel* preceding an I/O transaction to any option module. In cycle 1, the system asserts *~sel*; for an I/O

write transaction, the system also asserts *~write*. Also in cycle 1, the system drives the appropriate I/O address, as described in Section 2.3, onto the *ad* channel. The address is held for just one cycle.

For an I/O write, the system drives the data onto the ad channel in cycle 2 and holds it until the option acknowledges completion of the write by asserting *~rdy*. In the case of an I/O read, the system waits for the option to drive the data onto the ad bus and to indicate the presence of data by asserting *~rdy*.

If the option is unable to respond to the I/O transaction because it is committed to a DMA transaction, then it responds to *~sel* by asserting *~conflict* along with *~rdy*.

If the option does not assert *~rdy* within a system-defined timeout period, the system aborts the transaction.

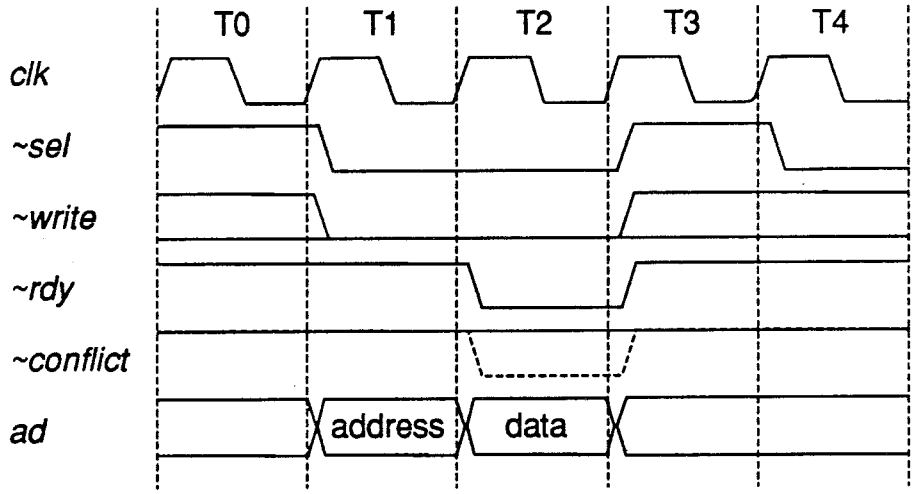

Figure 3. I/O Write Transaction

2.4.2 DMA transactions

TURBOchannel DMA transactions transfer blocks of 32-bit words; the blocks can be of any length up to a system-defined limit. The system-defined block-size limit must be a power of two, and at least 64 words. To economize on buffer space, an option may use small DMA blocks (or even no DMA at all). But since DMA bandwidth increases with block size, a system should allow larger DMA blocks to accommodate the options that can benefit from using them. For example, the DMA block-size limit is 128 words in the TURBOchannel implementations in the DECstation 5000 series of platforms.

Figure 4 shows the signal activity for a DMA read and Figure 5 shows the signal activity for a DMA write. In both cases, the option initiates the transaction by asserting *~rReq* or *~wReq* and continues to assert it until the system responds by asserting *~ack*, which may well happen in the same cycle. The option indicates the size of the

block to be transferred by prolonging the assertion of $\sim rReq$ or $\sim rReq$ after the $\sim ack$ assertion for a number of cycles equal to the number of words in the block.

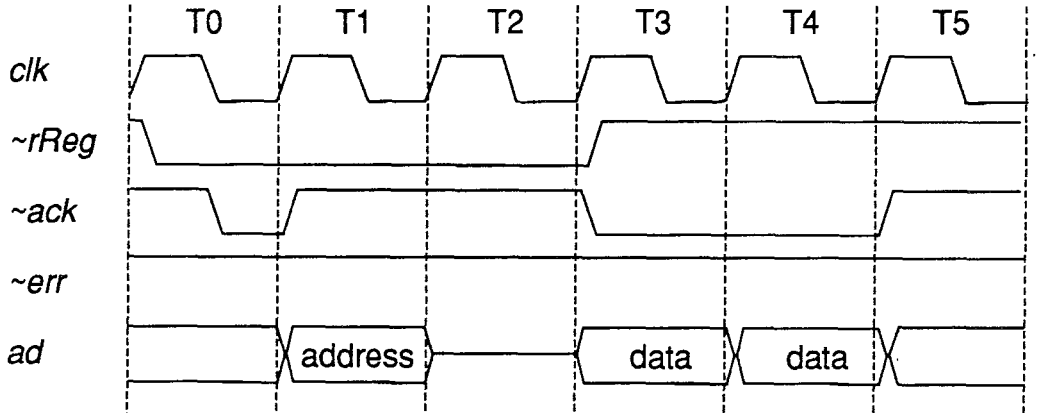

Figure 4. DMA 2-Word Read Transaction

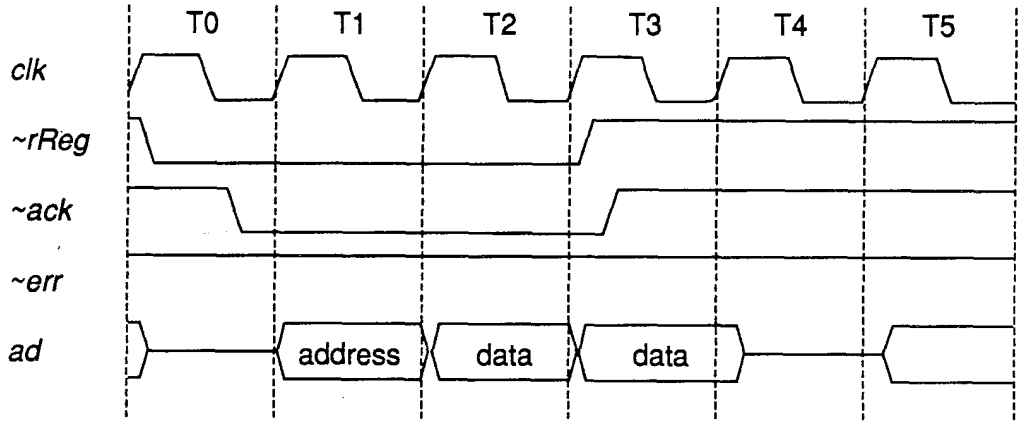

Figure 5 DMA 2-Word Write Transaction

In the cycle following the ~ack assertion, the system deasserts *~ack* and the option drives the DMA address, described above in Section 2.3, onto the *ad[31..0*] to just one cycle.

In the case of a DMA read, there will generally be a memory access latency delay before the system is able to drive the first data word onto *ad[31..0]*. For example, in the memory system of the DECstation 5000/200 this latency is nine cycles. The system

indicates the arrival of the first data word by reasserting ~ack when it drives the data onto the line. After the first word, the system continues to drive one word per cycle onto the ad lines and the option is obliged to accept them.

In the case of a DMA write, the option drives the first data word onto the ad bus in the cycle following the address cycle, and continues driving one word per cycle until reaching the block size indicated by the *~wReq* cycle count.

For more details on the TURBOchannel signal timing requirements, see the TURBOchannel Hardware Specification.

Arbitration of contention for DMA transactions is implementation dependent. Fixed priority schemes are easy and encouraged, but fair service priority disciplines also make sense. The DECstation 5000 series platforms use a fixed priority scheme: option slot 2 has the highest priority, slot 0 the lowest.

3. TURBOchannel Implementation Scalability

The TURBOchannel specification is scalable in the sense that it allows implementation variations that trade off cost for performance. The implementation variations relate primarily to TURBOchannel clock frequency, system memory architecture, and ad line parallelism.

In a low-end implementation, the clock would run at 12.5 Mhz to allow the use of a single bank of page-mode DRAMs. In such a system the TURBOchannel ad channel is embedded in the system bus. The I/O transaction control signals are radial from the processor to the options; the DMA control signals are radial from the memory to the options (see Figure 6). The processor and all the options share the bandwidth of the memory port. The DECstation 5000/100 provides an example of such a low-end TURBOchannel implementation.

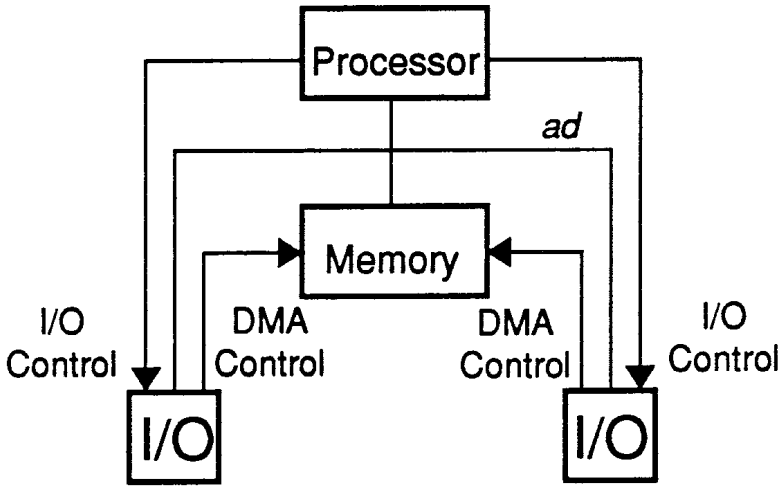

Figure 6. Low-cost Implementation

In a medium performance system, the clock would run at 25 MHz and the memory would have two-way interleaving with all options multiplexed onto a single memory port and the processor attached to a second memory port (see Figure 7). In the midrange implementation, all options share the bandwidth of the memory port. The DECstation 5000/200 provides an example of such a mi-range TURBOchannel implementation

A very high-performance system could use a radial point-to-point topology for the ad[31.. 0] lines as well as the control signals, effectively implementing an independent TURBOchannel for every option. To utilize the aggregate bandwidth of multiple TURBOchannel interconnects, the system would need a high performance memory architecture - highly interleaved multiple banks or perhaps a crossbar connection to the TURBOchannel ad channels (see Figure 8).

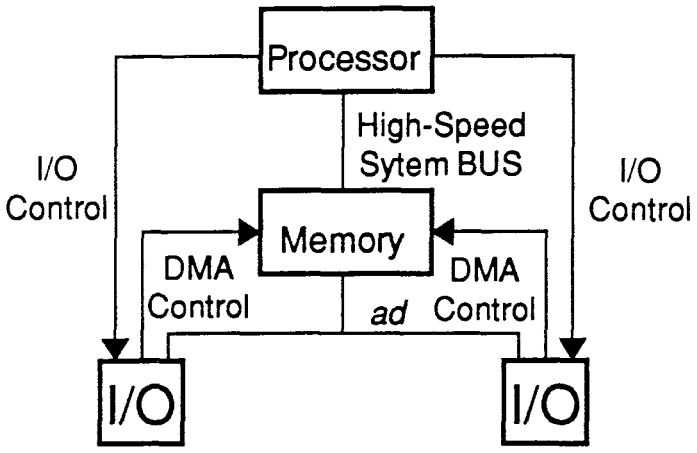

Figure 7. Midrange Implementation

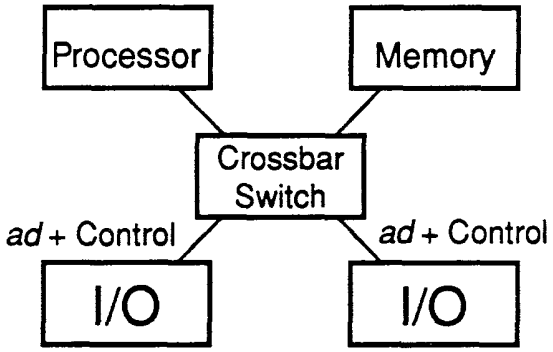

Figure 8. High-performance Implementation

4. TURBOchannel Performance

TURBOchannel data throughput will generally be proportional to the clock frequency. Because the TURBOchannel transfers four bytes per cycle, the absolute peak data rate at 25 MHz is 100 Mbytes/sec.

As with any communication channel, the upper limit for the data transfer rate averaged over a transaction is reduced by the number of overhead cycles. This reduction is moderated by the amount of data transferred per transaction, which amortizes the overhead. The maximum data transfer throughput averaged over multiple transactions may be further limited by software-related delays, such as interrupt service time, and other factors, such as DRAM refresh cycles. Because the TURBOchannel protocol is so compact and simple, both I/O transactions and block DMA transfers have relatively few overhead cycles and so the TURBOchannel performance is substantially higher than other interconnect schemes using the same underlying technology. The exact transaction overhead cycle counts are both system-dependent and option-dependent because of memory access latencies. The amount of data transferred per transaction is just one 32-bit word in I/O transactions and is determined by the option, up to a system-imposed limit, in DMA transactions.

Table 2 summarizes TURBOchannel bandwidth as limited by the protocol and memory latency, and taking software and memory refresh delays into account. The first column of Table 1 is determined by the cycle counts shown in the signal activity diagrams of Section 2.4, assuming a minimal one-cycle memory access latency and 128-word DMA blocks. The second column uses the actual memory access latency of the DECstation 5000/200 implementation. The third column reports throughput observed in laboratory tests on the DECstation 5000/200, using a SCSI controller option for the I/O transactions and a special test option for the DMA transactions.

Table 2. TURBOchannel Bandwidths

	Protocol Limited (Mbytes/sec)	Memory Latency Limited (Mbytes/sec)	Memory Latency and Software Limited (Mbytes/sec)
DMA write	98.5	94.1	78.1
DMA read	97.7	91.4	76.7
I/O write	33.3	33.0	19.3
I/O read	25.0	12.5	10.4

In the case of I/O transactions, the rate at which data can be moved between system memory and the option is probably more relevant than the data transfer rate averaged over a transaction. The former rate involves a processor load or store instruction in addition to the I/O transaction. Laboratory measurements on the DECstation 5000/200 give the following results for the data transfer rate between memory and option using I/O transactions:

Memory to Option 11.4 Mbytes/sec

Option to memory 8.9

5. TURBOchannel Software

As with any peripheral, the use of a TURBOchannel option requires appropriate device driver software installed in the operating system. Section 5.1 discusses the problem of writing a TURBOchannel device driver, which is a significant part of the task of developing a TURBOchannel option. Section 5.2 describes the ROM that resides on TURBOchannel options and carries identifying information as well as certain low-level host software needed in connection with the option.

5.1 Drivers for TUBBOchannel Devices

Device drivers are software modules that provide the software interfaces to peripheral devices. Application programs access peripherals through appropriate system calls to the operating system kernel. The kernel, in turn, passes standard functional requests on to the device driver. The operating system may also call on the driver independently of any direct I/O requests from applications, for example, for standard autoconfiguration and initialization functions, or for handling interrupts. Conversely, the driver may call on kernel support routines for such tasks as process scheduling and buffer management. The operating system defines standard driver functions of generic device classes; the driver contains all device-specific code.

Thus, device drivers lie at a low level of the software hierarchy and are necessarily operating system-specific as well as device-specific. The driver is also specific to the I/O channel or bus that connects the system to the device -the TURBOchannel in the case of TURBOchannel option devices. The driver controls the option device and ascertains its status via TURBOchannel I/O transactions; data transfer may occur through either I/0 transactions or DMA transactions.

Porting drivers to TURBOchannel options from other bus architectures, such as Unibus, is generally quite straight forward. The Guide to Writing and Porting TURBOchannel Device Drivers gives a detailed discussion of the TURBOchannel-specific driver requirements and sample code for some of the functionality which may have to be added.

5.2 Option ROM Information

Every TURBOchannel option module must have a ROM containing information about the option needed for automatic configuration. The ROM information is always

located at offset 3E0H from the base of the option's assigned address space. It begins with a header containing information about the ROM itself followed by information identifying the option (by vendor, name, firmware revision, etc.) sufficient to determine uniquely the appropriate device driver for the option. An important piece of the header information is a standard signature pattern by which the system can ascertain that the slot is indeed occupied by a TURBOchannel option. The header information may be followed by needed option-specific firmware.

Prior to booting the operating system, the system is controlled by ROM Executive (REX) firmware contained in system ROM. REX accepts a certain small set of commands for testing, initialization, configuration display, and booting. REX commands can be entered from the default system console or contained in scripts in option ROM. By performing all module-specific functions using firmware or scripts from the modules themselves, REX remains device independent and allows options to extend the functionality of TURBOchannelbased systems.

The information following the header in option ROM is organized into named containers called ROM objects. A ROM object may contain ASCII-encoded REX scripts, firmware code, or symbolic links to other objects. Most options will contain ROM objects for self-test, initialization, and displaying option configuration. Options that control boot devices will also contain ROM object routines for booting from the device

For more detailed information on TURBOchannel firmware, see the TURBOchannel Firmware Specification.

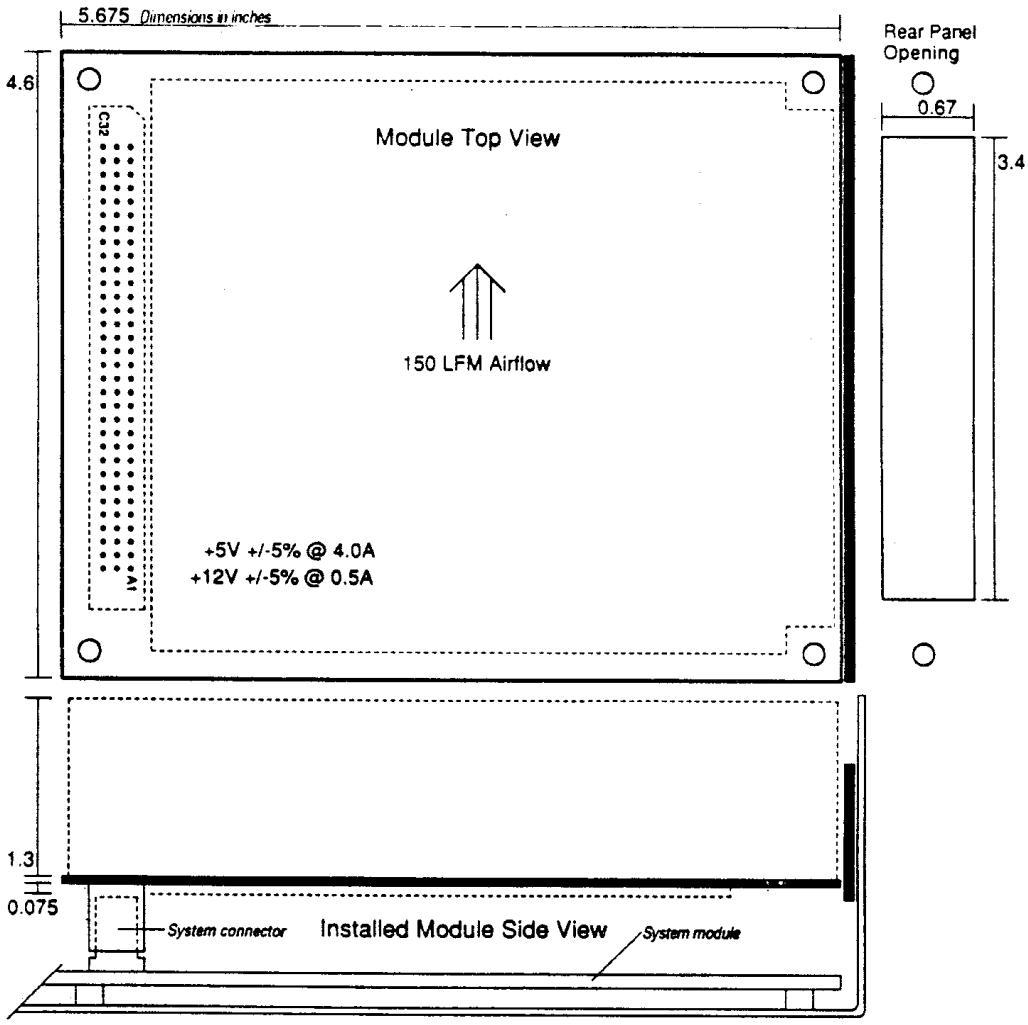

Figure 9 Option Mechanical Outline

6. TURBOchannel Option Implementation

TURBOchannel options may be integral to the system unit or may reside on boards plugged into expansion slots. This section describes salient features of the specification for board-resident options. For more detail see the TURBOchannel Hardware Specification.

6.1 Option Mechanical Specification

Figure 9 depicts the standard TURBOchannel board form factor. Option boards mount parallel to the system module in order to facilitate low profile desktop system

enclosures. Each option connects to the system module via a 96pin DIN connector and has its own bulkhead access in the rear panel of the system enclosure. There is sufficient vertical clearance to accommodate doublesided surface mount.

More complex subsystems can be implemented as double- or triple-width options.

The specification calls for the system to supply the option boards with 150 LFM

6.2 Electrical Specification

The TURBOchannel uses standard TTL signal levels, allowing implementation with either standard logic components or low-cost ASIC components. TURBOchannel timing requirements may be met with current generation bipolar PAL and fast CMOS octal bus devices or low-cost ASIC components. Digital has designed an ASIC providing a standard TURBOchannel interface, described below in Section 6.4.

6.3 TURBOchannel Option Power

The system provides each option slot with 26 watts of power at two voltage levels, +5V and +12V. A further benefit of the low signal count is that 27 pins of the connector are used for power and ground, a feature that contributes to signal integrity.

Multiple width options may draw power from all the system connectors but must connect signals to only one connector.

6.4 TURBOchannel Interface ASIC

Digital has designed a TURBOchannel Interface (TCI) ASIC, which will substantially simplify the design of a TURBOchannel option. The TCI design has been licensed to multiple sources for low-cost implementation. The 160-pin TCI supports all TURBOchannel functions over the full range of TURBOchannel clock frequencies. Independent subsystems provide support for programmed I/O transaction functions and for the DMA functions. The system controls a TCIbased option through a set of TCI control and status registers occupying the first 64 bytes of the the option address slot.

The TCI provides synchronization for I/O transactions, so that the option can operate at a different frequency than the TURBOchannel, but it also has a synchronous mode, which avoids introducing any extra delay when the option is run from the TURBOchannel clock.

The TCI features supporting DMA include variable option-word size, optionword to TURBOchannel word packing and unpacking, selectable byte ordering, automatic management of buffer pointer in paging, and generation of appropriate processor interrupts.

For further information on the TURBOchannel Interface ASIC, see the TURBOchannel Interface ASIC Specification available from the TRI/ADD Program.

7. TRI/ADD Program Support

Digital's TRI/ADD Program (ADDing THIRD party value)provides technical and marketing support to third-party hardware developers using the SCSI, TURBOchannel, VME, ACCESS.bus, and Futurebus+ interconnects. Vendors are encouraged to develop innovative TURBOchannel options or new TURBOchannel system implementations. Membership in the TRI/ADD Program is free. Services include automatic notification of available documentation.

Contact the TRI/ADD Program using the numbers on the first sheet of this document to obtain TURBOchannel Specification, containing the documentation necessary to support development. Also available is a set of software and hardware features that will expedite the development process. In addition, there is a 62-minute video tape presenting a TURBOchannel overview, as well as some detailed design hints from Digital and third-party engineers.

Marketing support includes listing of TRI/ADD products in Digital's catalogs sent to a customer base of 300,000 and in the TRI/ADD Shippable Products Catalog. In addition, TRI/ADD products can be supported by Digital's world-wide field service organization.