TABLE OF CONTENTS

Information in this MULTIBUS® Handbook subject to change without notice.

The following are trademarks of Intel Corporation and its affiliates and may be used only to identify Intel products:

AEDIT
BITBUS
BXP
COMMputer
CREDIT
i
iATC
I^2ICE
ICE
iCS
iDBP
iDIS
iLBX
im
iMMX
Insite
Intel
IntelBOS
Intelevision
inteligent Identifier
inteligent Programming
Intellec
Intellink
iOSP
iPDS
iRMX
iSBC
iSBX
iSDM
iSXM
Library Manager
MCS
Megachassis
MICROMAINFRAME
MULTIBUS
MULTICHANNEL
MULTIMODULE
Plug-A-Bubble
PROMPT
Promware
Ripplemode
RMX/80
RUPI
System 2000
UPI

MULTIBUS®-IEEE 796
SYSTEM BUS SPECIFICATION

Bibliography

Intel Multibus® Specification

CONTENTS

Page

CHAPTER 1
GENERAL

CHAPTER 2
FUNCTIONAL DESCRIPTION

CONTENTS (continued)

Page

CHAPTER 5
LEVELS OF COMPLIANCE

FIGURES

TABLES

CHAPTER 1
GENERAL

IEEE MICROCOMPUTER SYSTEM BUS STANDARD (796 BUS)

1.1 SCOPE

One of the most important elements in a computer system is the bus structure that supplies the interface for all the hardware components. This bus structure contains the necessary signals to allow the various system components to interact with each other. It allows memory and I/O data transfers, direct memory accesses, generation of interrupts, etc. This document provides a detailed description of all the elements and features that make up the 796 Bus.

The bus supports two independent address spaces: memory and I/O. During memory cycles, the bus allows direct addressability of up to 16 megabytes using 24-bit addressing. During I/O bus cycles, the bus allows addressing of up to 64K I/O ports using 16-bit addressing. Both memory and I/O cycles can support 8-bit or 16-bit data transfers.

The bus structure is built upon the master-slave concept where the master device in the system takes control of the bus and the slave device, upon decoding its address, acts upon the command provided by the master. This handshake (master-slave relationship) between the master and slave devices allows modules of different speeds to be interfaced via the bus. It also allows data rates up to five million transfers per second (bytes of words) to take place across the bus.

Another important feature of the bus is the ability to connect multiple master modules for multiprocessing configurations. The bus provides control signals for connecting multiple masters in either a serial or parallel priority fashion. With either of these two arrangements, more than one master may share bus resources.

This document has been prepared for those users who intend to evaluate or design products that will be compatible with the 796 system bus structure. To this end, the necessary signal definitions and timing and electrical specifications have been covered in detail.

This standard deals only with the interface characteristics of microcomputer devices: not with design specifications, performance requirements, and safety requirements of modules.

Throughout this standard, the term "system" denotes the byte or word interface system that, in general, includes all the circuits, connectors, and control protocol to effect unambiguous data transfer between devices. The term "device" or "module" denotes any product connected to the interface system that communicates information via the bus, and that conforms to the interface system definition.

1.2 OBJECT

This standard is intended:

(1) To define a general purpose microcomputer system bus.

(2) To specify the device-independent electrical and functional interface requirements that a module shall need in order to interconnect and communicate unambiguously via the system.

(3) To specify the terminology and definitions related to the system.

(4) To enable the interconnection of independently manufactured devices into a single functional system.

(5) To permit products with a wide range of capabilities to be interconnected to the system simultaneously.

(6) To define a system with a minimum of restrictions on the performance characteristics of devices connected to the system.

1.3 DEFINITIONS

The following general definitions apply throughout this standard. More detailed definitions can be found in the appropriate section.

1.3.1 General System Terms

Compatibility. The degree to which devices may be interconnected and used without modification, when designed as defined in Sections 2 and 3 of this standard. Section 5 introduces the notion of levels of compliance and the corresponding notation.

Bus cycle. The process whereby digital signals effect the transfer of data bytes or words across the interface by means of an interlocked sequence of control signals. "Interlocked" denotes a fixed sequence of events in which one event must occur before the next event can occur.

Interface. A shared boundary between two systems, or between parts of systems, through which information is conveyed.

Interface system. The device-dependent electrical and functional interface elements necessary for communication between devices. Typical elements are: driver and receiver circuits, signal line descriptions, timing and control conventions, and functional logic circuits.

Override. A bus master overrides the bus control logic when it is necessary to guarantee itself back-to-back bus cycles. This is called "overriding" or "locking" the bus, temporarily preventing other masters from using the bus.

System. A set of interconnected elements which achieve a given objective through the performance of a specified function.

1.3.2 Signals and Paths

Bus. A signal line or a set of lines used by an interface system to connect a number of devices, and to transfer information.

Byte. A group of eight adjacent bits operated as a unit.

Word. Two bytes or sixteen bits operated as a unit.

High state. The more positive voltage level used to represent one of two logical binary states.

Low state. The more negative voltage level used to represent one of two logical binary states.

Signal. The physical representation of data.

Signal level. The relative magnitude of a signal when compared to an arbitrary reference. Signal levels in this standard are specified in volts.

Signal line. One of a set of signal conductors in an interface system used to transfer messages among interconnected devices.

Signal parameter. That element of an electrical quantity whose values or sequence of values convey information.

CHAPTER 2
FUNCTIONAL DESCRIPTION

This section provides an overall understanding of how the 796 Bus functions, and describes the elements that connect to the bus, the signals that provide the interface to the bus, and the different types of operations performed on the bus.

In this section, as well as throughout the specification, a clear and consistent notation for signals has been used. The Memory Write Command (MWTC) will be used to explain this notation. The terms one:zero and true:false can be ambiguous, so their use will be avoided. In their place, we will use the terms electrical High and Low (H and L). A nathan (asterisk) following the signal name (MWTC*) indicates that the signal is active low as shown:

MWTC* = Asserted at 0 volts

The signal (MWTC*) driven by a three state driver will be pulled up to V_{CC} when not asserted. The following is used to further explain the notation used in this specification.

		Definition	
Function	**Electrical**	**Logic**	**State**
MWTC	H	1 True	Active, Asserted
	L	0 False	
MWTC*	L	1 True	Active, Asserted
	H	0 False	

2-1. 796 BUS ELEMENTS

This subsection describes the elements (masters and slaves) that interface to the bus and the 796 Bus signal lines that comprise this interface.

2.1.1 Masters

A master is any module having the ability to control the bus. The master exercises this control by acquiring the bus through bus exchange logic and then generating command signals, address signals, and memory or I/O addresses. To perform these tasks, the master is equipped with either a central processing unit or logic dedicated to transferring data over to the bus to and from other destinations. Figure 1 depicts a system that includes a master and two slave models.

The 796 Bus architecture can support more than one master in the same system, but in order to do this, there must be a means for each master to gain control of the bus. This is accomplished through the bus exchange logic (see 2.4).

Masters may operate in one of two modes of operation. Modes 1 and 2 are defined as follows:

Mode 1: Masters are limited to single bus transfers per bus connect. If all masters are Mode 1, system timing is rendered deterministic by conformance with a maximum bus busy period. That period is limited by the parameter t_{BYSO} max. (see 3.2.5).

Mode 2: Masters are unlimited in this bus control. They may invoke bus override. Bus timeouts are allowed. Conformance with the maximum busy period is not required.

The last classification is included to allow for a very broad class of operations, giving users maximum flexibility in meeting these applications' needs. The first mode of operation is defined to allow system designers to predict the overall performance of their systems without concern for uncontrolled timing parameters such as bus timeout. For Masters which can only operate in Mode 2, their specification shall state "Mode 2 master only."

2.1.2 Slaves

Another type of module that can interface to the bus is the slave. Slave modules decode the address lines and act upon the command signals from the masters. The slaves are not capable of controlling the bus. Some examples of bus slaves are shown in Figure 1.

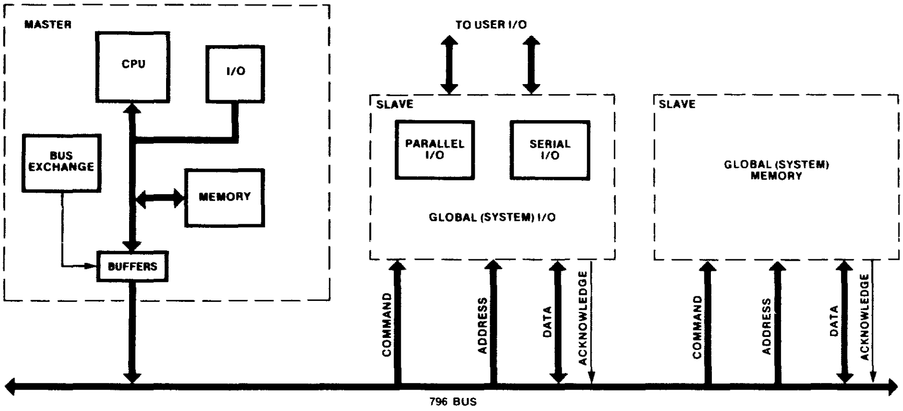


Figure 1 796 Bus Master and Slave Example

2.1.3 796 Bus Signals

Signals transferred over the bus can be grouped into several classes based on the functions they perform. The classes are:

(1) Control lines

(2) Address and Inhibit Lines

(3) Data Lines

(4) Interrupt Lines

(5) Bus Exchange Lines

The following subsections explain the different classes of 796 Bus signals.

2.1.3.1 Control Lines

The following signals are classified as control lines:

Class	Function	Signal
Clocks	Constant Clock	CCLK*
	Bus Clock	BCLK*
Commands	Memory Write	MWTC*
	Memory Read	MRDC*
	I/O Write	IOWC*
	I/O Read	IORC*
Acknowledge	Transfer Acknowledge	XACK*
Initialize		INIT*
Lock		LOCK*

2.1.3.1.1 Clock Lines.

(1) _Bus Clock (BCLK*)._ A periodic signal used to synchronize the bus contention logic; it may be slowed, stopped or single stepped. The Bus Clock shall be generated by one and only one source within the system. This means that each standalone bus master must have the capability of generating an acceptable clock that can optionally be connected to, or disconnected from, the bus. In a multimaster system, only one of the masters shall have its clock connected to the bus.

(2) _Constant Clock (CCLK*)._ A periodic signal of constant frequency, which may be used by masters or slaves as a master clock. The Constant Clock shall be generated by one and only one source within the system. This means that each bus master must have the capability of generating an acceptable clock that can optionally be connected to, or disconnected from, the bus. In a multimaster system, only one of the masters shall have its clock connected to the bus.

2.1.3.1.2 Command Lines (MWTC*, MRDC*, IOWC*, IORC*)

The command lines are elements of a communication link between the masters and slaves. There are two command lines for memory and two command lines for I/O. An active command line indicates to the slave that the address lines are carrying a valid address, and that the slave is to perform the specified operation. In a data write cycle, the active command line (MWTC* or IOWC*) additionally indicates that the data is valid on the bus. In a data read cycle, the transition of the command (MRDC* or IORC*) from active to inactive indicates that the master has received the data from the slave.

2.1.3.1.3 Transfer Acknowledge Line (XACK*)

This line is used by the slaves to acknowledge commands from the master. XACK* indicates to the master that the requested action is completed, and that data has been placed on, or accepted from, the data lines.

2.1.3.1.4 Initialize (INIT*)

The INIT* signal is generated to reset the entire system to a known internal state. This signal is usually generated prior to starting any operations on the system. INIT* may be generated by any or all of the bus masters or by an external source such as a buffered and debounced front panel switch.

2.1.3.1.5 Lock (LOCK*)

The LOCK signal is generated by the master in control of the bus to indicate the bus is locked. LOCK* is used to extend mutual exclusion to multiple port RAM designs.

2.1.3.2 ADDRESS AND INHIBIT LINES

The address and inhibit lines are used for the following signals:

Function	Signal
Address Lines	ADR0*-ADR17* (0-9, A-F, 10-17 in hexadecimal)
Byte High Enable	BHEN*
Inhibit Lines	INH1* and INH2*

2.1.3.2.1 Address Lines (24 lines)

These lines, which specify the address of the referenced memory location or I/O device, allow a maximum of 16 megabytes (16,777,216 bytes) of memory to be accessed. When addressing an I/O device, a maximum of 16 address lines (ADR0*-ADRF*) are used, thus allowing the addressing of a maximum of 64K devices. An I/O module must also be able to be configured to decode only 8 address lines (ADR0*-ADR7*) and ignore the upper 8 lines (see 2.2.2.3).

2.1.3.2.2 Byte High Enable Line (BHEN*)

This byte control line is used to enable the upper bytes (bits 8-F) of a 16-word bit word to drive the bus. The signal is used only on systems that incorporate 16-bit memory modules.

2.1.3.2.3 Inhibit Lines (INH1* and INH2*)

The inhibit lines can be invoked for any memory read or memory write operation (MRDC* or MWTC*). An inhibit line is asserted by a slave to inhibit another slave's bus activity during a memory read or write operation. The inhibit signal generated by the inhibiting slave is derived from decoding the memory address lines. The inhibiting slave can decode a single address, a block of addresses, or any combination of single and block addresses.

When it detects the specific address during an actual command (MRDC* or MWTC*), the inhibiting slave generates an inhibit signal, which is sensed by the inhibited slave. When so inhibited, this slave module disables its drivers from all data, address, and acknowledge bus lines, although it may actually perform internal operations. (All modules that may be inhibited must have completed internal operations within 1.5 microseconds from the start of the command line. This interval [1.5 microseconds] is also the minimum acknowledge timing for modules issuing inhibits. This guarantees that inhibited modules have enough time to return to their normal state before the current bus command is completed.)

2.1.3.3 DATA LINES (DAT0*-DATF*)

These 16 bidirectional data lines transmit and receive information to and from a memory location or an I/O port. (DATF* is the most-significant bit and DAT0* is the least-significant bit). In 8-bit systems, only lines DAT0*-DAT7* are valid.

2.1.3.4 INTERRUPT LINES

The interrupt lines consist of the following signals:

Function	Signal
Interrupt Requests	INT0*-INT7*
Interrupt Acknowledge	INTA*

2.1.3.4.1 Interrupt Request Lines (INT0*-INT7*)

Interrupts are requested by activating one of the eight interrupt request lines. INT0* has the highest priority and INT7* has the lowest priority.

2.1.3.4.2 Interrupt Acknowledge (INTA*)

In response to an Interrupt Request signal, an Interrupt Acknowledge signal can be generated by a bus master with bus vectored interrupt capability. The Interrupt Acknowledge signal is used to freeze the interrupt status and request the placement of the interrupt vector address on the bus data lines.

2.1.3.5 BUS EXCHANGE LINES

The bus exchange lines are used by the following signals:

Function	Signal
Bus Clock	BCLK*
Bus Request	BREQ*
Bus Priority	BPRN*, BPRO*
Bus Busy	BUSY*
Common Bus Request	CBRQ*

A master gains control of the bus through the manipulation of these signals.

2.1.3.5.1 Bus Request (BREQ*)

A signal used by the bus masters in a priority resolution circuit to indicate a request for control of the bus.

2.1.3.5.2 Bus Priority (BPRN* and BPRO*)

The priority functions allow masters to break deadlocks that occur when more than one master concurrently requests the bus. The Bus Priority In (BPRN*) signal indicates to a particular master that no higher priority master is requesting use of the bus. The Bus Priority Out (BPRO*) signal is used in serial (daisy chain) bus priority resolution schemes. In such a scheme, BPRO* is passed by one master to the BPRN* input of the master with the next lower bus priority; when active, the BPRO* signal indicates that the higher priority master not require control of the bus.

2.1.3.5.3 Bus Busy (BUSY*)

A signal activated by the master in control of the bus to indicate that the bus is in use. This prevents other masters from gaining control of the bus.

2.1.3.5.4 Common Bus Request (CBRQ*)

A signal that maximizes a master's data transfer rate to the bus by sensing the absence of other bus requests. The CBRQ* signal does this by serving two functions. It indicates to the master controlling the bus whether or not another master needs to gain control of the bus. To the other masters, it is a means of notifying the controlling bus master that it must relinquish control of the bus if it is not using the bus.

2.2 DATA TRANSFER OPERATION

The primary function of the 796 Bus architecture is to provide a path for the transfer of data between modules on the bus. The following subsections describe the different types of data transfers and the means by which they are implemented using the signals previously described. Figure 2 can be referenced during the following discussion.

The discussion of the data transfer operation of the bus is covered in three parts:

(1) An overview of the operation

(2) A detailed description of the signals used in the transfer

(3) A discussion of the specifics pertaining to the different transfers

It is assumed in this discussion that there is only one master on the bus, and therefore no bus contention exists. (The bus exchange logic is discussed in 2.4).

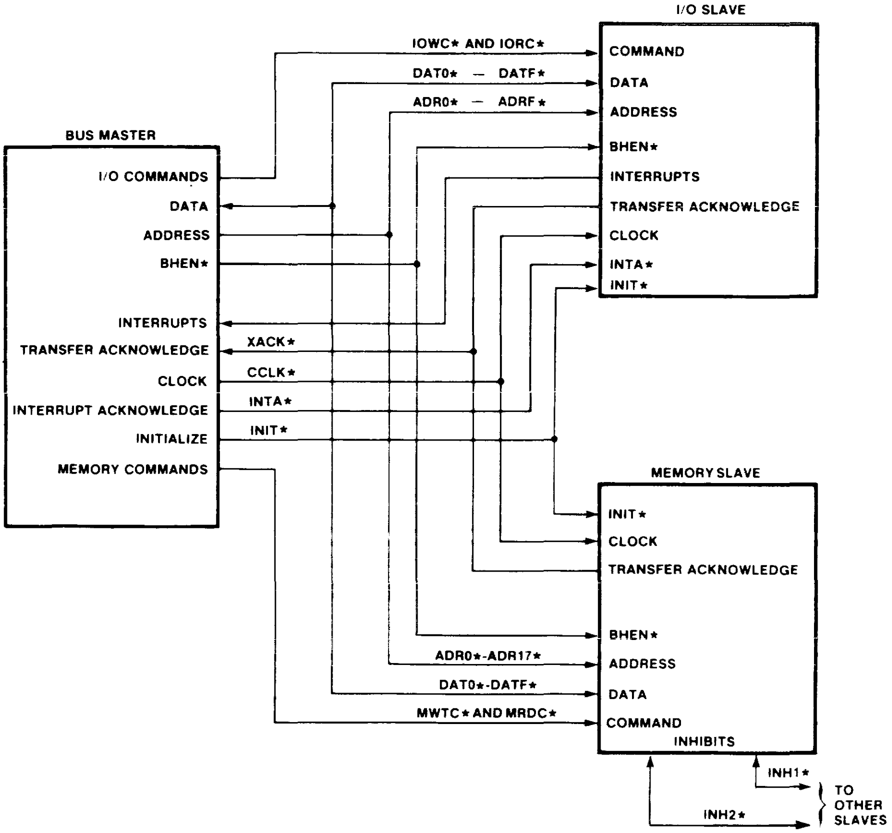


Figure 2 796 Bus Interface Lines

2.2.1 Data Transfer Overview

A data transfer is accomplished as follows. First the bus master places the memory address or I/O port address on the address lines. (If the operation is a write, the data would also be placed on the data lines at this time.) The bus master then generates a command (I/O read or write, or memory read or write), which activates the appropriate bus slave. The slave accepts the draft if it is a write operation, or places the data on the data lines if it is a read operation. A Transfer Acknowledge signal is then sent to the bus master by the bus slave, allowing the bus master to complete its cycle by removing the command from the command line and then clearing the address and data lines. Figure 3 and Figure 4 show the basic timing for a read and write data transfer operation.

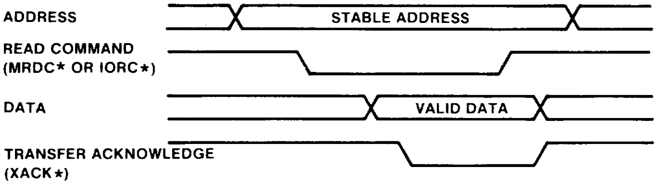


Figure 3 796 Bus Read Operation

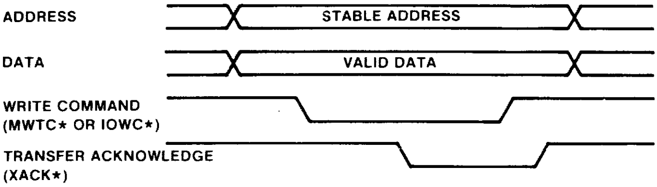


Figure 4 796 Bus Write Operation

2.2.2 Signal Descriptions

This subsection provides a detailed description of the 796 Bus signals. Included are timing, signal origination, and other information pertaining to the specific function that each signal performs in the data transfer operation.

2.2.2.1 INITIALIZE (INIT*)

Prior to any operation of the bus, all system modules should be reset to a known internal state. This can be accomplished by an INIT* signal initiated by one of three sources:

(1) A power-on clear circuit (RC network), which holds INIT* low until the power supplies reach their specific voltage outputs

(2) A reset button, which is sometimes provided on the system front panel for operator use. Note that this button must be debounced.

(3) A software command that can be implemented to pull down the INIT* line.

The INIT* line is driven by open-collector gates and requires signal conditioning to meet the electrical specifications of the bus.

2.2.2.2 CONSTANT CLOCK (CCLK*)

The Constant Clock signal, which is driven by only one source, provides a timing source for any or all modules on the bus. CCLK* is a periodic signal with a specified frequency and is driven by a clock driver circuit.

2.2.2.3 ADDRESS LINES (ADR0*-ADR17*)

The address lines are used to specify the address of the memory locations or the I/O device that is being referenced by the command. There are 24 address lines, binary coded, to allow up to 16,777,216 bytes of memory to be referenced. These lines are driven by three-state drivers and are always controlled by the master using the bus.

For the I/O bus cycles, master modules have the option of generating 8-bit or 16-bit addresses. Because of this, all I/O slaves must be capable of being configured to decode 8 address bits (ADR0*-ADR17*) and ignore the upper address bits or to decode all 16 bits of address (ADR0*-ADRF*). Note that in a system using 8-bit I/O addresses, the value of the upper 16-its of address is unknown. A master generating only 8-bit address may set the upper 16 address bits to any arbitrary value.

Refer to Figure 5 for an example of address line usage.

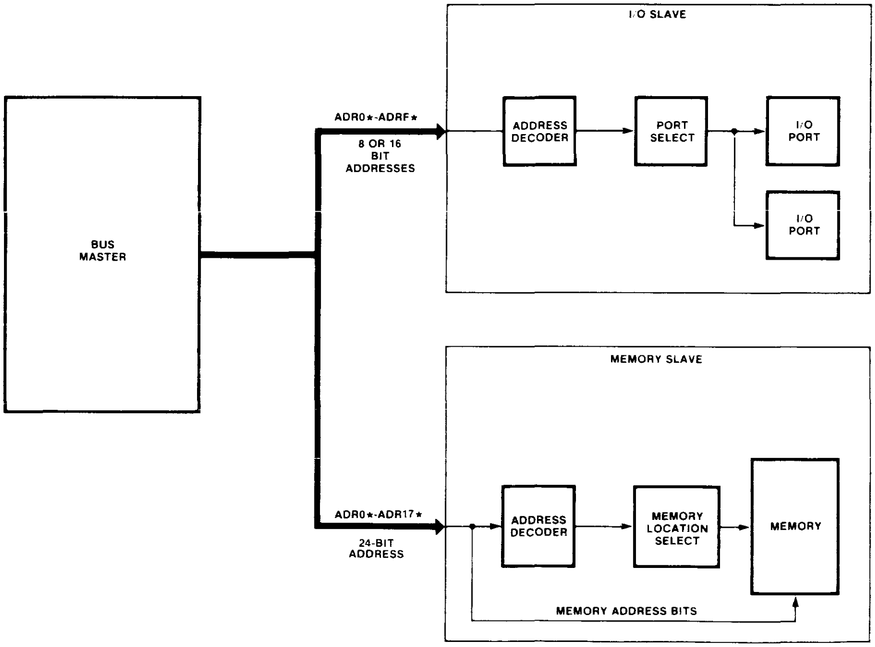


Figure 5 796 Bus Address Line Usage

2.2.2.4 DATA LINES (DAT0*-DATF*)

These are 16 bi-directional data lines used to transmit and receive information to and from a memory location or I/O port. The 16 lines are driven by the master on write operations and by the addressed slave (memory or I/O) on read operations. Both 16-bit and 8-bit transfers can be accomplished by using only lines DAT0*-DAT7* (with DAT0* being the least-significant bit).

There are three types of transfers that take place across the bus:

(1) Transfer of low (even) byte on DAT0*-DAT7*

(2) Transfer of high (odd) byte on DAT0*-DAT7* (using swap byte function)

(3) Transfer of a 16-bit word

Figure 6 shows the data lines, and the contents of these lines for the three types of transfers mentioned.

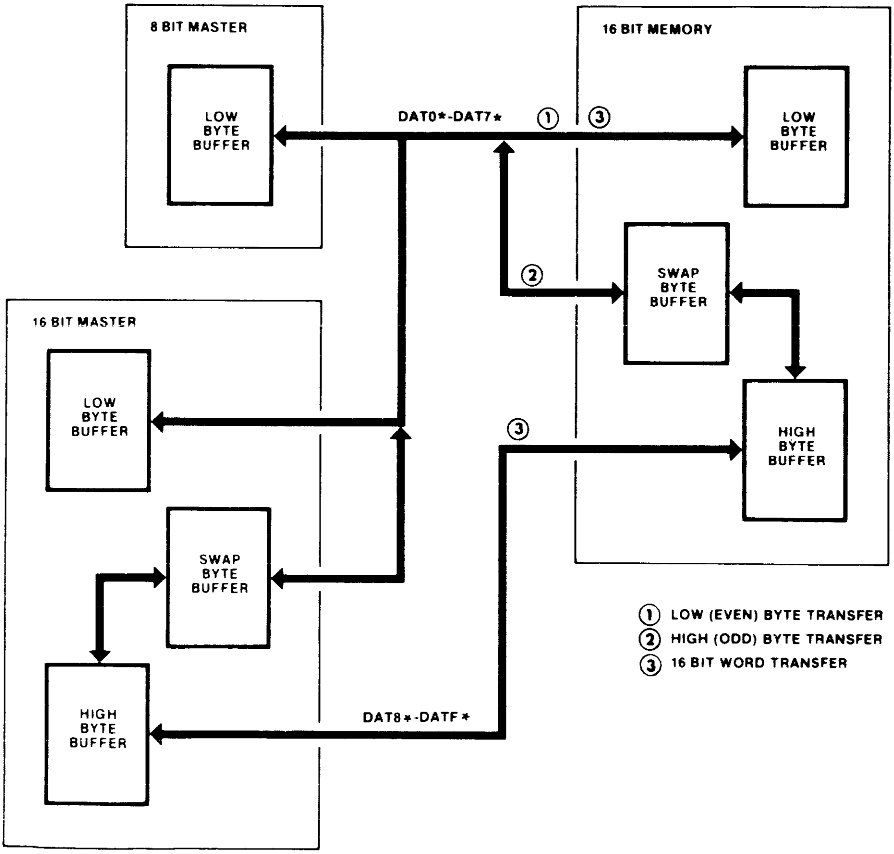


Figure 6 796 Bus Data Line Usage

Two signals control the data transfers. Byte High Enable (BHEN*) active indicates that the bus is operating in the 16-bit mode, and the address bit 0 (ADR0*) defines an even-byte or odd-byte transfer.

For an even byte-transfer, BHEN* and ADR0* are inactive, indicating the transfer of an even-byte. The transfer takes place across data lines DAT0*-DAT7*.

For an odd-byte transfer, BHEN* is inactive and ADR0* is active, indicating the transfer of an odd-byte. On this type of transfer, the odd-(high) byte is transferred through the Swap Byte Buffer to DAT0*-DAT7*. The high-(odd) byte is transferred across on DAT0*-DAT7* to make 8-bit and 16-bit systems compatible.

For a 16-bit transfer, BHEN* is active and ADR0* is inactive. On this type of transfer, the low-(even) byte is transferred on DATO*-DAT7* and the high-(odd) byte is transferred across the bus on DAT8*-DATF*.

The 796 Bus data lines are always driven by three-state drivers.

2.2.2.5 796 BUS COMMANDS

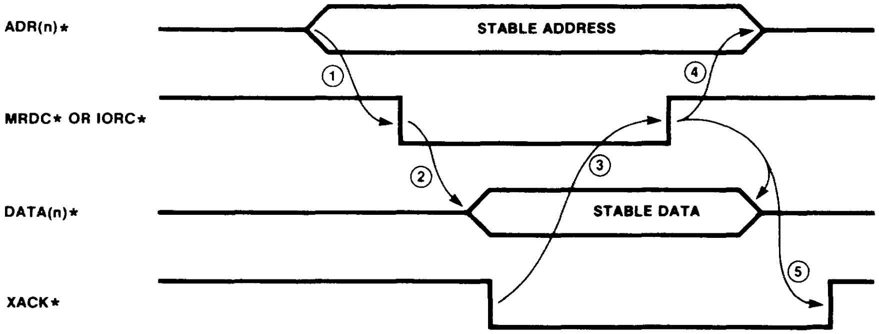
In this subsection we will discuss the command lines and how they work in conjunction with other lines to accomplish a read or a write operation. There are four command lines:

Function	Line
Memory Read Command	MRDC*
I/O Read Command	IORC*
Memory Write Command	MWTC*
I/O Write Command	IOWC*

The command lines, which are driven by three-state drivers on the bus master, indicate to the slave the action that is being requested.

2.2.2.5.1 Read Operation

The two read commands (MRDC* and IORC*) initiate the same basic type of operation. The only difference is that MRDC* indicates that the memory address is valid on the address lines, whereas IORC* indicates that the I/O port address is valid on the address lines. This address (memory or I/O port) must be valid on the bus 50 nanoseconds prior to the read command being generated. When the read command is generated, the slave module (memory or I/O port) places the data on the data lines and returns a Transfer Acknowledge (XACK*) signal, indicating that the data is on the bus. When the bus master receives the acknowledge, it strobes in the data and removes the command (MRDC* or IORC*) from the bus. The slave address (memory or I/O port remains valid on the bus a minimum of 60 nanoseconds after the read command is removed. XACK* must be removed from the bus within 65 nanoseconds after the command is removed to allow for the next bus cycle. Figure 7 shows the timing for the Memory Read or I/O Read command.



① ADDRESS SETUP TIME: 50 NANOSECONDS MINIMUM.

② TIME REQUIRED FOR SLAVE TO GET DATA ONTO BUS IN ACCORDANCE WITH SETUP TIME REQUIREMENT. XACK* CAN BE ASSERTED AS SOON AS DATA IS ON BUS.

③ TIME REQUIRED FOR MASTER TO REMOVE COMMAND.

④ ADDRESS AND DATA HOLD TIME; 50 NANOSECONDS MINIMUM.

⑤ XACK* AND DATA MUST BE REMOVED FROM THE BUS A MAXIMUM OF 65 NANOSECONDS AFTER THE COMMAND IS REMOVED.

Figure 7 Memory or I/O Read Timing

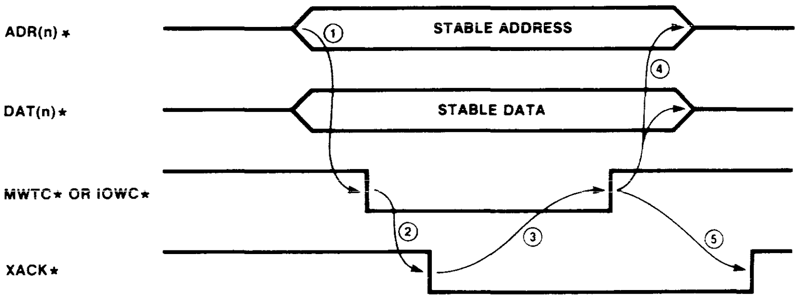
2.2.2.5.2 Write Operation

The write commands (MWTC* and IOWC*) initiate the same basic type of operation. MWTC* indicates that the memory address is valid on the address lines, whereas IOWC* indicates that the I/O port address is valid on the address lines. The address (memory or I/O) and data must be valid on the bus 50 nanoseconds prior to the write command being generated. This requirement allows data to be latched on either the leading or trailing edge of the command. When the write command (MWTC* or IOWC*) is asserted, the data on the data lines is stable and can be accepted by the slave. The slave indicates acceptance of the data by returning a Transfer Acknowledge (XACK*), allowing the bus master to remove the command, address, and data from the bus, XACK* must be removed from the bus within 65 nanoseconds to allow for the next bus cycle. Figure 8 shows the timing for the Memory Write or I/O Write command.

2.2.2.6 TRANSFER ACKNOWLEDGE (XACK*)

The Transfer Acknowledge (XACK*) signal is the response from the bus slave (memory or I/O) indicating that the commanded read or write operation is complete and that the data has been placed on, or accepted from, the data lines. In effect, this signal (XACK*) allows the master to complete the current bus cycle.

If a bus master addresses a non-existent or malfunctioning memory or I/O module, an acknowledge will not be returned to the master. If this should occur, the bus master would normally wait indefinitely for an acknowledge and would therefore never relinquish control of the 796 Bus. To avoid this possibility, a bus timeout function can optionally be implemented on a bus master to terminate a bus cycle after a preset interval, even if no acknowledge has been received. A bus timeout can therefore be defined as any data transfer cycle terminated by the master before the Transfer Acknowledge (XACK*) signal is received. The minimum allowable bus timeout interval is 1.0 millisecond.



1 ADDRESS AND DATA SETUP; 50 NANOSECONDS MINIMUM.
2 TIME REQUIRED FOR SLAVE TO ACCEPT DATA.
3 TIME REQUIRED FOR MASTER TO REMOVE COMMAND FROM BUS.
4 ADDRESS AND DATA HOLD TIME; 50 NANOSECONDS MINIMUM.
5 XACK* MUST BE OFF THE BUS 65 NANOSECONDS AFTER COMMAND.

Figure 8 Memory or I/O Write Timing

2.2.2.7 INHIBIT (INH1* AND INH2*)

The inhibit lines can be invoked for any memory read or memory write operation (MRDC* or MWTC*). An inhibit line is asserted by a slave to inhibit another slave's bus activity during a memory read or write operation. The inhibit signal generated by the inhibiting slave is derived from decoding the memory address lines (t_{ID} = 100 nanoseconds maximum). The inhibiting slave can decode a single address, a block of addresses, or any combination of single and block addresses.

When it detects the specific address during the actual command (MRDC* or MWTC*), the inhibiting slave generates an inhibit signal, which is sensed by the inhibited slave. When so inhibited, this slave module disables its drivers from all data, address, and acknowledge bus lines, although it may actually perform internal operations. (All modules that may be inhibited must have completed internal operations with 1.5 microseconds from the start of the command line. This interval [1.5 microseconds] is also the minimum acknowledge (t_{ACC}) timing for modules issuing inhibits. This guarantees inhibited modules enough time to return to their normal state before the current bus command is completed.)

The slaves involved in the inhibit operation fall into three inhibit classes: top (inhibit) priority, middle priority, and bottom priority. In reference to the above paragraphs, a higher priority slave module would be the inhibiting slave and a lower priority slave would be the inhibited slave. INH1* is asserted (during the appropriate address) by a middle priority slave (such as a read-only memory module or memory-mapped I/O module) to inhibit the bus activity of a bottom priority slave (such as a read/write RAM module). INH2* is asserted (at the appropriate address) by a top priority slave (such as an auxiliary or a bootstrap ROM module) to inhibit the bus activity of a middle priority slave. The top priority slave shall also assert INH1* so that a bottom priority slave will also be inhibited. The inhibit lines shall be asserted low by open collector (or equivalent) drivers. When both a middle and a top priority inhibiting slave are activated, INH1* will be asserted by drivers on both modules.

The use of the inhibit signals during memory reads (MRDC*) shall not cause any adverse effects within the inhibited slave module. That is, data in the inhibited slave shall not be altered and its status register, if any, shall not be affected.

The use of the inhibit signals during memory writes (MWTC*) shall be allowed, and might or might not affect the data within the inhibited slave. If the data is affected, it shall be only within the one byte (or word) that is being addressed. (No other data within the inhibited slave shall be altered.)

The inhibit signals, when issued, shall be generated within 100 nanoseconds (t_{ID}) after the address is stable. (See Figure 9.) A command may be generated as early as 50 nanoseconds (t_{AS}) after the address is stable. This timing can cause the inhibit to occur after the command has been received by the inhibited module. To prevent false acknowledges, modules that can be inhibited must not generate an acknowledge until the inhibit signals have had time to become valid (50 nanoseconds after the command).

Figure 9 shows the timing for an inhibit operation. In this example, the PROM and RAM have the same memory addresses; therefore, the PROM inhibits the RAM.

Although inhibit signals may be generated during IORC*, IOWC*, or INTA* operations, these signals are ignored by other slaves (including the slave that should respond to the INTA*, IORC*, or IOWC*).

2.2.2.8. LOCK (LOCK*)

The Lock line is driven by the master control of the bus when a locked bus access is required. A locked access is typically required in a read-modify-write semaphore operation to prevent another processor from accessing the memory between the read and the write. The busy line allows for this mutual exclusion on the 796 Bus. The Lock line allows mutual exclusion to be extended off of the bus. The Lock signal (LOCK*) must be active 100 nanoseconds prior to the read or write command going away. It must remain active a minimum of 100 nanoseconds after the falling edge of the command signal for the last locked memory cycle. The slave locks its multiple ported memory to the 796 Bus when it is addressed and the lock line is asserted. The lock signal must not be asserted for more than 12 microseconds continuously. This assures the processor on the other side of the multiple ported memory that it will gain access to the memory in a reasonable amount of time. The busy signal (BUSY*) must be active whenever the Lock line is asserted. Figure 11 shows the timing for the lock signal.

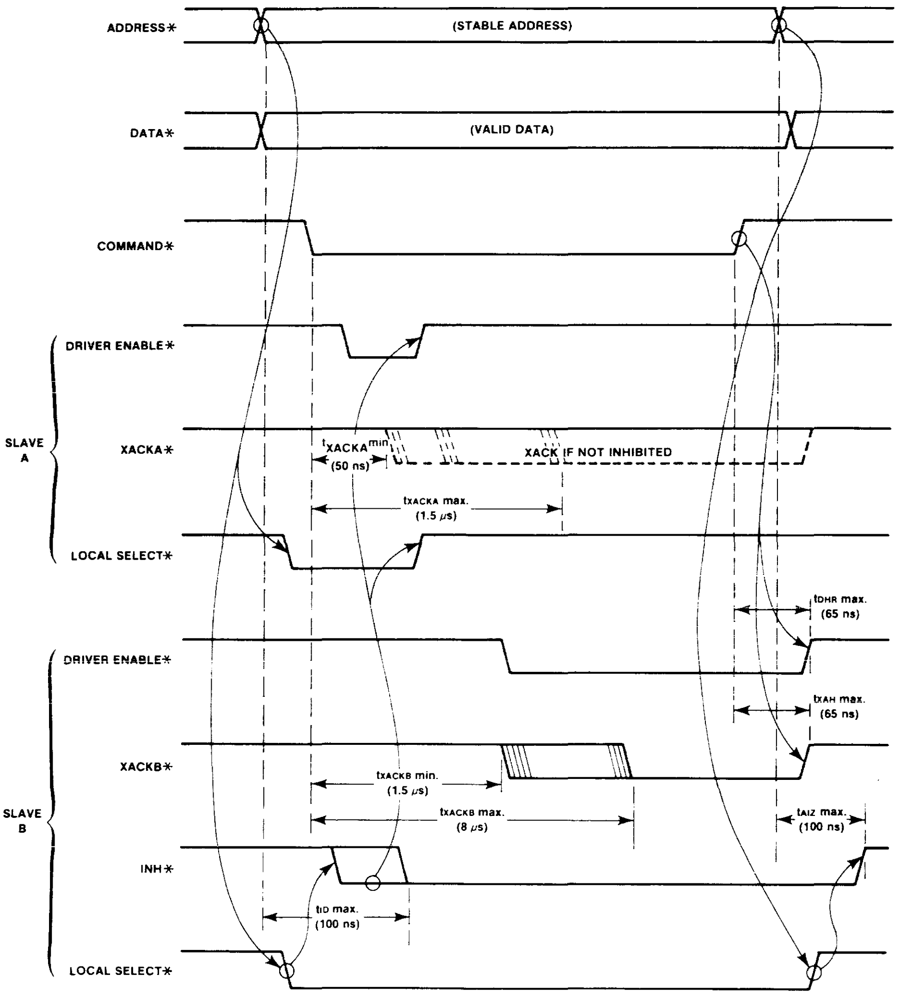


Figure 9 Inhibit Timing for Write Operation

2.3 INTERRUPT OPERATIONS

The following subsections explain the 796 Bus signal lines used in the interrupt operation, and the two different types of interrupt implementation. Refer to Section 5.1.4 for information on levels of compliance with respect to interrupt attributes.

2.3.1 Interrupt Signal Lines

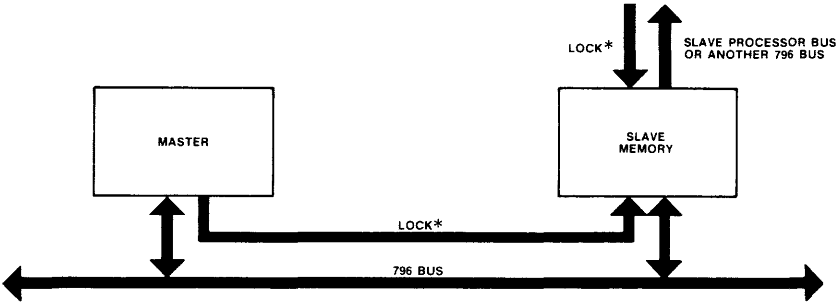


Figure 10 796 Bus Lock Usage

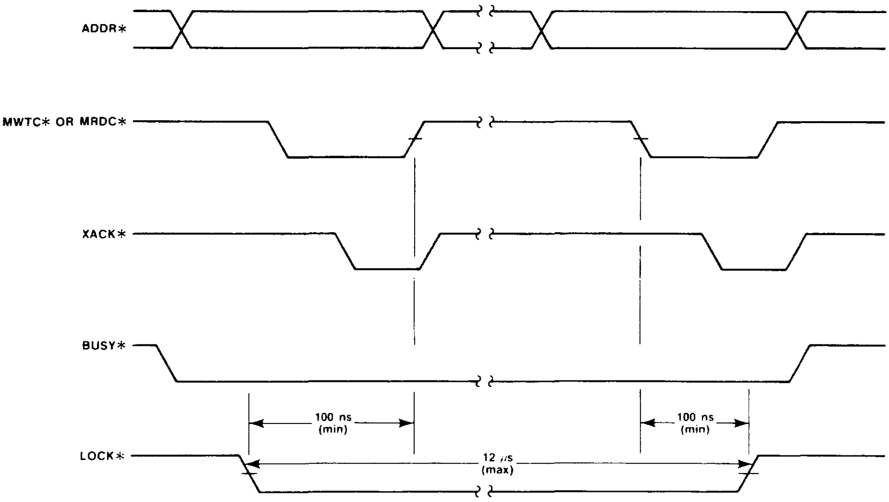


Figure 11 Lock Timing

2.3.1.1 INTERRUPT REQUEST LINES (INT0*-INT7*)

A set of interrupt request lines (INT0*-INT7*) is provided on the bus. An interrupt is generated by activating one of the eight interrupt request lines with an open-collector driver. All interrupts are level-triggered, rather than edge-triggered. Requiring no edge to trigger an interrupt allows several sources to be attached to each line. The interrupt request lines are prioritized, with INT0* having the highest priority and INT7* having the lowest priority.

2.3.1.2 INTERRUPT ACKNOWLEDGE (INTA*)

An interrupt acknowledge line (INTA*), driven by the bus master, requests the transfer of interrupt information on the bus. The specific information timed onto the bus depends on the implementation of the interrupt scheme. In general, the leading edge of INTA* indicates that the address bus is active; the trailing edge indicates that data is present on the data lines.

2.3.2 Classes of Interrupt Implementation

There are two types of interrupt implementation schemes: Non-Bus Vectored (NBV) and Bus Vectored (BV). The two schemes are explained in the following subsections.

2.3.2.1 NON-BUS VECTORED INTERRUPTS

Non-Bus Vectored (NBV) interrupts are those interrupts handled on the bus master that do not require the 796 Bus for transfer of the interrupt vector address. The interrupt vector address is generated by the interrupt controller on the master and transferred to the processor over the local bus. The slave modules generating the interrupts can reside on the master module or on other bus modules, in which case they use the 796 Bus interrupt request lines (INT0*-INT7*) to generate their interrupt requests to the bus master. When an interrupt request line is activated, the bus master performs its own interrupt operation and processes the interrupt. Figure 12 shows an example of NBV interrupt implementation.

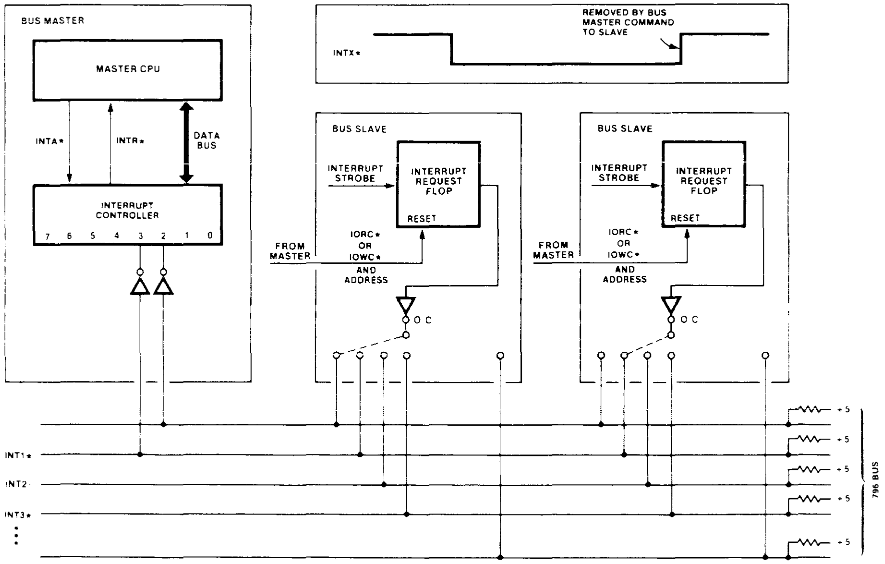


Figure 12 Non-Bus Vectored (BV) Interrupt Logic

2.3.2.2 BUS VECTORED INTERRUPTS

Bus Vectored (BV) interrupts are those interrupts that transfer the interrupt vector address over the 796 Bus from the slave to the bus master using the INTA* command signal.

When an interrupt request occurs, the interrupt control logic on the bus master interrupts its processor. The processor on the bus master generates the INTA command, freezing the state of the interrupt logic for priority resolution. The bus master also overrides (retains the bus between bus cycles) the 796 Bus to guarantee itself back-to-back bus cycles. After the first INTA* command, the bus master's interrupt control logic puts an interrupt code on the 796 Bus address lines. The interrupt code is the address of the highest priority active interrupt request line. At this point in the BV interrupt procedure, two different sequences can occur because the 796 Bus can support masters that generate either two or three INTA* commands.

If the bus master generates two INTA* commands, one more INTA* command will be generated. This second INTA* causes the bus slave interrupt control logic to transmit its interrupt vector address on the 796 Bus data lines. The address is used by the bus master to service the interrupt.

If the bus master generates three INTA* commands, two more INTA* commands will be generated. These two INTA* commands allow the bus slave to put its 2-byte interrupt vector address on the 796 Bus data lines (one byte for each INTA*). The interrupt vector address is used by the bus master to service the interrupt.

NOTE

> The 796 Bus can support only one type of Bus Vectored interrupt in a given system. However, the 796 Bus can support both Bus Vectored (BV) and Non-Bus Vectored (NBV) interrupts within the same system.
>
> Figure 13 depicts an example of the BV Interrupt implementation.

2.4 796 BUS EXCHANGE

The 796 Bus can accommodate several bus masters on the same system, each taking control of the bus as it needs to effect data transfers. The bus masters request bus control through a bus exchange sequence.

The discussion of the 796 Bus exchange will be separated into three parts. The first part explains the signals involved, the second part discusses the bus exchange priority techniques (serial and parallel), and the third part explains the implementation of the exchange logic.

2.4.1 796 Bus Exchange Signals

A set of six signals is used to implement the bus exchange operation. All bus exchange signals are synchronized by BCLK*.

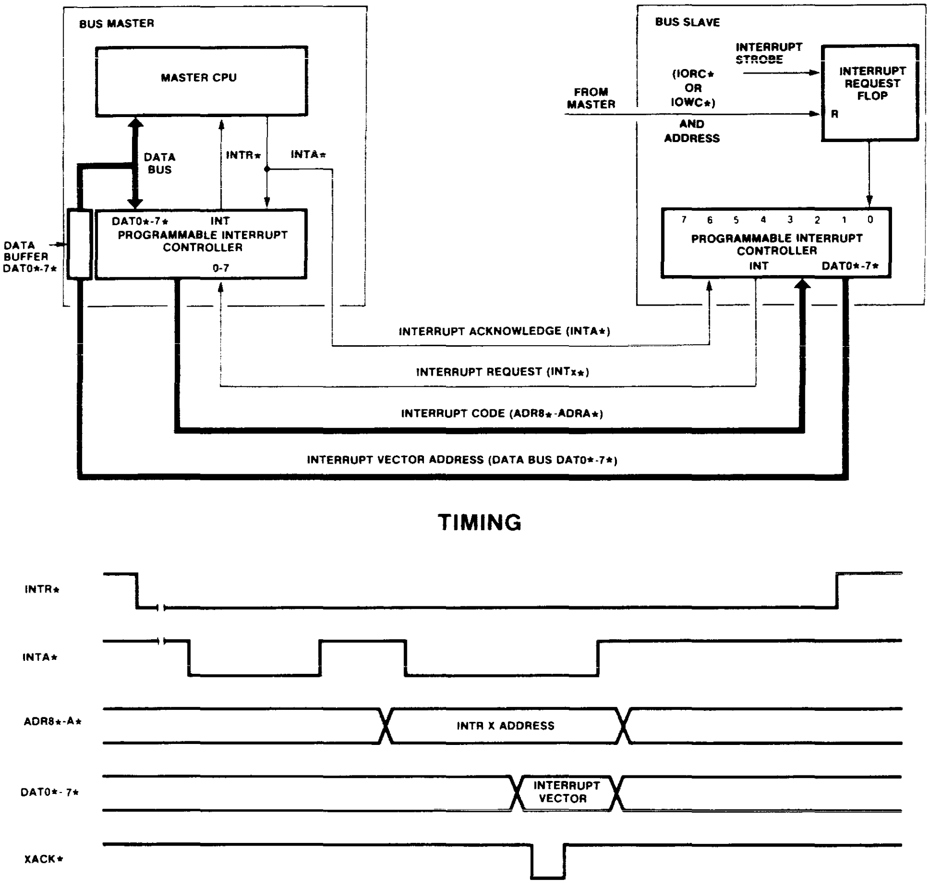


Figure 13 Bus Vectored (BV) Interrupt Logic

2.4.1.1 BUS CLOCK (BCLK*)

This periodic clock signal is used to synchronize the exchange logic, with synchronization occurring on the trailing (high-to-low) edge of the pulse. BCLK* has a duty cycle of approximately 50 percent, a maximum frequency of 10 MHz, and can be slowed, stepped, or stopped as called for by system design. There is no requirement for synchronization between BCLK* and CCLK*, but they may be derived from the same source. The BCLK* line is driven by a TTL clock driver.

2.4.1.2 BUS BUSY (BUSY*)

This signal is driven by the master in control of the bus. All other masters monitor BUSY* to determine the state of the bus. This bidirectional signal, which is driven by an open-collector gate, is synchronized by BCLK*.

2.4.1.3 BUS PRIORITY IN (BPRN*)

A non-bused signal that indicates to a master that no master of higher priority is requesting control of the bus. BPRN* is synchronized by BCLK* and driven by TTL gates. In a serial resolution scheme, this is the master's input from the priority chain. In a parallel resolution scheme, this is the master's input from the parallel priority circuit.

2.4.1.4 BUS PRIORITY OUT (BPRO*)

This non-bused signal, when activated by a bus master, indicates to the bus master of the next lower priority that it may gain control of the bus (i.e., no higher priority requests are pending for control of the bus). This signal is used only in a daisy-chained serial priority resolution scheme and should be connected to the Bus Priority In (BPRN*) input of the next lower priority bus master. BPRO* is driven by TTL gates and is synchronized by BCLK*.

NOTE

> Each bus master must allow its BPRO* signal to be disconnected from the BPRO* line on the 796 Bus so that, if desired, a parallel priority resolution scheme can be used. This capability is to allow some bus masters to have their BPRN* inputs driven by a central parallel resolution circuit instead of by the BPRO* of the next higher priority master.

2.4.1.5 BUS REQUEST (BREQ*)

The Bus Request (BREQ*) line is used with the parallel priority resolution scheme, and is a request of the master for 796 Bus control. The priorities of the BREQ* from each master are resolved in a parallel priority resolution circuit. The highest priority request enables the BPRN* input of that master, allowing it to gain control of the bus. BREQ* is synchronized by BCLK* and is a TTL output.

2.4.1.6 COMMON BUS REQUEST (CBRQ*) (OPTIONAL)

Any master that wants control of the 796 Bus but does not control it, can activate CBRQ* with an open-collector gate. If CBRQ* is high, it indicates to the bus master that no other master is requesting the bus and therefore the present bus master can retain the bus. There are times when this can save the bus exchange overhead for the current master. This is because quite often when a master is controlling the bus, there are no other masters that are requesting the bus. Without CBRQ*, only BPRN* indicates whether or not another master is requesting the bus and, for BPRN*, only if the other master is of higher priority. Between the master's bus transfer cycles, in order to allow lower priority masters to take the bus if they need it, the master must give up the bus. At the start of the master's next transfer cycle, the bus must be regained. If no other master has the bus, this can take approximately three BCLK* periods. To avoid this overhead of unnecessarily giving up and regaining the bus when no other masters need it, CBRQ* may be used. Any master that wants but does not have the bus, must drive this line low (true). The master that has the bus can, at the end of a transfer cycle, sense CBRQ*. If it is not low, then the bus does not have to be released, thereby eliminating the delay of regaining the bus at the start of the next cycle. (At any time before the master's next cycle, any other master desiring the bus will drive CBRQ* and cause the master to relinquish the bus at the time.)

Masters that use CBRQ* must be able to disable that function such that they can be used with masters that do not generate the CBRQ* signal.

2.4.2 Bus Exchange Priority Techniques

Two bus exchange priority techniques are discussed: a serial technique and a parallel technique. Figure 14 and Figure 15 illustrate these two techniques. Note that the parallel and serial schemes are compatible and therefore can be combined and used together on the same bus. The bus exchange implementation discussed in 2.4.3 is the same for both techniques.

2.4.2.1 SERIAL PRIORITY TECHNIQUE

Serial priority resolution is accomplished with a daisy-chain technique (See Figure 14). With such a scheme, the bus priority output (BPRO*) of each master is connected to the bus priority input (BPRN*) of the next lower priority master. The BPRN* of the highest priority master in the serial chain shall either be always active or connected to a central Bus Arbiter as described in 2.4.2.2. The latter connection would be used if a parallel-serial priority structure were used.

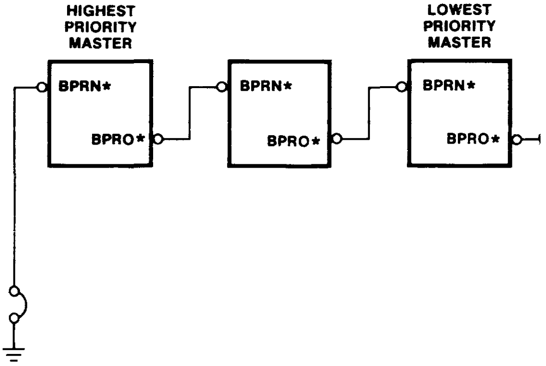


Figure 14 Serial Priority Technique

Serial priority resolution is accomplished in the following manner. The BPRO* output for a particular master is asserted if and only if its BPRN* input is active and that master is not requesting control of the bus. Thus, if a master requests control of the bus, it shall set its BPRO* high, which in turn disables the BPRN* of all lower priority masters. The number of masters that can be linked in a serial chain is limited by the fact that the BPRN* signal must propagate through the entire chain within one BCLK* cycle. If the maximum BCLK* of 10 MHz is used, then the number of masters in a serial chain is limited to three.

2.4.2.2 PARALLEL ARBITRATION TECHNIQUE

In the parallel technique, the bus allocation is determined by a Bus Arbiter (See Figure 15). This may be a priority scheme, which determines the next master by a fixed priority structure or some other mechanism for allocation (e.g., sequential). The BREQ* lines are used by the arbiter to signal the next master on the appropriate BPRN* line. The BPRO* lines are not used in the parallel allocation BPRN* scheme.

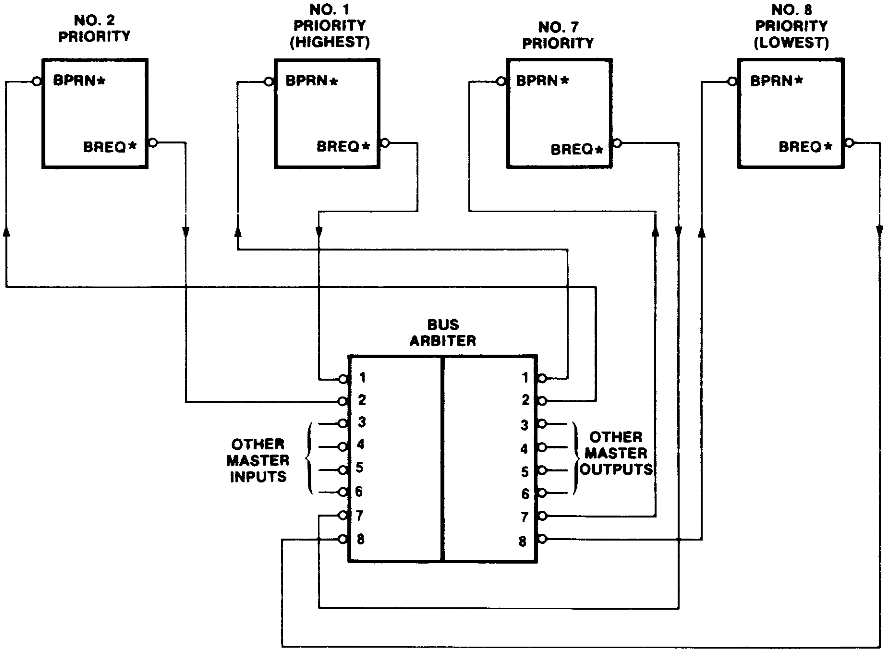


Figure 15 Parallel Priority Technique

CHAPTER 3
ELECTRICAL SPECIFICATIONS

This section presents the electrical specifications for the 796 Bus as follows:

(1) General bus considerations of the state relationships, signal line characteristics, and power supplies

(2) Timing specifications for the bus signals

(3) Specifications for the signal line drivers and receivers, as well as the electrical termination requirements

When electrical specifications indicate minimum or maximum values for the bus, they must be measurable at any point on the bus.

Note that a particular implemented bus could have any amount of bus propagation delay and ringing (before setup times), as long as all bus parameters (e.g., setup, hold, and other times) are met at all points on the bus. However, to facilitate the design of a compatible set of modules (masters and slaves) that use the bus, the standard maximum bus propagation delay will be specified as t_{PD} (max).

3.1 GENERAL BUS CONSIDERATIONS

3.1.1 Logical and Electrical State Relationships

The signal names indicate whether or not the signal lines on the 796 Bus are active high or active low. If the signal name ends with a nathan ("*", a non-superscript asterisk), then the signal is active low and its logical-electrical state relationship for that signal is:

Logical State	Electrical Signal Level	At Receiver	At Driver
0	H=TTL High state	5.25V≥H≥2.0V	5.25V≥H≥2.4V
1	L=TTL Low state	.8V≥L≥−.5V	.5V≥L≥0V

If the signal name has no nathan (no "*"), then the signal is active high and its logical-electrical state relationship for that signal is:

Logical State	Electrical Signal Level	At Receiver	At Driver
0	L=TTL Low state	.8V≥L≥−.5V	.5V≥L≥0V
1	H=TTL High state	5.25V≥H≥2.0V	5.25V≥H≥2.4V

These specifications are based on TTL, where the power source is 5V±5%, referenced to logic ground (GND).

When specified, current flow into a node has a positive sign and current flow out of a node has a negative sign.

3.1.2 Signal Line Characteristics

The following subsections describe two types of requirements. The first includes the requirements on the signal line that are measured when the signal line is in use. The second type includes those that are measured under special test conditions.

3.1.2.1 IN-USE SIGNAL LINE REQUIREMENTS

During normal use, the rise and fall times of the signals depend on which type of driver is used (See 3.3). Typical rise and fall times are:

	Open Collector	Totem Pole	3-State
Rise Time	—	10 ns	10 ns
Fall Time	10 ns	10 ns	10 ns

The maximum signal propagation delay on the bus is t_{PD} (max). This is measured from the edge of any one board plugged into the 796 Bus to any other board plugged into the 796 Bus:

$$t_{PD}\ (\text{max}) = 3\ \text{ns}$$

These dynamic signal parameters can be tested by using 74S20 gates as drivers.

NOTE

For all boards plugged into the bus, the setup, hold, or any other times are measured at the edge of the board where it is plugged into the bus. This means that all board-internal delays must be taken into account, while still providing for the setup, hold, and other times.

After Power-Up, the following specifications apply:

(1) Bus termination required for each signal line (See 3.3).

(2) Setting time for all command line signals (See 2.2.2.5) after a transition is zero.

On these lines the ringing cannot go beyond the noise immunity levels — i.e., high, minimum; or low, maximum. This also applies to the Transfer Acknowledge and Inhibit lines.

For all address lines (see 2.2.2.3) the signals must be stable (settled) at least 50 nanoseconds before any command line goes active (setup time). This settling requirement means there can be no ringing beyond the noise immunity levels (high, minimum; low, maximum). These requirements also apply to the data lines (See 2.2.2.4) during any write operations.

For all data lines during read operations, the setup time is zero before the Transfer Acknowledge (XACK*) signal goes active; and the hold time is zero after the read-type command goes inactive.

The setup, hold, and command ringing are summarized and graphically presented in Figure 16.

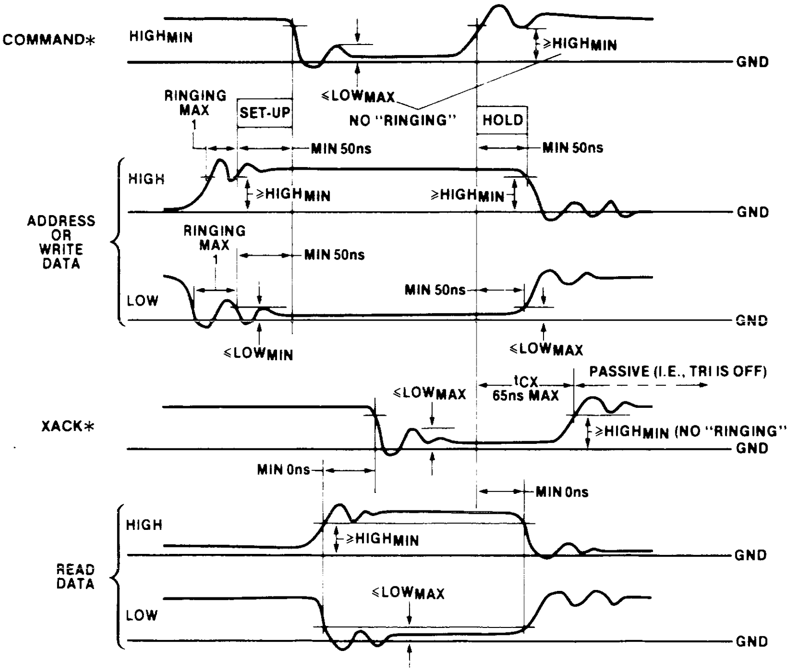


Figure 16 Setup, Hold, and Command Ringing Summary

3.1.2.2 BACKPLANE SIGNAL TRACE CHARACTERISTICS

Requirements for line-to-line coupling characteristics are shown in Figure 17. The specific test conditions under which the specifications are to be met are also shown.

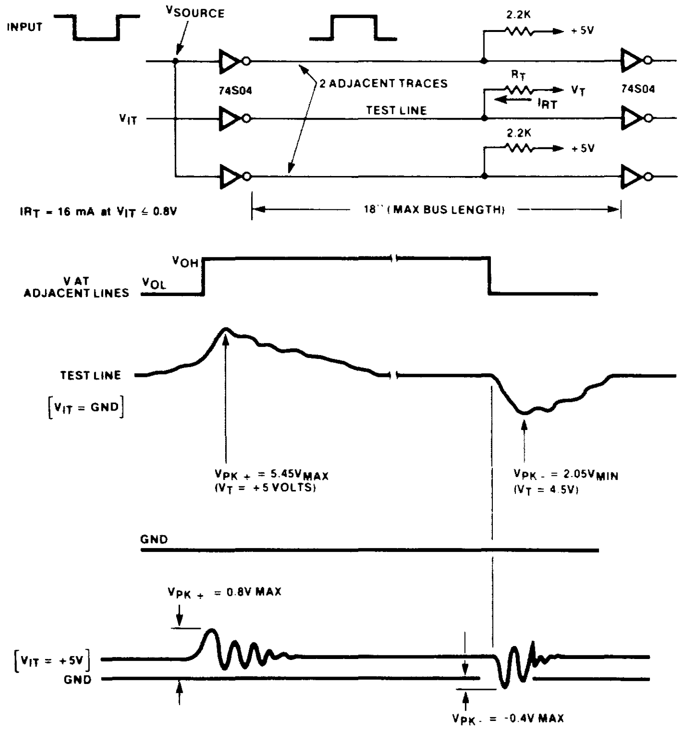


Figure 17 Line-to-Line Coupling Characteristics

3.1.3 Power Supply Specifications

Table 1 provides all power supply specifications. All voltages not shown in Table 1 that may be required on a board plugging into the 796 Bus should be derived from one of the standard voltages (+5V, +12V, −12V).

Table 1 796 Bus Power Supply Specifications

	Standard[1]			
Parameter	**Ground**	**+5**	**+12**	**−12**
Mnemonic	GND	+5V	+12V	−12V
Bus Pins	P1-1, 2, 11, 12, 75, 76, 85, 86	P1-3, 4, 5, 6, 81, 82, 83, 84	P1-7,8	P1-79,80
Toierance	Ref.	±1%	±1%	±1%
Combined Line & Load Reg	Ref.	0.1%	0.1%	0.1%
Ripple (Peak to Peak)	Ref.	50 mV	50 mV	50 mV
Transient Response (50% Load Change)		100 μs	100 μs	100 μs

[1] Point of measurement is at connection point between motherboard and power supply. At any card edge connector a degradation of 2% maximum (e.g. voltage tolerance ±2%) is allowed.

3.1.4 Temperature and Humidity

Bus specifications should be met with temperature and humidity within the following ranges:

Temperature:	0-55°C (32-150°F); free-moving air across modules and bus
Relative Humidity:	90% maximum (no condensation)

This represents standard environment for the 796 Bus. It may be desirable to create more (or less) severe environmental restrictions in some applications.

3.2 TIMING

This subsection describes all timing specifications on the 796 Bus. It does not present descriptions or functional relationships (which are given in Section 2); however, this section does imply the functionality when relating two signals.

Table 2 summarizes the timing specifications in this section. For detailed descriptions, refer to the specific subsection(s) in the right-hand column.

The timing diagrams shown in this section usually show the minimum or maximum values required for each parameter. However, for clarity, the diagrams in this specification do not usually show both the minimum and maximum parameters. For this reason, the bus timing specification (Table 2) should be referenced for complete information. The timing diagram show how all of the parameters are defined in relation to the signals involved.

Table 2 796 Bus Timing Specifications Summary

Parameter	Description	Minimum	Maximum	Units	Reference
t_{AH}	Address Hold Time	50		ns	3.2.1, 3.2.2, 3.2.4
t_{AIZ}	Address to Inhibit High Delay	0	100	ns	3.2.3
t_{AS}	Address Setup Time (at slave board)	50		ns	3.2.1, 3.2.2, 3.2.4
t_{BCY}	BCLK* Period	100	∞	ns	3.2.5
t_{BPRNO}	BPRN* to BPRO*	0	30	ns	3.2.5
t_{BPRNS}	BPRN* to ↓BCLK* Setup Time	22		ns	3.2.5
t_{BPRO}	↓BCLK* to BPRO*	0	40	ns	3.2.5
t_{BPREQH}	↓BLCK* to BREQ* High Delay	0	35	ns	3.2.5
t_{BPREQL}	↓BCLK* to BREQ* Low Delay	0	35	ns	3.2.5
t_{BSYO}	CBRQ*●BUSY* to ↑BUSY	—	12	μs	3.2.5
t_{BUSY}	BUSY* delay from ↓BCLK*	0	70	ns	3.2.5
t_{BUSYS}	BUSY* to ↓BCLK Setup Time	25		ns	3.2.5
t_{BW}	BCLK* Width	$0.35t_{BCY}$	$0.65t_{BCY}$		3.2.5
t_{CBRO}	↓BCLK* to CBRQ*	0	60	ns	3.2.5
t_{CBRQS}	CBRQ* to ↓BCLK* Setup Time	35		ns	3.2.5
t_{CCY}	CCLK* Period	100	110	ns	3.2.6
t_{CMD}	Command Pulse Width	100	t_{TOUT}	ns	3.2.1, 3.2.2
t_{CMPH}	Command Hold Time	20		ns	3.2.1, 3.2.2
t_{CPM}	Central Priority Module Resolution Delay (parallel priority)	0	$t_{BCY}-t_{BREQ}$ $-2t_{PD}-t_{BPRNS}$ $-t_{SKEW}$		3.2.5
t_{CESP}	Command Separation	100		ns	3.2.4, 3.2.6
t_{CW}	CCLK* Width	$0.35t_{CCY}$	$0.65t_{CCY}$	ns	3.2.6

Table 2 796 Bus Timing Specifications Summary (continued)

Parameter	Description	Minimum	Maximum	Units	Reference
t_{DHR}	Read Data Hold Time	0	65	ns	3.2.1, 3.2.4
t_{DHW}	Write Data Hold Time	50		ns	3.2.2
t_{DS}	Write Data Setup Time	50		ns	3.2.2
t_{DXL}	Read Data Setup Time to XACK*	0		ns	3.2.1, 3.2.4
t_{IAD}	XACK* Disable from Inhibit (internal parameter on an inhibited slave; used to determine t_{XACKA} min.)	0	100 (arbitrary)	ns	2.3.2
t_{ID}	Inhibit Delay	0	100 (recommend <100 ns)	ns	3.2.3
t_{INIT}	INIT* Width	5		ms	3.2.6, 3.2.7
t_{INTA}	INTA* Width	250		ns	3.2.4
t_{LCKH}	LOCK* hold time from command active	100		ns	3.2.6
t_{LCKS}	LOCK* to command setup time	100		ns	3.2.6
t_{LOCK}	LOCK* Width		12	μs	3.2.6
t_{OUT}	Timeout Delay	1	dc (∞)	ms	—
t_{PD}	Standard Bus Propagation Delay		3	ns	3.1.2, 3.2.5
t_{SKEW}	BCLK* Skew		t_{PD}		3.2.5
t_{XACK}	XACK* Time (for slaves without inhibit function)	0	8	μs	3.2.1, 3.2.2, 3.2.4
t_{XACKA}	XACK* Time of an Inhibited Slave	t_{IAD} +50 ns	1500	ns	3.2.3
t_{XACKB}	XACK* Time of an Inhibiting Slave	1500	8000	ns	3.2.3
t_{XAH}	XACK* Hold Time	0	65	ns	3.2.1, 3.2.2, 3.2.4
Serial Priority	See 3.2.5				

3.2.1 Read Operations (I/O and Memory)

A read operation transfers data from memory or from I/O to the master that is controlling the bus (see 2.2). The lines involved and timing specifications for a read operation are shown in Figure 18. See the special inhibit operation in 3.2.3.

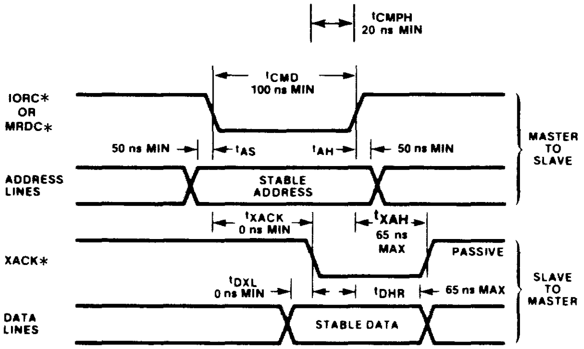


Figure 18 Read AC Timing

3.2.2 Write Operations (I/O and Memory)

A write operation transfers data from the master controlling the bus to memory or I/O (see 2.2). Timing for a write operation is shown in Figure 19. See 3.2.3 for inhibit operation.

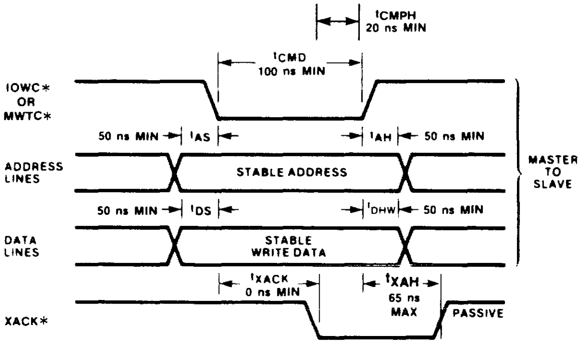


Figure 19 Write AC Timing

3.2.3 Inhibit Operations

An inhibit operation may accompany any memory read or memory write operation. The main effect is for one slave to inhibit another slave from driving the data lines and from returning (driving) an acknowledge (XACK*). I/O address cannot be inhibited. Although inhibit signals may be generated during IORC*, IOWC*, or INTA* operations, these signals are ignored by other slaves (including the slave that should respond to the INTA*, IORC*, or IOWC*). Inhibit timing is illustrated in Figure 20. Related subsections are:

Subsection	Number
Functional Descriptions	2.1.3.2.3
Timing Specification Summary	3.2
Read Operations	3.2.1
Write Operations	3.2.2
Interrupt Implementations	3.2.4

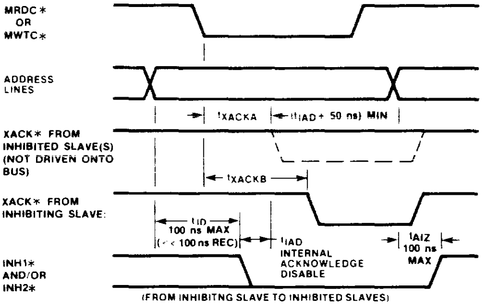


Figure 20 Inhibit AC Timing

3.2.4 Interrupt Implementations

There are two types of interrupt implementation schemes: Non-Bus Vectored (NBV) and Bus Vectored (BV).

3.2.4.1 NBV INTERRUPTS

NBV interrupts are handled on the bus master and do not require the 796 Bus for transfer of an interrupt vector address. The slave modules generating the interrupts may reside on the master module or on other bus modules, in which case they use the 796 Bus interrupt request lines (INT0*-INT7*) to generate interrupt requests to the bus master. When an interrupt request line is activated, the bus master performs its own internal interrupt operations and then processes the interrupt.

3.2.4.2 BV INTERRUPTS

BV interrupts are those interrupts that transfer the interrupt vector address along the 796 Bus from the slave to the bus master in response to the INTA* command signal. BV interrupt timing is shown in Figure 21.

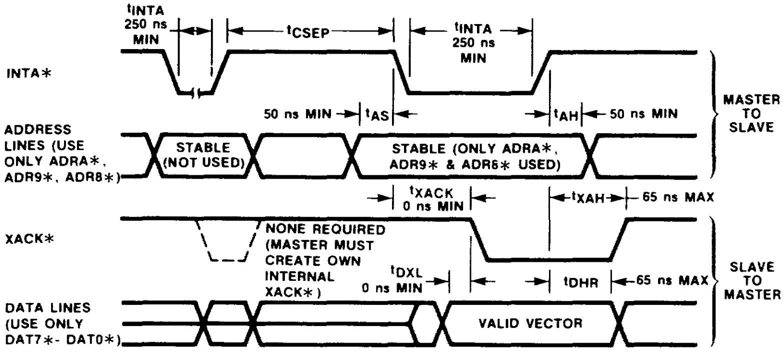


Figure 21 Bus Vectored (BV) Interrupt AC Timing

When an interrupt request occurs, the interrupt control logic on the bus master interrupts its processor. The processor on the bus master generates an INTA* command, which freezes the state of the interrupt logic on the 796 Bus for priority resolution. The bus master also locks the 796 Bus (retains the bus between bus cycles) to guarantee itself back-to-back bus cycles. After the first INTA* command, the bus master's interrupt control logic puts an interrupt code onto the 796 Bus address lines. The interrupt code is the address of the highest priority active interrupt request line. At this point in the BV interrupt procedure, two different sequences could take place. The difference occurs because the 796 Bus can support masters that generate either two or three INTA* commands during the interrupt process.

If the bus master generates two INTA* commands, one more INTA* command will be generated. This second INTA* causes the bus slave interrupt control logic to transmit its interrupt vector address on the 796 Bus data lines. The address is used by the bus master to service the interrupt.

If the bus master generates three INTA* commands, two more INTA* commands will be generated. These two INTA* commands allow the bus slave to put its 2-byte interrupt vector address onto the data lines (one byte for each INTA* command). The interrupt vector address is used by the bus master to service the interrupt.

NOTE

The 796 Bus can support only one type of BV interrupt in a given system. However, it can support both BV and NBV interrupts in the same system.

Subsections related to BV and NBV interrupts are:

Subsection	Number
Functional Descriptions	2.3.2.2
Timing Specification Summary	3.2

3.2.5 Bus Control Exchanges

A bus control exchange takes control of the bus (i.e., the ability to do read, write, and interrupt acknowledge operations) from one master and gives it to another master. See 2.4 for a functional description of this process.

For a master that does not use the bus signal CRBQ* (Common Bus Request), the timing specifications in Figure 22 apply.

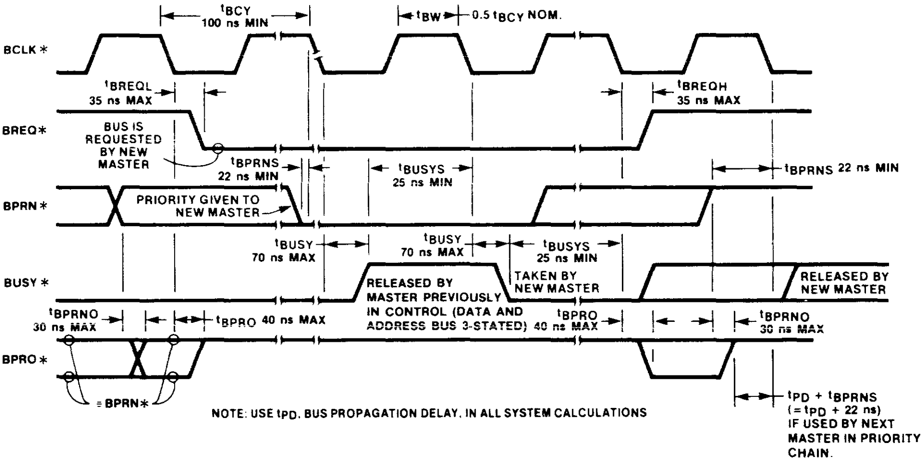


Figure 22 Bus Exchange AC Timing

For a system using CRBQ* (Common Bus Request), each master must also satisfy the timing requirements illustrated in Figure 23. Note that before "releasing the bus" (i.e., releasing BUSY*), the hold times, etc., of any ending cycle must still be met as described in the previous subsections of this chapter. Likewise, after "taking the bus" (i.e., driving BUSY* low), it is necessary to satisfy all applicable setup and other timing parameters for a cycle just beginning.

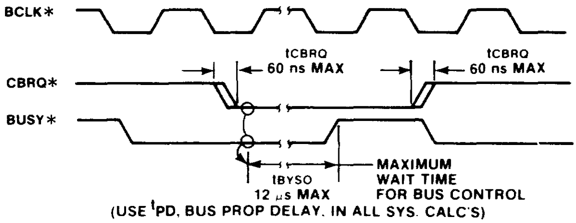


Figure 23 Common Bus Request AC Timing

3.2.5.1 SERIAL PRIORITY

For a system that uses a serial priority scheme (i.e., daisy-chain BPRO*s to BPRN*s) (see 2.4), the timing specifications in Figure 24 apply.

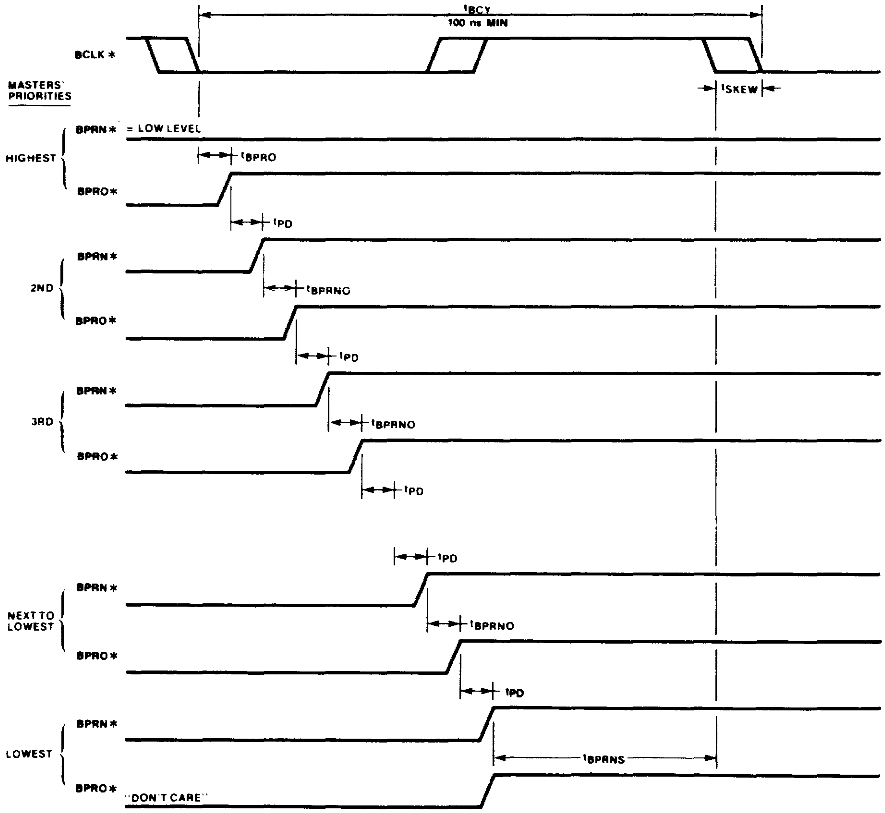
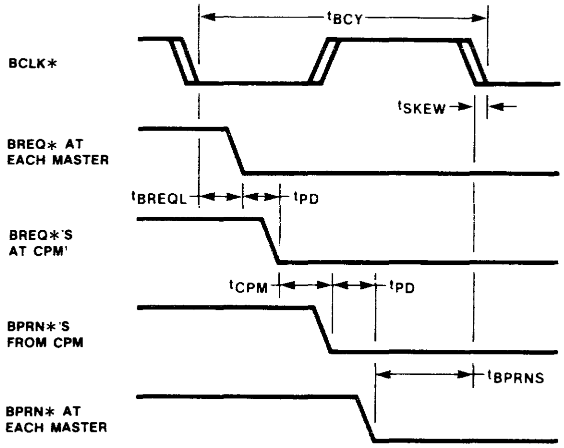


Figure 24 Serial Priority AC Timing

3.2.5.2 PARALLEL PRIORITY

For a system that uses a parallel priority scheme (i.e., a central priority resolver) (see 2.4), the following system and CPM (Central Priority Module) timing specifications of Figure 25 apply.



SPEC:
$t_{CPM(MAX)} \leqslant t_{BCY_{MIN}}$ $t_{BREQ_{MAX}}$ $2t_{PD_{MAX}}$ $t_{BPRNS_{MAX}}$ $t_{SKEW_{MAX}}$

[1] CPM IS CENTRAL PRIORITY MODULE

Figure 25 Parallel Priority AC Timing

3.2.6 Miscellaneous Timing

The timing diagrams in Figures 26, 27, 28, and 29 show the timing of Constant Clock (CCLK*), Command Separation (t_{CSEP}), Initialize (t_{INIT}), and Lock (LOCK*), respectively.

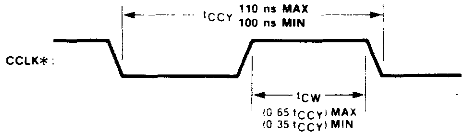


Figure 26 Constant Clock AC Timing

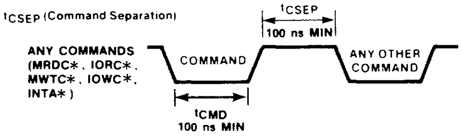

tCSEP (Command Separation)
tCSEP
100 ns MIN
ANY COMMANDS (MRDC*, IORC*, MWTC*, IOWC*, INTA*)
COMMAND
ANY OTHER COMMAND
tCMD
100 ns MIN

Figure 27 Command Separation AC Timing

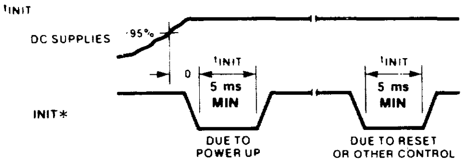


Figure 28 Initialize AC Timing

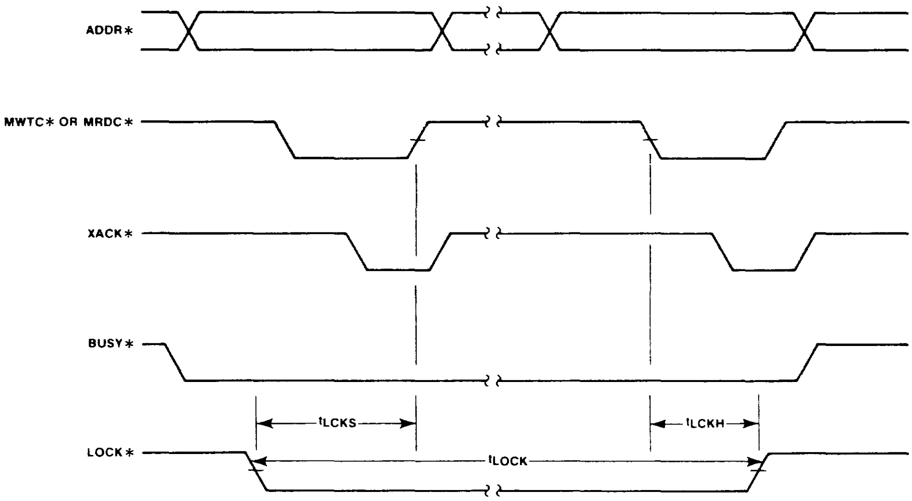


Figure 29 Lock AC Timing

3.3 RECEIVERS, DRIVERS AND TERMINATIONS

Non-timing specifications unique to each signal line or to groups of signal lines are presented in Table 3. The requirements for the signal line receivers, drivers, and bus terminations, and the locations of the receiver, driver, and termination for each signal are given.

Table 3 Bus Drivers, Receivers, and Terminations

	Driver[1,3]						Receiver[2,3]				Termination			
Bus Signals	Location	Type	I_{OL} Min_{mA}	I_{OH} $Min_{\mu a}$	I_{OH} $Max_{\mu a}$	C_O Min_{pf}	Location	I_{IL} Max_{mA}	I_{IH} $Max_{\mu a}$	C_I Max_{pf}	Location[5]	Type	R	Units
DAT0* - DATF* (16 lines)	Masters & Slaves	TRI	16	−2000	—	300	Masters & Slaves	−0.8	125	18	1 place	Pullup	2.2	KΩ
ADR0* - ADR17* BHEN* (25 lines)	Masters	TRI	16	−2000	—	300	Slaves	−0.8	125	18	1 place	Pullup	2.2	KΩ
MRDC*, MWTC*	Masters	TRI	32	−2000	—	300	Slaves (Memory; Memory-Mapped I/O)	−2	125	18	1 place	Pullup	1	KΩ
IORC*, IOWC*	Masters	TRI	32	−2000	—	300	Slaves (I/O)	−2	125	18	1 place	Pullup	1	KΩ
XACK*	Slaves	TRI	32	−400	—	300	Masters	−2	125	18	1 place	Pullup	510	Ω
INH1*, INH2*	Inhibiting Slaves	OC	16	—	−250	300	Inhibited Slaves (RAM, PROM, ROM, Memory-Mapped I/O)	−2	50	18	1 place	Pullup	1	KΩ
BCLK*	1 place (Master usually)	TTL	48	−3000	—	300	Masters	−2	125	18	Mother-board	To +5V To GND	220 330	Ω Ω
BREQ*	Each Master	TTL	10	−200	—	60	Central Priority Module	−2	50	18	Central Priority Module (not req.)	Pullup	1	KΩ
BPRO*	Each Master	TTL	3.2	−200	—	60	Next Master in Serial Priority Chain at its BPRN*	−1.6	50	18	(not req.)			
BPRN*	Parallel: Central Priority Module Serial: Prev Masters BPRO*	TTL	16	−400	—	300	Masters	−4	100	18	(not req.)			
LOCK* BUSY* CBRQ*	Master All Masters	TRI OC	32 20	−2000 —	— −250	300 300	All All Masters	−2 −2	125 50	18 18	1 place 1 place	Pullup Pullup	1 1	KΩ KΩ
INIT*	Master	OC	32	—	−250	300	All	−2	50	18	1 place	Pullup	2.2	KΩ
CCLK*	1 place	TTL	48	−3000	—	300	Any	−2	125	18	Mother-board	To +5V To GND	220 330	Ω Ω
INTA*	Masters	TRI	32	−2000	—	300	Slaves (Interrupt-ing I/O)	−2	125	18	1 place	Pullup	1	KΩ
INT0*-INT7* (8 lines)	Slaves	OC	16	—	−250	300	Masters	−1.6	40	18	1 place	Pullup	1	KΩ

[1] Driver Requirements:
I_{OH} = High Output Current Drive
I_{OL} = Low Output Current Drive
C_O = Capacitance Drive Capability
TRI = 3-State Drive
OC = Open Collector Driver
TTL = Totem-pole Driver

[2] Receiver Requirements
I_{IH} = High Input Current Load
I_{IL} = Low Input Current Load
C_I = Capacitive Load

[3] For low and high voltage specifications see 3.1.1.

[4] ±5%, ¼W Resistors

[5] All termination resistors specified as "1 place" are typically located on the motherboard.

CHAPTER 4
MECHANICAL SPECIFICATIONS

This section describes all the physical and mechanical specifications that a designer must be concerned with when designing a 796 Bus backplane or when designing printed circuit boards that will plug into the 796 Bus interface.

4.1 BACKPLANE CONSIDERATIONS

This section is a discussion of the things that the designer must consider when designing a 796 Bus backplane.

The maximum length of the backplane connecting modules is 18 inches. Extender boards used within the system will not be supported by the bus unless the overall resulting length of the bus including the extender card is less than the 18-inch maximum.

4.1.1 Board to Board Relationships

The following printed circuit board specifications must be adhered to when designing 796 Bus compatible boards which are to operate in a 0.6-inch board to board spacing backplane.

a. *Board to Board Spacing (L_C)* — center to center of boards when plugged into backplane must be at least 0.6 inches ±0.02.

b. *Board Thickness (L_T)* — the typical board thickness is 0.062 ±0.005 inches.

c. *Component Lead Length (L_L)* — the length of the component leads below the printed circuit board cannot exceed 0.093 inches.

d. *Component Height (L_H* — the following equation is used to determine the maximum height of the components above the printed circuit board:

$L_H < L_C - L_T - L_L$

$L_H < 0.58'' - 0.067'' - 0.093''$

$L_H < 0.420$ inches (including board warpage)

Electrically conductive components require L_H to be decreased to 0.40 inches.

An example of a typical backplane and the components necessary to implement it are shown in Figure 31.

This section contains only the mechanical specifications for designing a 796 Bus interface. The designer must also take into consideration the electrical specifications in Section 3.

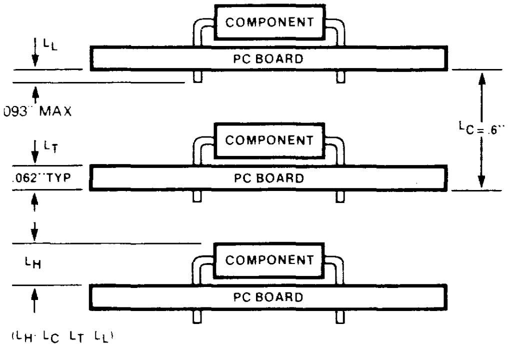
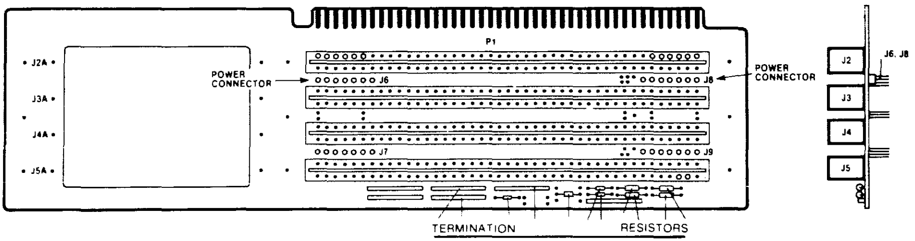


Figure 30 796 Bus Backplane Card to Card Separation



PARTS LIST

1 PWB TERMINATION BACKPLANE
2 7 POST WAFER CONNECTORS (.156 PIN CENTERS) (J6 AND J8)
4 EDGE BOARD CONNECTORS 43/86 PINS ON .156" CENTERS (J2-J5)
12 WIRE WRAP POSTS
4 10 PIN, 2.2K, 9 RES, 1.5W RESISTOR PACKS (RP1-RP4)
1 10 PIN, 1K, 9 RES, 1.5W RESISTOR PACK (RP5)
1 10 PIN, 1.1K, 9 RES 1.5W RESISTOR PACK (RP6)
1 1K RESISTOR, 1/8W, ±5% (R1)
1 2.2K RESISTOR, 1/8W, ±% (R5)
2 220Ω RESISTORS, 1/4W ±5% (R9, R11)
2 330Ω RESISTORS, 1/4W ±5% (R10, R12)
2 510Ω RESISTORS 1/8W ±5% (R7 R8)

Figure 31 Typical 796 Bus Backplane

4.1.2 796 Bus Pin Assignments

Printed circuit boards which are designed to interface with the 796 Bus have two connectors which plug into the backplane. P1 (Primary) and P2 (Auxiliary). Table 4 shows the pin/signal assignments for the P1 connector on the printed circuit boards. Reserved signals on the P1 connector must be bussed as normal signal lines on the backplane. Table 5 shows the pin/signal assignments for the P2 connector on the printed circuit boards. If a backplane is used then the "Reserved and bussed" signals must be bussed as normal signal lines, and the "Reserved but not bussed" signals shall have no connections.

4.2 796 BUS BOARD FORM FACTORS

Certain 796 Bus characteristics must be taken into consideration when designing printed circuit boards that interface to it. The designer will ensure himself of 796 Bus compatibility if the specifications discussed in this chapter are followed.

4.2.1 Connector Naming and Pin Numbering Standards

The connectors on the printed circuit boards designed for the 796 Bus interface should adhere to the following standards (See Figure 31).

a. The connectors on the bus side of the board will be called P1, P2, P1 is the 86 pin main connector, and P2 is the 60 pin auxiliary connector.

b. Mating connectors on the motherboard (796 Bus) backplane will be called J1, J2, etc.

c. Pins should be numbered with odd number pins on the component side of the board, and in ascending order when going counterclockwise around the board (as shown in Figure 32).

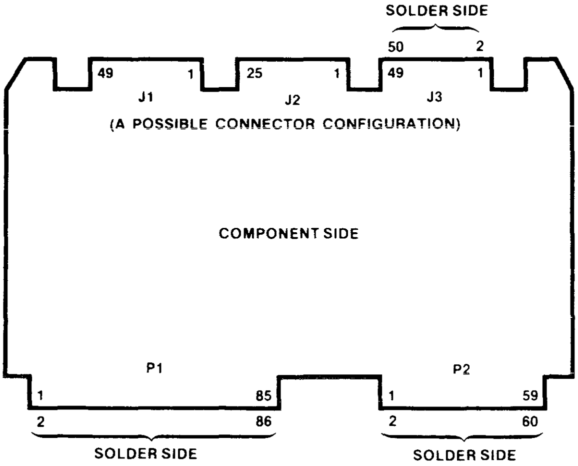


Figure 32 Connector and Pin Numbering

Table 4 Pin Assignment of Bus Signals on 796 Bus Board Connector (P1)

	Pin	(Component Side)		Pin	(Circuit Side)	
		Mnemonic	Description		Mnemonic	Description
Power	1	GND	Signal GND	2	GND	Signal GND
Supplies	3	+5V	+5Vdc	4	+5V	+5Vdc
	5	+5V	+5Vdc	6	+5V	+5Vdc
	7	+12V	+12Vdc	8	+12V	+12Vdc
	9		Reserved, bussed	10		Reserved, bussed
	11	GND	Signal GND	12	GND	Signal GND
Bus	13	BCLK*	Bus Clock	14	INIT*	Initialize
Controls	15	BPRN*	Bus Pri. In	16	BPRO*	Bus Pri. Out
	17	BUSY*	Bus Busy	18	BREQ*	Bus Request
	19	MRDC*	Mem Read Cmd	20	MWTC*	Mem Write Cmd
	21	IORC*	I/O Read Cmd	22	IOWC*	I/O Write Cmd
	23	XACK*	XFER Acknowledge	24	INH1*	Inhibit 1 (disable RAM)
Bus	25	LOCK*	Lock	26	INH2*	Inhibit 2 (disable PROM or ROM)
Controls	27	BHEN*	Byte High Enable	28	AD10*	
and	29	CBRQ*	Common Bus Request	30	AD11*	Address
Address	31	CCLK*	Constant CLK	32	AD12*	Bus
	33	INTA*	Intr Acknowledge	34	AD13*	
Interrupts	35	INT6*	Parallel	36	INT7*	Parallel
	37	INT4*	Interrupt	38	INT5*	Interrupt
	39	INT2*	Requests	40	INT3*	Requests
	41	INT0*		42	INT1*	
Address	43	ADRE*		44	ADRF*	
	45	ADRC*		46	ADRD*	
	47	ADRA*	Address	48	ADRB*	Address
	49	ADR8*	Bus	50	ADR9*	Bus
	51	ADR6*		52	ADR7*	
	53	ADR4*		54	ADR5*	
	55	ADR2*		56	ADR3*	
	57	ADR0*		58	ADR1*	
Data	59	DATA*		60	DATF*	
	61	DATC*		62	DATD*	
	63	DATA*	Data	64	DATB*	Data
	65	DAT8*	Bus	66	DAT9*	Bus
	67	DAT6*		68	DAT7*	
	69	DAT4*		70	DAT5*	
	71	DAT2*		72	DAT3*	
	73	DAT0*		74	DAT1*	

Table 4 Pin Assignment of Bus Signals on 796 Bus Board Connector (P1) (cont.)

	Pin	(Component Side)		Pin	(Circuit Side)	
		Mnemonic	Description		Mnemonic	Description
Power Supplies	75	GND	Signal GND	76	GND	Signal GND
	77		Reserved, Bussed	78		Reserved, bussed
	79	−12V	−12Vdc	80	−12V	−12Vdc
	81	+5V	+5Vdc	82	+5V	+5Vdc
	83	+5V	+5Vdc	84	+5V	+5Vdc
	85	GND	Signal GND	86	GND	Signal GND

All Reserved pins are reserved for future use and should not be used if upwards compatibility is desired.

Table 5 Pin Assignment of Bus Signals on 796 Bus Board Connector (P2)

	Pin	(Component Side)		Pin	(Circuit Side)	
		Mnemonic	Description		Mnemonic	Description
	1		Reserved, Not Bussed	2		Reserved, Not Bussed
	3		Reserved, Not Bussed	4		Reserved, Not Bussed
	5		Reserved, Not Bussed	6		Reserved, Not Bussed
	7		Reserved, Not Bussed	8		Reserved, Not Bussed
	9		Reserved, Not Bussed	10		Reserved, Not Bussed
	11		Reserved, Not Bussed	12		Reserved, Not Bussed
	13		Reserved, Not Bussed	14		Reserved, Not Bussed
	15		Reserved, Not Bussed	16		Reserved, Not Bussed
	17		Reserved, Not Bussed	18		Reserved, Not Bussed
	19		Reserved, Not Bussed	20		Reserved, Not Bussed
	21		Reserved, Not Bussed	22		Reserved, Not Bussed
	23		Reserved, Not Bussed	24		Reserved, Not Bussed
	25		Reserved, Not Bussed	26		Reserved, Not Bussed
	27		Reserved, Not Bussed	28		Reserved, Not Bussed
	29		Reserved, Not Bussed	30		Reserved, Not Bussed
	31		Reserved, Not Bussed	32		Reserved, Not Bussed
	33		Reserved, Not Bussed	34		Reserved, Not Bussed
	35		Reserved, Not Bussed	36		Reserved, Not Bussed
	37		Reserved, Not Bussed	38		Reserved, Not Bussed
	39		Reserved, Not Bussed	40		Reserved, Not Bussed
	41		Reserved, Bussed	42		Reserved, Bussed
	43		Reserved, Bussed	44		Reserved, Bussed
	45		Reserved, Bussed	46		Reserved, Bussed
	47		Reserved, Bussed	48		Reserved, Bussed
	49		Reserved, Bussed	50		Reserved, Bussed
	51		Reserved, Bussed	52		Reserved, Bussed
	53		Reserved, Bussed	54		Reserved, Bussed
Address	55	ADR16*	Address Bus	56	ADR17*	Address Bus
	57	ADR14*		58	ADR15*	
	59		Reserved, Bussed	60		Reserved, Bussed

All Reserved Pins are reserved for future use and should not be used if upwards compatibility is desired.

4.2.2 Standard Outline of Printed Wiring Boards

Figure 33 shows the standard outline for 796 Bus-compatible boards (Printed Wire Boards and Printed Circuit Boards). The non-bus edge of the board is not restricted. The remainder of the board including connectors P1 and P2 must adhere to the dimensions shown in Figure 33. Only the basic boards' standard vertical height is currently specified.

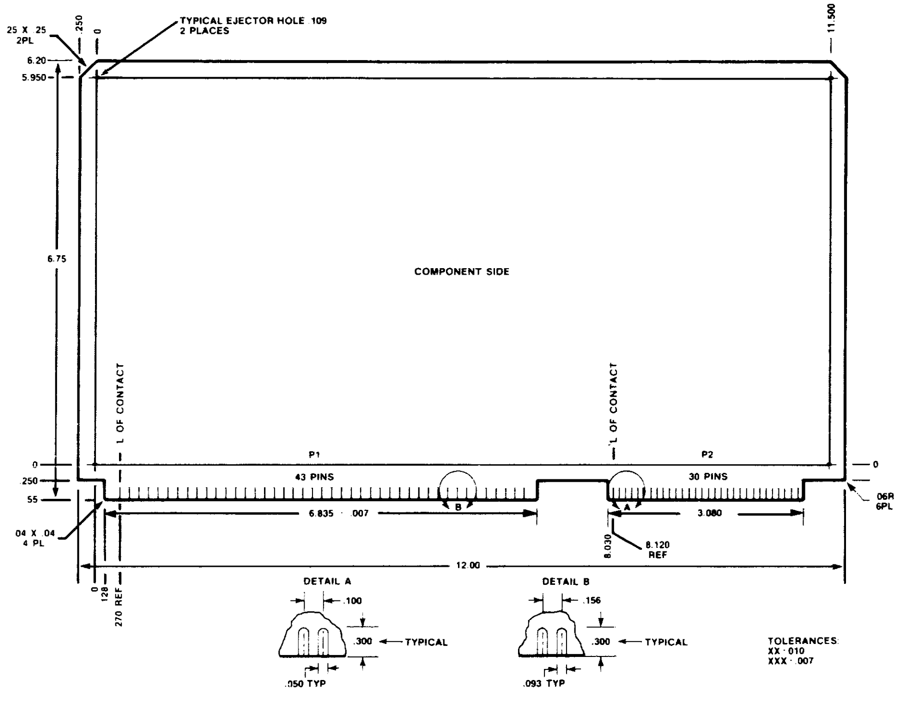


Figure 33 Standard Printed Wiring Board Outline

4.2.3 Bus Connectors

The 796 Bus backplane has connectors that mate to the P1 (43/86 pin) board edge connector. The backplane uses 43/86 pin on 0.156" centers connectors.

The P2 connector is a 30/60 pin board edge connector with 0.100" pin centers.

CHAPTER 5
LEVELS OF COMPLIANCE

This section presents the concept and notation of levels of compliance with the 796 Bus Standard as follows:

(1) Variable Elements of capability composing the essence of 796 Bus standard compliance.

(2) General discussion of compliance relationship for masters and for slaves.

(3) Notation for describing level of compliance with the 796 Bus Standard.

The notation levels of compliance is introduced to facilitate the use of 796 Bus products of varying capability manufactured by diverse vendors. It bounds the variability allowed within the 796 Bus specification and provides a succinct and convenient notation for these variables.

5.1 VARIABLE ELEMENTS OF CAPABILITY

The 796 Bus is very versatile allowing systems to be constructed with boards of varying capability. The 796 Bus allows for variations in data path width, I/O address path width, and interrupt attributes. In addition it is recognized that some vendors products have differing memory address path width.

5.1.1 Data Path

The 796 Bus allows for both 8- and 16-bit data path products. The 16-bit data path products use the byte swapping technique described in section 2.2.2.4, thus allowing the 8- and 16-bit products to work together.

5.1.2 Memory Address Path

The 796 Bus standard designates a 24-bit address path. In many systems a 16- or 20-bit address path may be sufficient, though not fully 796 Bus compatible.

5.1.3 I/O Address Path

The 796 Bus allows for both 8- and 16-bit I/O address paths. The 16-bit path products must also be configurable to act as 8-bit path products.

5.1.4 Interrupt Attributes

The 796 Bus (section 2.3) allows for considerable variety in interrupt attributes. A product may support no interrupts, Non Bus Vectored (NBV) interrupts, two cycle bus vectored interrupts, and three cycle bus vectored interrupts. There are two methods of interrupt sensing: the preferred level-triggered; and for historical compatibility only, edge-level-triggered.

Level Triggered. The active level of the request line indicates an active request. Requiring no edge to trigger an interrupt allows several sources to be attached to a single request line. Sources for level triggered sense inputs should provide a programmatic means to clear the interrupt request.

Edge-Level-Triggered. The transition from the inactive to the active level indicates an active request if and only if the active level is maintained at least until it has been recognized by the master. The requirement for a transition precludes multiple sources on a request line. But, Edge-Level triggering removes the requirement that the source have a programmatic means to clear the interrupt request.

NOTE

Edge-Level-Triggering is described only to allow for historical compatibility. New designs shall use level-triggered interrupt sensing.

A master may support either or both of the above interrupt sensing methods. It is necessary to configure the system such that the sources of the interrupt requests correspond to the interrupt sensing method of the master. Note that a source which is compatible with Level-Triggering is also compatible with Edge-Level triggering.

5.2 MASTERS AND SLAVES

When constructing 796 Bus systems it is not necessary that all modules have identical capabilities. One may for instance have a master with an 8-/16-bit data path and a slave with an 8-bit data path. The system is completely functional, though the application must restrict itself to 8-bit access to the slave.

The key concept when constructing a 796 Bus system is that of required capability versus supplied capability. Each product will provide some set of capability. A transaction between two such products will be restricted to use that capability which is the intersection of the sets of capability of the two products. In some cases the intersection may be null implying fundamental incompatibility. It is the responsibility of the system designer to assure the viability of this intersection.

5.3 COMPLIANCE LEVEL NOTATION

A notation is introduced which allows a vendor to succinctly and accurately specify a product's level of compliance with the 796 Bus standard. For boards which may act as either masters or slaves, the compliance levels must be specified for both cases. Increasing levels of compliance subsume lesser levels for data path width, memory address path width and I/O address path width. Interrupt attributes are listed separately as they are independent of one another. The lack of an element (i.e., no I/O address path) specification normally implies no capability for this element.

5.3.1 Data Path

D8 represents an 8-bit data path

D16 represents an 8-/16-bit data path

5.3.2 Memory Address Path

M16 represents a 16-bit memory address path

M20 represents a 20-bit memory address path

M24 represents a 24-bit memory address path

5.3.3 I/O Address Path

I8 represents an 8-bit I/O address path

I16 represents an 8- or 16-bit I/O address path

5.3.4 Interrupt Attributes

V0 represents Non Bus Vectored interrupt requests

V2 represents two cycle bus vectored interrupt requests

V3 represents three cycle bus vectored interrupt requests

E represents Edge-Level triggering only

L represents Level triggering

EL represents Level or Edge-Level triggering

The interrupt attributes notation can be concatenated to represent multiple capabilities.

5.3.5 An Example

A versatile combination I/O and memory slave board which supports an 8-16-bit data path, a 20-bit memory address, an 8- or 16-bit I/O address, NBV interrupt requests, two and three cycle bus vectored interrupt requests would be specified as follows:

796 Bus Compliance: Slave D16 M20 I16 VO23 L

5.3.6 Compliance Marking

The compliance levels of a card shall be clearly marked on the printed circuit board as well as in the printed specifications.

iSBX™-IEEE P959 I/O BUS SPECIFICATION

II

CONTENTS

CONTENTS (continued)

CHAPTER 4 Page

MECHANICAL SPECIFICATIONS

CHAPTER 5

CONNECTER SPECIFICATIONS

CHAPTER 6

LEVELS OF COMPLIANCE

APPENDIX A

APPENDIX B

FIGURES

TABLES

CHAPTER 1 GENERAL

1.1 SCOPE

One of the most important parts of any computer system is the I/O. Many systems require different types of I/O depending on the specific application. In addition, the system designer is faced with a significant challenge to keep up with new technology in I/O device types, interfaces and the LSI components designed to support them. The iSBX Bus is designed to address these needs. Using this standard interface allows small increments of I/O to be added or modified in an easy, cost effective way.

The iSBX Bus is specified independent of processor or board type. Each expansion interface supports up to sixteen 8-bit I/O ports directly. Enhanced addressing capability is available using slave processors or FIFO devices. In addition, each expansion interface may optionally support a DMA channel capable of data rates up to 2 Mwords/sec. These features are supported for both 8- and 16-bit data paths.

This specification has been prepared for those users who intend to design or evaluate products that will be compatible with the iSBX Bus. For this purpose, functional, electrical and mechanical specifications are covered in detail. The intent of the specification is to guarantee compatibility between baseboards and expansion modules while not restricting the actual designs any more than necessary. Additional design considerations that are not part of the actual specification are presented in the appendices.

1.2 DEFINITIONS

The following paragraphs contain definitions that are used throughout this specification. More detailed definitions can be found in the appropriate section of this specification.

1.2.1 General System Term Definitions

Compatibility. The degree to which devices may be interconnected and used without modification, when designed as defined in sections 2, 3, and 4 of this specification. Section 6 discusses the various levels of compliance that the specification allows.

Operation. The process whereby digital signals effect the transfer of data across the interface by means of a sequence of control signals. Operations may be either interlocked or full-speed.

Interface. A shared boundary, between two systems or parts of systems, through which information is transferred.

System. A set of interconnected elements which achieve a given objective through performing a specific function.

Arbitration. The process of determining which requesting device will gain access to a resource.

1.2.2 Signals and Paths

Bus. A signal line or set of signal lines used by an interface system to connect a number of devices and to transfer data.

Byte. A group of eight adjacent bits of data that are operated on as a unit.

Word. Two bytes operated on as a unit.

Data Path. Signal lines on a bus associated with data.

High State. The more positive voltage level; used to represent one of two logical binary states (see section 3.1.1 for more details).

Low State. The more negative voltage level; used to represent one of two logical binary states (see section 3.1.1 for more details).

Signal. The physical representation of a logical value.

Signal Level. The relative magnitude of a signal when compared to an arbitrary reference. Signal levels in this specification are specified in volts.

Signal Line. One of a set of signal conductors in an interface system used to transfer messages among interconnected devices.

Active-High Signal. A signal for which the logical-true (activated) state is represented by the high electrical state, and the logical-false (deactivated) state is represented by the low electrical state. In this specification, the active-high signal can be identified by the absence of an asterisk (*) at the end of the signal name (refer to section 3.1.1 for more details).

Active-Low Signal. A signal for which the logical-false (deactivated) state is represented by the high electrical state, and the logical-true (activated) state is represented by the low electrical state. In this specification, the active-low signal can be identified by the presence of an asterisk (*) at the end of the signal name (refer to section 3.1.1 for more details).

Two-directional Signal Line. A signal line that may be defined in either direction across an interface, and that cannot be defined in both directions simultaneously. The direction of operation for a two-directional signal line in a system is a configuration option.

Bi-directional Signal Line. A signal line that may be defined in either direction across an interface. The direction is determined by control signals for each operation.

CHAPTER 2
FUNCTIONAL DESCRIPTION

The iSBX Bus concept allows low cost, highly flexible I/O expansion for computer boards. This section provides an overall understanding of the concept by describing the elements of the bus system, the bus interface signals and the bus operations.

2.1 iSBX™ BUS ELEMENTS

There are two basic elements in the iSBX Bus; the baseboard and the expansion module.

2.1.1 Baseboard

A baseboard is any board which provides one or more I/O expansion interfaces (connectors) that meet the electrical and mechanical requirements of this specification. Logically, the baseboard is always the master device, making it responsible for generating all addresses, chip selects, and commands. As a master device for DMA transfers, the baseboard is required to provide the DMA controller function. Mechanically, the baseboard shall supply the latching connector(s) and mounting hole(s) for attaching the expansion module with nylon screws and spacers. See figures 1 and 2 for mounting details.

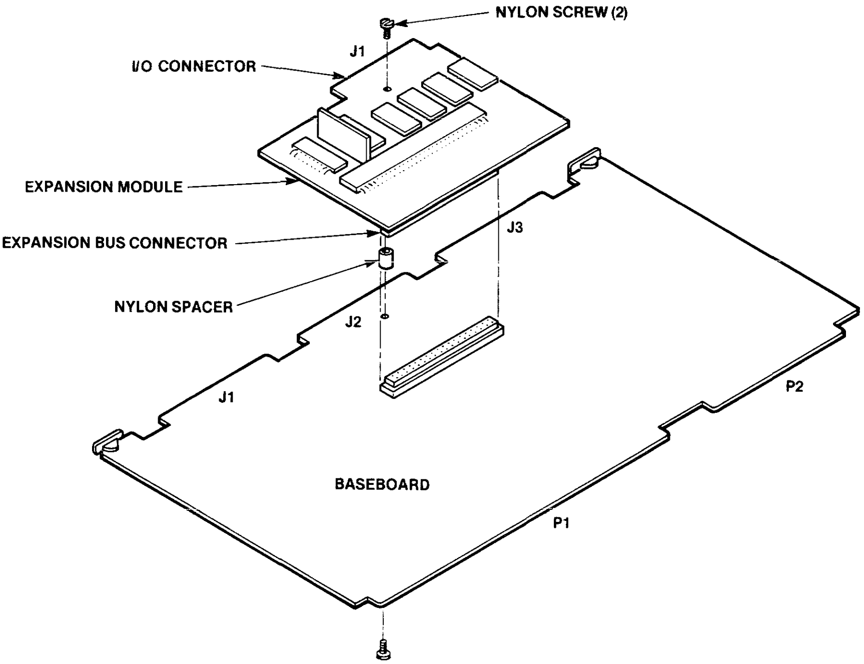


Figure 1 iSBX™ Bus System with Single-Wide Expansion Module

2.1.2 Expansion Module

An expansion module is a small specialized I/O board which attaches to a baseboard. Each expansion module can be one of two standard sizes, single wide and double wide; each is shown in figures 1 and 2 respectively. The purpose of an expansion module is to convert the general bus interface into a specific I/O interface. An example of this would be an RS232 serial interface module. The iSBX Bus is specifically designed to simplify the expansion module interface. In many cases, LSI peripheral components can connect to this interface directly.

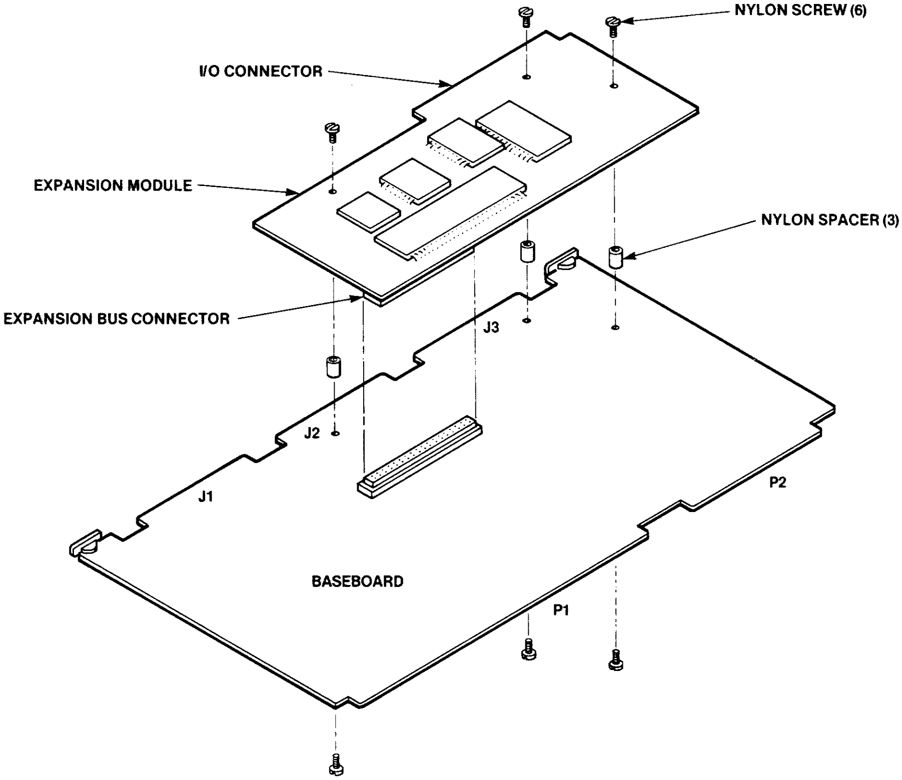


Figure 2 iSBX™ Bus System with Double-Wide Expansion Module

2.2 iSBX^{TM} BUS SIGNALS

The iSBX Bus signals can be grouped into seven classes based on the functions they perform. The classes are:

(1) Address and Chip Select Lines
(2) Data Lines
(3) Control Lines
(4) Interrupt Lines
(5) Option Lines
(6) Power Lines
(7) Reserved Lines

2.2.1 Address and Chip Select Lines.

These lines can be broken into two groups.

Function	**Signals**
Address	MA0, MA1, MA2
Chip Select	MCS0*, MCS1*

The function of these signals is somewhat unique. A typical bus would simply have address lines and require the slave device to create its own chip selects. The iSBX Bus requires the baseboard to generate chip selects. This feature simplifies the expansion modules with only minimal impact to the baseboard.

2.2.1.1 ADDRESS LINES (MA0, MA1, MA2).

The address lines are active high signals generated by the baseboard from the low order address lines. MA0 is the least significant address bit. For an 8-bit baseboard, these lines connect directly to the least significant address bits A0, A1, and A2. For a 16-bit baseboard, the address line connections are: MA0 to A1, MA1 to A2, and MA2 to A3 (A0 is used along with byte high enable to control byte/word transfers). During DMA operations, the state of the address lines is undefined.

2.2.1.2 CHIP SELECTS (MCS0*, MCS1*).

The chip selects are active low signals generated by the baseboard to enable communication with the expansion module. These lines may transition when they are not required to be valid. It is the responsibility of the expansion module to qualify the chip select signals with commands (note that this is done internally in many LSI components). Chip selects shall remain high during a DMA operation.

The chip select lines are defined differently for 8- and 16-bit baseboards as shown in figure 3. For 8-bit baseboards, each chip select enables 8 consecutive ports as specified by the three address lines. For 16-bit boards, there are two cases; 8-bit expansion module and 16-bit expansion module.

For a 16-bit baseboard driving an 8-bit expansion module, the chip select operation is similar to the 8-bit baseboard. The 8-bit expansion module connects to the low byte on the data bus which makes it only accessible through even ports (odd ports transfer over the high byte). Each chip select therefore enables 8 consecutive even ports as determined by the three address lines. Note that this mode also allows each chip select signal to enable 8 consecutive 16-bit ports for 16-bit expansion modules.

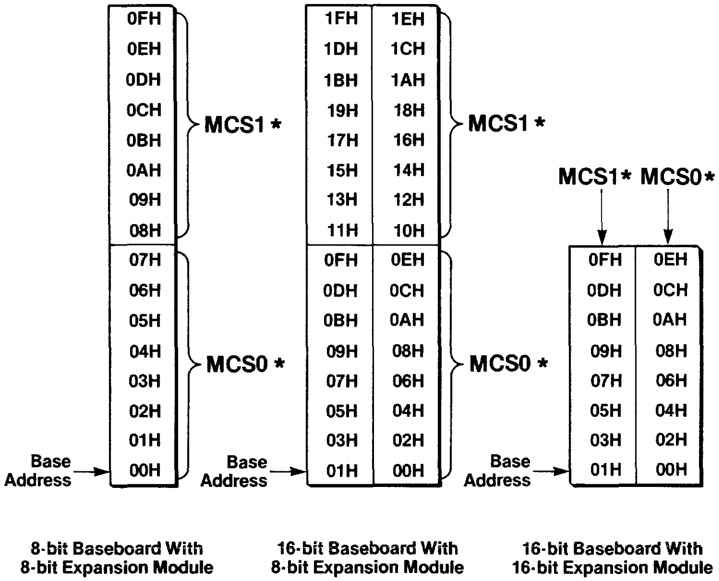


Figure 3 Chip Select Address Ranges

The chip selects on a 16-bit baseboard with a 16-bit expansion module have two functions. In addition to enabling communication, they control high byte, low-byte, and word transfer modes. MCS0* is used to enable a low byte transfer (even ports), MCS1* is used to enable a high byte transfer (odd ports) and both are used for a word transfer. The three address lines determine which of eight 16-bit ports are being addressed.

2.2.2 Data Lines (MD0-MD15)

The data lines are used to transmit or receive data to or from the expansion module ports. The 16 active high, bi-directional lines are organized in two groups (bytes) of 8 lines each. The low byte (MD0-MD7) is used for all transfers on 8-bit baseboards and for all even byte transfers on 16-bit baseboards. MD0 is the least significant bit of this byte. The high byte (MD8-MD15) is used for all odd byte transfers on 16-bit baseboards. MD8 is the least significant bit of this byte. Table 1 presents a summary of the data bus functions for different configurations.

Table 1 Data Bus Function For Different Configurations

Configuration	Lines Used	
	MD0-MD7	MD8-MD15
8-bit baseboard	X	
16-bit baseboard with:		
8-bit expansion module	X	
16-bit expansion module - even byte	X	
16-bit expansion module - odd byte		X
16-bit expansion module - word	X	X

2.2.3 Control Lines

The following signals are classified as control lines:

CLASS	FUNCTION	SIGNAL
Commands	I/O Read	IORD*
	I/O Write	IOWRT*
Direct Memory Access (DMA)	DMA Request	MDRQT
	DMA Acknowledge	MDACK*
	Terminate DMA	TDMA
Initialize	Reset	RESET
Clock	Expansion Module Clock	MCLK
System Control	Expansion Module Wait	MWAIT*
	Expansion Module Present	MPST*

2.2.3.1 COMMAND LINES (IORD*, IOWRT*)

The command lines are active low signals generated by the baseboard. An active command line indicates to the expansion board that the address and chip select lines are valid and that the selected expansion module (i.e. one with active chip select) should perform the specified operation. The I/O read command (IORD*) is used by the baseboard to request data to be transferred from the expansion module. The I/O write command (IOWRT*) is used by the baseboard to transfer data to the expansion module.

2.2.3.2 DIRECT MEMORY ACCESS (DMA) LINES

The DMA lines control the communication link between a DMA controller on the baseboard and the expansion module. The use of these lines is optional on both the baseboard and the expansion module.

2.2.3.2.1 DMA Request (MDRQT)

This active high signal from the expansion module to the baseboard DMA controller requests that a DMA cycle be initiated. Upon receipt of this signal the DMA controller begins arbitration for the baseboard's local bus.

2.2.3.2.2 DMA Acknowledge (MDACK*)

This active low signal from the baseboard to the expansion module indicates that a DMA operation has been granted (DMA controller has the bus). Like a chip select, this signal may transition when not valid and therefore shall be qualified by commands on the expansion module.

2.2.3.2.3 Terminate DMA (TDMA)

This is an active high, two-directional DMA control signal. The specific direction shall be determined by configuration (i.e., jumper or tri-state driver). Once configured, TDMA may only operate in one direction. When generated by the expansion module, TDMA is used to terminate the DMA transfer. When generated by the baseboard, TDMA is used to signal the end of a DMA transfer to the expansion module. Baseboards that support DMA shall be configurable to support DMA in either direction. An expansion module may support or not support this signal as required.

2.2.3.3 INITIALIZE (RESET)

This active high signal is generated by the baseboard to put the expansion module into a known state. The baseboard shall generate a power up reset when power is applied. During normal operation after a power up reset sequence, the expansion module may be reinitialized with a standard reset signal (refer to section 3.2 for more details).

2.2.3.4 EXPANSION MODULE CLOCK (MCLK)

This input to the expansion module is a general purpose timing signal. The 9-10MHz clock is asynchronous to all other bus signals.

2.2.3.5 SYSTEM CONTROL

There are two system control signals (MWAIT* and MPST*) that are defined in the following paragraphs.

2.2.3.5.1 Expansion Module Wait (MWAIT*)

This active low signal is generated by the expansion module to extend the current data transfer operation. While MWAIT* is activated, the baseboard is forced to insert wait states into the current bus cycle, allowing the expansion module more time to complete the requested operation. The MWAIT* signal is generated or enabled by a combination of valid chip select(s) and addresses or by a valid DMA acknowledge signal. As these lines change, MWAIT* may transition, however it shall be stable no later than 75nsec after MCS0*, MCS1*, and MDACK* become stable.

Recognition of the MWAIT* signal by the baseboard is optional, however, it is strongly recommended. Only in cases where the processor does not support the signal should it be omitted, as it will limit the number of modules that are compatible with the baseboard. Baseboards that do not support the use of the MWAIT* signal may perform only the full-speed operations, as defined in sections 2.3.1 and 2.3.2. When it supports the MWAIT* signal, the baseboard shall guarantee (usually via a pull-up resistor) that the signal is inactive if not connected. Expansion modules shall support the MWAIT* signal if they cannot meet the full-speed operation specifications.

2.2.3.5.2 Expansion Module Present (MPST*)

This active low signal is driven by the expansion module to inform the baseboard that a module is installed. Typically, this signal is connected to ground on the expansion module and to decode logic on the baseboard. When the signal is not activated, the I/O address space normally reserved for the expansion module may be used elsewhere. One application of the MPST* signal is to allow an old product that did not include facilities for the iSBX to be upgraded with a new baseboard that includes iSBX capability. Without expansion modules, the I/O addressing for the new baseboard looks the same as that of the old one. If the new baseboard requires expansion modules, they can be added.

2.2.4 Interrupt Lines (MINTR0, MINTR1)

These active high lines from the expansion module make interrupt requests to the baseboard. An interrupt line driven from the expansion module shall remain active until the baseboard services it. These lines are asynchronous to all other bus signals.

2.2.5 Option Lines (OPT0, OPT1)

These lines are user defined. Their purpose is to provide connections for any unique system requirements. Examples would be additional interrupt lines or another DMA channel. Baseboards shall connect these lines to wirewrap posts for configurability.

2.2.6 Power Lines

All baseboards shall be capable of supplying +5VDC and ±12VDC.

2.2.7 Reserved Lines

These two lines are reserved for future enhancements made possible through new technology. Baseboards and the expansion modules shall leave these lines unconnected.

2.3 BUS OPERATIONS

The iSBX Bus supports I/O read, I/O write, DMA, and interrupt operations.

2.3.1 I/O Read Operations

There are two types of I/O read operations on the iSBX Bus; the full speed I/O read and the interlocked I/O read. Whether or not the expansion module generates the MWAIT* signal determines which type of operation is performed.

Figure 4 shows the full speed I/O read operation. The baseboard generates a valid address and chip select for the expansion module to initiate the operation. After the set up times are met, the baseboard activates the IORD* line for a minimum of 300 nsec. The expansion module shall generate valid data from the addressed I/O port within 250 nsec after the IORD* line is activated. The baseboard then reads the data and deactivates the IORD* line. The baseboard is now free to change the address and chip select lines for the next operation.

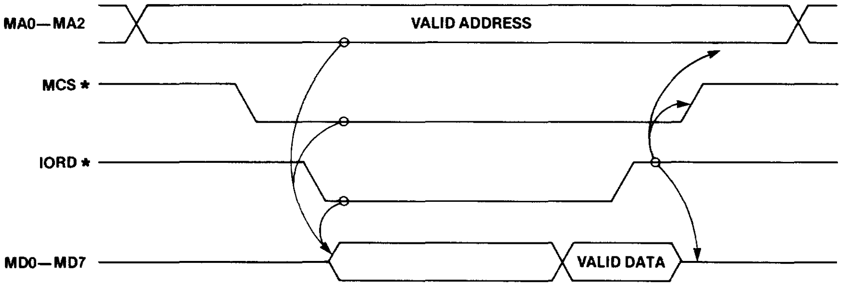


Figure 4 Full-Speed I/O READ Operation

Figure 5 shows the interlocked I/O read operation. The baseboard initiates the operation by generating a valid address and chip select, just as in the full speed I/O read. The expansion module then activates the MWAIT* signal which in turn inhibits the ready signal to the baseboard processor. The baseboard will then activate the IORD* line and insert wait states as long as the MWAIT* signal from the expansion module is active. The expansion module will remove the MWAIT* signal when data is valid on the bus. The baseboard then reads the data and deactivates the IORD* line. The baseboard is now free to change the address and chip select lines for the next operation. The interlocked I/O read operation shall be used by all expansion modules that require a read pulse width greater than 300 nsec or that cannot guarantee valid data on the data bus within 250 nsec after the IORD* line is activated.

2.3.2 I/O Write Operations

There are two types of I/O write operations on the iSBX Bus; the full speed I/O write and the interlocked I/O write. Whether or not the expansion module generates the MWAIT* signal determines which type of operation is performed.

Figure 6 shows the full speed I/O write operation. The baseboard generates a valid address and chip select to initiate the operation. After the set up times are met, the baseboard activates the IOWRT* line and enables data. The IOWRT* line will remain active for at least 300 nsec and the data will be valid for at least 250 nsec before the IOWRT* line is deactivated. After the IOWRT* line is deactivated, the baseboard is free to change the address and chip select lines for the next operation.

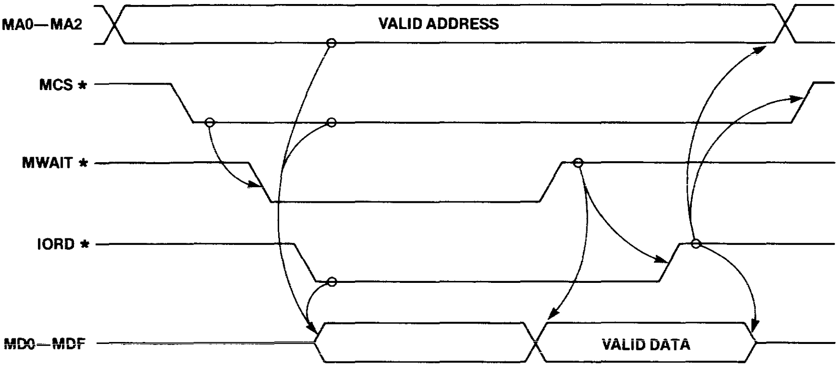


Figure 5 Interlocked I/O READ Operation

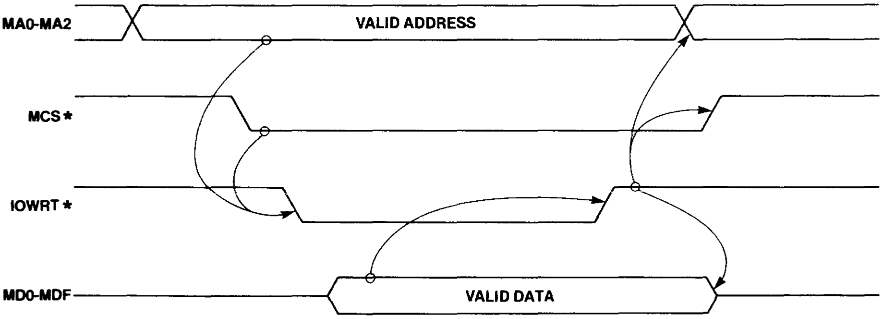


Figure 6 Full-Speed I/O WRITE Operation

Figure 7 shows the interlocked I/O write operation. The baseboard initiates the operation by generating a valid address and chip select, just as in the full speed I/O write. The expansion module then activates the MWAIT* signal which in turn inhibits the ready signal to the baseboard processor. The baseboard will then activate the IOWRT* line, enable data, and insert wait states as long as the MWAIT* signal is active. The expansion module will remove MWAIT* when it is ready to receive the data. The baseboard then deactivates the IOWRT* line and removes the data. Data shall be stable at least 250nsec before the IOWRT* signal is deactivated. The baseboard is now free to change the address and chip select lines for the next operation. The interlocked I/O write operation shall be used by all expansion modules that cannot guarantee proper operation with a 300 nsec write pulse width.

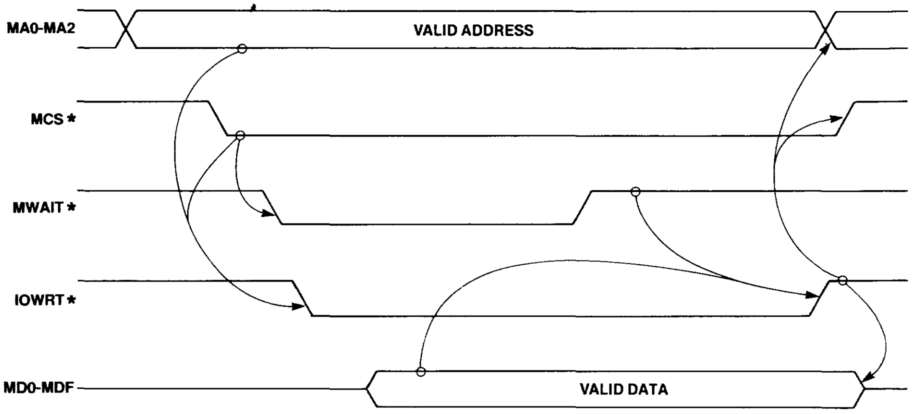


Figure 7 Interlocked I/O WRITE Operation

2.3.3 Direct Memory Access (DMA) Operations

DMA is a means of moving a block of data to or from an expansion module without the overhead of an interrupt service or processor polling routine. The source or destination of the data on the baseboard is typically a series of consecutive memory locations. For the expansion module, the source or destination is always an I/O port. The DMA process is always initiated by executing software on the baseboard that sets up the DMA controller and expansion module. This software determines the direction of data movement, the addresses of the source and destination, and the length of the transfer. Once set up, DMA operations will be done automatically by the DMA controller, as requested by the expansion module.

DMA operations are similar to I/O read and I/O write operations. Expansion modules that meet the requirements for full-speed I/O read and I/O write operations can perform full-speed DMA operations. Likewise, expansion modules that meet the requirements for interlocked I/O read and I/O write operations can perform interlocked DMA operations. Once the DMA controller is set up, a DMA operation is initiated by the expansion module activating MDRQT. The DMA controller then acquires (arbitrates) the local bus on the baseboard and activates the MDACK* line to the expansion module to acknowledge the request. The

MDACK* line functions as a chip select for DMA operations. During a DMA operation, both chip selects (MCS0*, MCS1*) are deactivated and the address lines (MA0-MA2) are undefined. The rest of the operation is the same as the I/O read and I/O write operations described in sections 2.3.1 and 2.3.2, respectively. Figures 8 and 9 show full speed and interlocked DMA operations respectively.

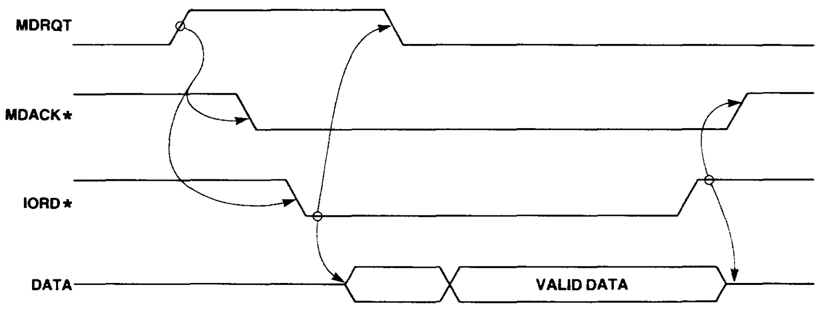


Figure 8 Full-Speed DMA Operation

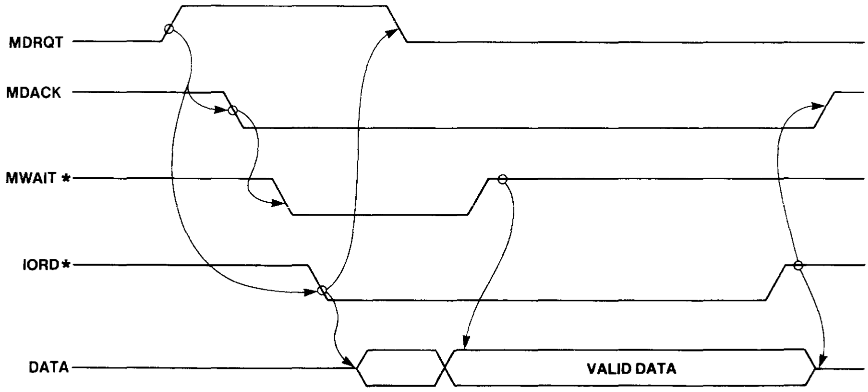


Figure 9 Interlocked DMA Operation

DMA operations can be done in two modes: cycle steal and burst. In the cycle steal mode, the expansion module will deactivate the MDRQT line every operation after receiving a command. The DMA controller, in turn, releases the bus after completing the operation. In the burst mode, the expansion module will hold the MDRQT line active until the last operation of a transfer. The burst mode will always attain equal or greater throughput, since the DMA controller only arbitrates for the bus once per transfer rather than once per operation.

The TDMA line is provided for additional DMA control. This line is configurable either as an input to the expansion module or as an output from the expansion module.

As an input, the TDMA line identifies the end of a DMA transfer. In this mode, the baseboard activates the TDMA line after the last valid DMA operation is completed. If the expansion module is holding MDRQT active when it receives an active TDMA line, it shall remove the MDRQT line as if it were a command. However, the baseboard need not wait for the MDRQT line to go active before activating the TDMA line.

As an output, the TDMA line shall terminate a DMA transfer in the baseboard. To accomplish this, the expansion module first deactivates the MDRQT line, and then activates the TDMA line.

2.3.4 Interrupt Operations

The expansion module initiates an interrupt operation by activating one of the interrupt request lines (MINTR0, MINTR1). This signal will interrupt the baseboard processor causing it to execute an interrupt service routine. The interrupt service routine performs two functions. First, it services the interrupt. This will typically consist of reading data from or writing data to the expansion module. Secondly, the service routine deactivates the interrupt line from the expansion module. In summary, from the expansion module's point of view, the expansion module initiates an interrupt by activating its interrupt line and removes the interrupt when the baseboard tells it to do so.

CHAPTER 3
ELECTRICAL SPECIFICATIONS

The electrical considerations required for the bus include the following: logical and electrical state relationships, environmental considerations, power supply specifications, signal line characteristics, timing specifications, and driver/receiver specifications.

3.1 GENERAL BUS CONSIDERATIONS

3.1.1 Logical and Electrical State Relationships

The signal name indicates whether the signal is active high or active low. If a signal name ends with an asterisk (*), then the signal is active low and has the logical-electrical state relationship as listed in Table 2.

Table 2 Active-Low Signal Characteristics

Logical State	Electrical State	Signal Level	
		At Receiver	At Driver
0 (deactivated)	H=TTL High State	5.25V≥H≥2.0V	5.25V≥H≥2.4V
1 (activated)	L=TTL Low State	0.8V≥L≥−0.5V	0.5V≥L≥0V

If the signal name does not end with an asterisk, then the signal is active high and has the logical-electrical state relationship as listed in Table 3.

Table 3 Active-High Signal Characteristics

Logical State	Electrical State	Signal Level	
		At Receiver	At Driver
0 (deactivated)	L=TTL Low State	0.8V≥L≥−0.5V	0.5V≥L≥0V
1 (activated)	H=TTL High State	5.25V≥H≥2.0V	5.25V≥H≥2.4H

These specifications are based on TTL, where the power source is 5V±5%, reference to logic ground.

3.1.2 Signal Line Characteristics

The settling time for commands, interrupts, MDRQT, TDMA, and MCLK, after a transition, is zero. These control lines are used to determine the state of the bus. Ringing beyond the noise immunity levels for these signal lines could cause system failures. Address lines, chip select lines, data lines, MDACK*, and MWAIT* can ring beyond the noise immunity levels as long as they remain stable for their specified set-up times.

3.1.3 Power Supply Specification

Table 4 gives the power supply specifications for the iSBX Bus. The maximum current specification is for one expansion module connector. A baseboard or system may optionally limit the total power to a group of connectors as long as these specifications are met for each individual expansion module connector. The voltage specifications at the connector are measured over the full current specification range.

Table 4 Power Requirements

Voltage			Current
Minimum Volts	Nominal Volts	Maximum Volts	Maximum Amps
+4.75	+5.00	+5.25	3.0
+11.4	+12.0	+12.6	1.0
−12.6	−12.0	−11.4	1.0
- - - -	GND	- - - -	6.0

3.1.4 Environmental

The electrical specifications shall be met within the following environmental conditions.

3.1.4.1 OPERATING CONDITIONS

Bus specifications should be met with operating environmental conditions in the following ranges:

Temperature:	0°C to 55°C (32°F to 131°F) with free moving air
Humidity:	0% to 95% without condensation

3.1.4.2 STORAGE

Bus specifications should be met with non-operating environmental conditions in the following ranges:

Temperature:	−40°C to 70°C (−40°F to 158°F)
Humidity:	0% to 95% without condensation

3.2 TIMING SPECIFICATIONS

This section provides all timing specifications on the iSBX Bus. Table 5 summarizes the timing specifications shown in the timing diagrams (Figures 10 through 16). All timing is measured at 0.8VDC for low and 2.0VDC for high over the full load impedance (resistance and capacitance).

Table 5 AC Specifications

Symbol	Parameter	Min (ns)	Max (ns)	Figure Reference
t1	Address stable before read	50		10
t2	Address stable after read	30		10
t3	Read pulse width	300		10
t4[2]	Data valid from read	0	250	10
t5	Data float after read	0	150	10
t6	Time between commands		Note 3	-
t7	Chip select stable before command	25		10,11
t8	Chip select stable after command	30		10,11
t9	Power-up reset pulse width	50 Msec		16
t10	Address stable before write	50		11
t11	Address stable after write	30		11
t12	Write pulse width	300		11
t13[2]	Data valid to write	250		11
t14	Data valid after write	30		11
t15	MCLK cycle	100	110	15
t16	MCLK width	35		15
t17[1]	MWAIT* pulse width	0	4 Msec	10,11
t18[5]	Reset pulse width	50 usec		16
t19[1]	Chip select to MWAIT* valid	0	75	10,11
t20	MDACK* set up to command	25		12
t21	MDACK* hold after command	30		12
t22[4]	Command or TDMA to MDRQT removed		150	12,13,14
t23	TDMA pulse width	300		13,14
t24[1]	MWAIT* to valid read data		0	10
t25[1]	MWAIT* to write command		0	11
t26	MDRQT inactive to TDMA			14

Note 1: Required only if MWAIT* is activated.
Note 2: If MWAIT* not activated.
Note 3: To be specified by each expansion module.
Note 4: Required in cycle steal mode and for last operation in burst mode.
Note 5: Use t9 for reset after power is applied.

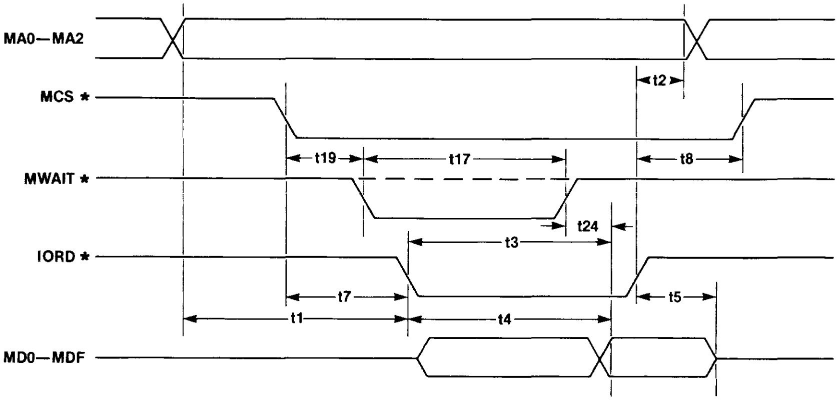


Figure 10 I/O READ Timing

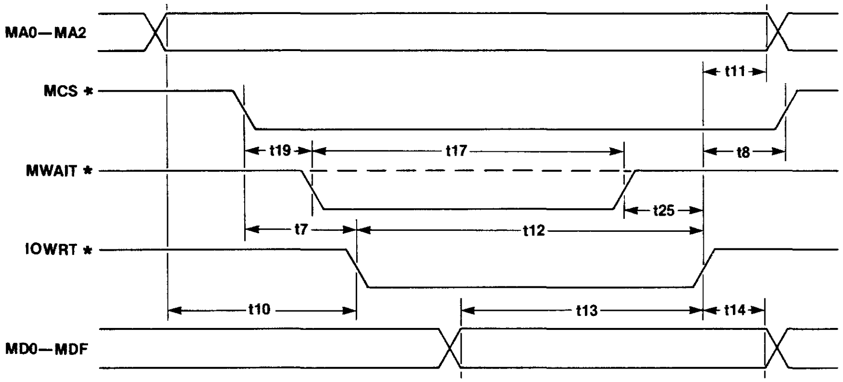


Figure 11 I/O WRITE Timing

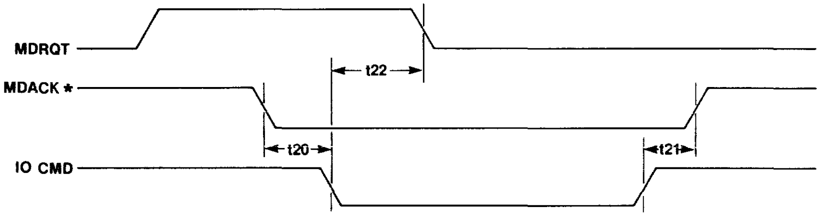


Figure 12 DMA Timing

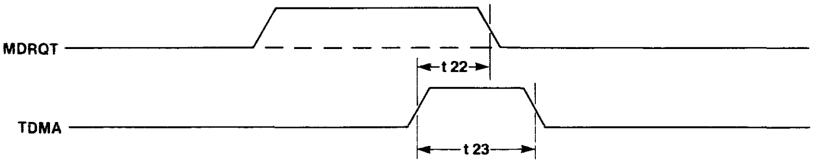


Figure 13 TDMA Timing From Baseboard

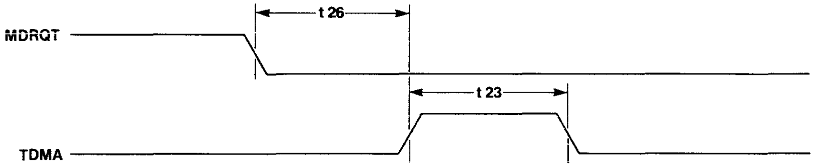


Figure 14 TDMA Timing From Expansion Module

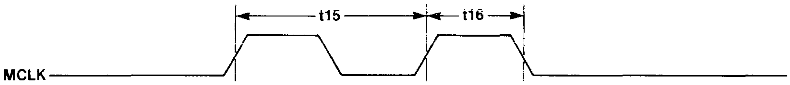


Figure 15 MCLK Timing

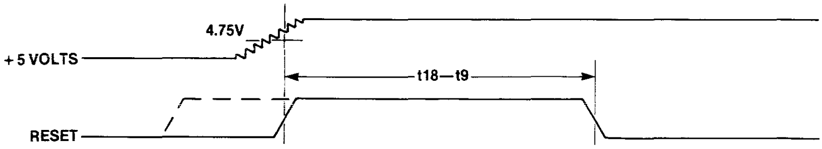


Figure 16 RESET Timing

3.3 DRIVER AND RECEIVER SPECIFICATIONS

This section specifies all driver types and DC specifications for the iSBX Bus. Table 6 summarizes the specifications.

Table 6 DC Specifications

Signal	Driver				Receiver		
	Type	I_{OL} (Min mA)	I_{OH} (Min uA)	C_O (Min pf)	I_{IL} (Max mA)	I_{IH} (Max uA)	C_I (Max pf)
MA0-MA2	TTL				−0.5	70	40
MCS0*,MCS1*	TTL				−1.0	100	40
MDACK*	TTL				−1.0	100	40
MD0-MD15	TRI	1.6	−300	130	−0.45	70	40
IORD*,IOWRT*	TTL				−1.0	100	40
MWAIT*	TTL	1.6	−50	40			
MDRQT	TTL	1.6	−50	40			
TDMA	TTL	1.6	−50	40	−1.0	100	40
RESET	TTL				−2.1	100	40
MCLK	TTL				−2.4	100	40
MINTR0,MINTR1	TTL	2.0	−100	40			
OPT0,OPT1[1]	TTL	2.0	−100	40	−2.0	100	40
MPST*	TTL	2.0	−100	40	−2.0	100	40

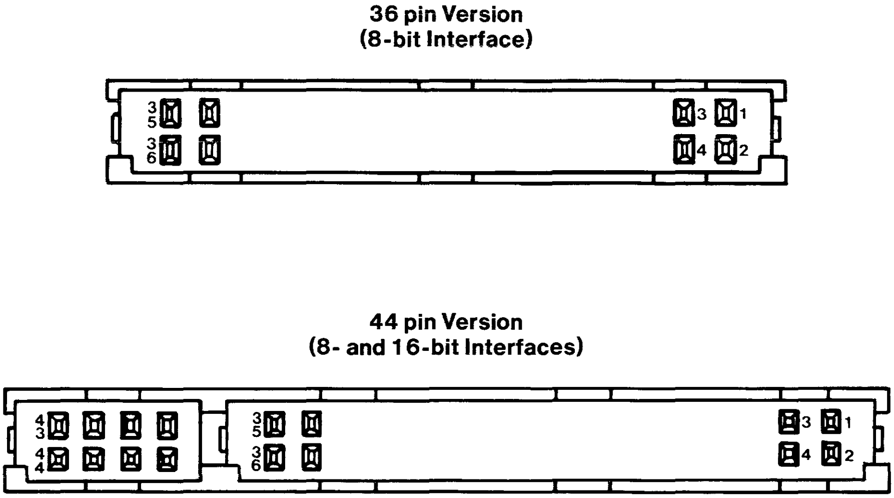
Note 1: These are recommended specifications. These lines are user defined so it is the responsibility of the user to ensure adequate drive.

CHAPTER 4
MECHANICAL SPECIFICATIONS

This section describes the physical and mechanical specification that a designer must be concerned with when designing a baseboard or an expansion module compatible with the iSBX Bus. Detailed specifications for the connector are contained in Section 5.

4.1 CONNECTOR PIN ASSIGNMENTS

There are two types of connector pairs used for the iSBX Bus; a 36 pin 8-bit version and a 44 pin 16-bit version. The 36 pin male connector on the expansion module will mate with both the 36 pin (8-bit) and the 44 pin (16-bit) female connectors on the baseboard. The 16-bit 44 pin male connector on the expansion module will mate with only the 44 pin (16-bit) female connector on the baseboard. The pin locations are shown in Figure 17. Table 7 lists the pin assignments.



36 pin Version
(8-bit Interface)

44 pin Version
(8- and 16-bit Interfaces)

(Top View of Baseboard Connector)

Figure 17 Connector Pin Numbering

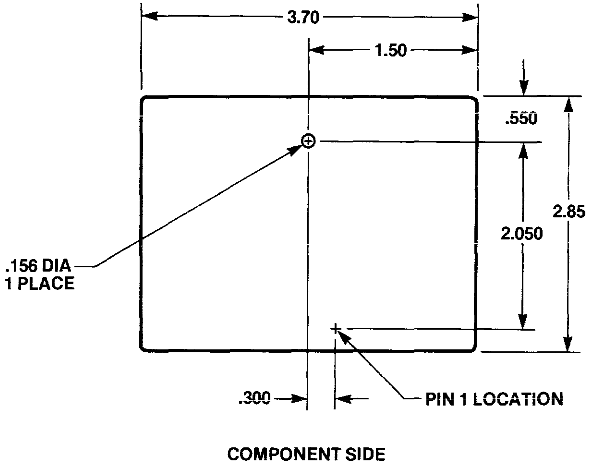
4.2 EXPANSION MODULE SPECIFICATIONS

There are two standard outlines for expansion modules. The single wide outline is shown in figure 18. The double wide outline is shown in figure 19. For each outline, either edge-finger connectors or pin-and-socket connectors may be used. When edge-finger connectors are used, they shall conform to the additional specifications contained in figure 20. For pin-and-socket connectors, there are no additional restrictions. Details of the I/O connector positioning are determined by the user with some examples provided in Appendix B.

Table 7 Pin Assignments

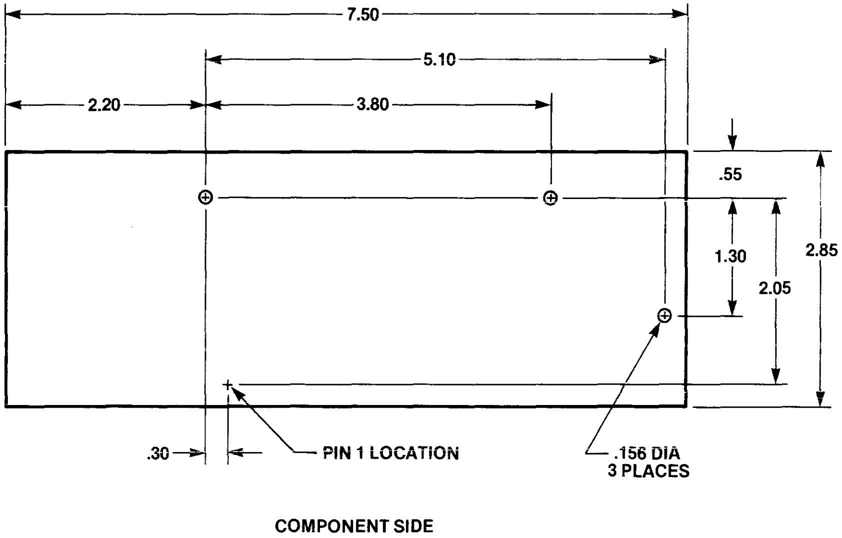
Pin	Mnemonic	Description	Pin	Mnemonic	Description
1	+12V	+12 Volts	2	−12V	−12 Volts
3	GND	Signal Ground	4	+5V	+5 Volts
5	RESET	Reset	6	MCLK	Expansion Module Clock
7	MA2	Address 2	8	MPST*	Expansion Module Present
9	MA1	Address 1	10		Reserved
11	MA0	Address 0	12	MINTR1	Interrupt 1
13	IOWRT*	I/O Write Command	14	MINTR0	Interrupt 0
15	IORD*	I/O Read Command	16	MWAIT*	Expansion Module Wait
17	GND	Signal Ground	18	+5V	+5 Volts
19	MD7	Data Bit 7	20	MCS1*	Chip Select 1
21	MD6	Data Bit 6	22	MCS0*	Chip Select 0
23	MD5	Data Bit 5	24		Reserved
25	MD4	Data Bit 4	26	TDMA	Terminate DMA
27	MD3	Data Bit 3	28	OPT1	Option 1
29	MD2	Data Bit 2	30	OPT0	Option 0
31	MD1	Data Bit 1	32	MDACK*	DMA Acknowledge
33	MD0	Data Bit 0	34	MDRQT	DMA Request
35	GND	Signal Ground	36	+5V	+5 Volts
37[1]	MD14	Data 14	38[1]	MD15	Data 15
39[1]	MD12	Data 12	40[1]	MD13	Data 13
41[1]	MD10	Data 10	42[1]	MD11	Data 11
43[1]	MD8	Data 8	44[1]	MD9	Data 9

Note 1: MD8-MD15 used only on 16-bit systems.
Note 2: Signals ending with an asterisk (*) are active-low. Signals ending without an asterisk are active-high.



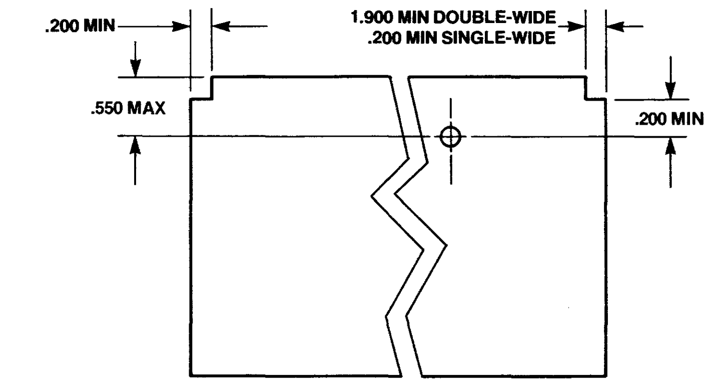
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 18 Standard Outline For Single-Wide Expansion MULTIMODULE™



Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 19 Standard Outline For Double-Wide Expansion MULTIMODULE™

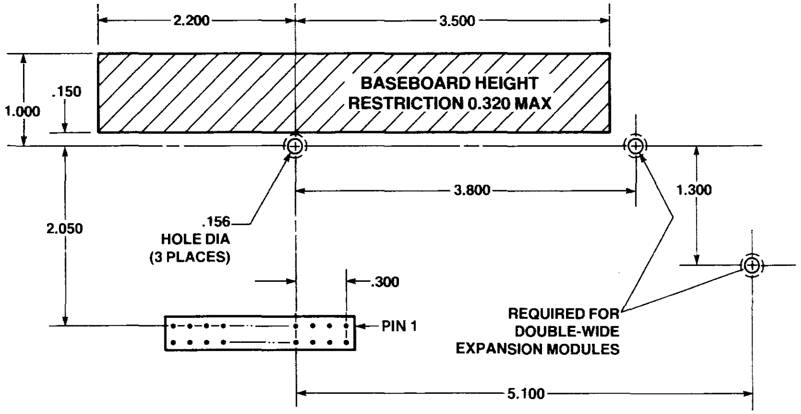


Note: 1. All dimensions are listed in inches unless otherwise specified.

Figure 20 Edge-Finger Connector Specifications

4.3 BASEBOARD SPECIFICATIONS

The absolute placement of the iSBX connector(s) on the baseboard is not part of this specification. Only the relative positions of the mounting holes with respect to the connector are specified. Figure 21 shows this relationship.

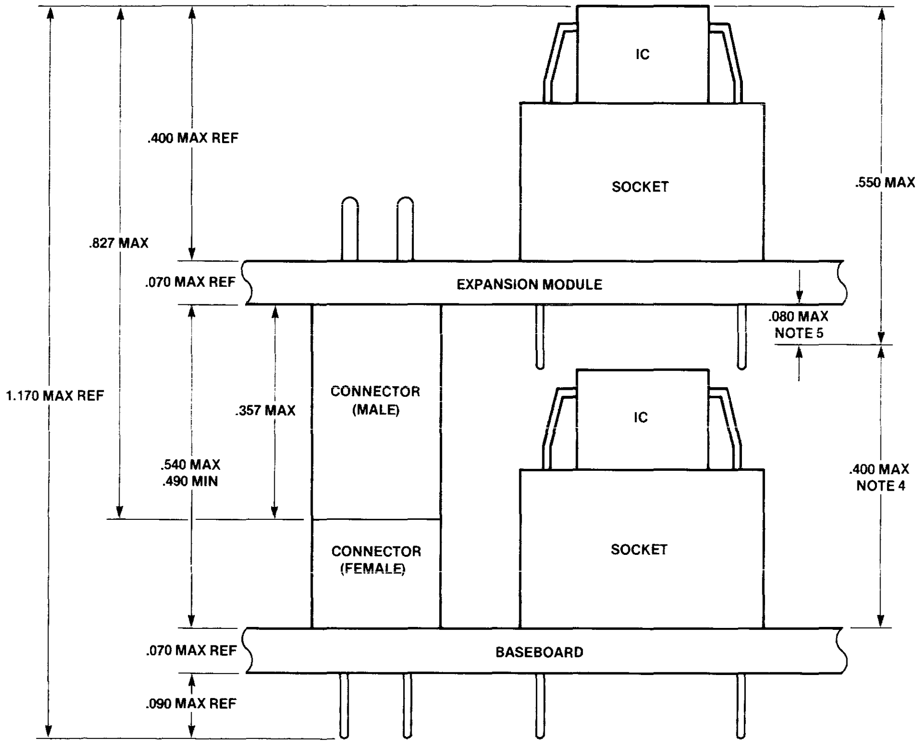


Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xxx ± 0.005 unless otherwise specified.

Figure 21 Relative Mounting Hole Placement For Baseboards

4.4 BOARD HEIGHT SPECIFICATIONS

Figure 22 shows worst-case height parameters for the iSBX Bus system. Figure 21 shows the baseboard region for which an additional height restriction applies (necessary for edge-finger connectors).



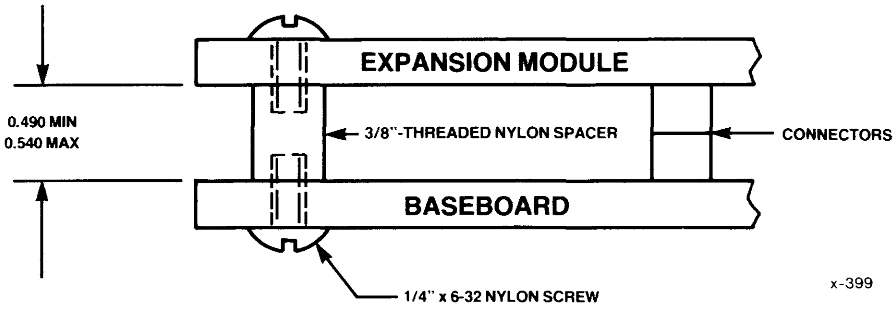
Notes:

1. All dimensions are listed in inches unless otherwise specified.
2. Maximum connector dimensions allow for 0.015 float above or below expansion module surface.
3. All parameters listed as reference are not part of this specification.
4. See additional restrictions for edge finger connectors, as listed in figure 20.

Figure 22 Maximum Height Requirement

4.5 MOUNTING TECHNIQUES

Expansion modules are mechanically secured to baseboards with the latching connector and with 6-32 nylon screws and threaded spacers as shown in figure 23.



Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 23 Mechanical Mounting Technique

CHAPTER 5
CONNECTOR SPECIFICATIONS

This section specifies the requirements of a miniature two-piece stacking connector to be used in both the 8- and 16-bit versions of the iSBX Bus. The connector parts mounted onto both the baseboards and the expansion modules shall meet all specifications.

5.1 REFERENCE DOCUMENTS

(1) MIL-G-45204 - Gold Plating, Electro-Deposited
(2) QQ-N-290 - Nickel Plating, Electro-Deposited
(3) UL-478 - Standard for Safety, EDP Units and Systems

5.2 MECHANICAL REQUIREMENTS

5.2.1 Materials

The material requirements are as follows:

(1) Insulators: Glass-reinforced nylon, type 6/6 or substance of equivalent mechanical and electrical properties.

(2) Contacts: Copper alloy, to meet all mechanical and electrical requirements of specification.

(3) Contact Finish: Gold plate in accordance with MIL-G-45204 Type II, Grade C, .000025 inch thick minimum over nickel plate in accordance with QQ-N-290, Class I, .000050 inch minimum in contacting area with tin lead or gold flash on the termination tail, or gold inlay in contact area with tin lead overlay in the termination area.

5.2.2 Number of Positions

8-bit - 18/36 dual row on 0.100 x 0.100 centers. 16-bit - 22/44 bridge dual row on 0.100 x 0.100 centers.

5.2.3 Durability

Mated pairs of connectors shall be capable of meeting all electrical and mechanical requirements of this specification during and after a minimum of 200 cycles of mating and unmating.

5.2.4 Contact Retention Force

Axial force in either direction which a contact shall withstand while remaining firmly fixed in its normal position within the insulator: 3 lbs. minimum.

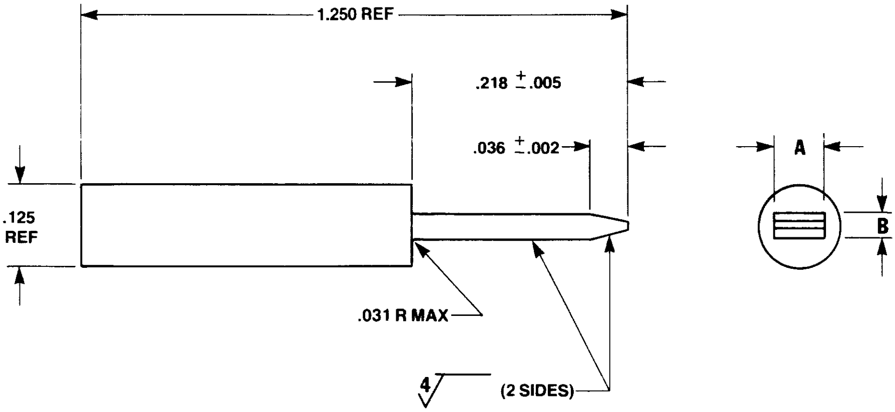
5.2.5 Connector Mating and Unmating Forces

A mating pair of connectors shall meet the following requirements when tested as follows:

(1) Mating: 20 lbs. maximum.
(2) Unmating: 5 lbs. minimum, 30 lbs. maximum.
(3) Mating and Unmating Force Test Method: The connector pair is to be tested with a floating fixture that allows one connector of the pair to align itself. The fixture shall maintain parallelism of the connectors to allow simultaneous engagement or disengagement of the connector contacts. The force transducer shall have a minimum resolution of 0.25 lbs. and a minimum accuracy of 0.1 lbs.

5.2.6 Contact Engagement and Separation Forces

When tested using gages listed in figure 24, the engagement force per female contact shall not exceed 10 oz. and separation force of 1 oz. minimum.



GAGE	A	B
ENGAGEMENT	$.0380^{+.0000}_{-.0001}$	$.0190^{+.0000}_{-.0001}$
SEPARATION	$.0320^{+.0001}_{-.0000}$	$.0170^{+.0001}_{-.0000}$

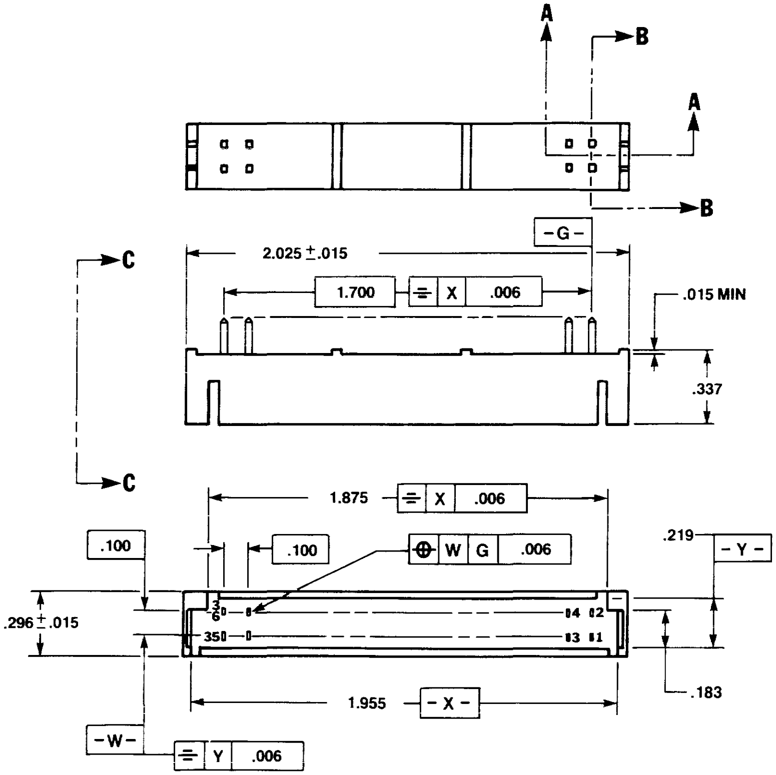
Figure 24 Contact Engagement and Separation Gages

5.2.7 Contact Identification

Contact positions should be identified.

5.2.8 Connector Requirements

All dimensions of connector halves shall conform to dimensions shown in Figures 25, 26, 27, and 28.



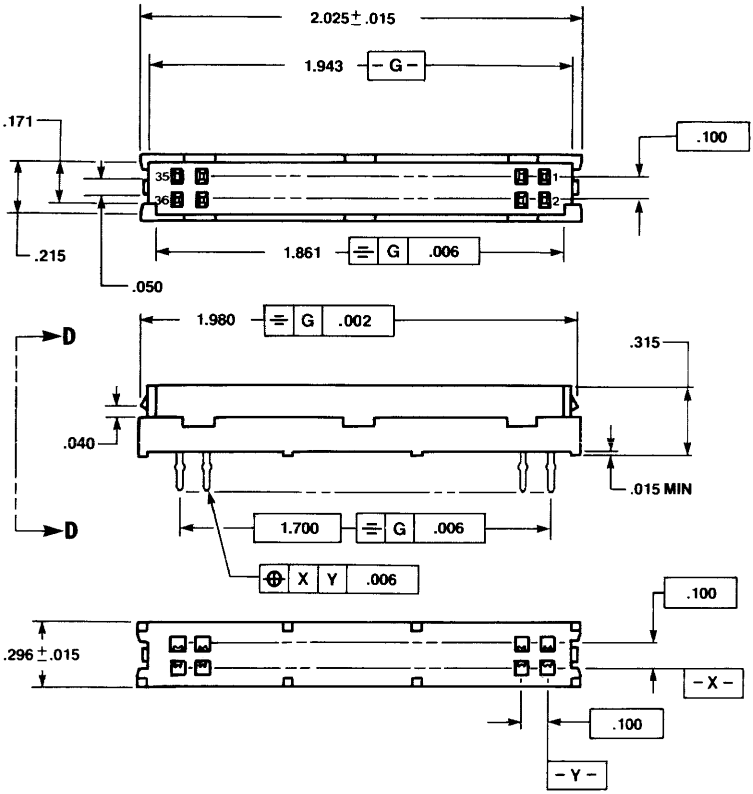
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 25 18/36 Pin Plug Assembly

The receptacle insulator shall provide closed entry feature and exclude the possibility of insertion of any device larger than 0.023 x 0.047 inches into the female contact.

The male contacts shall be completely shrouded to protect them from bending.

The connector plug and receptacle shall be polarized as per Figures 25, 26, 27, and 28 to prevent mismating.



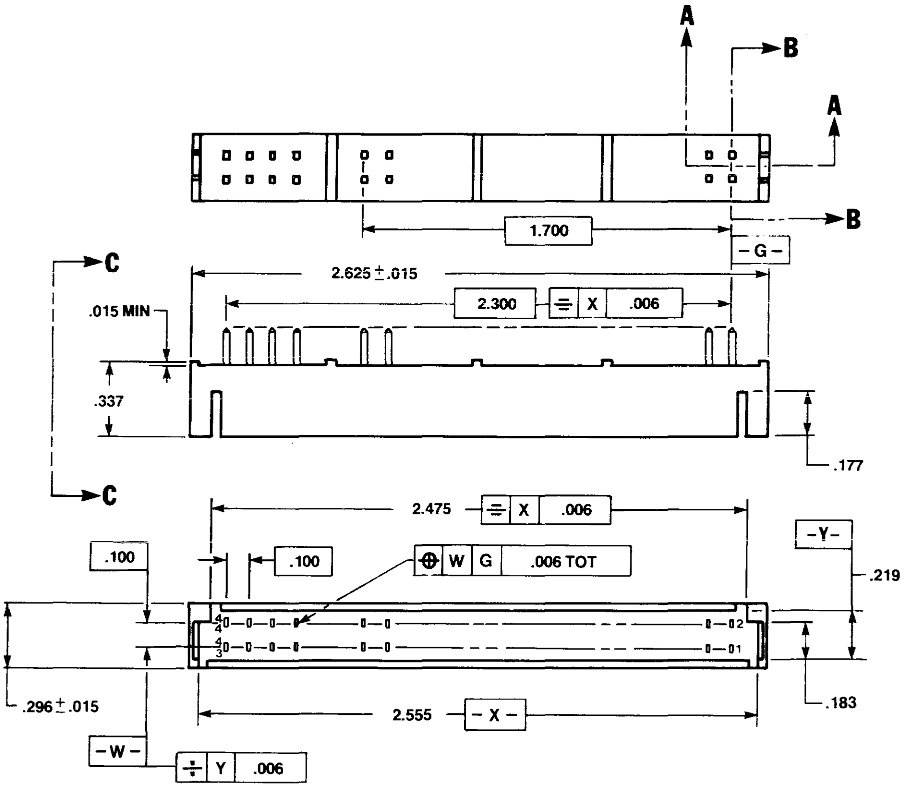
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 26 18/36 Pin Receptacle Assembly

5.3 ELECTRICAL REQUIREMENTS

The electrical specifications shall meet the following requirements:

(1) Current Rating:	3 amperes per contact.
(2) Dielectric Withstanding Voltage:	1200 V RMS minimum.
(3) Insulation Resistance:	1000 megohms minimum.
(4) Contact resistance:	0.01 ohms maximum.



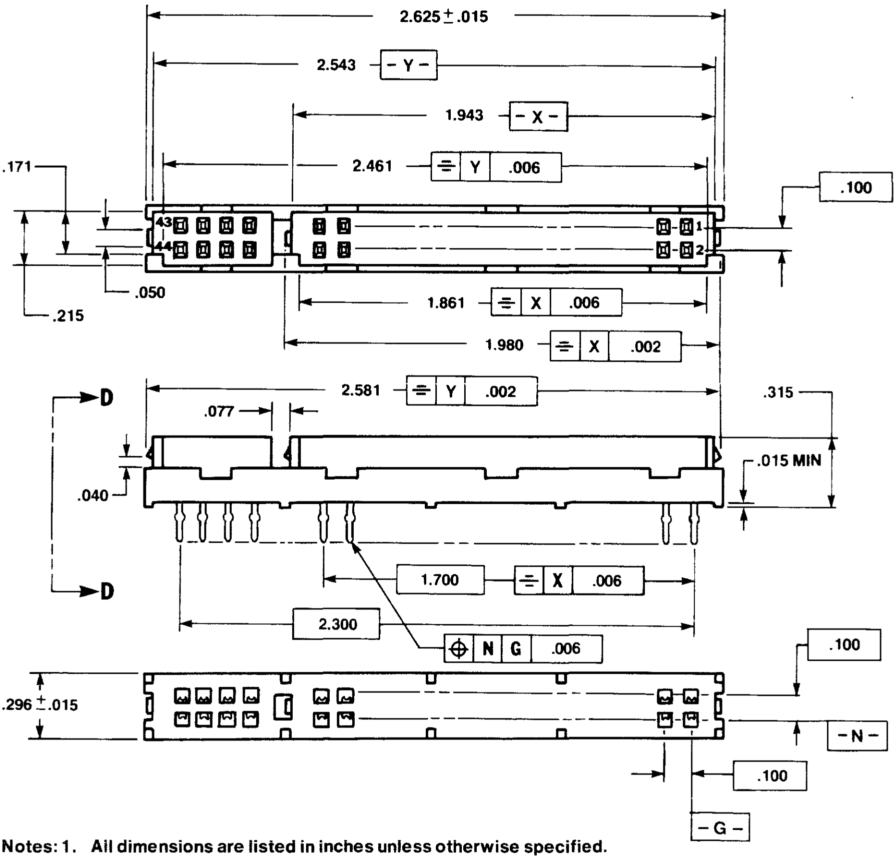
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 27 22/44 Pin Plug Assembly

5.4 ENVIRONMENTAL REQUIREMENTS

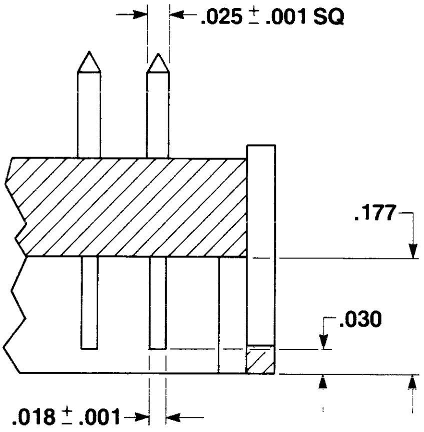
Connector shall meet all requirements of this specification without degradation due to environmental tests.

(1) Ambient Temperature Range:	−55°C to +122°C.
(2) Vibration:	Test criteria: ± 0.04 inch amplitude displacement, 15 one minute sweeps 10 to 65 Hz; 15 G, 65 to 2000 Hz, three minute dwell at resonant frequency in each plane.
(3) Shock:	Test criteria: 30G, 11 ± 1 millisecond with half sine wave shape, 3 times in each plane.
(4) Humidity:	0 to 95% without condensation, 0°C to 70°C (MIL STD 202, Method 106, less steps 7a and 7b)



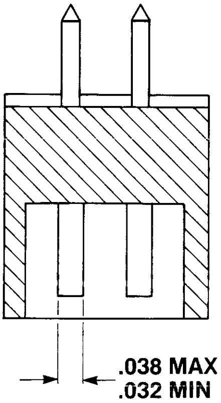
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 28 22/44 Pin Receptacle Plug Assembly



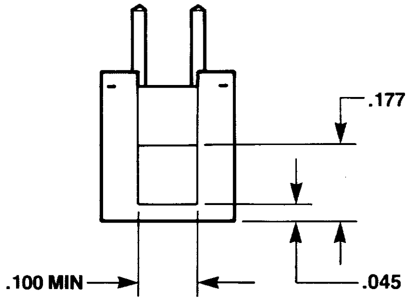
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 29 Section A-A For Figures 25 and 27



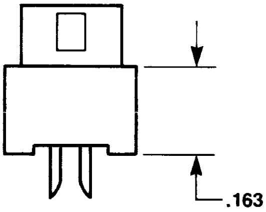
Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 30 Section B-B For Figures 25 and 27



Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 31 Section C-C For Figures 25 and 27



Notes: 1. All dimensions are listed in inches unless otherwise specified.
2. All tolerances are .xx ± 0.01 or .xxx ± 0.005 unless otherwise specified.

Figure 32 Section D-D For Figures 26 and 28

CHAPTER 6
LEVELS OF COMPLIANCE

This section presents the concept and notation of levels of compliance for the iSBX Bus. The notion of levels of compliance is included to allow the use of iSBX Bus products of varying capability manufactured by diverse vendors. It bounds the variability allowed within the iSBX Bus specification, and provides a succinct and convenient notation for these variables.

6.1 VARIABLE ELEMENTS OF CAPABILITY

The iSBX Bus allows for variation in the data path width, DMA support, and interlocked operation. Each is discussed in the following paragraphs.

6.1.1 Data Path

Data path variations exist for both expansion modules and baseboards. Expansion modules may either support 8- or 16-bit data paths. Sixteen bit baseboards may support only 8-bit data paths or both 8- and 16-bit data paths. Eight bit baseboards may support only 8-bit data paths.

6.1.2 DMA Support

Both expansion modules and baseboards may optionally support DMA operations. In order for DMA to be used in a system, both the expansion module and baseboard shall support the signals that are required (MDRQT, MDACK*, TDMA).

6.1.3 Interlocked Operation

Baseboards may optionally not support this feature. Expansion modules may optionally require this feature. The feature is implemented via the MWAIT* signal. Typically, baseboards will support the interlocked operation and expansion modules will not require it. The purpose for not requiring interlocked operation from all baseboards is to allow the use of low cost single-chip microcontroller devices that do not support a ready function.

6.2 BASEBOARDS AND EXPANSION MODULES

When constructing an iSBX Bus system, it is not necessary that all modules have identical capabilities. The only restriction is that the system will only support modes of operation supported by both the baseboard and the expansion module. For example, an expansion module that supports DMA cannot be plugged onto a baseboard that does not support DMA and result in a system that supports DMA. It is the responsibility of the system designer to guarantee that a required system feature is supported by all system components.

6.3 COMPLIANCE LEVEL NOTATION

The following notation allows a vendor to succinctly and accurately specify a product's level of compliance with the iSBX Bus specification. The lack of an element specification implies that either 1) there is no compatibility or 2) there is no requirement for that element.

6.3.1 Data Path

D8 represents an 8-bit expansion module

D16 represents a 16-bit expansion module

D8/8 represents an 8-bit baseboard that can support an 8-bit expansion module

D16/8 represents a 16-bit baseboard that can support an 8-bit expansion module

D16/16 represents a 16-bit baseboard that can support an 8- or 16- bit expansion module

6.3.2 DMA Support.

DMA represents a baseboard or expansion module that can support DMA operations

6.3.3 Interlocked Operation

F represents a baseboard that does not support interlocked operation (that is, only full speed operations are performed)

I represents an expansion module that requires interlocked operation (that is, requires the baseboard to support MWAIT*)

6.3.4 Examples

A 16-bit baseboard that supports both 8- and 16-bit expansion modules, that has DMA capability, and that supports the MWAIT* function would be specified as follows:

iSBX Bus Compliance: D16/16 DMA

An 8-bit expansion module that does not require the MWAIT* line and does not support DMA would be specified as follows:

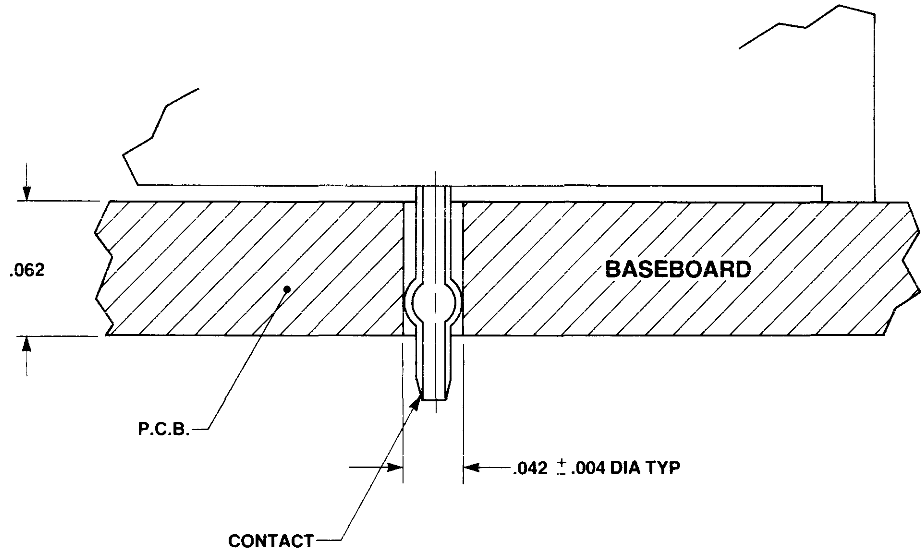
iSBX Bus Compliance: D8

6.3.5 Compliance Marking

The compliance levels of a product shall be documented in all product specifications and optionally marked on the printed circuit board.

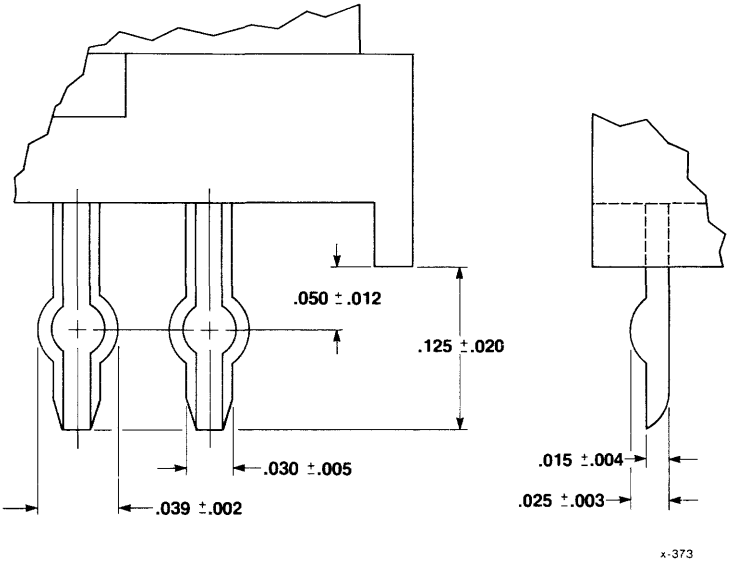
APPENDIX A
PRINTED CIRCUIT HOLE SIZE AND CONTACT SIZE

The following information is not part of this specification, however, is included for clarification. It presents an example of the requirements for the connector tail and printed circuit board hole requirements on a 0.062 inch thick printed circuit board.



All dimensions are listed in inches unless otherwise specified.

Figure A-1 Baseboard Hole Size and Retention

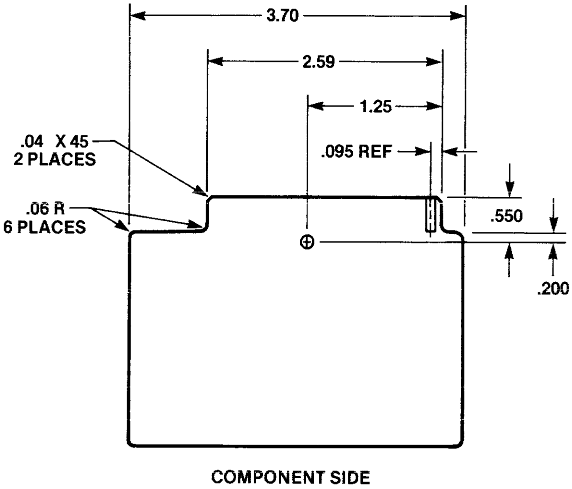


All dimensions are listed in inches unless otherwise specified.

Figure A-2 Baseboard Contact Tail Size and Retention

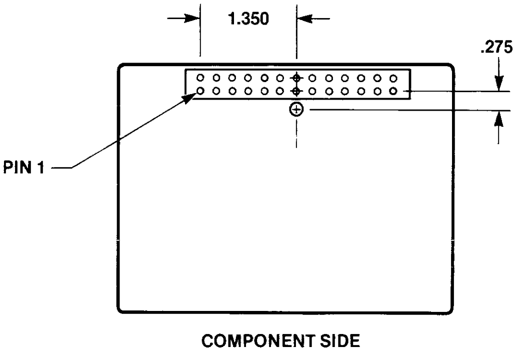
APPENDIX B
BASEBOARD CONSIDERATIONS

The following information is not part of this specification, however, is included for clarification. It presents an example of the requirements for connector positioning for different types of I/O connectors on the expansion modules.



All dimensions are listed in inches unless otherwise specified.

Figure B-1 Expansion Module With Edge-Type Connector



All dimensions are listed in inches unless otherwise specified.

Figure B-2 Expansion Module With Pin-and-Socket-Type Connector

iLBX™ EXECUTION BUS SPECIFICATION

III

PREFACE

The iLBX bus is one of the subsidiary buses within the overall Multibus Interface System. The iLBX bus is stand-alone to the extent that its interface and protocol do not require the existence of the general purpose Multibus interface or any of the other subsidiary buses. The iLBX bus uses the form factor of the Multibus P2 connector and imposes restrictions on board designs implementing both the general purpose Multibus interface and the iLBX bus.

This specification describes the operation protocol of the iLBX bus and defines the electrical and mechanical requirements of the iLBX bus. A section of design guideline examples provide additional insight for implementing the iLBX bus in a system. This specification does not duplicate specification information from the Multibus Interface Specification or any of the subsidiary bus specifications. Information on the Multibus interface or a subsidiary bus is provided in the following specifications.

- *Intel MULTIBUS® Interface Specification,* Order Number: 9800683
- *Intel iSBX™ Bus Specification,* Order Number: 142686
- *Intel MULTICHANNEL™ Bus Specification,* Order Number: 144330

This specification follows the general guidelines in the "Recommendations on Terminology for IEEE Computer Society Interface Standards" review draft dated September 9, 1981, and revised November 3, 1981, and June 3, 1982. In compliance with the terminology recommendations, this specification uses decimal notation when numbering bus lines with bit 0 as the least significant bit. This specification also uses the trailing asterisk to designate active Low signal lines. Where Multibus interface signal names (or subsidiary bus signal names) are used in this specification, these names are converted to comply with the terminology recommendations. For example, the Multibus address extension line ADR14/ is listed in this specification as ADR20*.

CONTENTS

CHAPTER 1 **Page**

INTRODUCTION

CHAPTER 2

FUNCTIONAL DESCRIPTION

CHAPTER 3
ELECTRICAL SPECIFICATION

CHAPTER 4
MECHANICAL SPECIFICATION

CHAPTER 5
DESIGN GUIDELINES AND SYSTEM APPLICATIONS

CONTENTS (continued)

CHAPTER 6
LEVELS OF COMPLIANCE

APPENDIX A
iLBX™ SIGNAL DESCRIPTIONS SUMMARY

III

FIGURES

TABLES

CHAPTER 1 INTRODUCTION

The Local Bus Extension (iLBX) bus is a specialized electrical and mechanical interfacing protocol operating within the overall Multibus interfacing system. The iLBX bus provides local memory expansion which is physically off-board, without loss of execution speed. A typical iLBX installation would have a master iSBC processor board attached to an additional memory board via the iLBX bus lines. Because of the increased execution speed of the iLBX bus, this off-board memory will be used as local on-board memory by the processor board. Up to five boards may use the iLBX bus in one system.

1.1 MULTIBUS® INTERFACE OVERVIEW

Figure 1 illustrates the overall Multibus interfacing system. The foundation of the Multibus interfacing system is the general purpose Multibus interface, the flexible bus structure used to interface the family of Intel's 80/86 products including both 8- and 16-bit products. The Multibus interface supports both 8- and 16-bit data transfers and direct addressability of up to 16 megabytes of memory. In many systems, the Multibus interface provides all of the required interconnect capability for the system.

As systems grow in complexity and performance, the throughput demands on the interconnect architecture increase. The Multibus interfacing system meets these demands by offloading specific interconnect needs to the following subsidiary buses:

- iSBX™ bus
- MULTICHANNEL™ bus
- iLBX™ bus

In a fully expanded Multibus interface system, the Multibus interface is used mainly for system control and low to medium-speed data transfers.

1.1.1 iSBX™ Bus

Increasing the number of functions residing on each system board attached to the Multibus interface increases the system performance. The improved system performance results because the resident functions are accessed without bus arbitration. The trade-off becomes choosing the resident functions when designing the system board. The iSBX bus extension of the Multibus interfacing system helps reduce the need to make design choices. The special functions are designed onto individual small boards and connected to a system board using the iSBX bus interface. When installed on the system board, the special function operates as though it were residing on the system board. Thus a system designer can have resident on the system boards those special functions most advantageous to his system.

1.1.2 MULTICHANNEL™ Bus

Reducing the impact of burst-type peripherals (e.g. most disk peripherals) on the Multibus interface provides a second means of increasing system performance. The actual data transfers from a burst type peripheral can saturate a general purpose interface such as the Multibus interface. Adding more burst-type peripherals to a system often decreases the computing performance of the system. The Multichannel bus extension to the Multibus interfacing system helps reduce the bus-saturation problem. The Multichannel bus protocol specifically accommodates burst-type data transfers. The full performance improvement requires use of dual port memory accessed over both the Multichannel bus and the Multibus interface.

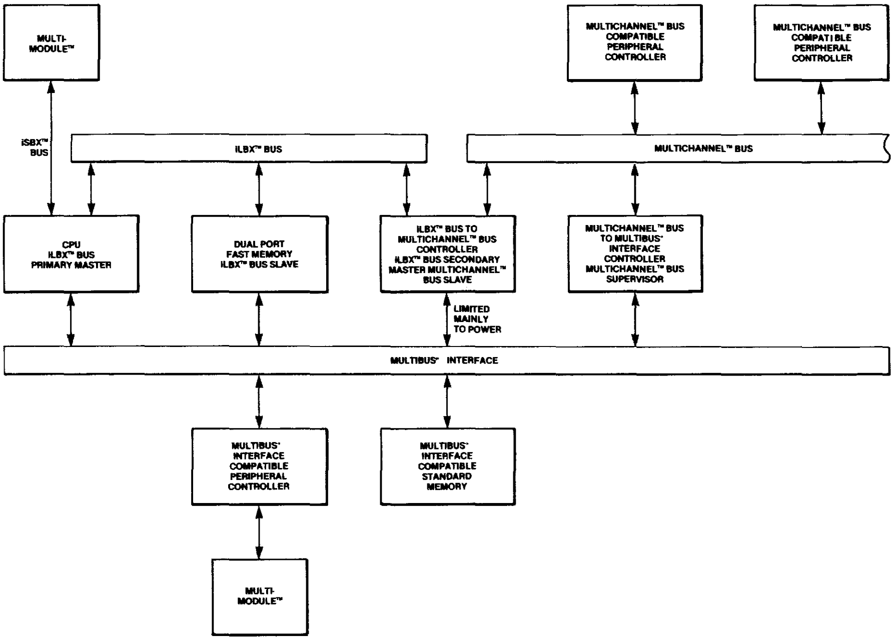

MULTI-MODULE™
MULTICHANNEL™ BUS COMPATIBLE PERIPHERAL CONTROLLER
MULTICHANNEL™ BUS COMPATI BLE PERIPHERAL CONTROLLER
iSBX™ BUS
ILBX™ BUS
MULTICHANNEL™ BUS
CPU ILBX™ BUS PRIMARY MASTER
DUAL PORT FAST MEMORY ILBX™ BUS SLAVE
ILBX™ BUS TO MULTICHANNEL™ BUS CONTROLLER ILBX™ BUS SECONDARY MASTER MULTICHANNEL™ BUS SLAVE
MULTICHANNEL™ BUS TO MULTIBUS* INTERFACE CONTROLLER MULTICHANNEL™ BUS SUPERVISOR
LIMITED MAINLY TO POWER
MULTIBUS* INTERFACE
MULTIBUS* INTERFACE COMPATIBLE PERIPHERAL CONTROLLER
MULTIBUS* INTERFACE COMPATIBLE STANDARD MEMORY
MULTI-MODULE™

Figure 1 MULTIBUS® Interface System

1.1.3 iLBX™ Bus

Dramatically increasing the local (on-board) memory resources of a high performance processor provides a third means of increasing system performance. As with other special functions, memory residing on the processor board improves system performance because the processor directly addresses the memory without waiting for bus arbitration. However, there is a physical limitation to the amount of memory that can reside on the processor board. The iLBX bus helps to reduce the physical space limitation. Using the iLBX bus, the additional memory no longer needs to be located on the processor board or on a multimodule attached to the processor board. The full 16 megabytes of memory addressable by the processor can be accessed over the iLBX bus and appear to the processor as though it were resident on the processor board. Dual porting the memory between the iLBX bus and the Multibus interface makes the same memory resources available to other system components.

1.2 iLBX^{TM} BUS GENERAL DESCRIPTION

The iLBX bus configuration uses the form-factor of the standard 60-pin Multibus P2 connector. It occupies 56 of the P2 connector pins (55 defined signal lines and one reserved signal line). The four Multibus address extension lines (pins 55 through 58) on the Multibus P2 connector retain the standard Multibus interface functions.

The iLBX bus is designed for direct high-speed Master-Slave data transfers and provides the following features.

- A minimum of two and up to five devices can be connected over the iLBX bus.
- Two (maximum) masters can share the bus, limiting the need for bus arbitration.
- Bus arbitration is asynchronous to the data transfers.
- Slave devices are defined as byte-addressed memory resources.
- Slave device functions are directly controlled from iLBX bus signal lines.

CHAPTER 2
FUNCTIONAL DESCRIPTION

III

2.1 INTRODUCTION

The Functional Description defines the various elements of the iLBX bus interface. These elements include descriptions of the device categories using the iLBX bus, the signal line grouping and functions, the timing requirements, and the bus communication protocol.

2.2 NOTATION CONVENTIONS

The general notational conventions used in this manual conform to the notational conventions used in the Multibus Specification. The following paragraphs summarize the notational conventions.The iLBX bus lines are assigned unique names and, for brevity, unique mnemonics. The signal line names are shown with initial capital letters when used in text. The corresponding signal line mnemonics are shown in all capital letters. Signal mnemonics for lines that are Active when electrically High (also called positive true) or Active when either electrically High or Low (the data lines for example) do not have a special terminating character as the last character in the mnemonic. Signal mnemonics for lines Active when electrically Low (also called negative true) have an asterisk (*) character as the last (terminating) character in the mnemonic.

The descriptions of electrical signal characteristics use the terms High and Low (in initial capital letters) corresponding to the relative voltage level of the signal. The terms true, false, 1, and 0 are avoided to reduce misinterpretation. Thus an Active Low signal is asserted when its relative voltage level is Low. Table 1 relates the electrical signal characteristics to the corresponding logical and state notations. The Example mnemonic, XMPL, illustrates the notational conventions.

The address and data bit numbering scheme used with the iLBX bus has bit 0 as the least significant bit, and decimal numbers are used to identify the lines. Thus, Data line 0 (DB0) carries the least significant bit and Data line 15 (DB15) carries the most significant bit.

Table 1 Notational Summary

SIGNAL NAME	NOTATION		
	ELECTRICAL	LOGICAL	STATE
XMPL	H, High L, Low	1, True 0, False	Active, Asserted Inactive
XMPL*	H, High L, Low	0, False 1, True	Inactive Active, Asserted

2.3 iLBX^{TM} BUS DEVICES

Three device categories interface to the iLBX bus. The device categories are the following:

- Primary Master
- Secondary Master
- Slave

At most, five devices can be simultaneously attached to the iLBX bus. The set of devices must include a *Primary Master* and one *Slave* device. The remaining three devices are optional and may include additional Slave devices and one *Secondary Master.* The Slave device(s) contain the memory resources used by the Primary Master and the Secondary Master. The combined, directly accessible, memory total for the Slave devices is 16 megabytes.

2.3.1 Primary Master

The Primary Master controls the iLBX bus and manages the Secondary Master access to the Slave memory resources. Every implementation of the iLBX bus must have a device performing all, or an allowed subset, of the Primary Master functions. A maximum of one Primary Master can be attached to the iLBX bus. Typically the system processor board includes the Primary Master function as an on-board function with the iLBX bus as an extension of the Primary Master's on-board local bus.

The Primary Master must perform three specific functions in addition to general iLBX-bus control. The Primary Master must actively drive all iLBX bus signal lines (except Slave Acknowledge and Secondary Master Request) unless it relinquishes signal line control to the Secondary Master. It must provide a +5VDC pull-up termination for those iLBX bus lines requiring termination. It must monitor the Secondary Master Request signal line and release control of the iLBX bus to a Secondary Master for data transfer.

Single-master implementations are allowed where a limited Primary Master does not monitor the Secondary Master Request signal line. When the Primary Master lacks the ability to monitor the Secondary Master Request signal line, a Secondary Master cannot be attached to the iLBX bus.

2.3.2 Secondary Master

The optional Secondary Master provides alternate access over the iLBX bus to the Slave resources. The Secondary Master transfers data over the iLBX bus in the same manner as a Primary Master; however, the Secondary Master must first request control and the Primary Master must acknowledge the request to pass iLBX bus control to the Secondary Master. The specified maximum of two masters (one Primary Master and one Secondary Master) reduces bus arbitration to a simple request and acknowledgement process. Bus control arbitration occurs asynchronously to the data transfers.

When the Secondary Master controls the iLBX bus, the Secondary Master must actively drive all signal lines (except Slave Acknowledge and Secondary Master Acknowledge) until it returns signal line control to the Primary Master. The Secondary Master must not provide additional signal line termination.

The Secondary Master must provide a means for varying the timing of its response to the Slave Acknowledge. The iLBX bus data-transfer-timing allows closely coupling (optimizing) the Slave device's data-transfer rate to the Primary Master's data-transfer rate. When the close coupling is implemented, part of the Slave device's access overhead occurs concurrently with the Primary Master's acknowledge acceptance overhead. When implementing the iLBX bus in a system, the Secondary Master's acknowledge acceptance timing must be adjusted to match the Primary Master's acknowledge acceptance timing to assure reliable data transfers.

2.3.3 Slave

The Slave device(s) contain the memory resources used by the Primary Master and the Secondary Master. The combined, directly accessible, memory total for the Slave devices is 16 megabytes. Any given iLBX bus implementation is limited to a maximum of four Slave devices.

The Slave devices must continually monitor the iLBX bus Address lines and the Address Strobe line. The Slave device detecting an assigned memory address on the Address lines assumes selection, and data transfer initiation occurs when the Slave device detects the Address Strobe signal. The iLBX bus protocol requires a positive response to complete any selection, and Slave devices must be able to actively drive the Slave Acknowledge line.

The iLBX-bus data-transfer-timing allows closely coupling (optimizing) the Slave device's data-transfer rate to the Primary Master's data-transfer rate. When the close coupling is implemented, part of the Slave device's access overhead occurs concurrently with the Primary Master's acknowledge acceptance overhead. Use of the optimized operation is optional, and a Slave device designed to implement optimized operation must provide a means for varying the timing of its Slave Acknowledge. Refer to the timing specifications and timing adjustment requirements in the hardware reference manual for iLBX bus compatible devices.

2.4 SIGNAL LINE DESCRIPTIONS

The following four signal line categories make up the physical structure of the iLBX bus interface:

- address and data lines
- control lines
- command lines
- bus access lines

2.4.1 Address and Data Lines

The bus has 16 data lines and 24 address lines.

2.4.1.1 DATA LINES (DB15 - DBO)

All 8-bit and 16-bit data transfers between the active bus master and the selected Slave device use the 16 bi-directional data lines exclusively. The 16-bit data transfers use all 16 data lines. The 8-bit data transfers within the 16-bit data frame use the appropriate low-order (DB7 through DBO) or high-order (DB15 through DB8) data lines. Byte data is transferred between 8-bit devices using data lines DB7 through DBO. The state of unused data lines during 8-bit data transfers is undefined.

The general data signal line implementation specifications are as follows.

- The data lines require tri-state drivers and any iLBX bus device can drive the data lines.
- The data lines are positive true lines and only the *transmitting* device (master or slave) drives the lines. The *receiving* device and all inactive devices must hold their data line drivers in the high-impedance state during the data transfer.

2.4.1.2 ADDRESS LINES (AB23 - ABO)

The active bus master uses 24 address lines to select a Slave device and to specify a location in memory. The use of 24 address lines provides the ability to address 16-megabytes of memory space.

Only the active bus master drives the address lines in the iLBX bus. The condition of the address lines during the data time is undefined, and the Slave device must store (latch-in) the address information when the Slave device detects the Address Strobe signal (see Section 2.4.3.1).

The general address signal line implementation specifications are as follows.

- The address lines are positive true lines.
- Primary Masters and Secondary Masters must provide tri-state drivers for the address lines. Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use standard TTL drivers with drive characteristics comparable to the specified tri-state driver.

2.4.1.3 TRANSFER PARITY (TPAR*)

The Transfer Parity signal is an optional line and is available to improve data-transfer integrity. The Transfer Parity operates as an additional data line with identical timing requirements. The iLBX bus uses odd parity defined as follows: when there is an even number of one bits in the transfer element (byte, 16-bit word), the transmitting device drives the Transfer Parity line Low. Because the state of unused data lines is undefined, parity generation and checking is limited to the active data lines for the transfer element used. The iLBX bus does not provide a means for reporting a detected transfer parity error.

The general Transfer Parity signal line implementation specifications are as follows.

- The option must be available on all devices, both masters and slaves, if a transfer parity option is to be used on the iLBX bus.
- Any device designed with the parity option must provide a means for disabling recognition of a transfer parity error.
- All masters and Slave devices with the parity option must provide a tri-state driver for the Parity line.
- Only the transmitting device (master or slave) drives the Parity line. The receiving device must sample the Parity line and all inactive devices must hold their parity line driver in the high-impedance state during the data transfer.

2.4.2 CONTROL LINES

The active bus master specifies the data transfer parameters to the selected Slave device by using the three control lines.

2.4.2.1 READ-NOT-WRITE (R/$\overline{W}$)

The active bus master controls the direction of data transfer with the Read-Not-Write line. When driven Low, the active bus master transmits the data and the selected slave device receives the data. Driving the Read-Not-Write line High reverses the transfer direction.

The Read-Not-Write signal need not remain valid throughout the data transfer operation. Therefore, the Slave device must store (latch-in) the state of the Read-Not-Write signal line when the Slave device detects the leading (falling) edge of the Address Strobe (see Section 2.4.3.1).

The general Read-Not-Write signal line implementation specifications are as follows.

- Primary Masters and Secondary Masters must provide a tri-state driver for the Read-Not-Write line. Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use a standard TTL driver with drive characteristics comparable to the specified tri-state driver.

- Only the active bus master drives the Read-Not-Write line. The inactive master must hold the Read-Not-Write line-driver in the high-impedance state.

2.4.2.2 DATA ELEMENT SELECT CONTROL LINE (BHEN)

The iLBX bus data transfers take place within an overall data frame that is limited by the iLBX bus configuration. Within the data frame limits, transfer of an 8-bit (byte) and 16-bit (word) data element is allowed. The particular size data element and its location within a data frame must be specified to the slave device by the active bus master. Refer to Table 2 for element size and location within a 16-bit data frame.

The active bus master controls the type of data transfer (8-bit or 16-bit) using the Byte High Enable (BHEN) element select line and the low-order address bit (AB0). The four signal-level combinations of these two lines specify both the element size and the element location within the data frame. The signal-level combinations are shown in Table 3.

Table 2 16-Bit Data Frame

BITS 15 - 8	BITS 7 - 0
HIGH BYTE	LOW BYTE
WORD	

Table 3 Element Selection

ELEMENT	SIGNAL & LEVEL	
	BHEN	ABDO
HIGH BYTE	HIGH	HIGH
LOW BYTE	LOW	LOW
RESERVED	LOW	HIGH
WORD	HIGH	LOW

The BHEN signal is not required to remain valid throughout the data transfer operation. The Slave device must store (latch-in) the configuration of the signal line when the Slave device detects the Address Strobe signal.

The general BHEN signal line implementation specifications are as follows.

- Primary Masters and Secondary Masters must provide tri-state drivers. Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use a standard TTL driver with drive characteristics comparable to the specified tri-state driver.

- The Byte High Enable line must be implemented on all masters and Slave devices.

- Only the active bus master drives BHEN. The inactive master must hold the BHEN line driver in the high-impedance state.

2.4.3 Command Lines

The active bus master and the selected Slave device use three command lines to initiate, control, and terminate a data transfer.

2.4.3.1 ADDRESS STROBE (ASTB*)

The active bus master drives the Address Strobe line Low to initiate a data transfer cycle. The control line and address line signal levels must be valid prior to the bus master driving the Address Strobe line Low. The Slave device(s) decodes the address to determine if it is selected. The selected Slave device must store the control and address information at the leading (falling) edge of the Address Strobe signal. The selected Slave device then proceeds with the data transfer.

The general Address Strobe signal line implementation specifications are as follows.

- Primary Masters and Secondary Masters must provide a tri-state driver for the Address Strobe line. Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use a standard TTL driver with drive characteristics comparable to the specified tri-state driver.

- Only the active bus master drives the Address Strobe line. The inactive master must hold its address strobe line driver in the high-impedance state.

2.4.3.2 DATA STROBE (DSTB*)

The active bus master drives the Data Strobe line Low to set-up the actual transfer of data. The active bus master drives the Data Strobe line High after the data transfer is completed to terminate the data transfer cycle. The meaning of the Data Strobe signal varies depending on the direction of data transfer, from master to slave (write) or from slave to master (read).

During a read operation, the active bus master indicates when it is ready to accept data from the selected slave device by driving the Data Strobe line Low. The active bus master must put its data line (DB15 - DB0) tri-state drivers in the high-impedance state before driving the Data Strobe line Low. The selected slave device starts driving the required lines after detecting the leading (falling) edge of the Data Strobe signal.

During a write operation, the active bus master indicates the availability of data for the selected slave device by driving the Data Strobe line Low. The active bus master is allowed a set-up time for the data lines after it drives the Data Strobe line Low. The selected slave device samples the data after first detecting the leading (falling) edge of the Data Strobe signal plus the specified data set-up time.

The general Data Strobe signal line implementation specifications are as follows.

- Primary Masters and Secondary Masters must provide a tri-state driver for the Data Strobe line. Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use standard a TTL driver with drive characteristics comparable to the specified tri-state driver.
- Only the active bus master drives the Data Strobe line. The inactive master must hold its Data Strobe line driver in the high-impedance state.
- The active bus master must hold the Data Strobe line Low for the specified minimum time after it receives the Acknowledge signal from the selected slave device.

2.4.3.3 ACKNOWLEDGE (ACK*)

The selected slave device responds to the active bus master by driving the Acknowledge line Low.

The Acknowledge signal timing requirements allow flexibility. The flexibility allows overlapping the internal operation overhead of the selected master with the bus set-up time of the selected slave device. Thus during a read operation, the selected slave device can drive the Acknowledge line Low prior to completing bus set-up of the data lines. The slave device must allow sufficient time to complete bus set-up before the active bus master internally recognizes and responds to the Acknowledge signal.

During a write operation, the Slave device can drive the Acknowledge line Low any time after the leading (falling) edge of the Address strobe provided it can accept the data from the active bus master within the specified time and the active bus master meets the optimized write timing requirements. Overlapping bus set-up time with the Primary Master's internal overhead improves the data transfer performance on the iLBX bus.

The Slave device should provide a means for varying its Acknowledge response time for read operations. The variable Acknowledge timing allows optimizing the data-transfer timing. A Slave device that does not provide a means for adjusting the Acknowledge timing can not drive the Acknowledge line Low before it drives the data lines.

The Slave device should provide a means for including or excluding the Data Strobe signal state as a prerequisite for driving the Acknowledge line Low to allow for optimized or non-optimized write operation. The exclusion of the Data Strobe going Low as a prerequisite allows the close coupling and overhead overlap to optimize the write data transfer operation. A Slave device that does not provide the choice of including the Data Strobe qualification must wait for the leading edge of the Data Strobe before driving the Acknowledge line Low.

The general Acknowledge signal line implementation specifications are as follows.

- Each Slave device must provide an open collector driver for the Acknowledge line.

- When a master on the iLBX bus cannot meet the optimized timing requirements, the Slave device(s) must be configured for non-optimized operation.

- Both Primary and Secondary Masters should provide a timer to force an Acknowledge locally and avoid a system lock-up if the addressed Slave device fails to respond. The iLBX bus does not provide a means for reporting a slave response failure.

2.4.4 Bus Access Lines

The Primary Master and the Secondary Master use the three bus access lines to transfer bus control between the master devices and to control alternate access to dual ported memory on a Slave device.

2.4.4.1 LOCK (LOCK*)

The active bus master restricts access through the alternate port to dual port RAM memory on a Slave device by driving the Lock line Low. By locking the alternate access, the active bus master assures that shared data stored in the Slave device is not disturbed until the active bus master completes its use of the data.

The Slave device accepts the Lock signal in conjunction with a data transfer and the Slave device remains locked until Lock signal goes inactive. The Slave device must be selected by the active bus master before the Slave device accepts the state of the Lock signal. To lock a given data transfer cycle to the following data transfer cycle, the active bus master must drive the Lock signal Low before the end of the first data transfer cycle (before the trailing edge of the Data Strobe signal) and keep the Lock signal active until after the start of the last locked data transfer cycle (after the leading edge of the Address Strobe).

Dual ported iLBX Slave devices that recognize both the iLBX bus Lock and the Multibus interface Lock present special design considerations. When alternate access to dual port memory can be locked from either port, the system can become deadlocked through access contention to the dual port memory. By limiting the lock application to the selected Slave device, the possibility of an access contention deadlock can be eliminated. However, the lock limitation allows the possibility of corrupting data if the logical memory in use crosses the physical boundary between two Slave devices. Because the logical memory space crosses the physical boundary, only part of the logical memory space is actually locked at any one time. A Slave device designed to recognize the iLBX bus Lock signal any time Lock is active avoids any risk of corrupting memory but risks system deadlock. A Slave device designed with the type of recognition optional allows the system environment to dictate which form of Lock recognition to use.

The general Lock signal line implementation specifications are as follows.

- Primary Masters and Secondary Masters must provide a tri-state driver for the Lock line. Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use a standard TTL driver with drive characteristics comparable to the specified tri-state driver.

- Only the active bus master drives the Lock line. The inactive master must hold the Lock line driver in the high-impedance state.

- Slave devices with single ported memory (iLBX bus only) and Slave devices with ROM memory need not recognize Lock.

- The selected dual ported slave device must recognize the Lock signal. The selected slave device must exclude alternate access until the Lock signal is driven High.

2.4.4.2 SECONDARY MASTER REQUEST (SMRQ*)

The Secondary Master requests use of the iLBX bus from the Primary Master by driving the Secondary Master Request line Low. Once the Secondary Master has control of the iLBX bus and completes its bus operation, it returns control of the iLBX bus to the Primary Master by driving the Secondary Master Request line High.

The Secondary Master must provide a TTL driver for the Secondary Master Request line and the Primary Master must provide a TTL receiver.

2.4.4.3 SECONDARY MASTER ACKNOWLEDGE (SMACK*)

The Primary Master allows use of the iLBX bus by the Secondary Master by driving the Secondary Master Acknowledge line Low after the Secondary Master drives the Secondary Master Request line Low. The Primary Master must continue to drive the Secondary Master Acknowledge line Low until after the Secondary Master drives the Secondary Master Request line High. When making the bus control transfer from the Primary Master to the Secondary Master, the Primary Master first grants bus use to the Secondary Master and then must put all tri-state drivers in the high-impedance state. When making the bus control transfer from the Secondary Master to the Primary Master, the Secondary Master first puts all tri-state drivers in the high-impedance state and then returns bus use to the Primary Master. The Primary Master must provide a TTL driver for the Secondary Master Acknowledge line and the Secondary Master must provide a TTL receiver.

2.5 iLBX^{TM} BUS PIN ASSIGNMENTS

The iLBX-bus configuration uses the form-factor of the standard 60-pin Multibus P2 connector and occupies 56 of the P2 connector pins. Table 4 lists the iLBX bus pin assignments for the 60-pin P2 edge connector. The four Multibus address extension lines (pins 55 through 58 on the P2 connector) retain the standard Multibus interface functions.

2.6 iLBX^{TM} BUS OPERATION PROTOCOL

The operation protocol for the iLBX bus includes the following three main operations:

- bus control access
- write data to memory
- read data from memory

The iLBX bus operations use asynchronous protocol with positive responses. Thus, specified signal level interactions must occur during an operation for the operation to proceed. Most iLBX bus timing parameters list only a minimum time or a maximum time for efficient use of the asynchronous protocol. The iLBX bus timing specifies a minimum abort time of 1 millisecond, within which a given bus transaction should be completed.

The following sections describe the different operations and the iLBX bus signal lines involved in each type of bus operation. The data-transfer operation descriptions cover the differences between 8-bit and 16-bit data transfers.

Table 4 iLBX™ Bus Pin Assignments, P2 Edge Connector

Component Side			Solder Side		
Pin	**Mnemonic**	**Signal Name**	**Pin**	**Mnemonic**	**Signal Name**
1	DB0	DATA LINE 0	2	DB1	DATA LINE 1
3	DB2	DATA LINE 2	4	DB3	DATA LINE 3
5	DB4	DATA LINE 4	6	DB5	DATA LINE 5
7	DB6	DATA LINE 6	8	DB7	DATA LINE 7
9	GND	GROUND	10	DB8	DATA LINE 8
11	DB9	DATA LINE 9	12	DB10	DATA LINE 10
13	DB11	DATA LINE 11	14	DB12	DATA LINE 12
15	DB13	DATA LINE 13	16	DB14	DATA LINE 14
17	DB15	DATA LINE 15	18	GND	GROUND
19	AB0	ADDRESS LINE 0	20	AB1	ADDRESS LINE 1
21	AB2	ADDRESS LINE 2	22	AB3	ADDRESS LINE 3
23	AB4	ADDRESS LINE 4	24	AB5	ADDRESS LINE 5
25	AB6	ADDRESS LINE 6	26	AB7	ADDRESS LINE 7
27	GND	GROUND	28	AB8	ADDRESS LINE 8
29	AB9	ADDRESS LINE 9	30	AB10	ADDRESS LINE 10
31	AB11	ADDRESS LINE 11	32	AB12	ADDRESS LINE 12
33	AB13	ADDRESS LINE 13	34	AB14	ADDRESS LINE 14
35	AB15	ADDRESS LINE 15	36	GND	GROUND
37	AB16	ADDRESS LINE 16	38	AB17	ADDRESS LINE 17
39	AB18	ADDRESS LINE 18	40	AB19	ADDRESS LINE 19
41	AB20	ADDRESS LINE 20	42	AB21	ADDRESS LINE 21
43	AB22	ADDRESS LINE 22	44	AB23	ADDRESS LINE 23
45	GND	GROUND	46	ACK*	SLAVE ACKNOWLEDGE
47	BHEN	BYTE HIGH ENABLE	48	R/$\overline{W}$	READ NOT WRITE
49	ASTB*	ADDRESS STROBE	50	DSTB*	DATA STROBE
51	SMRQ*	SECONDARY MASTER REQUEST	52	SMACK*	SECONDARY MASTER ACKNOWLEDGE
53	LOCK*	ACCESS LOCK	54	GND	GROUND
55	ADR22*	MULTIBUS® ADDRESS EXTENSION LINE 22	56	ADR23*	MULTIBUS® ADDRESS EXTENSION LINE 23
57	ADR20*	MULTIBUS® ADDRESS EXTENSION LINE 20	58	ADR21*	MULTIBUS® ADDRESS EXTENSION LINE 21
59	RES	RESERVED	60	TPAR*	TRANSFER PARITY

2.6.1 Bus Access

The iLBX bus uses a request and acknowledgement process to pass control between the two masters. A maximum of two masters can share the iLBX bus access to memory.

The specified maximum of two masters (one Primary Master and one Secondary Master) reduces bus arbitration to a simple request and acknowledgement process. Bus control arbitration occurs asynchronously to the data transfers.

The Primary Master controls bus access on the iLBX bus. A Primary Master must monitor the Secondary Master Request line and drive the Secondary Master Acknowledge line. The Secondary Master must drive the Secondary Master Request line and monitor the Secondary Master Acknowledge line. An iLBX bus master device designed to operate as either a Primary or a Secondary Master must be exclusively configured as one or the other when implementing the iLBX bus.

The Primary Master assumes control of the iLBX bus as the default configuration for iLBX bus control. The Secondary Master must drive the Secondary Master Request line High and the Primary Master must drive the Secondary Master Acknowledge line High during system initialization to set the default control configuration.

Following initialization, the Secondary Master requests control of the iLBX bus by initiating the transfer process. The bus access timing is illustrated in Figure 2. The Secondary Master drives the Secondary Master Request line Low indicating the need to control the bus. The Secondary Master can drive the request line Low anytime. The time to surrender control of the iLBX bus depends on the design implemented for the Primary Master. The time the Primary Master can retain control of the iLBX bus is not specified. Typically, the Primary Master releases control of the iLBX bus immediately if a data transfer is not in progress. When the Primary Master is actively transferring data, it retains control of the iLBX bus until completing the data transfer(s). The Primary Master relinquishes control of the iLBX bus by driving the Secondary Master Acknowledge line Low. The Primary Master is allowed a maximum of 35 ns after driving the Acknowledge line Low to put its tri-state drivers in the high-impedance state. The Secondary Master must wait a minimum of 35 ns after receipt of the Acknowledge signal before enabling its tri-state drivers out of the high-impedance state.

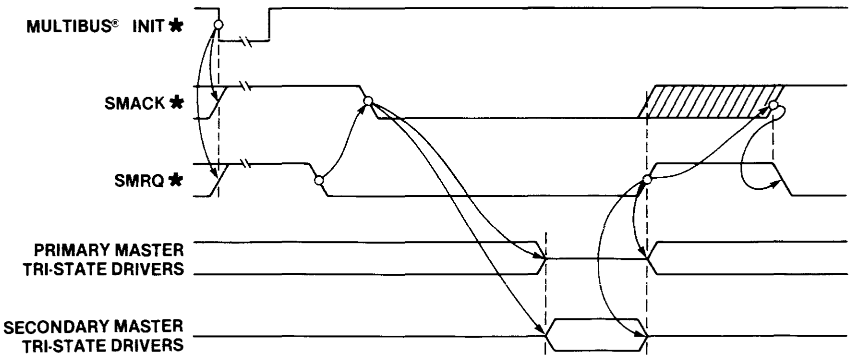


Figure 2 iLBX^{TM} Bus Granting Timing Relationship

The Secondary Master retains iLBX bus control until it completes the data transfer operation(s). The Secondary Master retains control by continuing to drive the Secondary Master Request line Low. The time the Secondary Master can retain control of the iLBX bus is not specified. This allows the Secondary Master the option of making a series of data transfers without returning control of the bus to the Primary Master; however, the Secondary Master typically surrenders control of the iLBX bus after completing the data transfer(s). The Secondary Master surrenders control of the iLBX bus by driving the Secondary Master Request line High. The Secondary Master must put its tri-state drivers in the high-impedance state before driving the request line High. The Primary Master can enable its tri-state drivers out of the high-impedance state when the request signal goes High. Concurrently, the Primary Master should drive the Secondary Master Acknowledge line High. After releasing control of the iLBX bus, the Secondary Master must detect the Secondary Master Acknowledge going High before it can again drive the Secondary Master Request line Low.

2.6.2 Data Transfer Operations

The data transfer operations all take place between the active bus master and the selected slave device. Because bus control-access occurs asynchronously to data transfers, it is not mentioned in the data transfer operation descriptions. Both types of data transfer operations, write and read, are similar with the main difference being the device that places the data on the iLBX bus data lines. Both write and read data transfers allow for both optimized and non-optimized operation. The optimized operation imposes additional timing requirements and considerations.

2.6.2.1 WRITE DATA-TO-MEMORY

The description of the write data-to-memory operation assumes full completion of any previous data transfer operation before the start of the write operation. The description also assumes the same master device is making another data transfer immediately following the operation described. The subsequent operation could be either read or write and is included here to show the operation to operation timing relationships. The active bus master could be either the Primary Master or the Secondary Master with the same resulting operation. Figure 3 illustrates the optimized 16-bit write data-to-memory timing relationships. The non-optimized write data-to-memory timing uses fixed signal sequences, described in the text, to assure a valid data transfer.

The active bus master initiates the write data-to-memory operation by placing the memory address on the address lines and a control configuration on the control lines. The active bus master must drive the various lines for the specified minimum set-up times before driving the Address Strobe line Low. The selected slave device stores the address information, including the data element selection information, when it detects the leading (falling) edge of the Address Strobe.

2.6.2.1.1 Optimized Operation

All devices attached to the iLBX bus must meet the additional timing requirements for optimized operation to implement optimized operation. The optimized write data-to-memory operation depends the master(s) meeting the specific maximum timing requirements from the leading edge of the Address Strobe signal. The active bus master must provide valid data a maximum of 80 ns from the leading edge of the Address Strobe signal and it must drive the Data Strobe line Low a maximum of 95 ns after the leading edge of the Address Strobe to meet the optimized operation timing requirements.

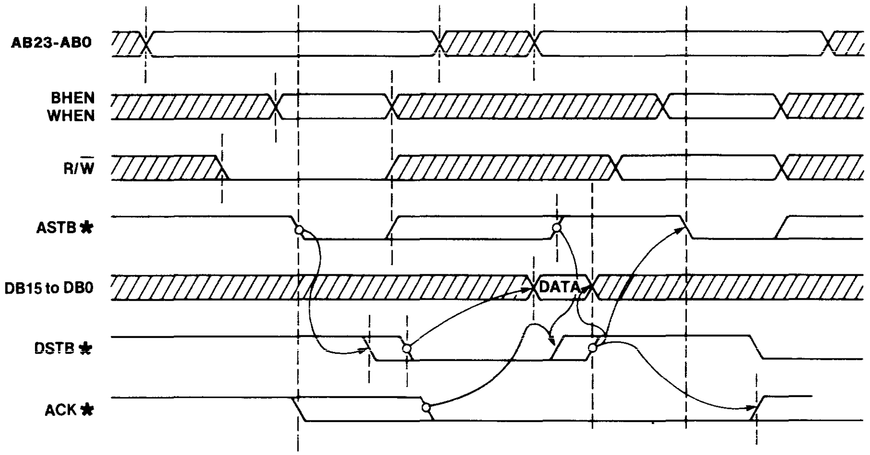


Figure 3 Write Data-To-Memory, 16-Bit Transfer Timing

Because the optimized operation timing specifies a data valid time relative to the leading edge of the Address Strobe, the Slave device can drive the Acknowledge line Low anytime after the leading edge of the Address Strobe, provided the Slave device can accept the data from the active bus master within the next 80 ns.

The master device must wait for the Acknowledge before completing the data transfer. The time coupling between the slave device and the master requires the master wait until the selected slave drives the Acknowledge line Low, and then the master must wait an additional 80 ns before driving the Data Strobe line High to complete the data transfer. If the Acknowledge is not received within 1 millisecond, the master can abort the operation by driving the Data Strobe line High.

2.6.2.1.2 Non-Optimized Operation

The non-optimized write data-to-memory operation is used when the master(s) can not meet the specific maximum timing requirements from the leading edge of the Address Strobe signal. For the non-optimized write data-to-memory operation, the leading (falling) edge of the Data Strobe becomes the critical timing element. The active bus master must now delay driving the Data Strobe line Low until it can provide valid data within 45 ns after the leading edge of the Data Strobe. The selected Slave device is restricted from pre-acknowledging the Data Strobe, and the Slave device must wait until the leading edge of the Data Strobe before driving the Acknowledge line, ACK*, Low. It can drive the Acknowledge line Low anytime after the leading edge of the Data Strobe provided it can accept the data from the active bus master within the next 80 ns.

The master device must wait for the Acknowledge before completing the data transfer. The time coupling between the slave device and the master requires the master wait until the selected slave drives the Acknowledge line Low, and then the master must wait an additional 80 ns before driving the Data Strobe line High to complete the data transfer. If the Acknowledge is not received within 1 millisecond, the master can abort the operation by driving the Data Strobe line high.

2.6.2.1.3 Operation Completion

The active master device completes the write data-to-memory operation by driving the Data Strobe line High.

When the write data-to-memory transfer is a single event transfer, the active master stops driving the data and address lines. The selected Slave stops driving the Acknowledge line and internally goes to the deselected state.

For sequential cycles of write data-to-memory operation, the active bus master can start address set-up for the next data transfer before driving the Data Strobe line High. The amount of data transfer overlap is limited by the minimum time of 25 ns from the rising edge of the Data Strobe to the falling edge of the next Address Strobe.

The overlap provides a potential cycle time of 105 ns per data transfer or a transfer rate of 9.5 MHz. Assuming 16-bit transfers, the resulting throughput is approximately 19 megabytes per second.

2.6.2.2 READ DATA-FROM-MEMORY

The description of the read data-from-memory operation assumes full completion of any previous data transfer operation before the start of the read operation. The description also assumes successive data transfers by a single master device. The active bus master could be either the Primary Master or the Secondary Master with the same resulting operation. The read data-from-memory operation uses roughly the same timing relationships as the write data-to-memory. The level of the Read-Not-Write signal and the device driving the data lines constitute the main differences between the two operations.

The description of the read data-from-memory operation includes the special considerations for optimizing the data transfer rate between the active bus master and the selected slave. Because the active bus master retains control of the data lines in the write data-to-memory operation, the bus master is the key device in determining the level of data transfer performance. However, during the read data-from-memory operation, the active bus master surrenders control of the data lines to the Slave device for part of the operation. Thus, the level the cooperation between the active bus master and the selected Slave is the key element in determining the level of data transfer performance. Figure 4 illustrates the optimized 16-bit read data-from-memory timing relationships. The non-optimized read data-from-memory timing uses fixed signal sequences, described in the text, to assure a valid data transfer.

The active bus master initiates the read data-from-memory operation by placing the memory address on the address lines and a control configuration on the control lines. The active bus master must drive the various lines for the specified minimum set-up times before driving the Address Strobe line Low. The selected slave device stores the address information, including the data element selection information, when it detects the leading (falling) edge of the Address Strobe.

The selected slave then drives the Acknowledge line, ACK*, Low anytime after the leading edge of the Address Strobe signal. When the selected slave device drives the Acknowledge line Low, it must meet the timing coupling requirements of the active master device and present valid data on the data lines before the active master device samples the data lines.

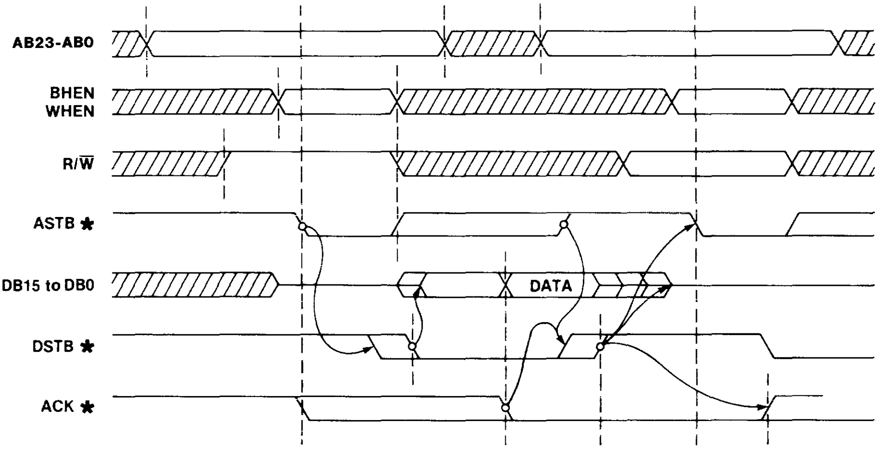


Figure 4 Read Data-From-Memory, 16-Bit Transfer Timing

2.6.2.2.1 Optimized Operation

The optimized read data-from-memory operation uses the signal overlap prohibited in the non-optimized operation. A Slave device designed for optimized operation must provide an adjustment for the length of time between when it drives the Acknowledge line Low and when it presents valid data on the data lines. The Primary Master determines the maximum allowed amount of signal overlap based on its acknowledge acceptance overhead time. The Slave device Acknowledge timing is then set to an overlap value equal to or less than the Primary Master's acknowledge acceptance overhead time. For any given Slave device, the length of time Acknowledge can precede valid data ranges from coincident (data valid at the same time the Slave device drives the Acknowledge line Low) to the maximum internal memory access time of the Slave device (immediate upon detecting the leading edge of the Address Strobe signal). Regardless of the allowed signal overlap with the optimized operation, the selected slave device must wait until the leading edge of Data Strobe before driving the data lines.

A minimum of 80 ns after detecting the Acknowledge line Low, the active bus master completes the optimized read data-from-memory operation by driving the Data Strobe line High. If the Acknowledge is not received within 1 millisecond, the master can abort the operation by driving the Data Strobe line High.

2.6.2.2.2 Non-Optimized Operation

The non-optimized read data-from-memory operation is used when a Slave device does not have variable Acknowledge to read data valid timing. For the non-optimized read data-from-memory operation, the selected Slave device must have data valid before driving the Acknowledge line Low. Because the leading edge of Data Strobe signals the selected Slave device that it can start driving the data lines, in non-optimized mode the selected Slave device must also wait for the Data Strobe before driving the Acknowledge line Low.

A minimum of 80 ns after detecting the Acknowledge line Low, the active bus master completes the non-optimized read data-from-memory operation by driving the Data Strobe line High. If the Acknowledge is not received within 1 millisecond, the master can abort the operation by driving the Data Strobe line High.

2.6.2.2.3 Operation Completion

The active bus master completes the read data-from-memory operation by driving the Data Strobe line High. When the read data-from-memory transfer is a single event transfer, the selected slave stops driving the data lines and the Acknowledge line and internally goes to the deselected state.

For sequential cycles of read data-from-memory, the active bus master can start address set-up for the next data transfer before driving the Data Strobe line High. The amount of data transfer overlap is limited by the minimum time of 25 ns from the rising edge of the data strobe to the falling edge of the next Address Strobe. The signal overlapping provides a total potential cycle time of 105 ns per data transfer or a transfer rate of 9.5 MHz. Assuming 16-bit data transfers, the resulting throughput is approximately 19 megabytes per second.

CHAPTER 3
ELECTRICAL SPECIFICATION

3.1 INTRODUCTION

This section defines the electrical requirements of the iLBX bus. The descriptions include the types of drivers and receivers required, the method, type, and location of line termination, general signal characteristics, and electrical timing.

3.2 ELECTRICAL STATE RELATIONSHIPS

The electrical state relationships used in this manual conform to the conventions used in the Multibus Specification. The iLBX bus uses commercial grade TTL components for all drivers and receivers. Table 5 relates the general industry voltage level standards for TTL components to the signal line notation conventions used in this manual. The specifications in Table 5 assume a power source of +5 Vdc, ±5 percent, referenced to logic ground. The iLBX bus does not include provision for system power and the electrical specification assumes that all power is drawn from the Multibus P1 connector power lines.

Table 5 Notational Summary

TTL ELECTRICAL LEVEL	TTL VOLTAGE RANGE	
	AT RECEIVER	AT DRIVER
H, High	+2.0 TO +5.25 Vdc	+2.4 TO +5.25 Vdc
L, Low	−0.5 TO +0.8 Vdc	0 to +0.5 Vdc

3.3 ENVIRONMENTAL REQUIREMENTS

The electrical specifications for the iLBX bus must be met under the following environmental conditions. The specifications list the ambient temperature requirements and the non-condensing requirements for humidity.

OPERATING

Temperature	0 to 55 degrees C
Relative humidity	0 to 85 percent

3.4 DC SPECIFICATIONS

Table 6 lists the iLBX bus DC specifications for the signal line drivers and the receiver loads presented to the signal lines. The DC specifications listed assume the use of devices typically associated with the standard 16-bit implementation of the iLBX bus. Refer to Section 6, Levels of Compliance, for the driver types required with the various allowed subsets of the iLBX bus. The drive and load requirements presented in Table 6 apply regardless of the iLBX bus subset implemented.

The drive requirements include the load capacitance an output driver must drive and for tri-state drivers the requirements assume four slave device loads. The load requirements include the maximum allowable input capacitance that any device can present to the signal line. The specifications assume the High drive signals are measured at +2.4 Vdc and the Low drive signals at +0.5 Vdc.

Table 6 DC Specifications

Signal Name	Driver Type	DC Termination (To +5 VDC)	Minimum Driver Requirements			Maximum Receiver Requirements		
			High	Low	Load Cap	High	Low	Load Cap
DB15-0	TRI-STATE	10K OHMS	0.6 ma	9 ma	75 pf	0.15 ma	2 ma	18 pf
TPAR*	TRI-STATE	10K OHMS	0.6 ma	9 ma	75 pf	0.15 ma	2 ma	18 pf
AB23-0	TRI-STATE	NONE	0.4 ma	20 ma	120 pf	0.10 ma	5 ma	30 pf
R/$\overline{W}$	TRI-STATE	NONE	0.2 ma	8 ma	75 pf	0.05 ma	2 ma	18 pf
BHEN	TRI-STATE	NONE	0.2 ma	8 ma	75 pf	0.05 ma	2 ma	18 pf
LOCK*	TRI-STATE	NONE	0.2 ma	8 ma	75 pf	0.05 ma	2 ma	18 pf
SMRQ*	TTL	10K OHMS	0.05 ma	2 ma	20 pf	0.05 ma	2 ma	18 pf
SMACK*	TTL	NONE	0.05 ma	2 ma	20 pf	0.05 ma	2 ma	18 pf
ASTB*	TRI-STATE	10K OHMS ¶	0.2 ma	9 ma	75 pf	0.05 ma	2 ma	18 pf
DSTB*	TRI-STATE	10K OHMS ¶	0.2 ma	9 ma	75 pf	0.05 ma	2 ma	18 pf
ACK*	OPEN COLL.	330 OHMS	N.A.	20 ma	45 pf	0.05 ma	2 ma	18 pf

Note: ¶ = Additional AC termination for both ASTB* and DSTB* lines are required on each slave device. each terminator is a series RC (100 ohm, 10 picofarad) network between the signal line and ground. The location of the termination network should be as close as possible to the receiver component input.

3.5 TERMINATION

DC and AC termination requirements are listed in Table 6. The DC termination for a particular line consists of a resistor connecting the line to +5VDC. The location of each resistor depends on the applicable signal line. Signal lines driven by a tri-state or open collector driver (that is, DB15 - DB0, TPAR*, ASTB*, DSTB*, and ACK*) has the termination resistors located at the Primary Master. If a Primary Master is not implemented on the iLBX bus a Secondary Master, operating as a limited Primary Master, must have the capability to provide the resistors. For the signal line, SMRQ*, the resistor is located at the Primary Master, and never at the Secondary Master.

An additional AC termination is required for lines ASTB* and DSTB* and is located at each slave device. Each slave device must provide a series RC network connecting the signal line to logic ground. The network should be placed in close proximity to the receiver component on the slave device.

3.6 AC SPECIFICATIONS

Table 7 lists the iLBX bus timing parameters for the signal lines. The table provides a reference designator for each timing parameter, a description of the timing parameter, the minimum and maximum timing requirements, and the source device where the timing parameter must be implemented. Figures 5 and 6 are timing charts that illustrate the timing relationships for the iLBX bus and the timing specifications on the timing charts use the reference designators from Table 7. Table 7 does not specify the typical transition rise and fall times for the iLBX bus drivers, however, the bus drivers should have slew rates less than 1 volt/nanosecond with less than 24 milliamps drive. Use of drivers having higher slew rates may degrade signal characteristics to an undesirable waveform. Figure 7 depicts a typical acceptable signal waveform during high-to-low and low-to-high transitions.

The following are the general and specific notes for Table 7.

General Notes:

- All times listed are nanoseconds unless otherwise noted
- TPAR* timing is the same as DB15 - DB0
- M refers to the current active bus master
- S refers to the currently selected slave device
- PM refers to the Primary Master
- SM refers to the Secondary Master

Specific Notes:

1. Board designs can implement either of two transfer rates, optimized and non-optimized, based on the degree of close coupling desired between the master and slave devices. Two factors determine the coupling and the degree of optimization realized when implementing the iLBX bus: the acknowledge acceptance time of the master device and the range of variability in the slave device to pre-acknowledge the data transfer. A master device designed for optimized operation must meet the t17 maximum time for write operation and the t11 maximum time for read operations. When the master devices meet the required times, the Slave device is allowed to drive the Acknowledge line Low any time after the leading edge of the Address Strobe. A master device that does not meet the maximum write time requirements, by default, transfers data using non-optimized timing, and the Slave device must wait for the leading edge of the Data Strobe before driving the Acknowledge line Low. See Note 3 for the Slave device timing restrictions.

2. The selected slave device must stop driving the Acknowledge line Low immediately upon detection of the trailing edge of the Data Strobe. The 45 ns maximum hold over time listed for the Acknowledge signal allows for the assumed input-to-output delay for the Acknowledge driver of 15 ns and the typical pull-up charge time through a 330 ohm resistor required to bring the Acknowledge signal from 0.2 Vdc to 2.4 Vdc assuming a worst case capacitive load of 45 pf.

3. The slave device should be provided with variable (typically discrete) timing capabilities for driving the Acknowledge line Low. For write operations, the slave device can drive the Acknowledge line Low anytime after the leading edge of the Address Strobe signal subject to the limitations listed in Note 1. For read operations, the slave device can pre-acknowledge the data transfer by driving the Acknowledge line Low before it provides valid data on the data lines. Pre-acknowledgement is subject to the limitations listed in Note 1. The amount of variability provided should range from 0.0 ns (data valid when the slave drives the Acknowledge line Low) to the maximum access time of the slave's memory resources (t_{acc}). If the board designer chooses not to provide variable timing, the slave device must have data valid at the time it drives the Acknowledge line Low.

4. The minimum t31 guarantees that a master does not start to drive the data bus (write cycle) until the slave has stopped driving the data bus (previous read cycle).

5. The t26 timing does not apply during system initialization (for example, when the Primary Master receives the Multibus interface initialization).

6. The t9 time used for computing t20 is the actual t9 time of the master. The t20 time can range from 0 to 20 nanoseconds.

7. The minimum operation abort time is 1 millisecond.

Table 7 iLBX^{TM} Bus Timing Parameters

REF	PARAMETER DESCRIPTION	TIMING		SOURCE	NOTE
		MIN	MAX		
t1	ASTB* Duration (Width)	25		M	
t2	Address Setup to Leading Edge of ASTB*	40		M	
t3	ADDRESS Hold After Leading Edge of ASTB*	40		M	
t4	BHEN, When Setup to Leading Edge of ASTB*	5		M	
t5	BHEN, When Hold After Leading Edge of ASTB*	5		M	
t6	$R/\overline{W}$ Setup to Leading Edge of ASTB*	20		M	
t7	$R/\overline{W}$ Hold After Leading Edge of ASTB*	25		M	
t8	Trailing Edge of ASTB* to Leading Edge of ASTB*	10		M	
t9	Leading Edge of ASTB* to Trailing Edge of ASTB*	25		M	
t10	DSTB* Duration (Width)	50		M	
t11	Leading Edge of ASTB* to Leading Edge of DSTB*	20	95	M	1
t12	ACK* Hold After Trailing Edge of DSTB*	0	45	S	2
t13	Leading Edge of ACK* to Read Data Valid	0	tacc	S	3
t14	Read Data Hold Time After Trailing Edge of DSTB*	0	45	S	
t15	Leading Edge of ACK* to Trailing Edge of of DSTB*	80		M	
t16	Leading Edge of DSTB* to Read Data Valid	0		S	
t17	Leading Edge of ASTB* to Write Data Valid		80	M	1
t18	Leading Edge of DSTB* to Write Data Valid		35	M	
t19	Write Data Hold Time After Trailing Edge of DSTB*	20		M	
t20	Leading Edge of ASTB* to First Sample of ACK* Line	45-t9		M	6
t21	LOCK* Setup to Trailing Edge of DSTB*	15		M	
t22	LOCK* Hold After Trailing Edge of DSTB*	15		M	
t23	SMACK* Low to Tri-State Drivers in High Impedance State		35	PM	
t24	SMACK* Low to Tri-State Drivers Out of High Impedance State	35		SM	
t25	SMRQ* High to Tri-State Drivers in High Impedance State		0	SM	
t26	SMRQ* High to Tri-State Drivers Out of High Impedance State	0		PM	5
t27	SMRQ* High to SMACK* High	0		PM	
t28	SMRQ* Low to SMACK* Low	0		SM	
t29	SMACK* High SMRQ* Low	0		PM	
t30	Leading Edge of ASTB* to Trailing Edge of DSTB* (Abort)	1 ms		M	7
t31	Write Data Active After Trailing Edge of DSTB*	45		M	4

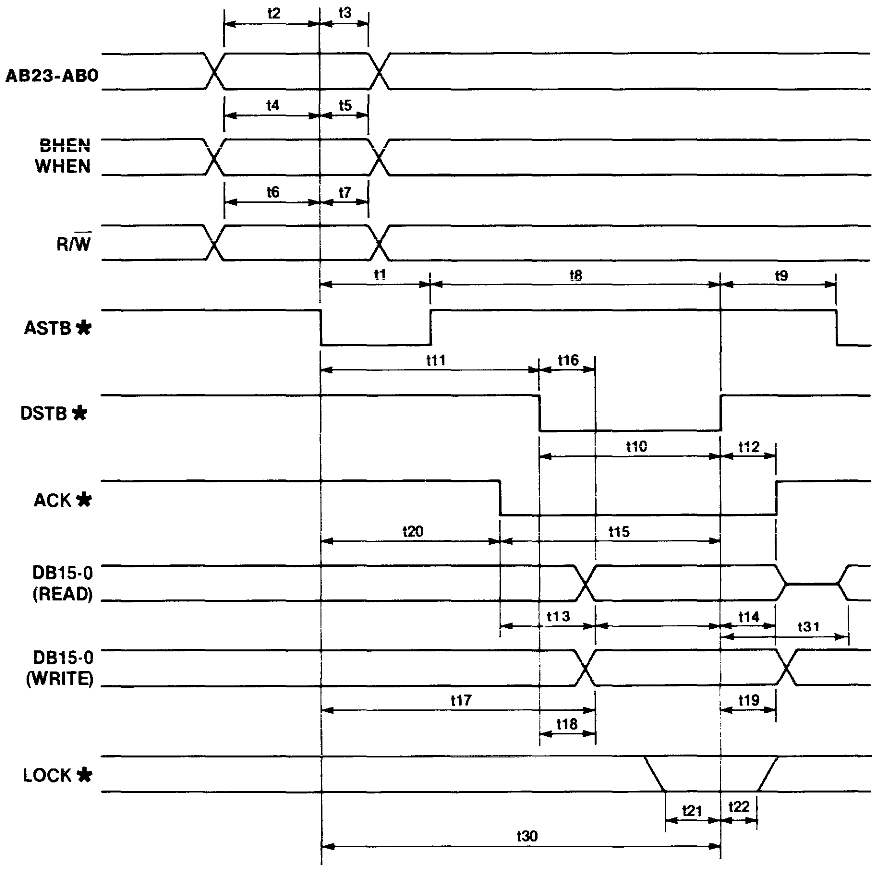


Figure 5 iLBX™ Bus Data Transfer Timing Chart

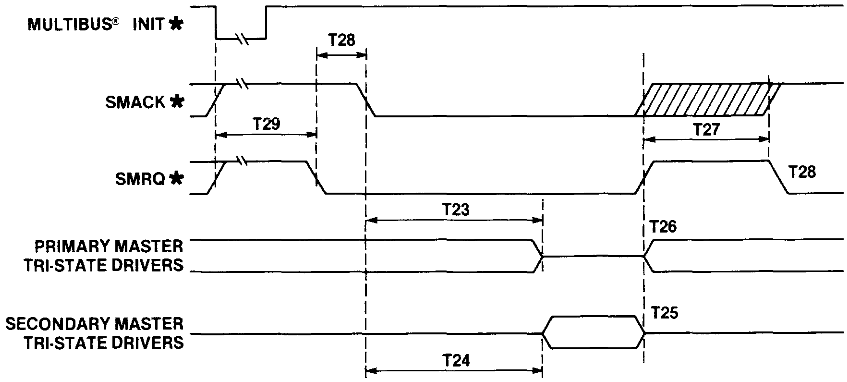
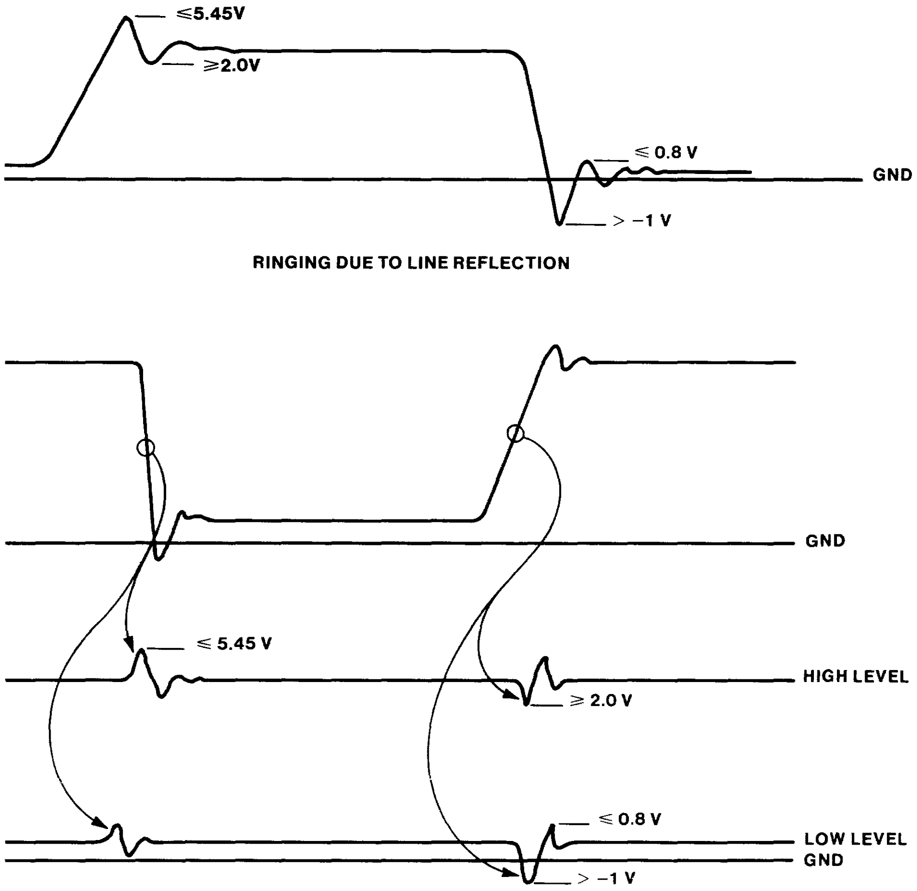


Figure 6 iLBX™ Bus Control Transfer Timing Chart



RINGING DUE TO LINE REFLECTION

LINE-TO-LINE COUPLING

Figure 7 Typical Acceptable Waveform

CHAPTER 4
MECHANICAL SPECIFICATION

4.1 INTRODUCTION

This section defines the physical and mechanical requirements that must be considered when designing iLBX bus compatible printed circuit boards or when implementing the iLBX bus in a system. The descriptions include the form factor requirements specific to the iLBX bus, the method, type, and location of connectors, and connector keying. Implementation of the iLBX bus on Multibus-compatible printed circuit boards is assumed. The iLBX bus Mechanical Specifications generally are limited to those specifications different from or in addition to the Multibus interface mechanical specifications.

4.2 iLBX™ BUS FORM FACTOR

Because of cable length restrictions, any board implementing the iLBX bus typically is installed in a Multibus chassis. A partial interface of the iLBX bus compatible device to the Multibus P1 connector is assumed because the iLBX bus does not provide any power connections, initialization signals, or interrupt capabilities. The iLBXbus specification requirements for board-to-board spacing, board thickness, component lead length, and component height above the board remain the same as in the Multibus specification. Refer to the *Intel MULTIBUS® Specification* for details on the general Multibus interface mechanical specifications.

4.2.1 Connector Locations and Board Outline

The Multibus interface P1 connector is unchanged for implementation of the iLBX bus. The iLBX bus resides on the Multibus form factor P2 connector and supercedes the Multibus interface definitions for the P2 signals. The 8- and 16-bit iLBX bus implementations use the same, 0.1 inch center, 60-pin P2 connector as the standard Multibus interface. Figure 8 illustrates the standard printed circuit board outline for the Multibus interface modified to accommodate the iLBX bus.

Figure 8 is limited to the basic form factor information for the iLBX bus implemented on a Multibus board. Refer to the Multibus Specification for additional information on the Multibus board form factor.

4.2.2 Pin Numbering Convention

The iLBX bus specification uses the same connector pin numbering convention as the Multibus specification. The numbering convention specifies locating pin 1 on the component side of the board so that it is at the left end of the connector when you face the connector with the component side of the board up. Pin 2 is located immediately under pin one on the solder side of the board. The pins are then numbered in ascending order from left to right with the odd numbered pins located on the component side of the board and the even numbered pins located on the solder side of the board. Figure 9 illustrates the iLBX bus P2 connector pin numbering convention.

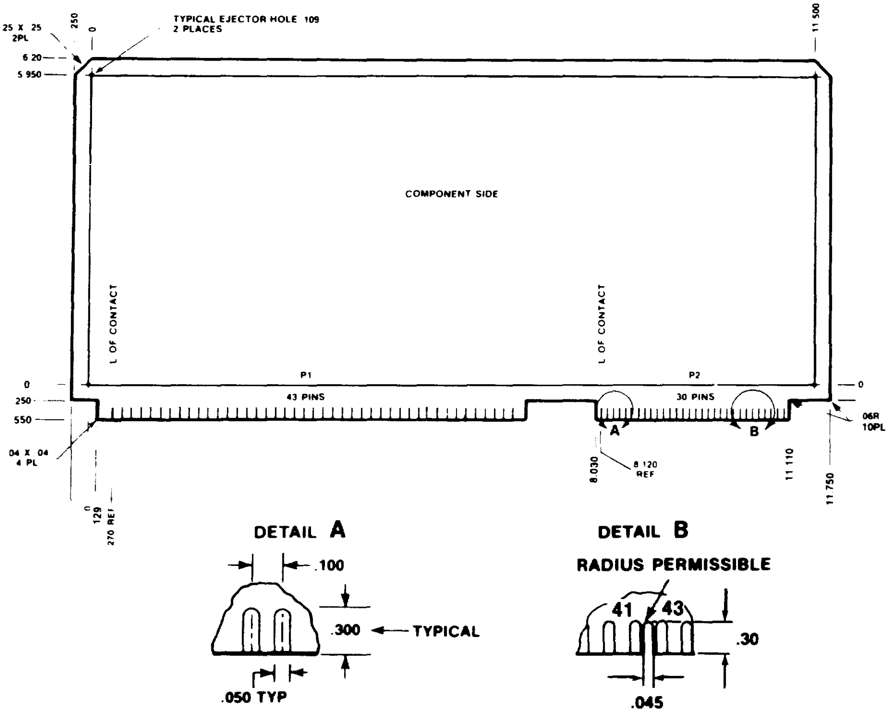


Figure 8 iLBX™ Bus Standard Printed Circuit Board Outline

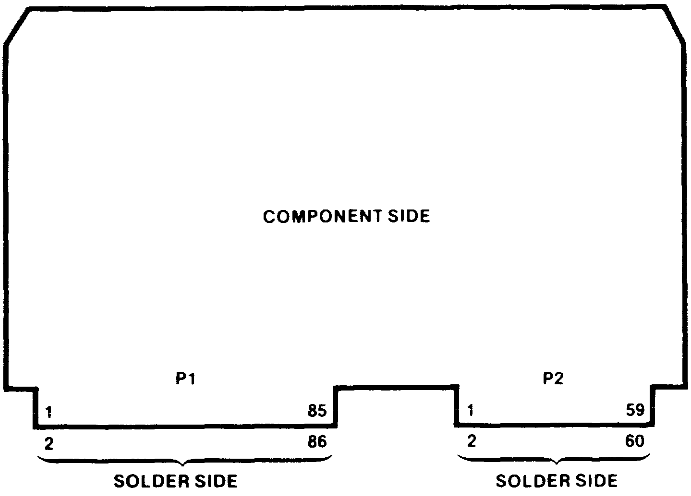


Figure 9 iLBX™ Bus Connector And Pin Numbering Conventions

4.2.3 Component Layout Considerations

To maintain the electrical signal quality of the iLBX bus signals, care must be taken when the iLBX bus drivers, receivers, and transceivers are positioned, relative to the P2 connector, on a board implementing the iLBX bus. Any device, driver or receiver, directly connected to an iLBX bus signal line should be located close to the P2 connector. The printed circuit board trace connecting a driver or receiver pin to the corresponding P2 connector pin should not exceed 5 cm (2 in) in length. On a Primary Master, include any additional trace required to connect the terminating resistor to the signal line trace when calculating the maximum trace lengths.

The iLBX bus interface components (integrated circuits containing drivers, receiver, or transceivers directly connected to an iLBX bus signal line) must have adequate connection to signal ground. On boards with a ground interlayer, the interlayer should be solid under the iLBX bus interface components with the ground connections made directly to the interlayer. On boards without a ground interlayer, the ground trace to the iLBX bus interface components should be 1.27 mm (0.05 in) wide (assuming 1 oz copper plate) and be directly connected to the main signal ground for the board.

4.2.4 iLBX™ Bus Pin Assignments

The iLBXbus configuration uses the form-factor of the standard 60-pin Multibus P2 connector and occupies 56 of the P2 connector pins.

Table 8 lists the iLBX bus pin assignments for the 60-pin P2 edge connector. The four Multibus address extension lines (pins 55 through 58 on the Multibus P2 connector) retain the standard Multibus interface functions. Information on designing a P2 layout with iLBX bus or Multibus P2 compatibility (limited to two front panel lines) is located in Section 5.

4.2.5 Connector Key Slot

The P2 keyslot for the iLBX bus is located between P2 pins 41 and 43 for 8- and 16-bit compatible boards. Figure 8 includes the routing specifications for the iLBX bus P2 connector key slot. All iLBX bus compatible boards must be key slotted to assure the board is not plugged into a P2 connector with Multibus P2 connector compatible battery back-up signals.

4.3 BATTERY BACK-UP AND FRONT PANEL INTERFACE

Implementation of the iLBX bus on a printed circuit board supercedes the full interface of the Multibus specification of the battery back-up and front panel interface signals to the P2 connector. Information on designing a P2 layout with iLBX bus or Multibus P2 compatibility (limited to two front panel lines) is located in Section 5. The iLBX bus specification introduces the use of an auxiliary right-angle connector (JX) mounted on top of the board to connect the battery back-up and front panel signals to the board. There are 14 signals assigned to the JX connector divided into two subset groups: the battery back-up signals (pins 1 through 6); and the front panel interface signals (pins 7 through 14). A full implementation of the JX connector can be made of either of the two subsets if board space is limited or when the additional signals are not used on the board. For example, the Slave devices typically do not require use of the front panel interface signals.

Table 8 iLBX™ Bus Pin Assignments, P2 Edge Connector

Component Side			Solder Side		
Pin	**Mnemonic**	**Signal Name**	**Pin**	**Mnemonic**	**Signal Name**
1	DBO	DATA LINE 0	2	DB1	DATA LINE 1
3	DB2	DATA LINE 2	4	DB3	DATA LINE 3
5	DB4	DATA LINE 4	6	DB5	DATA LINE 5
7	DB6	DATA LINE 6	8	DB7	DATA LINE 7
9	GND	GROUND	10	DB8	DATA LINE 8
11	DB9	DATA LINE 9	12	DB10	DATA LINE 10
13	DB11	DATA LINE 11	14	DB12	DATA LINE 12
15	DB13	DATA LINE 13	16	DB14	DATA LINE 14
17	DB15	DATA LINE 15	18	GND	GROUND
19	ABO	ADDRESS LINE 0	20	AB1	ADDRESS LINE 1
21	AB2	ADDRESS LINE 2	22	AB3	ADDRESS LINE 3
23	AB4	ADDRESS LINE 4	24	AB5	ADDRESS LINE 5
25	AB6	ADDRESS LINE 6	26	AB7	ADDRESS LINE 7
27	GND	GROUND	28	AB8	ADDRESS LINE 8
29	AB9	ADDRESS LINE 9	30	AB10	ADDRESS LINE 10
31	AB11	ADDRESS LINE 11	32	AB12	ADDRESS LINE 12
33	AB13	ADDRESS LINE 13	34	AB14	ADDRESS LINE 14
35	AB15	ADDRESS LINE 15	36	GND	GROUND
37	AB16	ADDRESS LINE 16	38	AB17	ADDRESS LINE 17
39	AB18	ADDRESS LINE 18	40	AB19	ADDRESS LINE 19
41	AB20	ADDRESS LINE 20	42	AB21	ADDRESS LINE 21
43	AB22	ADDRESS LINE 22	44	AB23	ADDRESS LINE 23
45	GND	GROUND	46	ACK*	SLAVE ACKNOWLEDGE
47	BHEN	BYTE HIGH ENABLE	48	R/$\overline{W}$	READ NOT WRITE
49	ASTB*	ADDRESS STROBE	50	DSTB*	DATA STROBE
51	SMRQ*	SECONDARY MASTER REQUEST	52	SMACK*	SECONDARY MASTER ACKNOWLEDGE
53	LOCK*	ACCESS LOCK	54	GND	GROUND
55	ADR22*	MULTIBUS® ADDRESS EXTENSION LINE 22	56	ADR23*	MULTIBUS® ADDRESS EXTENSION LINE 23
57	ADR20*	MULTIBUS® ADDRESS EXTENSION LINE 20	58	ADR21*	MULTIBUS® ADDRESS EXTENSION LINE 21
59	RES	RESERVED	60	TPAR*	TRANSFER PARITY

Figure 10 shows the iLBX bus board outline with the JX location area shaded. The JX connector must be located within the specified area to keep the mating cable lengths to a minimum. Figure 11 illustrates the height, pin spacing, and pin location requirements for the JX connector and Table 9 lists the pin assignments for the JX connector. The signal lines assigned to the JX connector are standard Multibus interface lines, and the descriptions and timing specifications are located in the Multibus Specification.

Table 9 Auxiliary (JX) Connector Pin Assignments

LOWER ROW			UPPER ROW		
PIN	MNEMONIC	SIGNAL NAME	PIN	MNEMONIC	SIGNAL NAME
1	+5	+5 VDC	2	GND	Ground
3	+5	+5 VDC	4	GND	Ground
5	MPRO*	Memory Protect	6	NVE*	Non-Volatile Enable
7	ALE	Address Latch Enable	8	GND	Ground
9	ARES*	Reset Switch	10	GND	Ground
11	INT	Front Panel INT	12	RES	Reserved
13	PFSN*	Power Fail Sense	14	PFIN*	Power Fail Interrupt

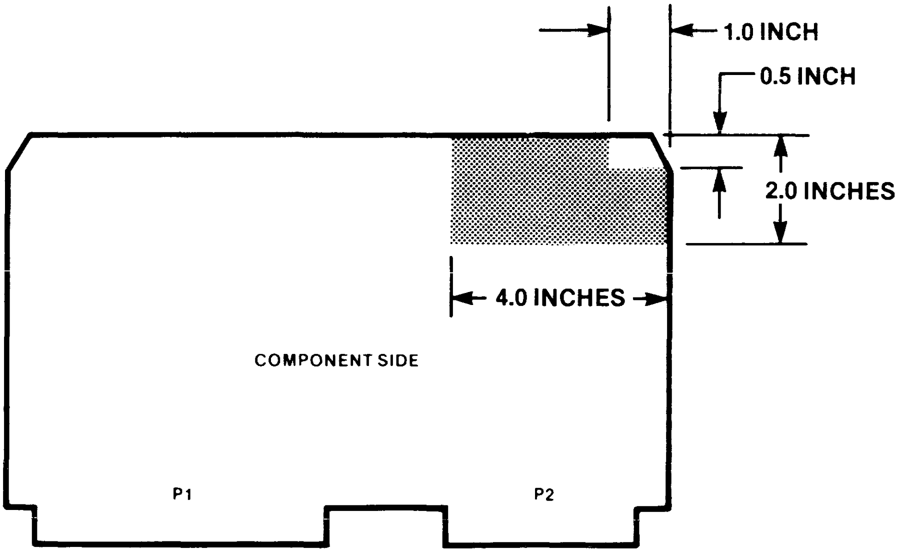
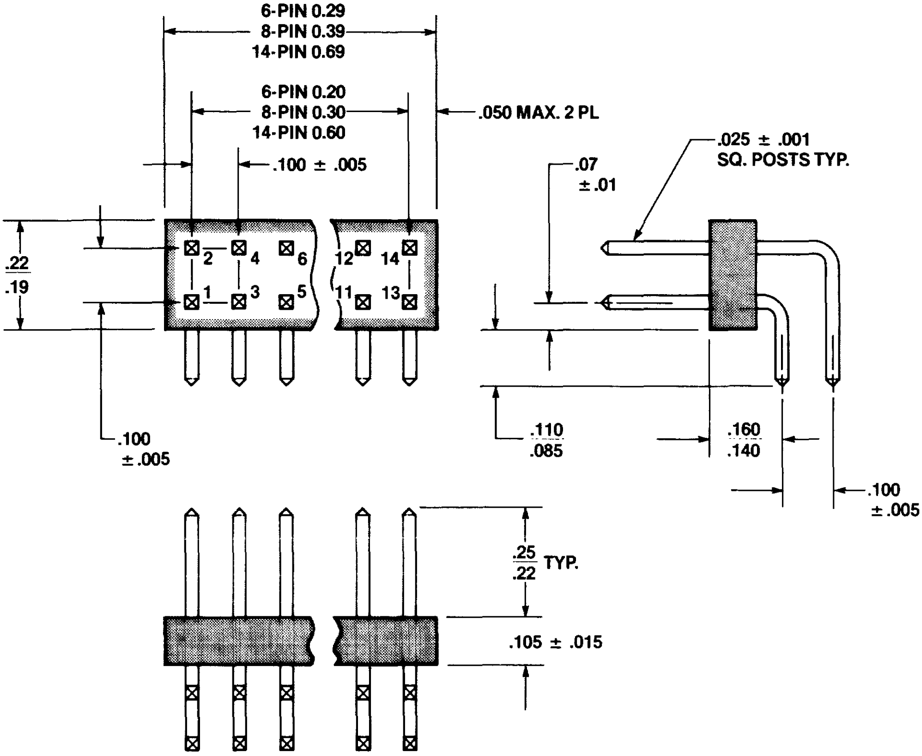


Figure 10 Auxiliary (JX) Connector Location Area



MATERIALS AND FINISHES

INSULATOR: GLASS FILLED POLYESTER OR EQUIVALENT.

CONTACT: PHOSPHOR BRONZE.

FINISH: .000020 IN. MIN. GOLD OVER .000050 IN. MIN. NICKEL PLATE.

Figure 11 Auxiliary Connector Mechanical Specifications

4.4 iLBX™ BUS CONNECTORS AND CABLING

The iLBX bus does not use a rigid backplane to interconnect the iLBX bus compatible boards but rather an interconnect cable assembly. This method of interconnection allows the system designer more freedom because the iLBX bus compatible boards are not required to be in adjacent board slots. The specification further simplifies system implementation by using ribbon cable and mass terminated connectors to make up the required interconnect cables. Table 10 lists suppliers that produce ribbon cable and connectors compatible with the iLBX bus. The table also lists the required installation hardware.

Table 10 Cable And Receptacle Vendors

iLBX BUS COMPATIBLE CABLE		
VENDOR	**VENDOR PART NUMBER**	**CONDUCTORS**
T & B Ansley	171-60	60
T & B Ansley	173-60	60
3M	3365/60	60
3M	3306/60	60
Berg	76164-060	60
Belden	9L28060	60
Spectrastrip	455-240-60	60
	iLBX BUS COMPATIBLE RECEPTACLES	
VENDOR	**VENDOR PART NUMBER**	**PINS**
KELAM	RF30-2803-5	60
KELAM*	110-10-001-37 (polarizing key)	
T & B Ansley**	A3020 (609-6026 modified)	60

Notes: * = Mounting hardware for KELAM consists of 2 sets of 5/8 inch 4-40 Philips round head screw, 1/8 inch 4-40 spacer, 4-40 internal tooth lock washer, 4-40 hex nut. ** = Mounting Hardware for T & B consists of 2 sets of 0.5 inch 4-40 Philips, Fillister head screw, 4-40 lock washer, and 4-40 hex nut.

4.4.1 iLBX™ Bus Cable

The iLBX bus interconnect cable uses 28 AWG, 60 conductor, flat ribbon cable for interconnecting 8- and 16-bit compatible boards. The maximum length for the interconnect cable is 10 centimeters (approximately 4 inches) for the interconnect cable. For best system performance and good electrical design practice, the interconnect cable should be kept as short as possible. The following are the general electrical and insulation specifications for iLBX bus compatible cable:

ELECTRICAL PROPERTIES

Impedance	100 ohms ±10%
Propagation velocity (maximum)	2.0 ns/ft
Capacitance (maximum)	15 pf/ft

INSULATION REQUIREMENTS

Voltage rating (minimum)	100 Vdc
Insulation resistance (minimum)	1×10^{10} ohms

4.4.2 iLBX™ Bus Connectors

The iLBX bus requires use of 60 pin, insulation displacement type female receptacles to attach the interconnecting cable to 8- and 16-bit compatible boards at the P2 edge connector. The female receptacle must have a key block compatible with the key slot specifications for the iLBX bus P2 connector.

4.4.3 iLBX™ Bus Cable Assembly

An iLBX bus cable assembly can have from two to five female receptacle connectors mass terminated to the flat ribbon cable. The spacing between the female receptacles assembled to the cable varies with the Multibus backplane used and any intervening, non-iLBX bus compatible, boards. Refer to the applicable chassis hardware reference manual for information on board spacing. Figure 12 illustrates an iLBX bus interface cable assembly. Figure 13 and 14 illustrate mounting of the insulation displacement type receptacle to the Multibus backplane, for the KELAM and T & B Ansley connectors.

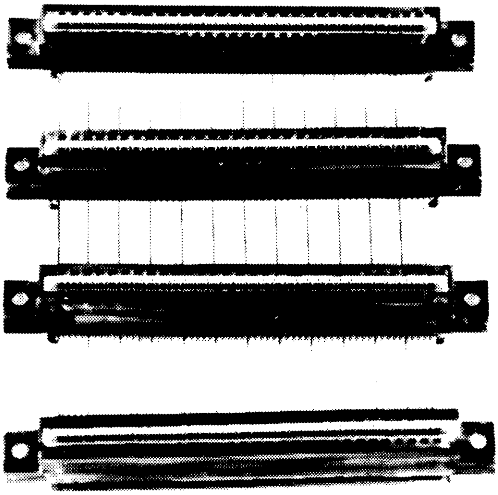


Figure 12 Typical iLBX™ Bus Interface Cable Assembly

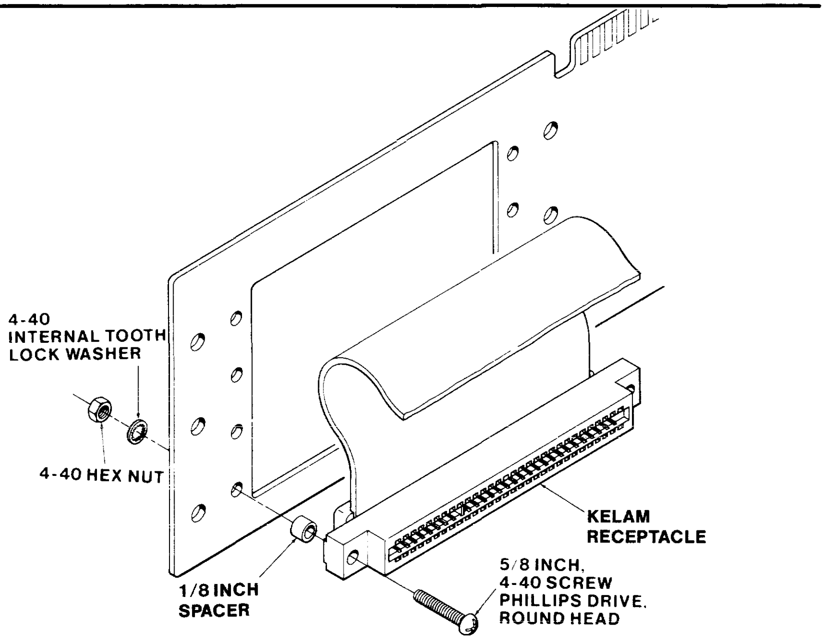


Figure 13 KELAM Connector Installation

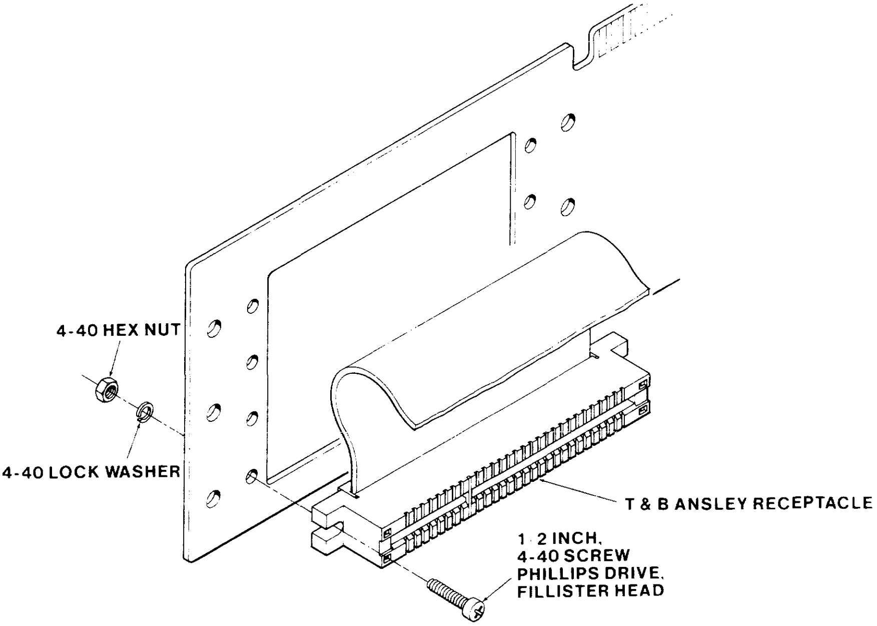


Figure 14 T & B Ansley Connector Installation

CHAPTER 5 DESIGN GUIDELINES AND SYSTEM APPLICATIONS

5.1 INTRODUCTION

This section provides examples of the typical circuits required to implement the interface between a given device and the iLBX bus. The circuit examples used in this chapter illustrate the main interface circuits. The examples do not show most of the on-board circuits that generate the timing of the control signals. Two full interface examples, a 16-bit Primary Master and a 16-bit Slave device, illustrate the general interfacing requirements for the iLBX bus. Additional partial examples show the special considerations for a Secondary Master, and an 8-bit device (master or slave). Wherever possible, the examples represent proven interface circuits.

The examples do not attempt to show fully optimized circuits. In most of the examples, the iLBX bus interface derives signals directly from the microprocessor or support component signals. For example, the Primary Master circuit example shows the Secondary Master Request signal applied, through an inverting buffer, to the HOLD signal input to the microprocessor. In turn, the microprocessor HOLDA signal, through an inverting buffer, drives the Secondary Master Acknowledge line. Thus in this example, when the Primary Master transfers control to the Secondary Master, the microprocessor waits in the standby mode until the Secondary Master returns control to the Primary Master.

III

5.2 PRIMARY MASTER DESIGN EXAMPLE - 16-BIT

The 16-bit Primary Master interface design example is for a full rather than limited Primary Master. Because the full Primary Master must be able to transfer control to a Secondary Master, the design example uses tri-state drivers for the address lines. The interface example also shows the jumpers required to allow the Primary Master to access the front panel control signals over the standard Multibus interface P2 connector. (The option is mutually exclusive. The Primary Master P2 connector must be interfaced to the iLBX bus or the Multibus interface. It cannot interface to both at the same time.) Figure 15 illustrates the 16-bit Primary Master interface example.

5.2.1 ADDRESS DECODE

The 16-bit circuit example uses a programmed array logic (PAL) component (U7) to decode the memory address from the processor and direct the address to the appropriate memory array. The PAL in the example shows four outputs: Multibus memory access (MBACESS*); on-board RAM access (OBRAM*); on-board ROM access (OBROM*); and iLBX bus access (LBXEN*). The PAL inputs shown are the address lines AE-A17, the memory I/O signal (M/$\overline{\text{IO}}$), and three select lines with programming jumpers. The memory I/O input limits PAL address decoding to memory accesses only. The address line inputs provide memory address decoding in 16k byte increments for the on-board RAM and Multibus memory access. The programmable select inputs specify the iLBX bus memory size in 512k byte increments. Table 11 lists the jumper configuration used to set the upper limit for the iLBX bus memory. The select jumpers in the circuit example relate directly to the address lines: E5-E6 to A13, E3-E4 to A14, and E1-E2 to A15. Installation of a jumper sets the select value to zero. The first two configurations in the Table 11 are reserved to avoid overlap with the standard Multibus memory address range.

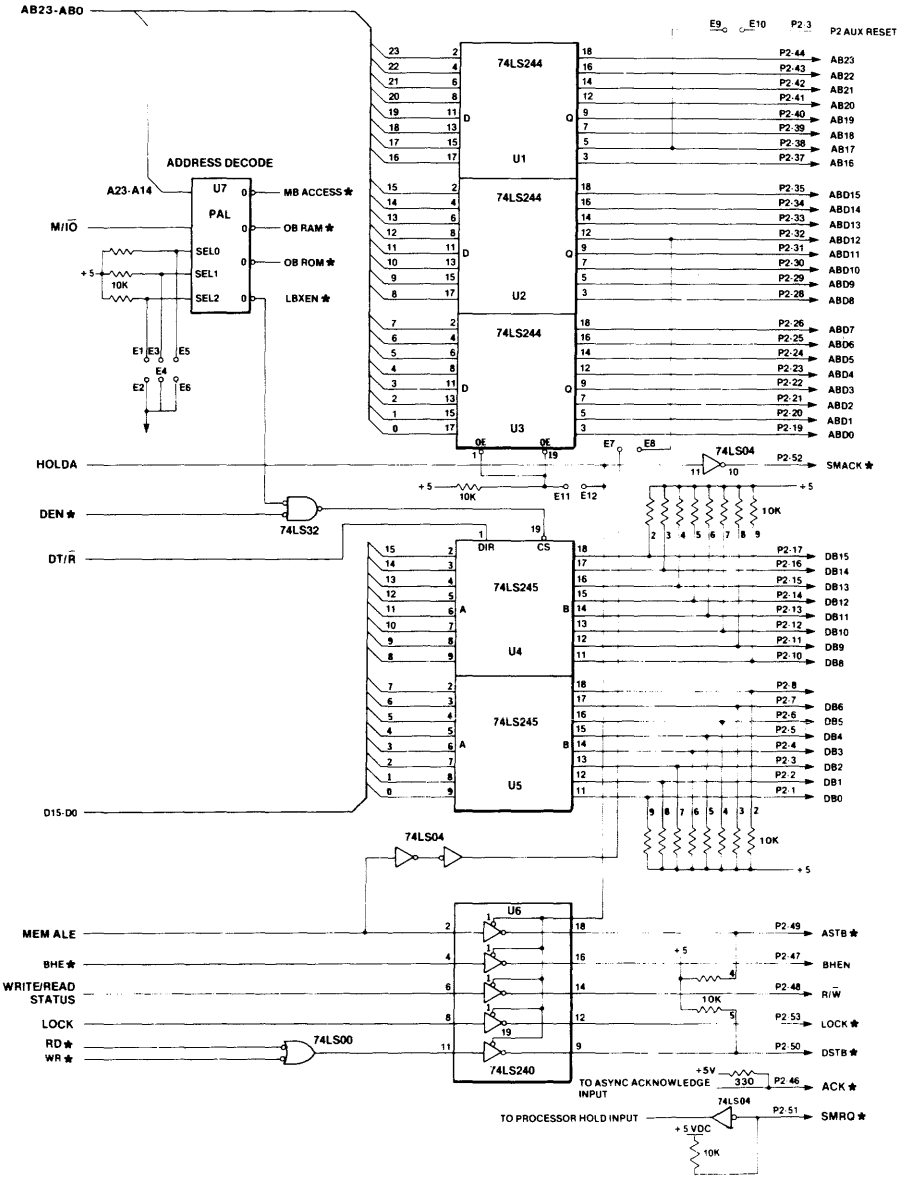


Figure 15 Interface Circuit Example - 16-Bit Primary Master

Table 11 iLBX™ Bus Address Range Select Jumpers

iLBX™ BUS UPPER LIMIT	SELECT JUMPERS		
	E1-E2	E3-E4	E5-E6
RESERVED	IN	IN	IN
RESERVED	IN	IN	OUT
17FFFFH	IN	OUT	IN
1FFFFFH	IN	OUT	OUT
27FFFFH	OUT	IN	IN
2FFFFFH	OUT	IN	OUT
37FFFFH	OUT	OUT	IN
DISABLE iLBX™ ACCESS	OUT	OUT	OUT

For example, assume the following system memory configuration: 16k bytes of on-board RAM (base address at 0H); 32k bytes of on-board ROM (base address at FF8000H); and jumper E1-E3 only installed corresponding to 1M byte of iLBX bus memory (base address at 100000H). The PAL drives the OBRAM* output active for memory addresses within the address range of 0H to 03FFFH. The PAL drives the OBROM* output active for memory addresses within the address range of FF8000H to FFFFFFH. The PAL drives the MBACCESS* output active for memory addresses within the address range of 04000H to 0FFFFFH. The PAL drives the LBXEN* output active for memory addresses within the address range of 100000H to 1FFFFFH.

5.2.2 Data Drivers

The 16-bit circuit example uses two 74LS245 octal tri-state transceivers (U4 and U5) to drive and receive the iLBX bus data lines DB15-DB0. The ANDed combination of the iLBX Bus Enable (LBXEN*) signal active from the address decode and Data Enable (DEN*) signal active from the processor circuit drives the chip select enable (CS) to the data transceivers. The processor's Data Transmit Not Receive (DT/$\overline{\text{R}}$)signal directly drives the transceiver direction control input (DIR).

5.2.3 Address Drivers

The 16-bit circuit example uses three 74LS244 octal tri-state buffers (U1, U2, and U3) to drive the iLBX bus address lines. Typically, the on-board processor keeps the address lines enabled to the iLBX bus Slave devices unless the iLBX bus interface is specifically disabled.

5.2.4 Control and Command Drivers

The 16-bit circuit example uses an 74LS240 octal tri-state inverting buffer (U6) to drive the iLBX bus control and command lines. The interface circuit example uses processor signal lines or direct derivatives from processor signal lines as inputs to the tri-state buffer. With this design, the processor uses the iLBX bus as though it were an extension of the processor's local bus and drives the iLBX bus control and command signals each time the processor accesses memory. Whenever the processor directs the memory access to non-iLBX bus memory, none of the iLBX Slave devices respond to the control signals and the processor aborts the iLBX bus portion of the operation when the non-iLBX bus memory responds.

In the example, the processor's Address Latch Enable signal (or a direct derivative) drives the iLBX bus Address Strobe line (ASTB*). The processor's Byte High Enable (BHE*) and Lock (LOCK*) signal directly drive the corresponding iLBX bus lines (BHEN and LOCK*). The ORed combination of the processor's Read strobe (RD*) and Write strobe (WR*) signals drives the iLBX bus Data Strobe line (DSTB*). The iLBX bus Read-Not-Write line does not have a directly equivalent processor signal and must be derived from the processor's status signals. To implement the circuit example with an 8086 family processor, derive the Read-Not-Write drive signal from the S1* status output.

Typically, the on-board processor keeps the control and command lines enabled to the iLBX bus Slave devices unless the iLBX bus interface is specifically disabled.

5.2.5 Interface Disabling

The circuit example depicts two conditions when the address, control, and command tri-state drivers would be placed in the high impedance state. The first condition occurs when the on-board processor drives the Secondary Master Acknowledge line (processor HOLDA or equivalent signal, iLBX bus SMACK*) Low in response to the Secondary Master's driving the Secondary Master Request line (SMRQ*) Low. The circuit example shows the Secondary Master Request signal connected through an inverting buffer to the processor's Hold request input. The second condition occurs as a result of configuring the board for non-iLBX bus operation. The configuration change requires three jumper changes. The removal of the Jumper E11-E12 disconnects the HOLDA signal from pins 1 and 19 (output enable) on the address line drivers, U1-U3. Pins 1 and 19 are pulled High and the buffers placed in the high-impedance state. The installation of Jumper E7-E8 connects the Address Latch Enable signal to P2 connector pin 32 and the installation of Jumper E9-E10 connects the Auxiliary Reset signal to P2 connector pin 38.

Neither disabling option is required under the following conditions.

- Limited Primary Masters that do not share the iLBX bus with a Secondary Master can use standard TTL drivers with drive characteristics comparable to the specified tri-state buffers for the line drivers. Limited Primary Masters are not required to monitor the Secondary Master Request line.

- An iLBX bus compatible DMA controller would not need the partial Multibus interface P2 compatibility.

5.2.6 Address Strobe Timing

Implementing the iLBX bus interface circuit example shown in Figure 16 on a board with an 8086/8088 processor requires the use of an additional timing support circuit. The 8086/8088 processor address line set-up time relative to its driving the Address Latch Enable signal active is shorter than the required address set-up for the iLBX bus. Thus, the processor's Address Latch Enable signal cannot directly drive the iLBX bus Address Strobe line. Figure 17 illustrates a T-state generator initialized by the processor's Address Latch Enable signal. The example uses the output of the T-state generator to develop a delayed Address Strobe drive signal from the Address Latch Enable signal.

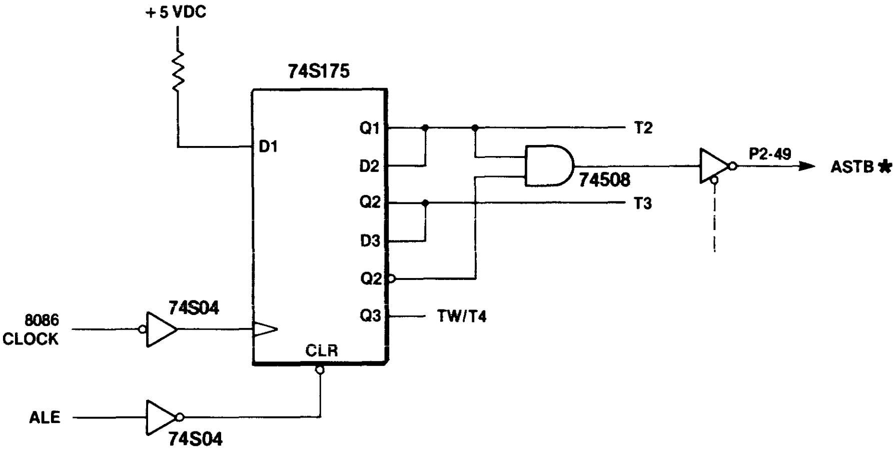


Figure 16 T-State Generator Circuit

Immediately following the Address Latch Enable signal going active, the T-state generator outputs a sequence of four valid signals directly related to the processor T-states. During the processor T1 time, the active Address Latch Enable (inverted) from the processor resets the 74S175 (D-type flip-flops) Q outputs Low. The processor deactivates the Address Latch Enable signal before the end of the T1 time. At the next falling edge of the processor clock, the D1 flip-flop sets and the Q1 output goes High signaling the start of the processor T2 time.

Because the T-state generator has the Q1 output connected to the D2 input, at the next fall of the processor clock, the Q2 output goes High signaling the start of the processor T3 time. Additional T-state signals can be generated by attaching the Q2 output to the D3 input and the Q3 output to the D4 input. Once the T-state generator completes the sequence, all outputs remain High until the next time the processor activates the Address Latch Enable signal.

The circuit example produces the iLBX bus Address Strobe drive signal by ANDing the Q1 output with the inverted Q2 output. At the start of the T2 time when the Q1 output goes High, the AND gate output goes High driving the iLBX bus Address Strobe line Low. At the start of the T3 time when the inverted Q2 output goes Low, the AND gate output goes Low driving the iLBX bus Address Strobe line High. Thus the circuit example generates the Address Strobe signal starting at the processor T2 time with a width of one processor clock cycle.

5.3 PRIMARY MASTER DESIGN EXAMPLE - 8-BIT

An 8-bit device is a master or a slave designed to perform all data transfers as 8-bit bytes. An iLBX compatible device capable of making 16-bit data transfers over the iLBX bus is classed as a 16-bit device, regardless of its internal architecture. When designing an 8-bit Primary Master, the level of 16-bit compatibility required must be determined. The data transceiver portion of the interface must accommodate the desired 8-bit data transfer format.

Designing an 8-bit Primary Master that operates with 8-bit Slave devices only, an iLBX bus interface circuit similar to the 16-bit Primary Master interface circuit can be used. Because the 8-bit interface does not use the high-order eight data lines, the 74LS245 tri-state transceiver, U4, can be eliminated and the connection from the processor Byte High Enable (BHE*) signal to the control and command driver can be eliminated. The remainder of the interface circuit example can be implemented as shown.

Designing an 8-bit Primary Master that operates with both 8-bit or 16-bit Slave devices requires a more extensive modification to the data transceiver circuit. Figure 17 shows a data transceiver circuit example that allows the 8-bit Primary Master to operate with 16-bit slave devices. An added feature of the circuit example, the jumper option for the A0 address bit, allows the interface to work with either 8-bit or 16-bit Slave devices.

When configured for compatibility with 16-bit Slave devices (E1-E2 connected), the A0 address bit from the processor controls transceiver selection and the iLBX bus Byte High Enable line. When the processor addresses an even numbered byte (A0 Low), the byte is placed on the low-order data lines DB7-DB0.

When the processor addresses an odd numbered byte (A0 High), the byte is placed on the high-order data lines DB15-DB8. In this implementation, the unused data transceiver holds the corresponding data lines in the high-impedance state.

When configured for compatibility with 8-bit Slave devices (E2-E3 connected), the A0 address bit line is opened and the selection circuit input connected to ground, forcing all data transfers to take place over the low-order data lines DB7-DB0.

5.4 SECONDARY MASTER DESIGN EXAMPLE

Because most of the iLBX bus interface circuits are the same as those used for the Primary Master, the Secondary Master interface design example concentrates on the following three circuits used specifically on the Secondary Master:

- the Secondary Master Request and Secondary Master Acknowledge circuit
- the adjustment circuit for the Secondary Master acknowledge acceptance time
- the Secondary Master to limited Primary Master conversion circuit requirements

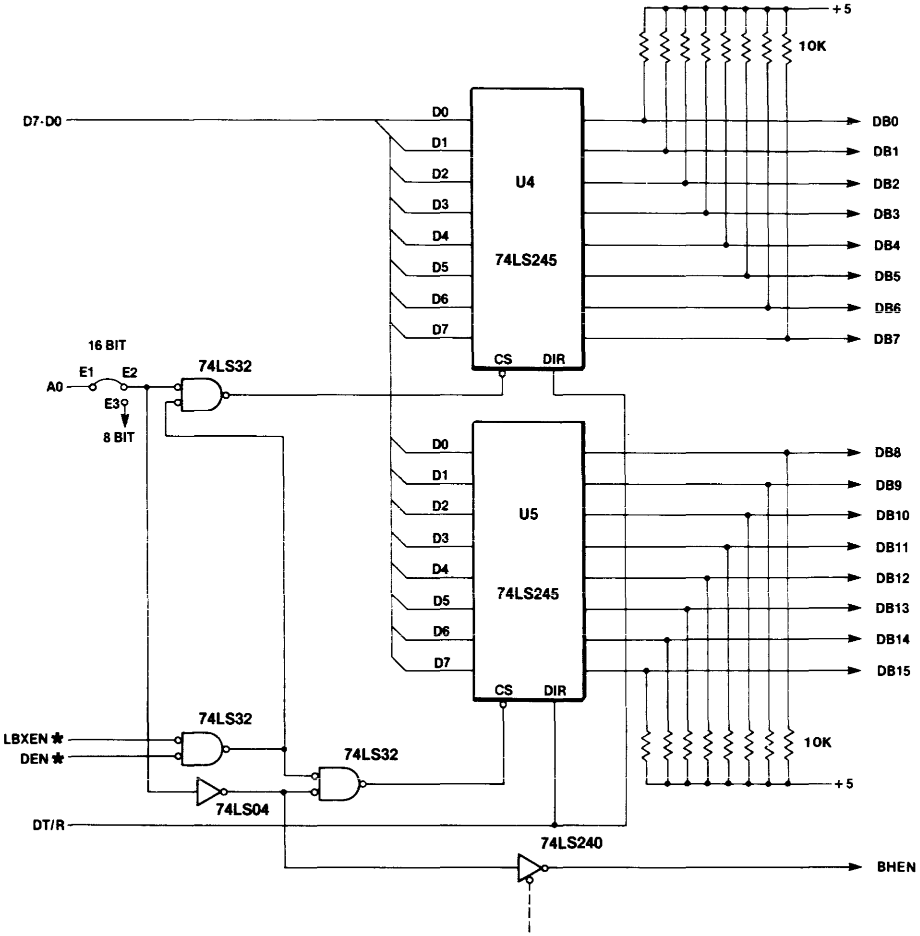


Figure 17 The 8-Bit Data Transceiver Circuit

Figure 18 illustrates the Secondary Master interface design example. Figure 18 does not repeat in detail the circuits shown in the Primary Master design example. These circuits are generalized in block diagram form to emphasize the circuits being described.

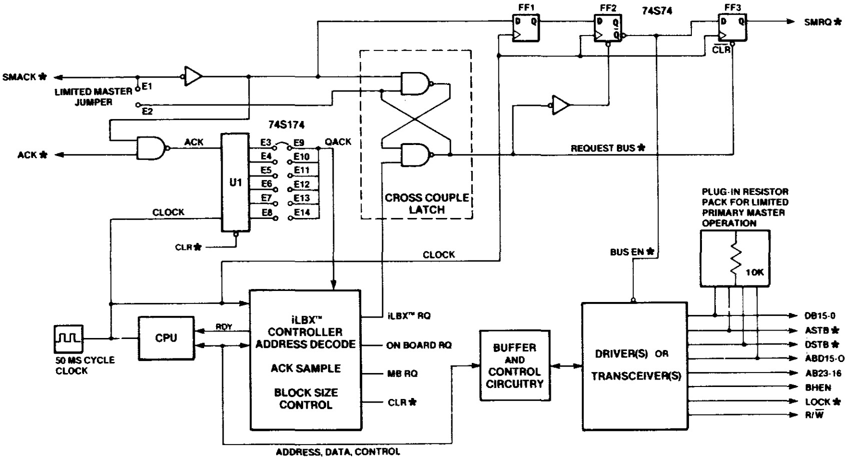


Figure 18 Interface Circuit Example - Secondary Master

5.4.1 Bus Request Circuit

The Secondary Master circuit example uses a cross coupling latch, along with timing derived from the on-board clock, to meet the bus request and release requirements. In the example, the cross coupling latch is wired-up from a pair of two input NAND gates. Any time the Primary Master has control of the bus, the Secondary Master Acknowledge (SMACK*) line is High. Inverted, the Secondary Master Acknowledge preconditions the cross coupling latch to accept an iLBX bus request from the on-board iLBX controller. With the cross coupling latch preconditioned, a valid iLBX request (LBXRQ) from the iLBX controller removes the reset clamp from the synchronizing flip-flop FF2 and resets FF3, driving the Secondary Master Request (SMRQ*) line Low. When the Primary Master responds by driving SMACK* Low, the synchronizing flip-flops, FF1 and FF2, delays the input for two clock pulses (50 to 100 ns) before the enabling the iLBX bus drivers.

The same circuit times the release of the iLBX bus back to the Primary Master. The Secondary Master initiates the release by driving the internal LBXRQ signal Low. This resets FF2 and immediately disables the iLBX bus drivers. At the next clock pulse, FF3 is reset to drive the Secondary Master request line High. At this time, the cross coupling latch is only partially restored to its preconditioned wait state. An iLBX request by the iLBX controller is blocked by the cross coupling latch and cannot be recognized until the Primary Master drives SMACK* High.

5.4.2 Programmable Acknowledge Delay Circuit

The Secondary Master can optionally provide an Acknowledge delay circuit for matching its acknowledge acceptance overhead time to that of the Primary Master. The Secondary Master example uses a 74175 hex D-type flip-flop wired as a delay line for the Acknowledge delay. Immediately following the Acknowledge (ACK*) going active, the delay line outputs a sequence of six Qualified Acknowledge (QACK) signals each delayed by 50 ns from the preceding signal. By connecting the output for the desired delay to the iLBX controller input, any levels of delay can be programmed for the Secondary Master.

5.4.3 Master Level Conversion

The Secondary Master can provide a configuration option to convert it from a Secondary Master to a limited Primary Master. In the Secondary Master example, two changes are required to make the conversion. The jumper from E1 to E2 connects the iLBX bus request from the cross coupling latch to the SMACK* input. Because the Secondary Master (now limited Primary Master) must provide termination for the bus lines, sockets are provided to accept the applicable plug-in resistor packs.

5.5 SLAVE DESIGN EXAMPLE

Because most of the iLBX bus interface circuits are similar to those used for the Primary Master, the Slave interface design example concentrates on the following circuits used specifically on the Secondary Master:

- the data element decode circuit
- the Acknowledge set up and pre-acknowledge adjustment circuit

Figure 19 illustrates the Slave interface design example. Figure 19 does not repeat in detail the circuits shown in the Primary Master design example. These circuits are generalized in block diagram form to emphasize the circuits being described.

5.5.1 Data Element Circuit

The Slave circuit example uses a pair of three input NAND gates to select the data element being transferred and enable the interface transceivers. The circuit example allows the Slave to accept or transmit a 16-bit word, or either the high-order or low-order byte. In the circuit example, the data line transceivers are disabled (in the high impedance state) except for the Data Strobe time when the Slave is selected.

5.5.2 Programmable Acknowledge Delay Circuit

The Slave can provide the choice of timing the driving the Acknowledge signal from Address Strobe or Data Strobe. The interface timing of the master(s) determines if Address Strobe can be used or if Data Strobe must be used. Connecting a jumper from E1 to E2 selects the Address Strobe as the clock to set the start cycle flip-flop, FF2. Connecting the jumper from E2 to E3 selects the Data Strobe as the start cycle clock. The clock and acknowledge timer circuit implements a delay line comparable to the delay line in the Secondary Master circuit example. Immediately following the Start Cycle going active, the delay line outputs a sequence of six signals each delayed by 50 ns from the preceding signal. By connecting the output for the desired delay to the Acknowledge driver, any level of delay can be programmed for the Acknowledge signal.

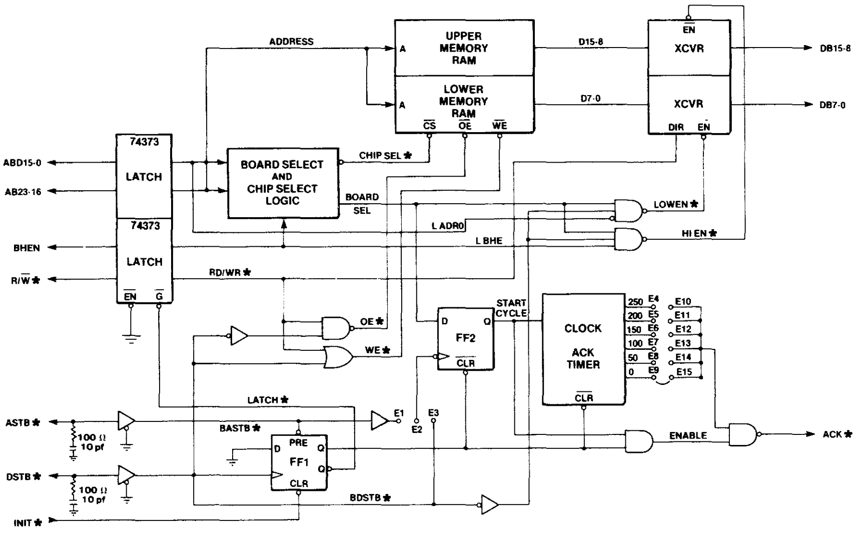


Figure 19 Interface Circuit Example - Slave

5.6 SYSTEM TIMING CONSIDERATIONS

Because the iLBX bus allows close coupling between the operation of the master and slave devices, the level of close coupling (optimization) used must be determined and set at system implementation time. The following descriptions examine the various read and write timing considerations for system implementation.

5.6.1 General Considerations

When the Primary and, if installed, the Secondary Master meet the optimized timing requirements for a master device, then the Slave device(s) can be configured to drive the Acknowledge line Low before the leading (falling) edge of the Data Strobe. Otherwise, the Slave device(s) must wait for the leading edge of the Data Strobe. A Slave device without the Data Strobe pre-acknowledgement capability can operate with the optimized timing from the Primary Master. Also, a mix of optimized and non-optimized Slave devices can operate successfully with an optimized master. If either the Primary Master or the Secondary Master cánnot meet the optimized timing, all the Slave devices must be configured for non-optimized operation. The optimized master continues to use the optimized timing, but it does not realize the full optimum performance.

When a Slave device has variable Acknowledge timing for the read operation, it can pre-acknowledge the read data transfer by driving the Acknowledge line Low before it provides valid data on the data lines. The amount of variability provided can range from 0.0 ns (data valid when the slave drives the Acknowledge line Low) to the maximum access time of the slave's memory resources (driving Acknowledge Line Low immediately after detecting the leading of the Address Strobe). The amount of pre-acknowledgement implemented must be less than the acknowledge acceptance overhead of the fastest master.

5.6.2 Read Timing Examples

The following examples are provided to better illustrate the timing considerations.

Assume an iLBX bus implementation with a Primary Master and a single Slave device. The Slave device memory access time is 150 ns, and the Slave has pre-acknowledge capabilities (variable Acknowledge timing in 10 ns increments). The Primary Master has a 100 ns internal acknowledge acceptance overhead time (from receipt of Acknowledge to the sampling of the data lines). The Figure 20 timing chart presents the non-optimized and optimized timing for a read data transfer from the leading edge of the Address Strobe until the time when the Primary Master samples the data lines (SAMPLE* on the timing chart). In the optimized configuration, there is a 90 ns improvement in the total data transfer time.

The example presented in Figure 20 is very simple and is used to demonstrate the basic optimized read data transfer considerations. The following example is more complex and assumes a Primary Master, a Secondary Master, and two Slave devices. Assume the Primary Master is relatively slow and has an acknowledge acceptance time of 350 ns. The Secondary Master is relatively fast with an acknowledge acceptance time of 100 ns. The Slave devices also have different memory access times. Slave A's access time is 100 ns with variable Acknowledge timing at 10 ns increments. Slave B's access time is 250 ns with variable Acknowledge timing at 25 ns increments.

Because the devices in this example do not have common timing parameters, the various timing considerations must be considered separately and in a specific order. Assume for now that the Secondary Master's acknowledge acceptance overhead time is fixed (not adjustable).

1. First compare the Secondary Master's overhead time with the Primary Master's overhead time. Because the iLBX bus master cannot indicate to the Slave device which master is accessing the Slave device, the amount of pre-acknowledgement implemented must be less than the acknowledge acceptance overhead of the fastest master.Thus, the overhead time of the Secondary Master (100 ns) becomes the standard master overhead time for this iLBX bus implementation.

2. Set Slave A's pre-acknowledge time to the first increment under 100 ns (90 ns).

3. Set Slave B's pre-acknowledge time to the first increment under 100 ns (75 ns).

The Figure 21 timing chart presents the optimized timing for a read data transfer from both Slave devices to both masters. Note that Slave A is close to fully optimized at 90 ns of pre-acknowledgement. However, Slave B is limited to 75 ns of pre-acknowledgement by the Secondary Master and system throughput is thus limited whenever Slave B is accessed.

The two preceding examples of read operation optimization resulted in direct system performance improvements when optimization is used. The next example adds a timing variable to the Secondary Master. This added level of variability means that overall system operation must be considered when and where to implement the read operation optimization.

Continuing to use the same devices, now assume the Secondary Master provides an adjustment to lengthen its acknowledge acceptance overhead time in 25 ns increments. Again, the various timing considerations must be considered in a specific order to arrive at the optimum settings.

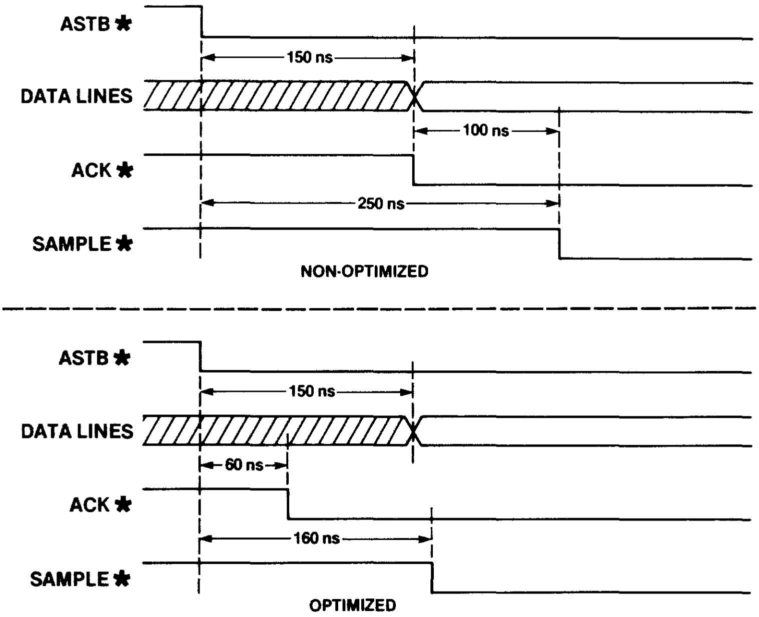


Figure 20. Read Data Optimization Timing - Basic Example

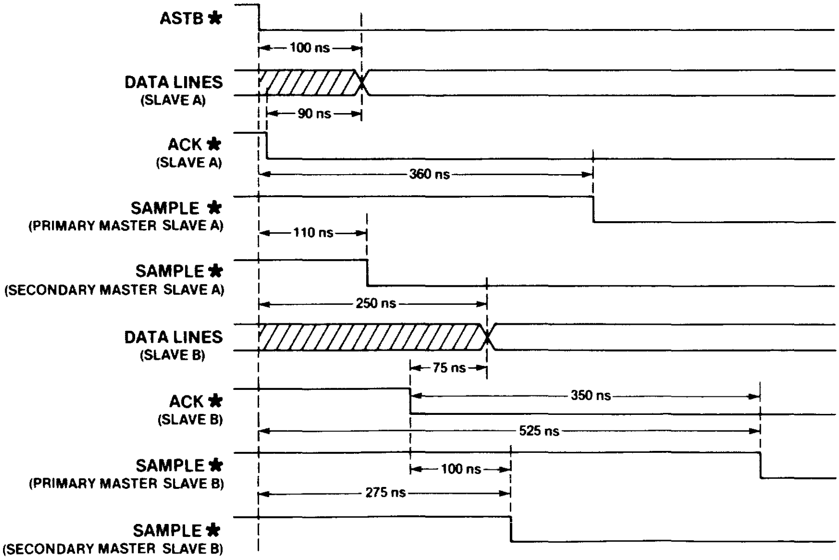


Figure 21 Read Data Optimization Timing - Non-adjustable, Secondary Master Example

1. Compare the Secondary Master's overhead time with the Primary Master's overhead time. The Primary Master has the longer overhead time so the Secondary Master's overhead time can be lengthened, possibly to 350 ns equaling the Primary Master's overhead time.
2. Compare the 350 ns overhead time with the Slave devices maximum access time. Because Slave B has a maximum memory access time of 250 ns that is shorter than the Primary Master's overhead time, the Secondary Master's overhead time should be set to a maximum of 275 ns.
3. Set Slave A for its maximum pre-acknowledge time of 100 ns.
4. Set Slave B for its maximum pre-acknowledge time of 250 ns.

The Figure 22 timing chart presents the optimized timing for a read data transfer from both Slave devices to both masters. In this example, both Slave devices are fully optimized. The coupling between the masters and the slaves is stabilized at 350 ns for the Primary Master and 275 ns for the Secondary Master. A comparison between the timing charts in Figure 21 and Figure 22 show that, taken on an individual basis, the throughput for some combinations remain about the same (Primary Master to Slave A and Secondary Master to Slave B), one combination (Primary Master to Slave B) was improved dramatically, and one combination (Secondary Master to Slave A) was much slower.

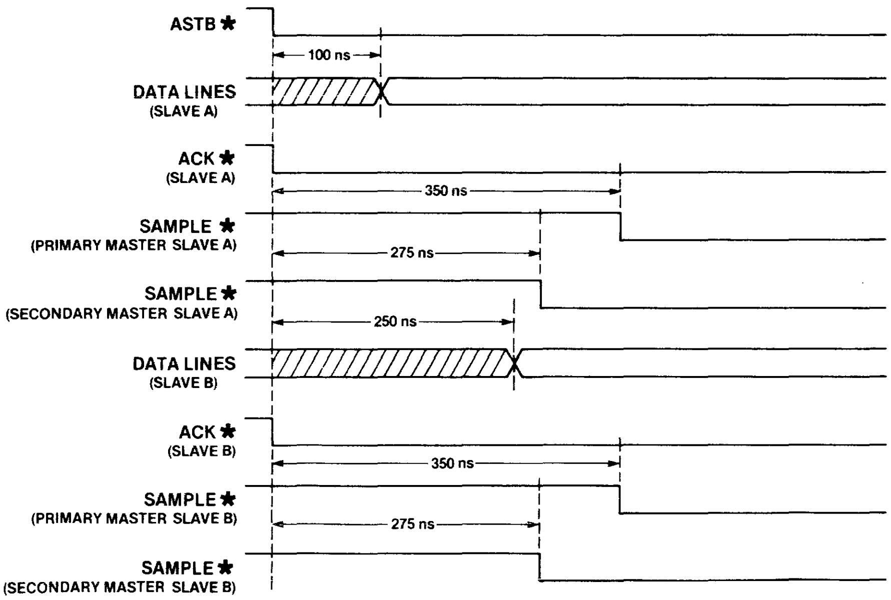


Figure 22 Read Data Optimization Timing, Adjustable Secondary Master Example

Thus, the final consideration becomes a system implementation question. For example, if the Primary Master accesses the memory on Slave A and Slave B equally and the Secondary Master almost exclusively accesses the memory on Slave B, then the fully optimized configuration (Figure 22 timing) would improve overall system throughput. Other combinations of use might dictate using the partially optimized timing represented by Figure 21.

CHAPTER 6
LEVELS OF COMPLIANCE

6.1 INTRODUCTION

This section bounds the variability allowed within the iLBX bus specification. The main purpose in bounding variability is to assure the maximum amount of upward compatibility. In most cases, mixing devices designed to different levels of compliance results in the more complex devices in the system operating at the level of the least complex device.

6.2 DATA PATH

The iLBX bus allows 8- and 16-bit data path devices. The 8- and 16-bit data path devices share the same mechanical implementation of the iLBX bus. When a system with a mixture of data path width devices is implemented, the width of a data transfer is limited to the narrower data path width of the master and slave transferring the data. For example, assume that a 16-bit data path Primary Master is connected to a 16-bit data path slave and an 8-bit data path slave. The master would be limited to 8- or 16-bit transfers with the 16-bit data path slave and 8-bit transfers only with the 8-bit data path slave.

6.3 ADDRESS PATH

There are 24 address lines (AB23 - AB0) defined for the iLBX bus. All devices, masters and slaves, must implement all 24 lines.

6.4 SIGNAL LINE CONNECTIONS

In general, an iLBX bus compatible device needs to connect to most of the iLBX bus lines. The iLBX bus includes a single optional line, Transfer Parity. The following is a list of additional exceptions to full iLBX bus connection:

- 8-bit data path devices need not connect to the Byte High Enable line.
- 8-bit data path devices need not connect to the high-order byte data lines (DB15 - DB8).
- Slave devices should not connect to the Secondary Master Request and Secondary Master Acknowledge lines.
- Single port Slave devices need not connect to the Lock line.
- ROM memory Slave devices need not connect to the Read Not Write line.
- Limited Primary Masters need not connect to the Secondary Master Request or the Secondary Master Acknowledge lines.

6.5 DOCUMENTATION

- range of pre-acknowledge adjustment and adjustment increments for a Slave device
- minimum data strobe active to the pre-acknowledge time, if greater than zero, for a slave device.
- maximum leading edge of Address Strobe to leading edge of Data Strobe time for a . master
- minimum Data Strobe to read data sample time for a master
- maximum Data Strobe to write data valid on the data lines for a master device
- maximum Data Strobe to read data valid on the data lines for a Slave device
- range of acknowledge acceptance delay adjustment and adjustment increments for a Secondary Master

6.6 COMPLIANCE MARKING

The compliance level of an iLBX bus compatible board should be clearly stated in the printed specifications. Omission of a capability denotes that the device does not support the capability. For example, a Primary Master that can communicate with 8- or 16-bit data path devices would be marked as follows: PM D8 D16.

A 16-bit data path Slave with parity support would be marked as follows: SL D16 P.

A 16-bit data path Secondary Master that can also operate as a Primary Master would be marked as follows: PM SM D16.

APPENDIX A iLBX™ SIGNAL DESCRIPTIONS SUMMARY

A.1 INTRODUCTION

This appendix provides a summary description of the iLBX bus signal lines. Refer to Section 2 for the full descriptions of the signal lines and the implementation parameters. Refer to Section 3 for the electrical specifications for the signal lines, and to Section 4 for the mechanical specifications. Table 12 lists the iLBX bus pin assignments for the 60-pin P2 edge connector.

Table 12 iLBX™ Bus Pin Assignments, P2 Edge Connector

Component Side			Solder Side		
Pin	**Mnemonic**	**Signal Name**	**Pin**	**Mnemonic**	**Signal Name**
1	DBO	DATA LINE 0	2	DB1	DATA LINE 1
3	DB2	DATA LINE 2	4	DB3	DATA LINE 3
5	DB4	DATA LINE 4	6	DB5	DATA LINE 5
7	DB6	DATA LINE 6	8	DB7	DATA LINE 7
9	GND	GROUND	10	DB8	DATA LINE 8
11	DB9	DATA LINE 9	12	DB10	DATA LINE 10
13	DB11	DATA LINE 11	14	DB12	DATA LINE 12
15	DB13	DATA LINE 13	16	DB14	DATA LINE 14
17	DB15	DATA LINE 15	18	GND	GROUND
19	ABO	ADDRESS LINE 0	20	AB1	ADDRESS LINE 1
21	AB2	ADDRESS LINE 2	22	AB3	ADDRESS LINE 3
23	AB4	ADDRESS LINE 4	24	AB5	ADDRESS LINE 5
25	AB6	ADDRESS LINE 6	26	AB7	ADDRESS LINE 7
27	GND	GROUND	28	AB8	ADDRESS LINE 8
29	AB9	ADDRESS LINE 9	30	AB10	ADDRESS LINE 10
31	AB11	ADDRESS LINE 11	32	AB12	ADDRESS LINE 12
33	AB13	ADDRESS LINE 13	34	AB14	ADDRESS LINE 14
35	AB15	ADDRESS LINE 15	36	GND	GROUND
37	AB16	ADDRESS LINE 16	38	AB17	ADDRESS LINE 17
39	AB18	ADDRESS LINE 18	40	AB19	ADDRESS LINE 19
41	AB20	ADDRESS LINE 20	42	AB21	ADDRESS LINE 21
43	AB22	ADDRESS LINE 22	44	AB23	ADDRESS LINE 23
45	GND	GROUND	46	ACK*	SLAVE ACKNOWLEDGE
47	BHEN	BYTE HIGH ENABLE	48	R/$\overline{W}$	READ NOT WRITE
49	ASTB*	ADDRESS STROBE	50	DSTB*	DATA STROBE
51	SMRQ*	SECONDARY MASTER REQUEST	52	SMACK*	SECONDARY MASTER ACKNOWLEDGE
53	LOCK*	ACCESS LOCK	54	GND	GROUND
55	ADR22*	MULTIBUS® ADDRESS EXTENSION LINE 22	56	ADR23*	MULTIBUS® ADDRESS EXTENSION LINE 23
57	ADR20*	MULTIBUS® ADDRESS EXTENSION LINE 20	58	ADR21*	MULTIBUS® ADDRESS EXTENSION LINE 21
59	RES	RESERVED	60	TPAR*	TRANSFER PARITY

A.2 SIGNAL LINE SUMMARY

DATA LINES (DB15 - DB0)
The 16 bi-directional data lines used to transfer data between the active bus master and the selected Slave device.

ADDRESS LINES (AB23 - AB0)
The 24 address lines used by the active bus master to select a Slave device and to specify a location in memory.

TRANSFER PARITY (TPAR*)
The optional Transfer Parity line operates as an additional data line to improve data-transfer integrity.

READ-NOT-WRITE (R/$\overline{W}$)
The active bus master controls the direction of data transfer with the Read-Not-Write line. When driven Low, the active bus master transmits the data and the selected slave device receives the data. Driving the Read-Not-Write line High reverses the transfer direction.

BYTE HIGH ENABLE (BHEN)
The active bus master in the 16-bit iLBX bus configuration controls the type of data transfer (8-bit or 16-bit) using the Byte High Enable (BHEN) element select line along with the low-order address bit (AB0).

ADDRESS STROBE (ASTB*)
The active bus master drives the Address Strobe line Low to initiate a data transfer cycle.

DATA STROBE (DSTB*)
The active bus master drives the Data Strobe line Low to set-up the actual transfer of data.

ACKNOWLEDGE (ACK*)
The selected slave device responds to selection by the active bus master by driving the Acknowledge line Low.

LOCK (LOCK*)
The active bus master restricts access through the alternate port to dual port RAM memory on a Slave device by driving the Lock line Low.

SECONDARY MASTER REQUEST (SMRQ*)
The Secondary Master requests use of the iLBX bus from the Primary Master by driving the Secondary Master Request line Low.

SECONDARY MASTER ACKNOWLEDGE (SMACK*)
The Primary Master allows use of the iLBX bus by the Secondary Master by driving the Secondary Master Acknowledge line Low after the Secondary Master drives the Secondary Master Request line Low.

MULTICHANNEL™ I/O BUS SPECIFICATION

IV

PREFACE

The Multichannel bus is one of the subsidiary buses within the overall Multibus Interface System. The Multichannel bus is free-standing to the extent that its interface and protocol do not require the existence of the general purpose Multibus interface or any of the other subsidiary buses. When used within the form factor of the Multibus board specification, the Multichannel bus attaches at the side of the circuit board opposite the P1 and P2 connectors and imposes restrictions on Multibus board designs implementing other subsidiary buses.

This specification describes the operation protocol of the Multichannel bus and defines the electrical and mechanical requirements of the Multichannel bus. A section of design guideline examples provide additional insight for implementing the Multichannel bus in a system. This specification does not duplicate specification information from the Multibus Interface Specification or any of the subsidiary bus specifications. Information on the Multibus interface or a subsidiary bus is provided in the following specifications.

- *Intel MULTIBUS® Interface Specification,* order number 9800683
- *Intel iSBX™ Bus Specification,* order number 142686
- *Intel iLBX™ Bus Specification,* order number 145695

This specification follows the general guidelines in the "Recommendations on Terminology for IEEE Computer Society Interface Standards" review draft dated September 9, 1981, and revised November 3, 1981, and June 3, 1982. In compliance with the terminology recommendations, this specification uses decimal notation when numbering bus lines with bit 0 as the least significant bit. This specification also uses the trailing asterisk to designate active Low signal lines. Where Multibus interface signal names (or subsidiary bus signal names) are used in this specification, these names are converted to comply with the terminology recommendations.

CONTENTS

CONTENTS (continued)

Page

CHAPTER 3
ELECTRICAL SPECIFICATION

CHAPTER 4
MECHANICAL SPECIFICATIONS

CHAPTER 5
DESIGN GUIDELINES AND SYSTEM APPLICATIONS

CONTENTS

Page

CHAPTER 3
ELECTRICAL SPECIFICATION

CHAPTER 4
MECHANICAL SPECIFICATIONS

CHAPTER 5
DESIGN GUIDELINES AND SYSTEM APPLICATIONS

CONTENTS (continued)

CHAPTER 6 Page

LEVELS OF COMPLIANCE

FIGURES

Figure Title Page

IV

TABLES

CHAPTER 1
INTRODUCTION

1.1 INTERFACE SYSTEM OVERVIEW

The Multichannel bus is a specialized electrical and mechanical interfacing protocol operating within the overall Multibus interfacing system. Figure 1-1 illustrates the Multibus interfacing system.

The foundation of the Multibus interface structure is the general purpose Multibus interface, the flexible bus structure used to interface the family of Intel's 80/86 products including both 8- and 16-bit products. The Multibus interface supports both 8- and 16-bit data transfers and direct addressability of up to 16 megabytes of memory. In many systems, the Multibus interface provides all of the required interconnect capability for the system.

As systems grow in complexity and performance, the throughput demands on the interconnect architecture increase. The Multibus interfacing system meets these demands by off-loading specific interconnect needs to the following subsidiary buses:

- iSBX^{TM} bus
- MULTICHANNEL^{TM} bus
- iLBX^{TM} bus

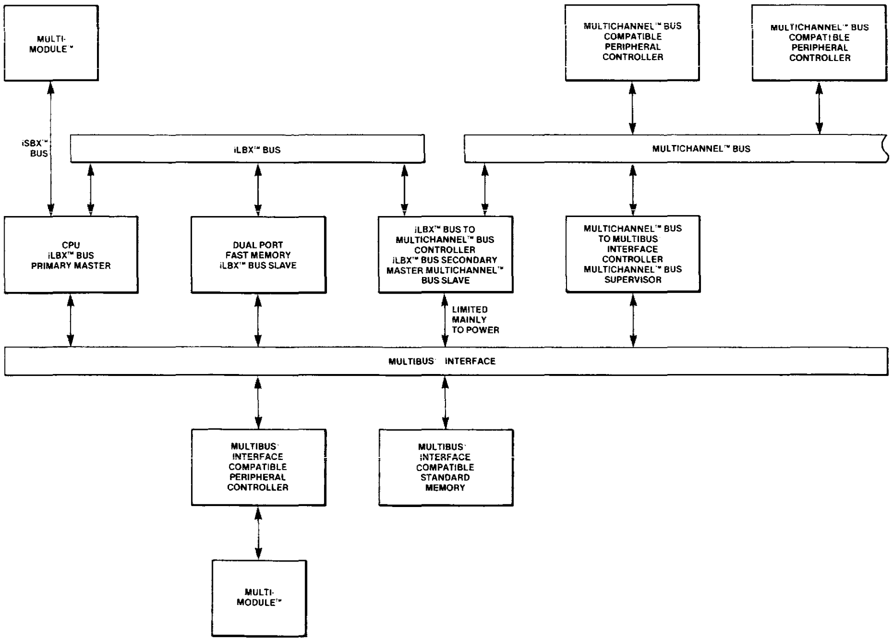

MULTI-MODULE™
MULTICHANNEL™ BUS COMPATIBLE PERIPHERAL CONTROLLER
MULTICHANNEL™ BUS COMPATIBLE PERIPHERAL CONTROLLER
iSBX™ BUS
iLBX™ BUS
MULTICHANNEL™ BUS
CPU iLBX™ BUS PRIMARY MASTER
DUAL PORT FAST MEMORY iLBX™ BUS SLAVE
iLBX™ BUS TO MULTICHANNEL™ BUS CONTROLLER iLBX™ BUS SECONDARY MASTER MULTICHANNEL™ BUS SLAVE
MULTICHANNEL™ BUS TO MULTIBUS INTERFACE CONTROLLER MULTICHANNEL™ BUS SUPERVISOR
LIMITED MAINLY TO POWER
MULTIBUS INTERFACE
MULTIBUS INTERFACE COMPATIBLE PERIPHERAL CONTROLLER
MULTIBUS INTERFACE COMPATIBLE STANDARD MEMORY
MULTI-MODULE™

Figure 1-1 MULTIBUS® Interface System

In a fully expanded Multibus interface system, the Multibus interface is used mainly for system control and low to medium-speed data transfers.

1.1.1 iSBX™ Bus

Increasing the number of functions residing on each system board attached to the Multibus interface increases the system performance. The improved system performance results because the resident functions are accessed without bus arbitration. The trade-off becomes choosing the resident functions when designing the system board. The iSBX bus extension of the Multibus interfacing system helps reduce the need to make design choices. The special functions are designed onto individual small boards and connected to a system board using the iSBX bus interface. When installed on the system board, the special function operates as though it were residing on the system board. Thus a system designer can have resident on the system boards those special functions most advantageous to his system.

1.1.2 iLBX™ Bus

Dramatically increasing the local (on-board) memory resources of a high performance processor provides a second means of increasing system performance. As with other special functions, memory residing on the processor board improves system performance because the processor directly addresses the memory without waiting for bus arbitration. However, there is a physical limitation to the amount of memory that can reside on the processor board. The iLBX bus helps to reduce the physical space limitation. Using the iLBX bus, the additional memory no longer needs to be located on the processor board or on a multimodule attached to the processor board. In fact, the full 16 megabytes of memory addressable by the processor can be accessed over the iLBX bus and appear to the processor as though it were resident on the processor board. Dual porting the memory between the iLBX bus and the Multibus interface makes the same memory resources available to other system components.

1.1.3 MULTICHANNEL™ Bus

Reducing the impact of burst-type peripherals (e.g. most disk peripherals) on the Multibus interface provides a third means of increasing system performance. The actual data transfers from a burst type peripheral can saturate a general purpose interface such as the Multibus interface. Adding more burst-type peripherals to a system often decreases the computing performance of the system. The Multichannel bus extension to the Multibus interfacing system helps reduce the bus-saturation problem.

The Multichannel bus protocol specifically accommodates burst-type data transfers. The full performance improvement requires use of dual port memory accessed over both the Multichannel bus and the Multibus interface.

1.2 MULTICHANNEL™ BUS GENERAL DESCRIPTION

The Multichannel bus is designed for high speed block data transfers between the Multibus system and the interconnected peripheral devices. The Multichannel bus devices typically are bus connected using a wire cable assembly. The maximum interconnect cable length is 15 meters (approximately 50 feet) inclusive. The Multichannel bus data path width can be 8-bits (byte wide) or 16-bits (word wide). A maximum of 16 Multichannel bus compatible devices can be interconnected by an implementation of the Multichannel bus. With the exception of the bus supervisor, each device attached to a Multichannel bus is assigned a specific device number from 0H to 0EH.

The Multichannel bus uses an addressing range of 2^{24}. The Multichannel bus memory data transfers use byte-addressing and this provides a memory addressing range of 16 megabytes. When the 16-bit data path is used, two bytes can be transferred by a single memory data transfer. The Multichannel bus registers are defined as 16-bit registers but they can be device limited to 8-bits.

The block data-transfer protocol used for memory data transfers on the Multichannel bus minimizes the data transfer overhead for data transfers where a series of bytes or words are transferred to or from consecutive memory locations. The Multichannel bus specification requires that both the sending and receiving device manage their own memory addressing and that after a pointer address is sent at the start of the transfer operation, each subsequent transfer does not include a memory address but is limited to data until the transfer sequence is completed. The Multichannel bus data transfers are asynchronous and use a positive handshake protocol along with transfer parity verification to assure the data transfer accuracy.

1.3 MULTICHANNEL™ BUS IMPLEMENTATION EXAMPLES

The following are typical examples of system environments where large blocks of data are passed between the devices in the system environment. Because these blocks of data are sequentially addressed, the Multichannel bus provides a fast, efficient means of interconnecting the system elements. The implementation examples do not attempt to show all the system components but only those applicable to the Multichannel bus. The examples also do not attempt to show all possible Multichannel bus implementation but rather a sampling to demonstrate the flexibility of the interfacing system.

Figure 1-2 shows a two device implementation of the Multichannel bus with the link being used to transfer large blocks of data between two Multibus based systems. In the example shown, one system could be a front-end processor that interfaces to the outside world and does preprocessing on the data before transferring the data to a second system for further processing.

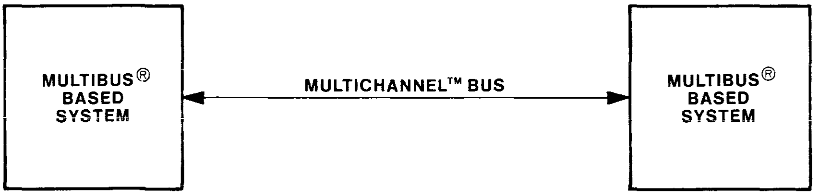


Figure 1-2 MULTICHANNEL™ Bus Implementation System Example

Figure 1-3 also shows a two device implementation of the Multichannel bus with the link being used to transfer blocks of data between a Multibus interface based system and a free standing array processor.

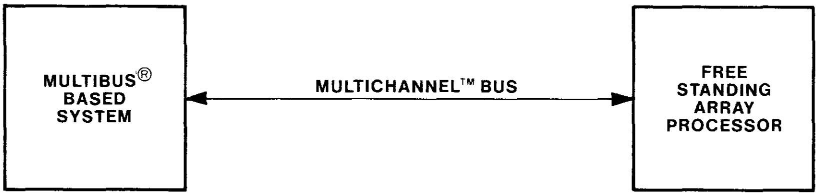


Figure 1-3 MULTICHANNEL™ Bus Implementation System Example

Figure 1-4 shows a four device implementation of the Multichannel bus. In this implementation, the intelligent workstation is the supervisor for the Multichannel bus. The video camera captures raw data, preprocesses, and digitizes the data for transfer to other devices in the system. For direct display, the workstation (supervisor) directs the video camera to transfer the data to the graphics terminal. For major processing of the captured data, the workstation (supervisor) directs the video camera to transfer the data to the Multibus based system and, following processing, from there to the graphics terminal for display. Additional operations are possible including the workstation directing the graphics terminal to modify the display (zoom, rotate, etc.) or the workstation turning supervisor control of the system over to the Multibus interface based system.

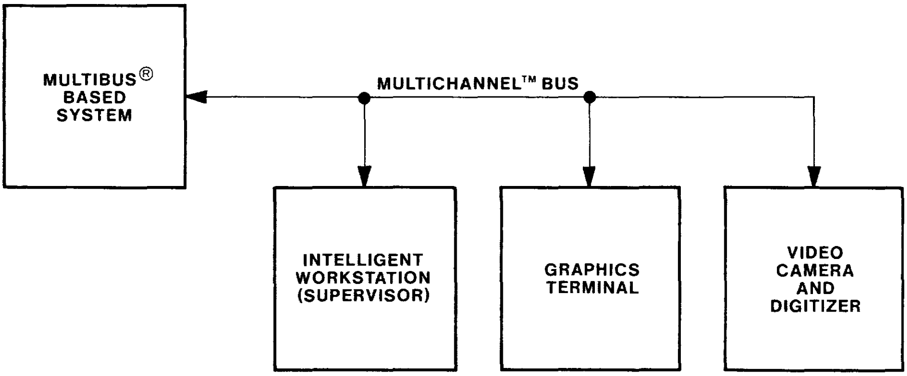

MULTIBUS® BASED SYSTEM
MULTICHANNEL™ BUS
INTELLIGENT WORKSTATION (SUPERVISOR)
GRAPHICS TERMINAL
VIDEO CAMERA AND DIGITIZER

Figure 1-4 MULTICHANNEL™ Bus Implementation System Example

Figure 1-5 shows the first three Multichannel bus implementations interconnected into a more complex system. This example particularly demonstrates that a Multibus based system can be attached to multiple implementations of the Multichannel bus.

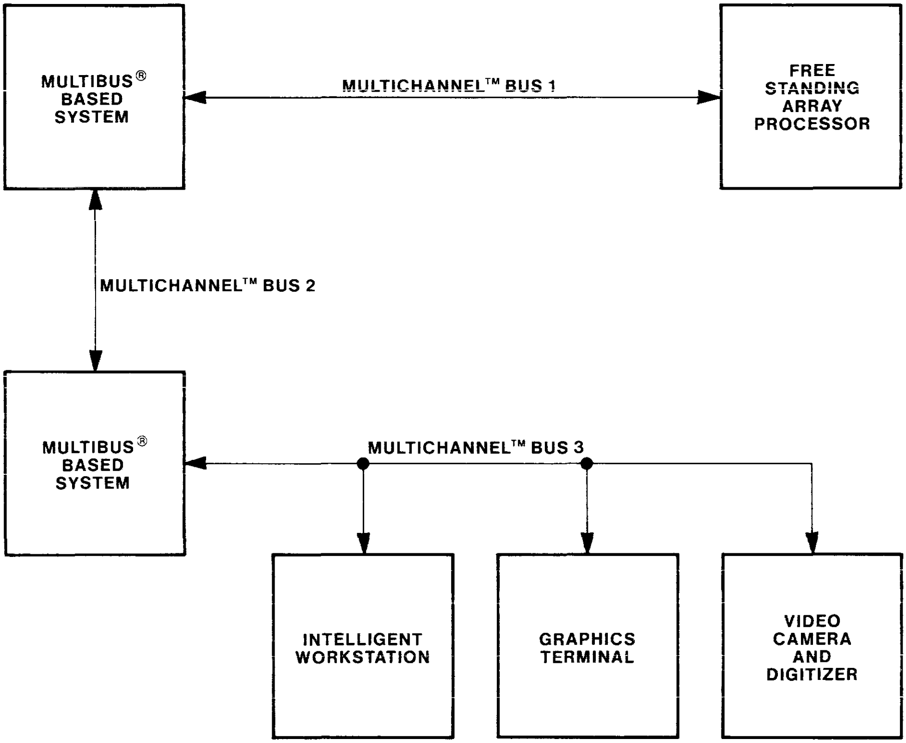


Figure 1-5 MULTICHANNEL™ Bus Implementation System Example

CHAPTER 2 FUNCTIONAL DESCRIPTION

2.1 INTRODUCTION

The Functional Description defines the various elements of the Multichannel bus interface. These elements include descriptions of the device categories using the Multichannel bus, the operational states for devices using the Multichannel bus, the signal line grouping and functions, the timing requirements, and the bus communication protocol.

2.2 NOTATION CONVENTIONS

The general notational conventions used in this manual conform to the notational conventions used in the Multibus Specification. The following paragraphs summarize the notational conventions.

The Multichannel bus lines are assigned unique names and, for convenience, unique mnemonics. The signal line names are shown with initial capital letters when used in text. The corresponding signal line mnemonics are shown in all capital letters.

Signal mnemonics for TTL driven lines that are Asserted when electrically High (also called positive true) do not have a special terminating character as the last character in the mnemonic. Signal mnemonics for TTL driven lines Asserted when electrically low (also called negative true) have an asterisk (*) character as the last (terminating) character in the mnemonic.

Signal mnemonics for the differential line pairs where the signal is Asserted when the output differential is positive (the non-inverted output positive relative to the inverted output) use a mnemonic without a special terminating character for the non-inverted output and the same mnemonic with a slash (/) as the terminating (last) character for the inverted output. Signal mnemonics for differential line pairs where the signal is Asserted when the output differential is negative use a mnemonic with an asterisk (*) character as the last character for the non-inverted output and the same mnemonic with an asterisk followed by a slash (*/) as the terminating (last) two characters for the inverted output in the mnemonic. The mnemonic also can include as descriptive elements a superscripted bar or a slash character any place in the mnemonic except as the last character. These do not apply directly to the signal characteristics.

The electrical signal characteristic descriptions use the terms Asserted and Non-asserted (in initial capital letters) to indicate the state of a TTL signal line or a differential pair. The terms High or Low, corresponding to the relative voltage level of a TTL signal, and Positive differential or Negative differential, corresponding to relative voltage difference between the non-inverted and inverted lines in a differential pair, are also used with reference to the signal lines. The terms true, false, 1, and 0 are avoided to reduce misinterpretation.

Table 2-1 relates the electrical signal characteristics to the corresponding logical and state notations. The Example mnemonic, XMPL, illustrates the notational conventions.

The address and data bit numbering scheme used with the Multichannel bus has bit 0 as the least significant bit, and decimal numbers are used to identify the lines. Thus, Address/Data line 0 (AD0) carries the least significant bit and Address/Data line 15 (AD15) carries the most significant bit.

Table 2-1 Notational Summary

SIGNAL NAME	NOTATION			
	ELECTRICAL		LOGICAL	STATE
	TTL	DIFFERENTIAL		
XMPL	H, High		1, True	Asserted
	L, Low		0, False	Non-asserted
XMPL, XMPL/		Positive	1, True	Asserted
		Negative	0, False	Non-asserted
XMPL*	H, High	0, False	Non-asserted	
	L, Low		1, True	Asserted
XMPL*, XMPL*/		Positive	0, False	Non-asserted
		Negative	1, True	Asserted

2.3 MULTICHANNEL™ BUS DEVICES

There are three levels of implementation defined for devices that interface to the Multichannel bus. The levels of implementation are Basic (minimum level), Controller (middle level), and Supervisor (highest level). The Controller is required to include all the capabilities of the Basic device. Because the Supervisor can not be the slave state device in a Multichannel bus transfer, it includes most, but not all, of the controller and Basic device capabilities. The number of devices interfaced by a Multichannel bus implementation is limited to 16 devices. The set of devices must include one Supervisor and at least one other device, either a Controller operating as a Basic device (slave state) or a Basic device. The remaining 14 devices are optional and may include additional Controllers and/or Basic devices; but, must not include another Supervisor.

2.3.1 Basic Device

The Basic device represents the minimum implementation level for compatibility with the Multichannel bus specification. A Basic device does not have any Multichannel bus control capability beyond the ability to assert Service Request (SRQ) or Supervisor Take Over (STO) to interrupt the Supervisor. A Basic device that supports SRQ or STO must both receive and transmit register data to the extent required for Multichannel bus interrupt control. Otherwise, the Basic device is not required to support register data transfers. The Basic device can receive memory data exclusively, transmit memory data exclusively, or both receive and transmit memory data. Up to 15 Basic devices can be attached to an implementation of the Multichannel bus, and the Basic devices can be assigned any device number from 0H to 0EH.

2.3.2 Controller Device

The Controller device represents the middle implementation level for compatibility with the Multichannel bus specification. The Controller device incorporates the full capabilities of a Basic device, and can direct data flow over the Multichannel bus. The Controller device cannot respond to Multichannel bus interrupts (Supervisor Take Over and Service Request) or assert the Supervisor Active signal. Up to 15 Controller devices can be attached to an implementation of the Multichannel bus, and the Controller devices can be assigned any device number from 0H to 0EH.

2.3.3 Supervisor Device

The Supervisor device represents the highest implementation level for compatibility with the Multichannel bus specification. The Supervisor device incorporates the full capabilities of a Controller device except those capabilities related to operation in the slave state. In addition, the Supervisor device monitors the interrupt signals (Supervisor Take Over and Service Request). When the Supervisor is not an active participant in a bus transfer (Supervisor Active non-asserted), the Supervisor can preempt the operation in progress by asserting the Supervisor Active signal. The Supervisor device also programs the Controller devices. Only one Supervisor device can be attached to the Multichannel bus to manage the bus implementation. The Supervisor device can not be the listener for an address mode transfer and does not have an assigned device number. The Multichannel bus protocol does not provide for the passing of supervisorship between devices.

2.4 DEVICE OPERATIONAL CONDITIONS

The devices physically attached to a Multichannel bus exhibit one of two operational conditions at all times: active condition and inactive condition.

A device is active when it is powered and available to participate in a Multichannel bus transaction. By definition, the Supervisor must be active to implement and manage the bus. The remaining devices can be active or inactive without affecting the ability of two active devices successfully completing a bus transaction.

A device is inactive when it is physically attached to the Multichannel bus cable but is logically not present. A device that is not powered is logically not present. All Basic and Controller devices should provide a means for the Supervisor to make the device inactive and logically not present. The ability to be made not present is a fault tolerant capability allowing the Supervisor to logically disconnect a device that is disrupting bus operation.

2.5 DEVICE OPERATING STATES

The active devices attached to an implementation of the Multichannel bus operate in one or more of the six operating states at all times. The following are the operating states:

- passive state
- selected state
- listener state
- talker state
- slave state
- master state

2.5.1 Passive State

The passive state is the least active state for any Multichannel bus slave state device that is logically present on the bus. A device in the passive state continually monitors the bus for an address mode transfer that specifies its assigned device number. Because the Supervisor device cannot accept an address mode transfer, it defaults to monitoring the interrupt lines, Service Request and Supervisor Take Over, anytime it is not Asserting the Supervisor Active line.

2.5.2 Selected State

A slave state device enters the selected state anytime it accepts an address mode transfer that specifies its assigned device number. Once selected, the device remains selected until it receives an address mode transfer that specifies Deselect (device number 0FH) or the Supervisor device Asserts the Reset line. The Supervisor device has selected status anytime it is Asserting the Supervisor Active line. During normal operation, there will be a minimum of one selected device and a maximum of two selected devices on the Multichannel bus. When the Supervisor device preempts control of the bus, there may be a brief period of time when there are three selected devices; however, upon preemption, the Supervisor immediately performs an address mode transfer specifying Deselect in preparation for a device poll to identify the interrupting device.

2.5.3 Listener State

A Multichannel bus device (Supervisor, Controller, or Basic device) is in the listener state anytime it is accepting information from the Address/Data lines. Each complete transaction over the Multichannel bus requires that at least one device must be in the listener state. When the Address-Not Data signal is asserted, all passive and slave state devices are in the listener state. When the Address-Not Data signal is non-asserted, only the selected slave state device is in the listener state.

2.5.4 Talker State

A Multichannel bus device (Supervisor, Controller, or Basic device) is in the talker state anytime it is Asserting the Address/Data lines. Each complete transaction over the Multichannel bus requires that one of the selected devices must be in the talker state.

2.5.5 Master State

The Supervisor device or a Controller device is in the master state when it has control of the Read-Not-Write and the Address-Not-Data differential control-line pairs. Thus the master is the controlling active participant in a data transfer operation (either memory or register). When the Supervisor is not an active participant in a bus transfer (Supervisor Active non-asserted), the Supervisor can preempt the operation in progress by asserting the Supervisor Active signal. The preempted devices must provide for an orderly termination of the preempted operation.

Only one device (the Supervisor or a Controller) can be in the master state at any one time. The Supervisor device must assume the master state by default when the system is initially powered. A Controller device must be programmed for master state operation by the Supervisor device before it can enter the master state.

2.5.6 Slave State

A Basic device or a Controller device is in the slave state when it is the controlled participant in any bus transaction. A Basic device is in the slave state continually. Until selected to exercise a higher level of control, a Controller device is in the slave state. A Controller device is in the slave state when it is selected and accepts control from the Read-Not-Write and the Address-Not-Data differential control-line pairs.

2.6 SIGNAL LINE DESCRIPTIONS

The following five signal line categories make up the physical structure of the Multichannel bus interface:

- address/data lines
- control lines
- handshake lines
- interrupt lines
- supervisor command lines

The Multichannel bus signal lines include four signals that are transferred over differential line pairs. In determining the number of bus signal lines, each differential line pair is considered a single signal. In the signal line descriptions, the differential line pairs are specifically described, the non-inverted and inverted signal mnemonics listed, and any special requirements noted. For simplicity of reference, the differential signal line pairs are referred to by specific name or by the non-inverted signal mnemonic.

2.6.1 Address/Data Lines

The address/data lines are subdivided into 16 Address/Data lines and the Parity Bit differential line pair. The 16 Address/Data lines are all functionally identical and the lines are not separately described.

2.6.1.1 ADDRESS/DATA LINES (AD15* - AD0*)

The selected talker for a given bus cycle uses the 16 Address/Data lines to transfer the required information to the listener.

The general Address/Data signal line implementation specifications are as follows.

- The address/data lines are negative true lines.
- All Multichannel bus devices capable of operating in the talker state must provide tristate drivers for the Address/Data lines.
- During a bus cycle, only the selected talker drives the Address/Data lines. The selected listener and all passive devices must hold the Address/Data line drivers in the high-impedance state.

2.6.1.2 PARITY BIT (PB*, PB*/)

The Parity Bit differential line pair is used to verify the integrity of the information transferred during a bus cycle. Parity must be developed over the full data path of the Multichannel bus regardless of the transfer element size (8-bit byte or 16-bit word). The Multichannel bus uses odd parity defined as follows: when there is an even number of one bits in the transfer element (8-bit byte or 16-bit word), the talker asserts the Parity Bit signal. A listener detecting a parity error stores a device defined non-zero value in the STO Register (register address 0H) and asserts the Supervisor Take Over line.

The general Parity Bit differential line pair implementation specifications are as follows.

- All Multichannel bus compatible devices must provide a tri-state differential driver and a receiver for the Parity Bit line pair.

- All Multichannel bus compatible devices should provide a means for masking the generation and/or the detection of parity.

- Only the selected talker asserts the Parity Bit line pair. The selected listener must sample parity and all passive devices must hold their Parity Bit differential driver in the high-impedance state during the bus cycle.

2.6.2 Control Lines

The selected master controls both the bus cycle and message information transfer parameters using the two control signals. The control signals are both carried on differential line pairs.

2.6.2.1 ADDRESS-NOT-DATA ($A/\overline{D}$, $A/\overline{D}/$)

The Address-Not-Data differential line pair is used by the master to identify the type of information transferred during a bus cycle. The master asserts the Address-Not-Data line pair to identify the transfer as address information. When Address-Not-Data is non-asserted, the transfer is data.

The general Address-Not-Data differential line pair implementation specifications are as follows.

- The Supervisor and all Controllers must provide a tristate differential driver and all Controllers and Basic devices must provide a receiver for the Address-Not-Data line pair.

- Only the selected master asserts the Address-Not-Data line pair. The selected slave state device must sample the Address-Not-Data line pair and all passive devices must hold their Address-Not-Data differential driver in the high-impedance state during the bus cycle.

2.6.2.2 READ-NOT-WRITE ($R/\overline{W}$, $R/\overline{W}/$)

The Read-Not-Write differential line pair is used by the master to identify the direction of information transfer during a bus cycle relative to the master. The master asserts the Read-Not-Write line pair to identify the transfer as a read operation. When Read-Not-Write is non-asserted, the transfer is a write. All address information transfers must be write operations. Data information transfers can be either read or write.

The general Read-Not-Write differential line pair implementation specifications are as follows.

- The Supervisor and all Controllers must provide a tri-state differential driver and all Controllers and Basic devices must provide a receiver for the Read-Not-Write line pair.

- Only the selected master asserts the Read-Not-Write line pair. The selected slave state device must sample the Read-Not-Write line pair and all passive devices must hold their Read-Not-Write differential driver in the high-impedance state during the bus cycle.

2.6.3 Transfer Handshake Lines

The selected talker and the listener(s) use the transfer handshake lines to strobe and acknowledge transfer of the information element during a bus cycle.

2.6.3.1 DATA READY (DRDY*, DRDY*/)

The Data Ready differential line pair is asserted by the talker to signal the listener that information is valid on the Address/Data lines.

The general Data Ready differential line pair implementation specifications are as follows.

- All Multichannel bus compatible devices with talker capabilities must provide a tri-state differential driver. All Multichannel bus compatible devices must provide a receiver for the Data Ready line pair.

- Only the selected talker asserts the Data Ready line pair. The selected listener must sample the Data Ready line pair and all passive devices must hold their Data Ready differential driver in the high-impedance state during the bus cycle.

2.6.3.2 ADDRESS MODE ACCEPT (AACC)

The Address Mode Accept line is used by all slave state devices to indicate receipt of the address information during an address mode bus cycle. All slave state devices must operate as listeners during address mode bus cycles. To assure that all devices receive the address information, the Address Mode Accept signal operates as a wired AND. In this configuration, so long as any one device is not asserting the Address Mode Accept line, the line remains non-asserted. Any device that has the line non-asserted masks the Address Mode Accept from the master. After all devices have asserted the address accept line, the master detects the address accept and continues with the address mode bus cycle.

The general Address Mode Accept line implementation specifications are as follows.

- A Controller in the master state for an address mode bus cycle must hold the Address Mode Accept line asserted to avoid masking acknowledgement from the slave state devices.

- All Multichannel bus devices must hold the Address Mode Accept line non-asserted during all data mode bus cycles.

- All slave state devices must hold the Address Mode Accept line non-asserted during all data mode bus cycles.

2.6.3.3 DATA MODE ACCEPT (DACC*)

The Data Mode Accept line is used by the selected listener during a data mode bus cycle to indicate receipt of the data information.

The general Data Mode Accept line implementation specifications are as follows.

- The talker for a data mode bus cycle and all non-selected devices must hold the Data Mode Accept line non-asserted during the data mode bus cycle.

- Only the selected listener state device may assert the Data Mode Accept line during a data mode bus cycle.

- All Multichannel bus devices must hold the Data Mode Accept line non-asserted during all address mode bus cycles.

2.6.4 Interrupt Lines

The Multichannel bus provides two interrupt lines. The interrupt lines are implemented as a wired OR function. The first device asserting an interrupt line causes the line to be asserted. Additional devices can also assert the line in which case, the line remains asserted until the last interrupt is serviced and all devices are non-asserting the interrupt line. Both specific Multichannel bus interrupt conditions and device defined conditions can be reported using the interrupt lines. Refer to the descriptions of the status registers for additional information. Interrupt servicing is performed by the supervisor exclusively.

2.6.4.1 SUPERVISOR TAKE OVER (STO*)

The Supervisor Take Over interrupt is used by a non-supervisor device to indicate task completion, parity errors, and other device errors. Any device that is able to assert the Supervisor Take Over line must also support the STO status register, register 0. The values assigned by the device to all bits in register 0, except bit 7, is device dependent. Bit 7 is the error flag bit and must be set to one whenever the STO interrupt is asserted due to detection of an error condition. The Supervisor services the interrupt by reading register 0 in the device asserting the Supervisor Take Over line. The interrupting device must respond with a non-zero .value, non-assert the Supervisor Take Over line, and then clear the STO status register to zero.

The general Supervisor Take Over line implementation specifications are as follows.

- The Supervisor Take Over line is a negative true, open collector, line.

- Support of the Supervisor Take Over line is optional; however, if a Basic or Controller device supports parity, it must also support the Supervisor Take Over line and the STO Status register.

2.6.4.2 SERVICE REQUEST (SRQ*)

The Service Request interrupt is used by a non-supervisor device to indicate device operational status and any other device specific conditions. Any device that is able to assert the Service Request line must also support the SRQ status register, register 1. Refer to the description of the SRQ status register for the specific bit values that must be used by a device to report status. The Supervisor services the interrupt by reading register 1 in the device asserting the Service Request line. The interrupting device must respond with a non-zero value, non-assert the Service Request line, and then clear the SRQ status register to zero.

The general Service Request line implementation specifications are as follows.

- The Service Request line is a negative true, open collector, line.
- Support of the Service Request line is optional; however, if a device supports the Service Request line, it must also support the SRQ Status register.
- All devices supporting the Service Request line must support the SRQ mask register.

2.6.5 Supervisor Command Lines

The Multichannel bus reserves two lines for the exclusive use of the supervisor. The supervisor uses these lines to initialize the Multichannel bus system and to preempt control of the bus.

2.6.5.1 SUPERVISOR ACTIVE (SA*)

The supervisor asserts the Supervisor Active line any time it is in the master state. The supervisor non-asserts the Supervisor Active line as the last step in relinquishing control of the bus to a selected master. The supervisor can preempt control of the bus at any time by asserting the Supervisor Active line. All non-supervisor devices on the bus must monitor the Supervisor Active line continuously and, upon detecting a transition of the Supervisor Active line from non-asserted to asserted, immediately non-assert all bus lines except the interrupt lines.

The general Supervisor Active line implementation specifications are as follows.

- The Supervisor Active line is a negative true, open collector, line.
- All non-supervisor devices must provide a receiver for the Supervisor Active line.

2.6.5.2 RESET (RESET*)

The supervisor asserts the Reset line to initialize all devices on the Multichannel bus to a known state. Typically, the supervisor asserts Reset only when the system is initially powered. All non-supervisor devices on the bus must monitor the Reset line continuously and, upon detecting the Reset line asserted, immediately place all tri-stateable devices in the high impedence state and internally initialize to the passive listener state. SRQ* and STO* lines may be left either asserted or non-asserted. AACC and DACC* lines must be non-asserted. At Reset, the Supervisor defaults to asserting Supervisor Active and Address-Not-Data.

The general Reset line implementation specifications are as follows.

- The Reset line is a negative true, open collector, line.
- All non-supervisor devices must provide a receiver for the Reset line.

2.7 MULTICHANNEL^{TM} BUS PROTOCOL CONCEPT

The Multichannel bus protocol is composed of two sub-protocols: the bus cycle protocol and the message protocol.

The bus cycle protocol is the control and handshaking required to move one data element (8-bit byte or 16-bit word) from the selected talker to the selected listener. Depending on the state of the control lines, the direction of data flow may be from master to slave or the reverse. A bus cycle is a complete entity with a definite start and end; however, a single bus cycle is not a complete message.

The message protocol determines the meaning given to a sequence of bus cycles. The minimum valid message is a two bus cycle address message. The Deselect message sent by the Supervisor immediately after it preempts bus control by asserting the Supervisor Active line is an example of a two cycle message. Figure 2-1 illustrates several message configurations including the minimum two cycle message.

The minimum valid data message requires five bus cycles:

1. a two cycle address mode transfer to select the slave
2. a one cycle data mode transfer
3. a two cycle address mode transfer to deselect the slave

The opening and closing two cycle address mode transfers are the same format as the minimum address mode message. A typical example of the minimum valid data message is the STO or SRQ register read operations conducted by the Supervisor during an STO or SRQ poll.

A valid data message can have any number of data mode cycles between the opening two cycle address mode transfer and the closing two cycle address mode transfer. Both types of data messages are illustrated in Figure 2-1.

The three message sequence in Figure 2-1 is another typical sequence. The sequence shows a portion of a data message, followed by a two cycle address mode message, followed by another data message. This sequence would be the typical sequence followed when the Supervisor resumes control of the bus following normal termination of a data message by the selected master.

2.8 MULTICHANNEL^{TM} BUS COMMUNICATION SEQUENCE

All transfers over the Multichannel Bus are asynchronous, with each device providing the transfer handshake capability. There are three modes of communication:

- Address Mode
- Data Mode
- Control Transfer Mode

MINIMUM VALID MESSAGE

BUS CYCLE 1 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 2 ADDRESS MODE ADDRESS WORD 2

MINIMUM VALID DATA MESSAGE

BUS CYCLE 1 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 2 ADDRESS MODE ADDRESS WORD 2	BUS CYCLE 3 DATA MODE DATA WORD 1	BUS CYCLE 4 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 5 ADDRESS MODE ADDRESS WORD 2

TYPICAL DATA MESSAGE

BUS CYCLE 1 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 2 ADDRESS MODE ADDRESS WORD 2	BUS CYCLE 3 DATA MODE DATA WORD 1	BUS CYCLE 4 DATA MODE DATA WORD 2	• • •	BUS CYCLE n-3 DATA MODE DATA WORD n-5	BUS CYCLE n-2 DATA MODE DATA WORD n-4	BUS CYCLE n-1 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE n ADDRESS MODE ADDRESS WORD 2

THREE MESSAGE SEQUENCE

CLE 3 ODE ORD 1	BUS CYCLE 4 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 5 ADDRESS MODE ADDRESS WORD 2	BUS CYCLE 1 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 2 ADDRESS MODE ADDRESS WORD 2	BUS CYCLE 1 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 2 ADDRESS MODE ADDRESS WORD 2	BUS CYCLE 3 DATA MODE DATA WORD 1	BUS CYCLE 4 ADDRESS MODE ADDRESS WORD 1	BUS CYCLE 5 ADDRESS MODE ADDRESS WORD 2

Figure 2-1 Bus Message Examples

2.8.1 Address Mode

The format for all address mode transfers is the same as the format of the minimum valid message. Each address mode transfer requires two information transfer elements. The information transfer elements can be two 16-bit words or two 8-bit bytes. When the information transfer elements are 8-bit bytes, the number of transfers remains the same. The 8-bit transfer elements must be the two low order bytes and the amount of information that can be passed is limited accordingly.

All slave state devices on the Multichannel bus must monitor all address mode transfers. Figure 2-2 shows the format for 16-bit address mode transfers. If the address mode transfers are device limited to 8-bit bytes, the information transfer elements are the low order bytes of the two words shown in Figure 2-2. During the first bus cycle of an address mode transfer, the master places the first word on the address/data lines and asserts the Data Ready line. Each slave state device on the Multichannel bus acknowledges receipt by asserting the Address Mode Accept line. The Address Mode Accept line operates as a wired AND function and the master does not detect Address Mode Accept as asserted until all devices are asserting the line. Therefore, the device with the slowest response time determines the timing for the address mode transfers. The same procedure is repeated for the second address mode bus cycle. Figure 2-3 illustrates the timing for the address mode transfer handshake.

	HIGH ORDER BYTE								LOW ORDER BYTE								
bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
	ADDRESS BITS 23 - 16								DEVICE NUMBER				RES	RES	$\overline{M}$/R	R/$\overline{W}$	WORD 1
	ADDRESS BITS 15 - 8								ADDRESS BITS 7 - 0								WORD 2

ADDRESS INFORMATION DESCRIPTIONS:

- ADDRESS BITS 23-16 – MOST SIGNIFICANT BYTE OF THE 24-BIT MEMORY ADDRESS.
- ADDRESS BITS 15-8 – MIDDLE BYTE OF THE 24-BIT MEMORY ADDRESS.
- ADDRESS BITS 7-0 – LEAST SIGNIFICANT BYTE OF THE 24-BIT MEMORY ADDRESS. BITS 4-0 ALSO USED FOR REGISTER ADDRESS.
- DEVICE NUMBER – ADDRESSES A DEVICE ON THE MULTICHANNEL BUS. NUMBERS 0H TO 0EH ARE USE FOR DEVICE ADDRESSING, NUMBER 0FH SPECIFIES THE ADDRESS MODE TRANSFER AS AN UNTALK SEQUENCE.
- RES – RESERVED BITS ALWAYS SET TO ZERO.
- $\overline{M}$/R – SPECIFIES TYPE OF ADDRESS. MEMORY ADDRESS WHEN 0, REGISTER ADDRESS WHEN 1.
- R/$\overline{W}$ – SPECIFIES DIRECTION OF TRANSFER. READ WHEN 0, WRITE WHEN 1. THIS BIT MUST MATCH THE OPERATION SPECIFIED BY THE STATE OF THE CORRESPONDING MULTICHANNEL BUS LINE.

Figure 2-2 Address Mode Parameter Block Format

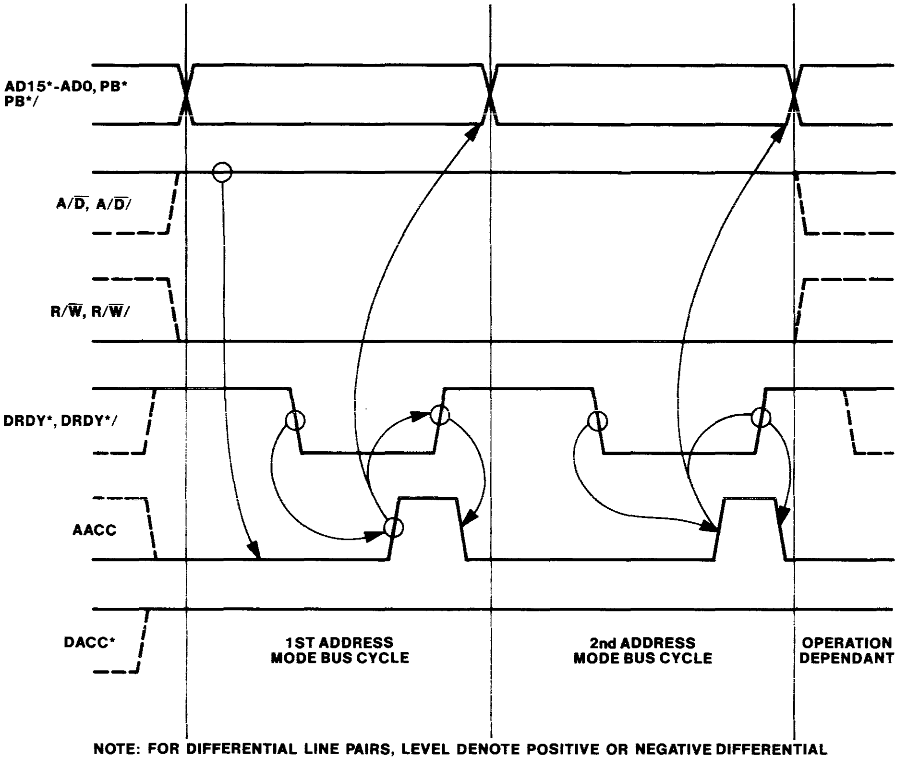

AD15*-AD0, PB*
PB*/
A/D̄, A/D̄/
R/W̄, R/W̄/
DRDY*, DRDY*/
AACC
DACC*
1ST ADDRESS MODE BUS CYCLE
2nd ADDRESS MODE BUS CYCLE
OPERATION DEPENDANT
NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTE POSITIVE OR NEGATIVE DIFFERENTIAL

Figure 2-3 Address Mode Bus Cycle Handshake Timing

The actions taken by the slave state devices receiving the address mode transfer depends upon the device number contained in the information transfer elements. If the device number is 0FH, the address mode transfer is a deselect message. When a slave state device receives the deselect message, it must terminate any operation in progress in an orderly manner and go to the passive state following the second address mode bus cycle in the deselect message. If the device number is in the range of 0H through 0EH, the master is attempting to select one of the slave state devices. Each slave state device must determine if it is the device being selected. If a slave state device is not selected, it remains in the passive state and continues to monitor the address mode transfers. The selected device must store and respond appropriately to the information in both of the address mode transfer elements.

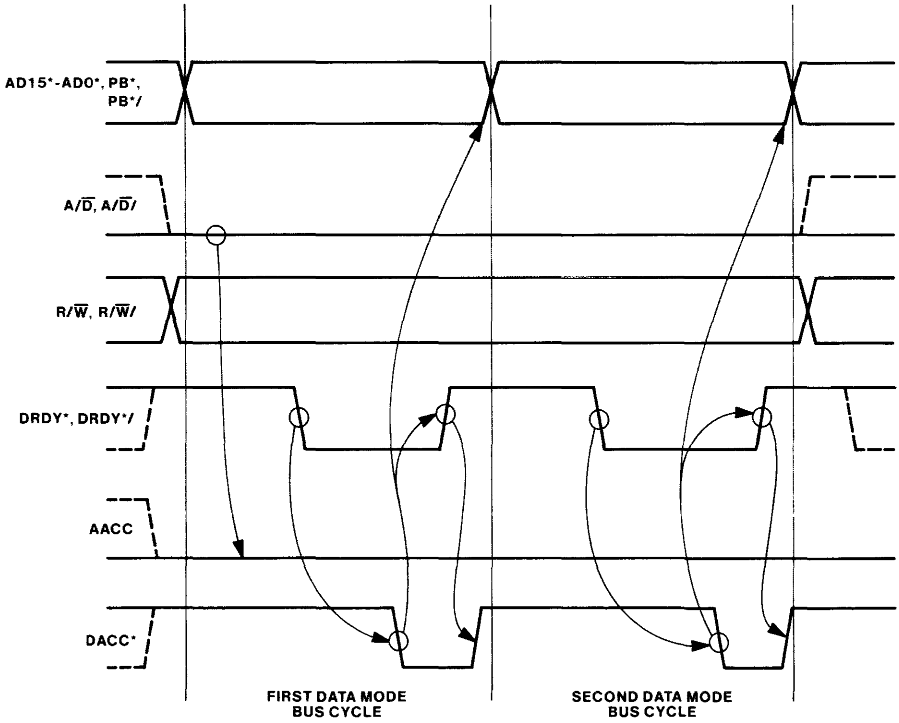
2.8.2 Data Mode

Data mode transfers, unlike address mode transfers, cannot be freestanding messages but are always part of larger messages. Any number of data mode transfers can be made between the opening address mode select sequence and the closing deselect address mode sequence. The talker for the data mode transfers in a data message can be the master (write) or the selected slave (read).

Data mode transfers can be 8-bit bytes or 16-bit words. When the data mode transfers are 8-bit bytes, the transfers are made over the address/data lines AD7* through AD0*.

The handshake sequence for the data mode transfers is the same as that used for the address mode transfers with one exception. In the data mode transfers, the listener acknowledges receipt of the information element by asserting the Data Mode Accept line. Figure 2-4 illustrates the timing for the data mode transfer handshake.

For write data messages, the master state device is the talker for the entire message. For read data messages, the master is the talker for the opening and closing address mode transfer sequences. However, the slave is the talker for the data mode bus cycles. This requires that control of the address/data lines and the Data Ready line must be transferred twice during the message. After the opening address mode sequence, the master passes talker control to the slave using the Address Not Data line and the Read Not Write line. The slave can anticipate the type of message, read or write, by examining the read not write bit in the first word (low order byte) of the opening address mode sequence. After all the data mode bus cycles are completed, the master assumes talker control from the slave using the Address Not Data line and the Read Not Write line.



NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTES POSITIVE OR NEGATIVE DIFFERENTIAL

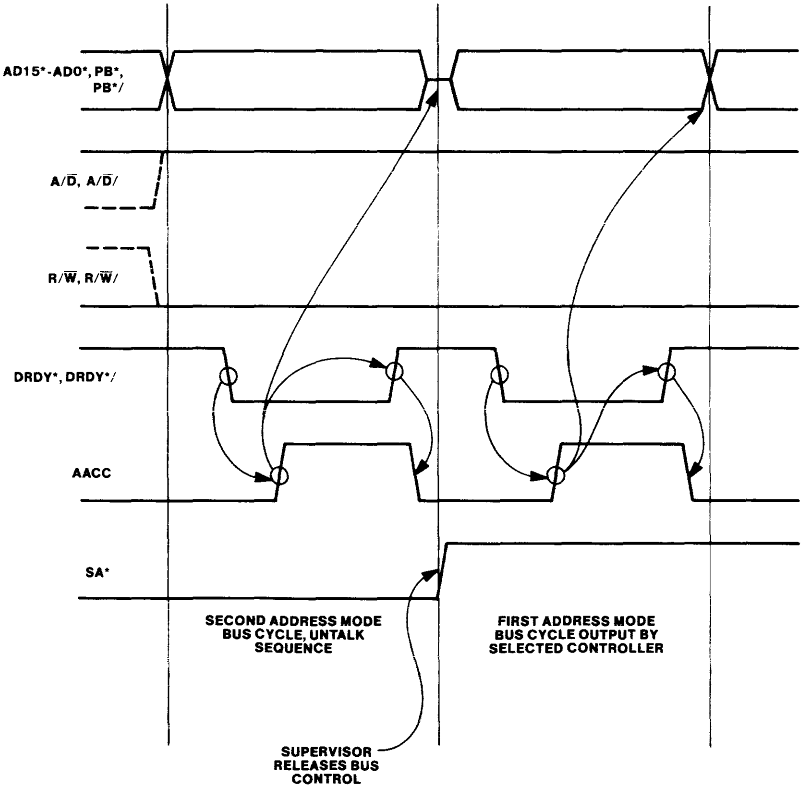
Figure 2-4 Data Mode Bus Cycle Handshake Timing

2.8.3 Control Transfer Mode

There are two types of control transfer: the Supervisor releasing control of the bus to a Controller and the Supervisor resuming control of the bus. The Supervisor releases control of the bus to a Controller after the Supervisor has programmed the Controller to perform one of a series of operations. The Supervisor may resume control of the bus as the result of an interrupt (Supervisor Take Over or Service Request) or it may choose to preempt control of the bus at anytime.

The Supervisor releases control of the bus by non-asserting the Supervisor Active line following a deselect address mode message. When the Supervisor non-asserts the Supervisor Active line, the programmed Controller assumes control of the Address Not Data and Read Not Write lines.

The Supervisor resumes control of the bus by asserting the Supervisor Active line. If any device is exercising control of the Address Not Data and Read Not Write lines, it must immediately release control of the lines. Upon taking control of the bus, the Supervisor outputs a deselect address mode message to deselect all devices on the bus.


AD15*-AD0*, PB*,
PB*/
A/D̄, A/D̄/
R/W̄, R/W̄/
DRDY*, DRDY*/
AACC
SA*
SECOND ADDRESS MODE BUS CYCLE, UNTALK SEQUENCE
FIRST ADDRESS MODE BUS CYCLE OUTPUT BY SELECTED CONTROLLER
SUPERVISOR RELEASES BUS CONTROL

NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTES POSITIVE OR NEGATIVE DIFFERENTIAL

Figure 2-5 Supervisor To Controller Bus Control Transfer

2.9 ADDRESSING

The Multichannel bus uses an addressing range of 2^{24}. The Multichannel bus memory data transfers use byte-addressing and this provides a memory addressing range of 16 megabytes. The Multichannel bus registers are defined as 16-bit registers but they can be device limited to 8-bits.

The Multichannel bus memory data mode transfers use byte-addressing and the maximum memory addressing capacity over the Multichannel bus is 16-megabytes. When the 16-bit data path is used, two bytes can be transferred by a single memory data mode transfer.

The Multichannel bus registers are defined as 16-bit registers and the register data mode transfers use word-addressing. The maximum register addressing capacity over the Multichannel bus is approximately 32-megabytes. Though defined as 16-bit registers, the register size can be device limited to 8-bits. Several of the system reserved registers are defined to be 8-bit registers.

The address mode parameter block (Figure 2-2) provides for 24 address bits when word transfers are made. The device selected by the device number must store the full address for use during the subsequent data mode transfers in a data message. Following the initial address mode transfer, the master does not update the address regardless of the number of data mode transfers made in the message. For memory data transfers, the slave must provide a means of incrementing the memory address after each data mode transfer. For register data transfers, the slave has the option of providing address incrementing after each data mode transfer.

2.10 RESERVED REGISTERS

The first 16 low order memory addresses, 0H through 0FH, are reserved for system use. The first three reserved registers (0H, 01H, and 02H) are the interrupt service registers. Registers 03H through 0BH are used by the Supervisor to pass operation parameters to a controller. The remaining registers, 0CH through 0FH, are undefined and reserved for future use. Support for the reserved registers is optional for a Basic device; however, if the Basic device supports the Supervisor interrupt capability, it must support the corresponding interrupt service register. A controller must support both the interrupt service and operation parameter registers.

2.10.1 Register 0H - STO Status

The STO status register is an 8-bit register provided by a non-supervisor device with the capability of interrupting the Multichannel bus supervisor with the Supervisor take Over interrupt. A non-supervisor uses the STO status register to report status and error conditions to the Multichannel bus Supervisor. Whenever the interrupting device detects an error condition, it writes a device defined code into the STO status register and asserts the Supervisor Take Over line. When the supervisor acknowledges the STO interrupt by conducting an STO poll, the interrupting device must output a non-zero value and clear the STO status register. Definition of the non-zero value is left to the interrupting device. Bit 7 of the STO register is designated the error flag bit and the interrupting device must set bit 7 to one to differentiate between error conditions and general device status.

2.10.2 Register 1H - SRQ Status

The SRQ status register is an 8-bit register provided by a non-supervisor device with the capability to interrupt the Multichannel bus supervisor with the Service Request interrupt. A non-supervisor device uses the SRQ status register to report operational status and other general status to the Supervisor. Whenever the operational status of the non-supervisor device changes or there is a device defined need to report status, it writes the appropriate code into the SRQ status register and asserts the Service Request line. When the supervisor acknowledges the SRQ interrupt by conducting an SRQ poll, the interrupting device must output a non-zero value and clear the SRQ status register. For reporting operational status, the content of the SRQ register is specifically defined. Figure 2-6 illustrates the reporting format of the SRQ register for both operational status and general status.

2.10.3 Register 2H - SRQ Mask Register

The SRQ mask register is an 8-bit register provided by a non-supervisor device with the capability of asserting the Service Request line. The Supervisor can disable the Service Request interrupt on the non-supervisor device by writing 01H to the SRQ mask register. During initialization, the non-supervisor device must set the SRQ mask register to 0H enabling the Service Request interrupt. This allows the non-supervisor device to interrupt the Supervisor at the completion of initialization and report the current operational status. The format for the SRQ mask register pair is shown in Figure 2-7.

2.10.4 Register 3H - Device Command

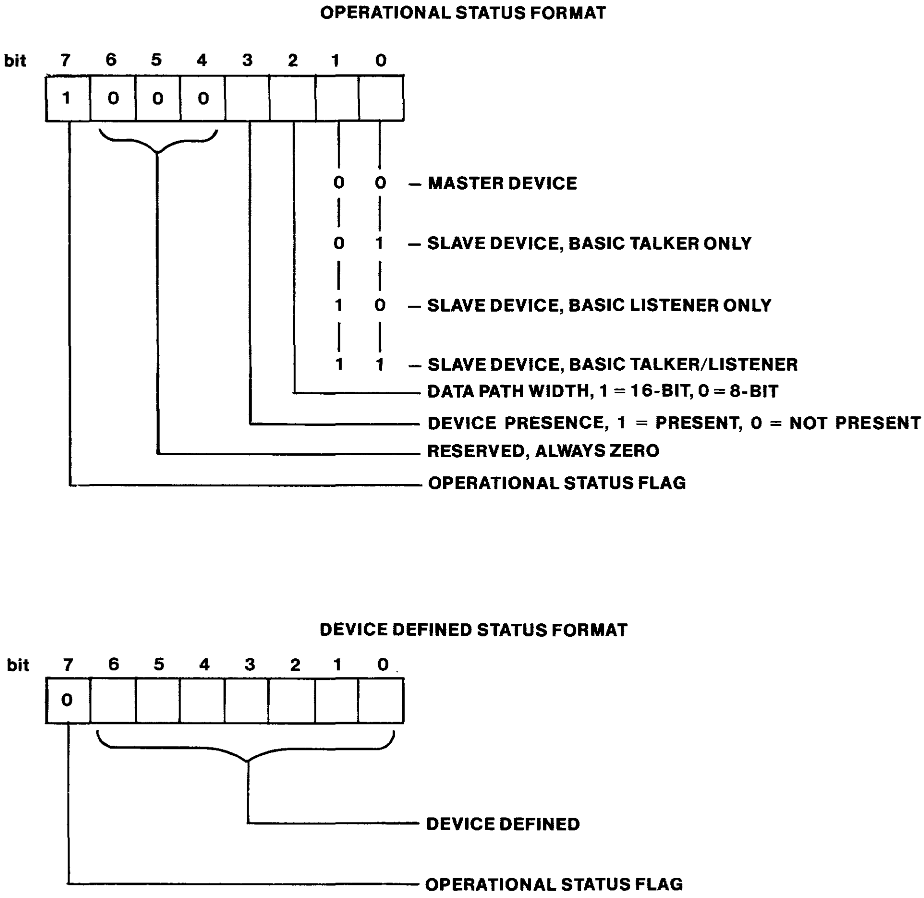
The device command register is an 8-bit register provided by a non-supervisor device that can operate in the master state (Controller). The functions or operations executed as a result of command codes stored in the device command register are device dependent. Bit 7 of the device command register determines the type of command executed. If bit 7 is set to one, the operation is local to the device and does not directly affect the Multichannel bus. Local commands are device defined and the device executes the commands off-line from the Multichannel bus. If bit 7 is set to zero, execution of the command requires use of the Multichannel bus. For Multichannel bus commands, the device must wait for two conditions: the deselect at the end of the message and the Supervisor Active line being non-asserted.

2.10.5 Register 4H - Device Parameters

The device parameters register is an 16-bit register provided by a non-supervisor device that can operate in the master state (Controller). The format for the device parameters register is the same as the format of the low order byte in Word 1 of the address mode parameter block. Refer to Figure 2-2. Using the device parameters register, the Supervisor specifies the slave state device and type of transfer operation to the Controller for a subsequent transfer operation. The use of the device parameters register is optional and the information contained is repeated in the data address register pair information.

2.10.6 Registers 5H and 6H - Data Address Pointer

The data address pointer registers are an 8- or 16-bit register pair provided by a non-supervisor device that can operate in the master state (Controller). The format for the data


OPERATIONAL STATUS FORMAT

bit 7 6 5 4 3 2 1 0
1 0 0 0
0 0 – MASTER DEVICE
0 1 – SLAVE DEVICE, BASIC TALKER ONLY
1 0 – SLAVE DEVICE, BASIC LISTENER ONLY
1 1 – SLAVE DEVICE, BASIC TALKER/LISTENER
DATA PATH WIDTH, 1 = 16-BIT, 0 = 8-BIT
DEVICE PRESENCE, 1 = PRESENT, 0 = NOT PRESENT
RESERVED, ALWAYS ZERO
OPERATIONAL STATUS FLAG

DEVICE DEFINED STATUS FORMAT

bit 7 6 5 4 3 2 1 0
0
DEVICE DEFINED
OPERATIONAL STATUS FLAG

Figure 2-6 SRQ Status Register Format

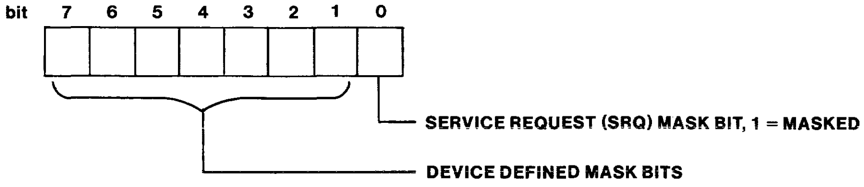


Figure 2-7 SRQ Mask Register Format

address pointer register pair is the same as the format of the address mode parameter block. Refer to Figure 2-2. Using the data address pointer registers, the Supervisor specifies the slave state device, type of transfer operation, and a memory address pointer to the Controller for a subsequent transfer operation. When the data address pointer registers are 8-bit registers, the data address pointer information is limited to the two low order bytes of the address mode parameter block.

2.10.7 Registers 7H and 8H - Block Length

The block length registers are an 8- or 16-bit register pair provided by a non-supervisor device that can operate in the master state (Controller). Using the block length registers, the Supervisor specifies the buffer size in the slave state device for a subsequent transfer operation. The format for the block length register pair is shown in Figure 2-8. The format of the block length register pair provides up to 24-bits for specifying the buffer size. Bits 15 through 8 are repeated in both registers to allow maximum access to the low order 16-bits regardless of the device data path or the register size provided.

2.10.8 Registers 9H and AH - Error Address

The error address registers are an 8- or 16-bit register pair provided by a non-supervisor device that can operate in the master state (Controller). The error address register pair is read by the Supervisor following an error reported by the controller. The information contained in the error address registers is device defined and could be the master or slave memory address where an error occurred. The format for the error address register pair is shown in Figure 2-8. The format of the error address register pair provides up to 24-bits for specifying an error address. Bits 15 through 8 are repeated in both registers to allow maximum access to the low order 16-bits regardless of the device data path or the register size provided.

2.10.9 Registers BH - Address Extension

The address extension register is an 8- or 16-bit register provided by a non-supervisor device that can operate in the master state (Controller). The information contained in the address extension registers is device defined. The address extension register can be used to extend the addressing range of 8-bit devices to 16-bits or to extend the addressing capacity of 16-bit devices beyond the 24-bits provided.

	HIGH ORDER BYTE	LOW ORDER BYTE	
bit	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	
	BITS 23 - 16	BITS 15 - 8	REGISTER 7H
	BITS 15 - 8	BITS 7 - 0	REGISTER 8H

Figure 2-8 Block Length and Error Address Register Pair Format

2.11 INTERRUPT POLLING

The Multichannel bus provides two separate interrupt lines (STO* and SRQ*) used by non-supervisor devices to request the attention of the Supervisor. This specification assigns some specific reporting to each of the interrupts. The SRQ interrupt is defined as maskable while the STO interrupt is defined as non-maskable. Otherwise, the conditions when a device would assert one of the interrupt lines is device defined. Response to an interrupt line being asserted is reserved to the Supervisor. The Supervisor services the interrupt by assuming control of the Multichannel bus and conducting an STO or an SRQ poll to determine the device asserting the interrupt. The Supervisor is not required to immediately respond to an interrupt and can choose to ignore either or both of the interrupts until any operation in progress is completed. The interrupting device must hold an interrupt asserted until it is serviced by the Supervisor.

The Supervisor initiates an interrupt poll by first determining if it must assume control of the Multichannel bus. If the Supervisor is already asserting Supervisor Active, it has control of the bus and can initiate the poll. If it is not asserting Supervisor Active, the Supervisor assumes control of the bus by asserting the Supervisor Active line. If any device is exercising control of the Address Not Data and Read Not Write lines, it must immediately release control of the lines. Upon taking control of the bus, the Supervisor outputs an deselect address mode message to deselect all devices on the bus.

The Supervisor conducts a poll by reading the appropriate register in each device on the bus in turn. The only difference between the STO and the SRQ polls is the register addressed by the Supervisor. The Supervisor addresses the STO status register (0H) when conducting an STO poll and the SRQ status register (1H) when conducting and SRQ poll. When the Supervisor reads the status register in the device asserting the interrupt line, the interrupting device must respond with a non-zero value, clear its status register, and non-assert the interrupt line. When the Supervisor reads the status register in a device that is not asserting the interrupt line, the interrupting device must respond with zero. The order in which the Supervisor polls the devices is determined by the Supervisor.

Any device that is not asserting the interrupt line must respond to the poll by the Supervisor with a zero value when its status register is read. In particular, if a device has the SRQ interrupt masked by the Supervisor and it has a pending service request, it must not respond with a non-zero value if the Supervisor conducts an SRQ poll. A device with the SRQ interrupt masked has the option of ignoring interrupt conditions while the interrupt is masked or storing the interrupt status until the Supervisor unmasks the SRQ interrupt.

CHAPTER 3
ELECTRICAL SPECIFICATION

3.1 INTRODUCTION

This section defines the electrical requirements of the Multichannel bus. The descriptions include the types of drivers and receivers required, the method, type, and location of line termination, general signal characteristics, and electrical timing.

3.2 ELECTRICAL STATE RELATIONSHIPS

The electrical state relationships used in this manual conform to the conventions used in the Multibus Specification. The Multichannel bus uses industrial grade components for all drivers and receivers. Table 3-1 relates the general industry voltage level standards for TTL and RS422/423 components to the signal line notation conventions used in this manual. The specifications in Table 3-1 assume a power source of +5 Vdc, +5 percent, referenced to logic ground. The Multichannel bus does not include provision for system power and the electrical specification assumes that all power is provided external to the Multichannel bus.

3.3 ENVIRONMENTAL REQUIREMENTS

The electrical specifications for the Multichannel bus must be met under the following environmental conditions. The specifications list the ambient temperature requirements and the non-condensing requirements for humidity.

OPERATING

Temperature	0 to 55 degrees C
Relative humidity	0 to 85 percent

Table 3-1 Notational Summary

ELECTRICAL LEVEL	VOLTAGE RANGE	
	AT RECEIVER	AT DRIVER
H, High	+2.0 TO +5.25 Vdc	+2.4 TO +5.25 Vdc
L, Low	−0.5 TO +0.8 Vdc	0 TO +0.5 Vdc
Positive Differential	+0.2 Threshold	+2.0 TO +5.25 Vdc
Negative Differential	−0.2 Threshold	−2.0 TO −5.25

3.4 DC SPECIFICATIONS

The Multichannel Bus signals are carried over twisted pair flat cable with a specified maximum length of 15 meters (approximately 50 feet). Three types of drivers are used to propagate the Multichannel bus signals: tri-state TTL, open collector TTL, and tri-state RS 422 differential. Figure 3-1 shows typical transmission line configurations for the three different types of bus driver and receiver combinations. Configuration A illustrates the typical tristate transmission line configuration. Configuration B illustrates the typical differential transmission line configuration. Configuration C illustrates the typical open collector transmission line configuration.

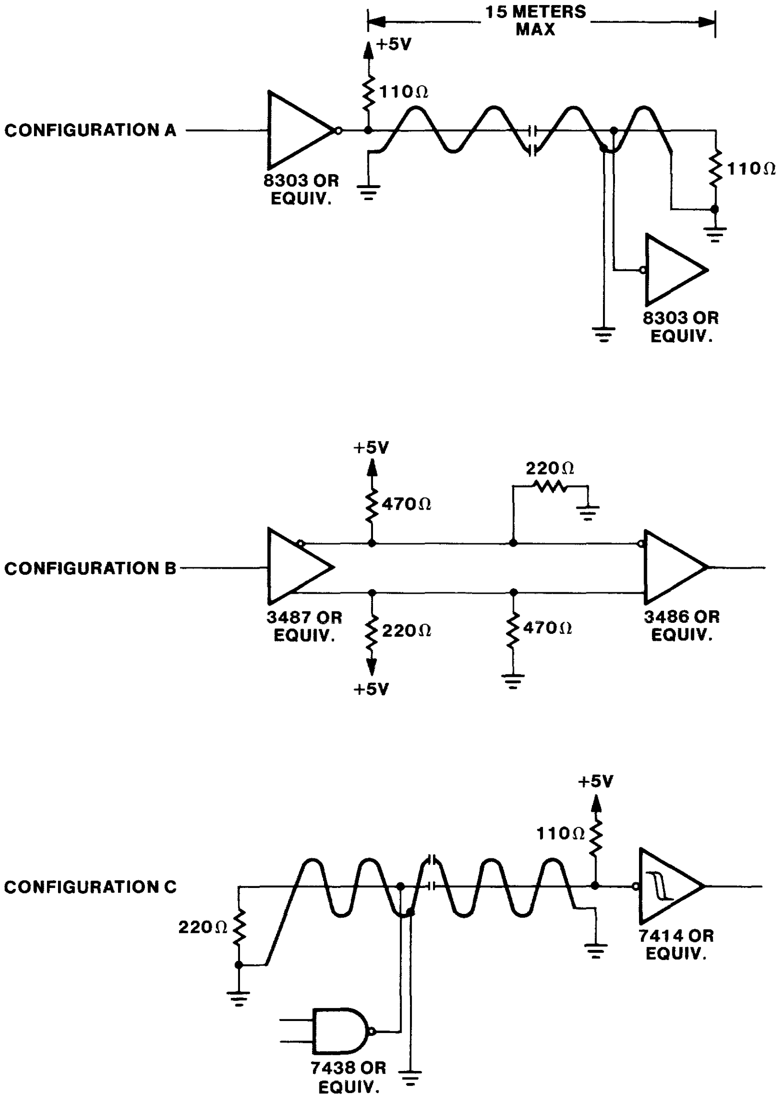


Figure 3-1 Multichannel Bus Transmission Line Configurations

Table 3-2 lists the Multichannel bus DC specifications for the signal line drivers and the receiver loads presented to the signal lines. Table 3-2 is divided into two sections, minimum driver requirements and maximum receiver requirements. The driver requirements must be met by a device when it is a talker on the Multichannel Bus. The receiver requirements must be met by a device when it is a listener on the Multichannel Bus.

Table 3-2 DC Specification

Signal Name	Driver Type	Termination (See Note)	Minimum Driver Requirements			Maximum Driver Requirements		
			High	Low	Load Cap	High	Low	Load Cap
AD15*-0*	Tri-State	110 OHMS	-5 ma	48 ma	300 pf	0.2 ma	0.8 ma	15 pf
SA*	Open Coll	110/220 OHMS	N.A.	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
RESET*	Open Coll	110/220 OHMS	300 pf	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
AACC	Open Coll	1k/2k OHMS	300 pf	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
DACC*	Open Coll	110/220 OHMS	300 pf	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
SRQ*	Open Coll	110/220 OHMS	300 pf	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
STO*	Open Coll	110/220 OHMS	300 pf	48 ma	300 pf	0.4 ma	0.6 ma	15 pf
R/$\overline{\text{W}}$	Dif, Non-Inv	220/470 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
R/$\overline{\text{W}}$/	Dif, Inv	470/220 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
A/$\overline{\text{D}}$	Dif, Non-Inv	220/470 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
A/$\overline{\text{D}}$/	Dif, Inv	470/220 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
PB*	Dif, Non-Inv	220/470 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
PB*/	Dif, Inv	470/220 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
DRDY*	Dif, Non-Inv	220/470 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf
DRDY*/	Dif, Inv	470/220 OHMS	-20 ma	40 ma	300 pf	0.5 ma	0.5 ma	15 pf

NOTES: 1. Termination provided only at the physically ends of the interconnect cable. Where the positive termination (pull up) resistance is different from the negative termination (pull down) resistance, the positive termination resistance is listed first.

2. The maximum receiver loading requirements assumes 16 load devices. The receiver on the Multichannel bus device corresponding to the driver for the line must be included as a load on the line.

3. All open collector driven lines must be received by a hysterisis device (74LS14 or equivalent) to reduce possible receiver oscillation.

4. The following are the abbreviations used for the driver types:
   - Dif, Non-Inv Differential driver, non-inverted line
   - Dif, Inv Differential driver, inverted line
   - Open Coll Open collector driver

3.5 AC SPECIFICATIONS

Table 3-3 lists the Multichannel bus timing parameters for the signal lines. The table provides a reference designator for each timing parameter, a description of the timing parameter, the minimum and maximum timing requirements, and the source device where the timing parameter must be implemented. Figures 3-2, 3-3, 3-4, and 3-5 are timing charts that illustrate the timing relationships for the Multichannel bus. Figure 3-2 shows the typical timing for an address mode bus cycle including the transition from data mode to address mode. Figure 3-3 shows the typical timing for a data mode bus cycle including the transition from address mode to data mode. Figure 3-4 shows the typical timing for a Supervisor preempting control of the Multichannel bus from a Controller. Figure 3-5 shows the typical timing for a Multichannel bus device asserting the Supervisor Take Over or Service Request interrupt lines. The timing specifications on the timing charts use the reference designators from Table 3-3.

The following are the general notes for Table 3-3.

- All times listed are nanoseconds unless otherwise noted
- All signals are shown as TTL type waveforms (for differential line pairs, the waveform applies to the TTL driver input or receiver output)
- T refers to the selected Talker for a bus cycle
- L refers to the (a) selected Listener for a bus cycle
- M refers to the selected Master for a bus cycle
- SL refers to the (a) selected Slave for a bus cycle
- SU refers to the system Supervisor

Specific Notes:

1. This timing parameter applies only when there is a message mode transition from address to data mode or from data to address mode. When the mode does not change, the Address Not Data line should be held at a constant level.

2. All timing relationships are relative to their presence at the receiving device. Thus the transfer handshake line specifications indicate a theoretical minimum handshake elapsed time of 0 ns. However, between each step of the handshake, there is a propagation delay induced by the physical distance separating the talker and listener devices.

3. These parameters apply in messages where the address mode talker is the listener for the data mode portion of the message.

4. These parameters apply when a transfer specific error such as incorrect parity is detected by a listener during a bus cycle. All other assertions of the interrupt lines can be asynchronous to the bus operation.

5. These parameters apply during the bus cycle when the STO or SRQ status register of the device asserting the interrupt line is read during an STO or SRQ poll.

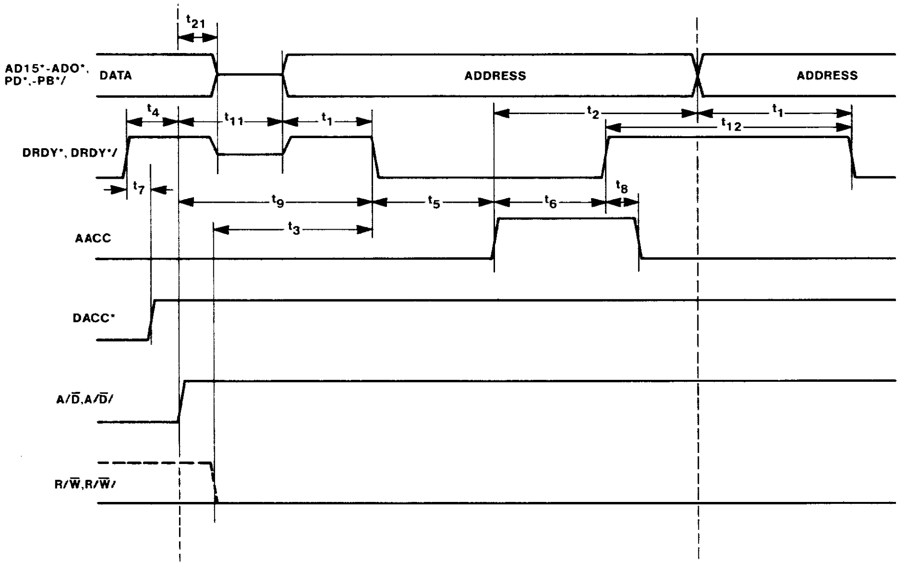
6. The minimum Reset pulse width is 5 milliseconds.

7. After the maximum time has elapsed, the trailing edge of DRDY* may occur independent of AACC or DACC*.

8. During SRQ register read transaction.

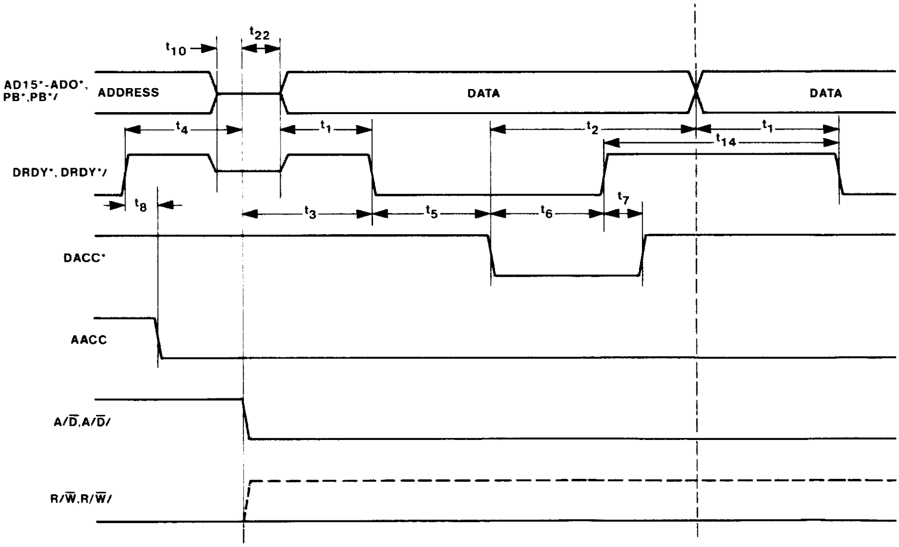
Table 3-3 MULTICHANNEL™ Bus Timing Parameters

REF	PARAMETER DESCRIPTION	TIMING (ns)		SOURCE	NOTE
		MIN	MAX		
t1	ADDRESS/DATA LINE SETUP TO LEADING EDGE OF DRDY*	60		T	
t2	ADDRESS/DATA LINE HOLD AFTER LEADING EDGE OF AACC OR DACC*	40		T	
t3	DATA MODE (A/$\overline{D}$ LOW) AND R/$\overline{W}$ SETUP TO LEADING EDGE OF DRDY*	60		T	
t4	A/$\overline{D}$, R/$\overline{W}$ HOLD AFTER TRAILING EDGE OF DRDY*	50		M	1
t5	LEADING EDGE OF DRDY* TO LEADING EDGE OF AACC OR DACC*	0	4ms 1ms	L	2, 7 2, 8
t6	LEADING EDGE OF AACC OR DACC* TO TRAILING EDGE OF DRDY*	0		T	2
t7	TRAILING EDGE OF DRDY* TO TRAILING EDGE OF DACC*	0		L	2
t8	TRAILING EDGE OF DRDY* TO TRAILING EDGE OF AACC	0	75	L	2
t9	ADDRESS MODE (A/$\overline{D}$ HIGH) TO LEADING EDGE OF DRDY*	200		M-T	
t10	AD15* - AD0*, DRDY* IN HIGH IMPEDANCE STATE TO A/$\overline{D}$ LOW, R/$\overline{W}$ HIGH		0	M	3
t11	A/$\overline{D}$ HIGH, R/MW LOW TO AD15* - AD0*, DRDY* OUT OF HIGH IMPEDANCE STATE	150		M	3
t12	TRAILING EDGE OF DRDY* TO LEADING EDGE OF DRDY* (A/$\overline{D}$ HIGH)	250		M-T	
t13	AD15* - AD0*, DRDY* A/$\overline{D}$, R/$\overline{W}$ IN HIGH IMPEDANCE STATE TO SA* HIGH		0	SU	
t14	TRAILING EDGE OF DRDY* TO LEADING EDGE OF DRDY* (A/$\overline{D}$ LOW)	100		SL-T	
t15	SA* LOW TO AD15* - AD0*, DRDY*, A/$\overline{D}$, R/$\overline{W}$ OUT OF HIGH IMPEDANCE STATE	175		SU	
t16	LEADING EDGE OF DRDY* TO LEADING EDGE OF STO*	0		L	4
t17	LEADING EDGE OF AACC OR DACC* TO LEADING EDGE OF STO*		0	L	4
t18	DATA MODE (A/$\overline{D}$ LOW) TO TRAILING EDGE OF STO* OR SRQ*	0		SL	5
t19	TRAILING EDGE OF STO* OR SRQ* TO TRAILING EDGE OF DRDY*		0	SL	5
t20	RESET PULSE WIDTH	5ms		SU	6
t21	A/$\overline{D}$ HIGH TO AD15* - AD0*, DRDY* IN HIGH IMPEDANCE STATE		75	SL-T	3
t22	A/$\overline{D}$ LOW TO AD15* - AD0*, DRDY* OUT OF HIGH IMPEDANCE STATE	0		SL-T	3
t23	SA* HIGH TO AD15* - AD0*, DRDY*, A/$\overline{D}$, R/$\overline{W}$ OUT OF HIGH IMPEDANCE STATE	0		M	
t24	SA* LOW TO AD15* - AD0*, DRDY*, A/$\overline{D}$, R/$\overline{W}$ IN HIGH IMPEDANCE STATE		75	M	



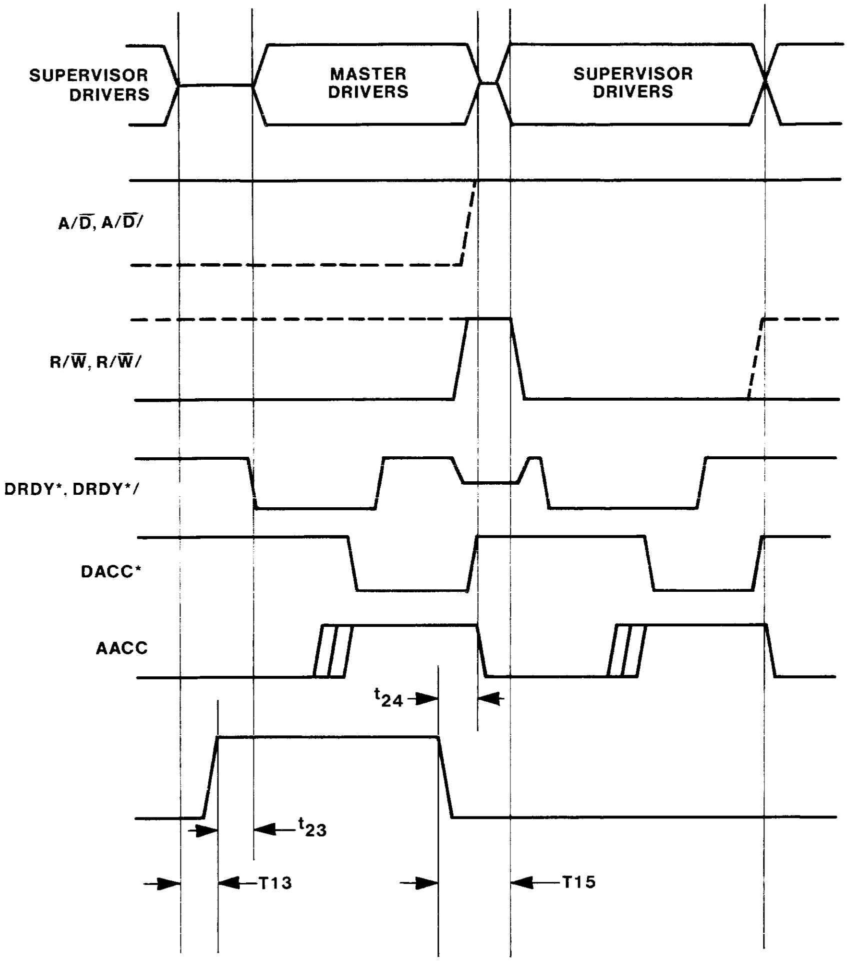
NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTES POSITIVE OR NEGATIVE DIFFERENTIAL

Figure 3-2 Address Mode Bus Cycle Timing Chart



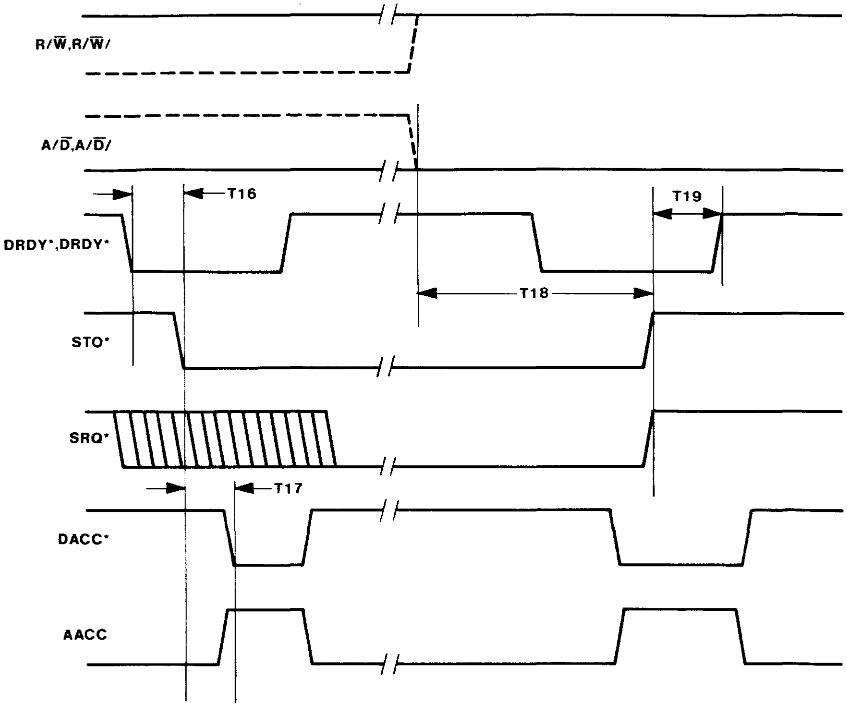
NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTES POSITIVE OR NEGATIVE DIFFERENTIAL

Figure 3-3 Data Mode Bus Cycle Timing Chart



NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTES POSITIVE OR NEGATIVE DIFFERENTIAL

Figure 3-4 Control Transfer Timing Chart



NOTE: FOR DIFFERENTIAL LINE PAIRS, LEVEL DENOTES POSITIVE OR NEGATIVE DIFFERENTIAL

Figure 3-5 Supervisor Interrupt Timing Chart

3.6 TYPICAL SIGNAL WAVEFORMS

Figures 3-6 through 3-9 illustrate the typical Multichannel bus signal waveforms. In most systems, the observed waveforms should not deviate significantly from the typical waveforms shown.

Figure 3-6 illustrates the typical waveform for the Address/Data lines, AD15* to AD0*. The waveform shown illustrates the transitions between the driven state and the high impedance state. When a burst of data transfers are made with the same device as the talker, the high impedance step at (2) is eliminated. The level at (2) represents the standard termination level for the address/data lines.

Figure 3-7 illustrates the typical waveform for the Address Accept line. The Address Accept line is driven by open collector devices and is non-asserted when electrically low. The waveform on the Address Accept line is typically a short pulse that rises toward the termination level on the bus of 3.33 Vdc. The Address Accept signal is typically recognized by the address mode talker before it fully rises to the termination level.

Figure 3-8 illustrates the typical waveform for the negative true lines driven by open collector devices. The non-asserted high level for these lines is the termination level for these lines.

Figure 3-9 illustrates the typical waveforms for the inverted and non-inverted lines driven by differential devices. The termination levels (2) are set for the line pairs to assure a positive differential when the drivers are in the high impedance state.

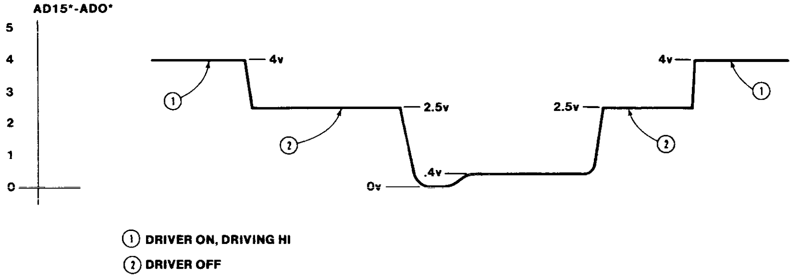
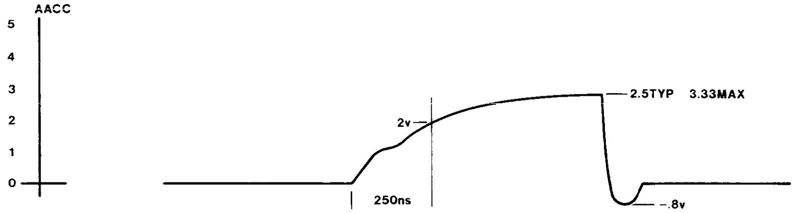


Figure 3-6 Address/Data Line Typical Waveform



ACTUAL PULSE WIDTH AND MAXIMUM SIGNAL LEVEL WILL BE DEPENDENT ON THE TIMING PARAMETERS t_6 AND t_8

Figure 3-7 Address Accept Line Typical Waveform

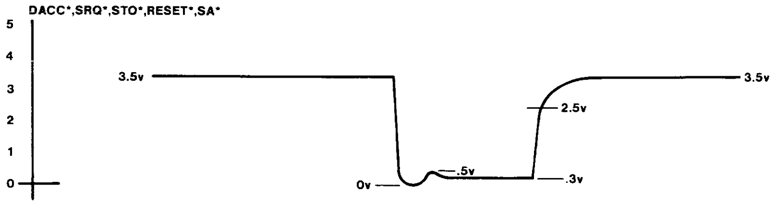


Figure 3-8 Negative True Open Collector Driven Lines Typical Waveform

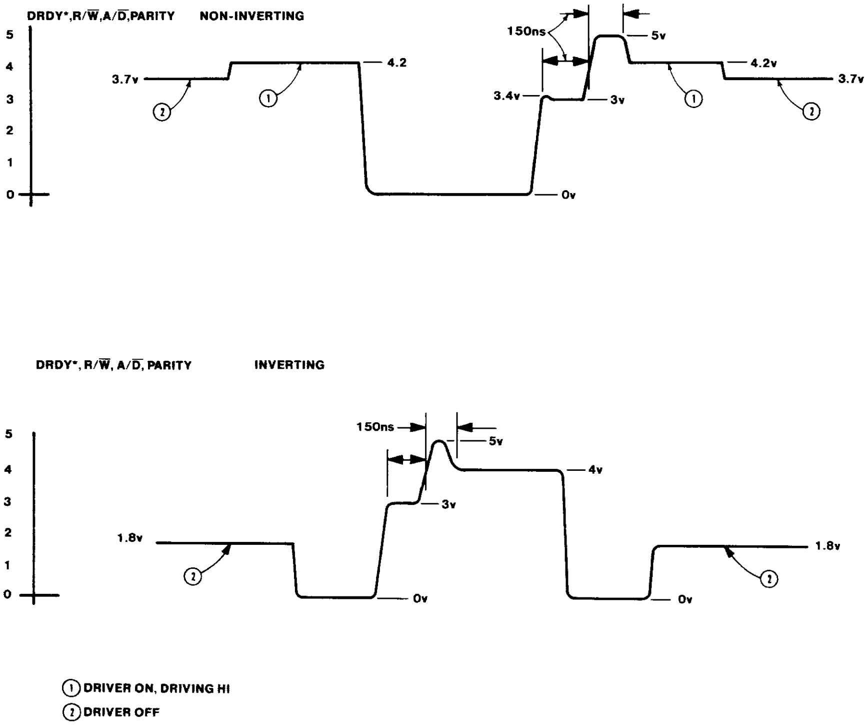


Figure 3-9 Differential Line Pairs Typical Waveforms

CHAPTER 4
MECHANICAL SPECIFICATION

4.1 INTRODUCTION

This section defines the physical and mechanical requirements that must be considered when designing Multichannel bus compatible printed circuit boards or when implementing the Multichannel bus in a system. The descriptions include the form factor requirements specific to the Multichannel bus, the interconnect cable requirements, type of connectors, and the location of connectors on Multibus-compatible printed circuit boards. The Multichannel bus Mechanical Specifications generally are limited to those specifications different from or in addition to the Multibus interface mechanical specifications.

4.2 MULTICHANNEL™ BUS FORM FACTOR

The Multichannel bus is intended to function as a part of the Multibus interface system. The printed circuit form factor information provided in this section is for the Multichannel bus implemented on a Multibus printed circuit board. The Multibus specification requirements for board-to-board spacing, board thickness, component lead length, and component height above the board remain the same as in the Multibus specification. Refer to the INTEL MULTIBUS SPECIFICATION for details on the general Multibus interface mechanical specifications. The Multichannel bus need not be implemented on a Multibus printed circuit board. In any non-Multibus board implementation, the connector and pin configuration must be retained for Multichannel bus compatibility. The Multichannel bus does not provide any power connections. For Multibus interface based systems, it is assumed that power is supplied over the Multibus P1 connector.

IV

When a Multichannel bus compatible device is implemented on a Multibus board, the male plug for connection to the Multichannel bus is located on the outer edge of the Multibus board. The Multichannel bus male plug can be located anywhere along the outer edge of the board; however, sufficient room must be allowed from the side of the board for operation of the Multibus board extractors and the retaining hooks on the sides of the plug.

4.3 MULTICHANNEL™ BUS CABLE

The Multichannel bus does not use a rigid backplane to interconnect the Multichannel bus compatible boards but rather an interconnect cable assembly. The specification further simplifies system implementation by using ribbon cable and mass terminated connectors to make up the required interconnect cables. Table 4-1 lists recommended vendors that produce ribbon cable and connectors compatible with the Multichannel bus.

Table 4-1 Cable And Receptacle Vendors

MULTICHANNEL™ BUS COMPATIBLE CABLE			
VENDOR	**TYPE**	**VENDOR NUMBER**	**CONDUCTORS**
Belden	Plain Flat Ribbon	9L28060	60
Belden	Twisted-Pair Ribbon	9V28060	60
Belden	Insulated Flat Ribbon	9L28260	60
Spectrastrip	Plain Flat Ribbon	455-240-60	60
Spectrastrip	Twisted-Pair Ribbon	455-248-60	60
Spectrastrip	Insulated Flat Ribbon	151-2830-060	60
MULTICHANNEL™ BUS COMPATIBLE CONNECTORS			
VENDOR	**TYPE**	**VENDOR NUMBER**	**PINS**
Berg	Male, Header	65823-103	60
Berg	Female, Mass-Terminated	65949-960	60
3M	Male, Header	3372-1302	60
3M	Female, Mass-Terminated	3334-6000	60

4.3.1 Cable Specifications

The Multichannel bus interconnect cable uses 60 conductor ribbon cable for interconnecting the Multichannel bus compatible devices. The maximum length for the interconnect cable is 15 meters (approximately 50 feet). The following are the general electrical and insulation specifications for Multichannel bus compatible cable:

PHYSICAL PROPERTIES

Conductors	28 AWG, 7/36 strand, tinned copper
Conductor insulation	0.010 inch wall, nominal
Conductor spacing, twisted pair	0.10 inch, nominal
Conductor spacing, flat	0.050 inch, ±10%
Cable thickness, flat	0.042 inch, nominal
Temperature rating	80 degrees celsius

ELECTRICAL PROPERTIES

Impedance (nominal)	105 ohms ±10%
Propagation velocity (nominal)	1.7 ns/ft
Capacitance (nominal)	22 pf/ft

INSULATION REQUIREMENTS

Voltage rating (minimum)	100 Vdc
Insulation resistance (minimum)	$1x10^{10}$ ohms

4.3.2 MULTICHANNEL™ Bus Connectors

The Multichannel bus requires use of mass terminating, 60 conductor, flat ribbon cable, with female receptacles to attach the interconnecting cable to the Multichannel bus compatible boards. The male and female connectors should provide a locating key to limit the possibility reversing the plug and receptacle.

4.3.2 MULTICHANNEL™ Bus Connectors

The Multichannel bus requires use of mass terminating, 60 conductor, flat ribbon cable, female receptacles to attach the interconnecting cable to the Multichannel bus compatible boards. The male and female connectors should provide a locating key to limit the possibility reversing the plug and receptacle.

4.3.3 Cable Assembly

A Multichannel bus cable assembly can have from two to sixteen female receptacles mass terminated to the flat ribbon cable. The spacing between the female receptacles assembled to the cable varies with the system layout. The compatible cables listed in Table 4-1 represent three different cable configurations: flat ribbon, flat ribbon with shield, and twisted pair ribbon. Each cable type presents different considerations for cable use and assembly. The plain, flat ribbon cable offers the greatest flexibility and is the easiest to use because the receptacles can be placed anywhere along the cable. For relatively benign (noise free) environments, the plain flat ribbon cable would be a good choice. For more hostile environments, either the shielded flat ribbon or the twisted pair ribbon cable should be used. These cable types have receptacle connection restrictions that must be considered when choosing the cable for a specific application. With the shielded flat ribbon cable, it is very difficult to install the receptacle anywhere except at the ends of the cable. The twisted pair cable does allow easy attachment of more than two devices; but, connection must be made at one of the locations where the cable reverts to plain flat ribbon for the attachment of mass terminated receptacles. The typical twisted pair ribbon cable has 18 inch sections of twisted pair separated by 2.5 inch sections of flat ribbon. Regardless to the cable type used, the cable length should be kept as short as possible without excess slack cable between devices.

4.4 PIN NUMBERING CONVENTION

The Multichannel bus connectors are configured with two parallel rows of pins. There are 30 pins in each row. The pin numbering convention assigns the odd numbered pins (1 through 59) to one row and the even numbered pins (2 through 60) to the other row. When the male plug is assembled on a printed circuit board, the row of odd numbered pins is adjacent to the printed circuit board. Pin 1 is located so that when you face the male plug with the odd row of pins on the top, pin 1 is at the upper right corner of the plug. Pin 2 is located immediately under pin one and the pins are then numbered in ascending order from right to left. Figure 4-1 illustrates the Multichannel bus pin numbering convention.

4.5 MULTICHANNEL BUS PIN ASSIGNMENTS

Table 4-2 lists the Multichannel bus pin assignments for the 60-pin Multichannel bus connector. The pin assignments are arranged so that mass termination of the Multichannel

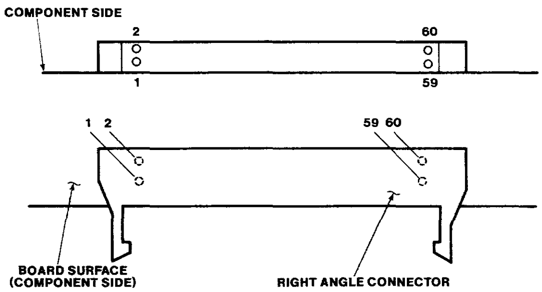


Figure 4-1 MULTICHANNEL™ Bus Male Plug Pin Numbering Convention

Table 4-2 MULTICHANNEL™ Bus Pin Assignments

LOWER ROW			UPPER ROW		
PIN	MNEMONIC	SIGNAL NAME	PIN	MNEMONIC	SIGNAL NAME
1	GND	Ground	2	AD0*	Address/Date Line 0
3	GND	Ground	4	AD1*	Address/Data Line 1
5	GND	Ground	6	AD2*	Address/Data Line 2
7	GND	Ground	8	AD3*	Address/Data Line 3
9	GND	Ground	10	AD4*	Address/Data Line 4
11	GND	Ground	12	AD5*	Address/Data Line 5
13	GND	Ground	14	AD6*	Address/Data Line 6
15	GND	Ground	16	AD7*	Address/Data Line 7
17	GND	Ground	18	AD8*	Address/Data Line 8
19	GND	Ground	20	AD9*	Address/Data Line 9
21	GND	Ground	22	AD10*	Address/Data Line 10
23	GND	Ground	24	AD11*	Address/Data Line 11
25	GND	Ground	26	AD12*	Address/Data Line 12
27	GND	Ground	28	AD13*	Address/Data Line 13
29	GND	Ground	30	AD14*	Address/Data Line 14
31	GND	Ground	32	AD15*	Address/Data Line 15
33	GND	Ground	34	RESET*	Reset
35	GND	Ground	36	AACC	Address Mode Accept
37	GND	Ground	38	SRQ*	Service Request
39	GND	Ground	40	STO*	Supervisor Take Over
41	GND	Ground	42	DACC*	Data Mode Accept
43	GND	Ground	44	SA*	Supervisor Active
45	PB*/	PARITY BIT (INV.)	46	PB*	PARITY BIT
47	R/$\overline{W}$/	Read-Not-Write (Inv.)	48	R/$\overline{W}$	Read-Not-Write
49	A/$\overline{D}$/	Address-Not-Data (Inv.)	50	A/$\overline{D}$	Address-Not-Data
51	DRDY*/	Data Ready (Inv.)	52	DRDY*	Data Ready
53	RES	Reserved	54	RES	Reserved
55	RES	Reserved	56	RES	Reserved
57	RES	Reserved	58	RES	Reserved
59	RES	Reserved	60	RES	Reserved

bus connector to the twisted pair ribbon cable places a ground line in the pair with each non-differential signal line (the non-inverted and inverted differential lines are paired together). On the flat ribbon cable, mass termination alternates the ground line and the signal lines.

4.6 COMPONENT LAYOUT CONSIDERATIONS

To maintain the electrical signal quality of the Multichannel bus signals, care must be taken to avoid noise on the bus lines. There are several factors relative to board layout that directly affect signal quality.

The Multichannel bus drivers, receivers, and transceivers directly connected to a Multichannel bus signal line should be located close to the Multichannel bus connector. The printed circuit board trace connecting a driver or receiver pin to the corresponding Multichannel bus connector pin should not exceed 5 cm (2 in) in length. When calculating the 5 cm trace length, include any trace required to connect the driver to the termination resistor when the board provides bus termination capabilities.

The Multichannel bus interface components (integrated circuits containing drivers, receivers, or transceivers directly connected to a Multichannel bus signal line) must have adequate connection to the signal ground of the Multichannel bus. On boards with a ground layer, the layer should be solid under the Multichannel bus interface components with the ground connections made directly to the ground layer. On boards without a ground layer, the Multichannel bus interface components and the Multichannel bus should be supplied with ground routed directly from the common board ground, that is, the point where ground is supplied to the board. This Multichannel bus interface component ground should not be shared with any other logic components on the board. The printed circuit trace connecting the common board ground to the area of the Multichannel bus interface should be as direct as possible and a minimum of 2.54 mm (0.10 in) wide (assuming 1 oz copper plate). Short ground traces connecting the Multichannel bus interface components to the main ground trace can be 1.27 mm (0.05 in) wide (assuming 1 oz copper plate) and be directly connected to the main signal ground for the board. The ground trace supplying the Multichannel bus and the interface components should be decoupled by 0.1 uf bypass capacitors between ground and +5 Vdc. Two bypass capacitors should be located at the board connection point to system ground with one additional capacitor approximately every 5 cm (2.0 in) along the ground trace. Two additional bypass capacitors should be located along the ground trace that actually supplies the interface components and the Multichannel bus ground lines.

To avoid unacceptable signal coupling between Multichannel bus signal lines, care must be taken when assigning signal lines to the Multichannel bus interface components. Avoid signal line assignments where one or two signal lines assigned to a Multichannel bus interface component are required to remain stable while the remaining lines change state concurrently. Refer to the Design Guidelines section for an example of signal line assignments to the Multichannel bus interface components.

4.7 CABLE TERMINATION REQUIREMENTS

All implementations of the Multichannel bus must have both pull-up and pull down termination. The termination must be located at the physical ends of the cable. The termination scheme for the Multichannel bus requires pull-up termination to +5 Vdc at one end of the cable and pull-down termination to ground at the other end of the cable. Figure 4-2 illustrates the termination scheme for the Multichannel bus. Any device attached to the Multichannel bus that is not located at the physical end of the cable must not provide any additional termination. Because the Multichannel bus does not include any +5 Vdc voltage lines, the pull-up termination must be provided by a Multichannel bus compatible device. The pull-down termination can be provided using a termination module that attaches to the end plug on the Multichannel bus with ground supplied through the Multichannel bus. The design guidelines section provides an example of a termination implementation that allows jumper selection of no termination, pull-up termination, or pull-down termination.

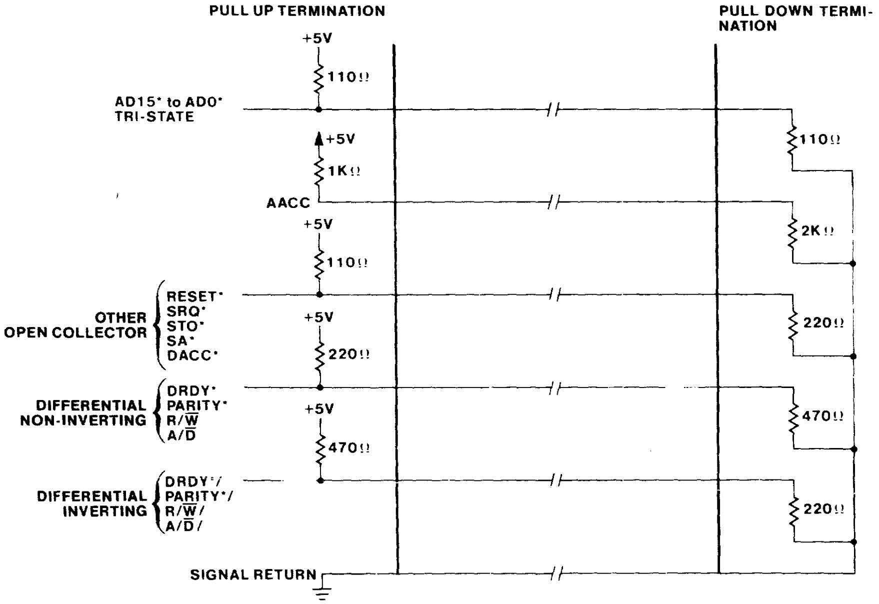


Figure 4-2 MULTICHANNEL™ Bus Termination Scheme

bus connector to the twisted pair ribbon cable places a ground line in the pair with each non-differential signal line (the non-inverted and inverted differential lines are paired together). On the flat ribbon cable, mass termination alternates the ground line and the signal lines.

4.6 COMPONENT LAYOUT CONSIDERATIONS

To maintain the electrical signal quality of the Multichannel bus signals, care must be taken to avoid noise on the bus lines. There are several factors relative to board layout that directly affect signal quality.

The Multichannel bus drivers, receivers, and transceivers directly connected to a Multichannel bus signal line should be located close to the Multichannel bus connector. The printed circuit board trace connecting a driver or receiver pin to the corresponding Multichannel bus connector pin should not exceed 5 cm (2 in) in length. When calculating the 5 cm trace length, include any trace required to connect the driver to the termination resistor when the board provides bus termination capabilities.

The Multichannel bus interface components (integrated circuits containing drivers, receivers, or transceivers directly connected to a Multichannel bus signal line) must have adequate connection to the signal ground of the Multichannel bus. On boards with a ground layer, the layer should be solid under the Multichannel bus interface components with the ground connections made directly to the ground layer. On boards without a ground layer, the Multichannel bus interface components and the Multichannel bus should be supplied with ground routed directly from the common board ground, that is, the point where ground is supplied to the board. This Multichannel bus interface component ground should not be shared with any other logic components on the board. The printed circuit trace connecting the common board ground to the area of the Multichannel bus interface should be as direct as possible and a minimum of 2.54 mm (0.10 in) wide (assuming 1 oz copper plate). Short ground traces connecting the Multichannel bus interface components to the main ground trace can be 1.27 mm (0.05 in) wide (assuming 1 oz copper plate) and be directly connected to the main signal ground for the board. The ground trace supplying the Multichannel bus and the interface components should be decoupled by 0.1 uf bypass capacitors between ground and +5 Vdc. Two bypass capacitors should be located at the board connection point to system ground with one additional capacitor approximately every 5 cm (2.0 in) along the ground trace. Two additional bypass capacitors should be located along the ground trace that actually supplies the interface components and the Multichannel bus ground lines.

To avoid unacceptable signal coupling between Multichannel bus signal lines, care must be taken when assigning signal lines to the Multichannel bus interface components. Avoid signal line assignments where one or two signal lines assigned to a Multichannel bus interface component are required to remain stable while the remaining lines change state concurrently. Refer to the Design Guidelines section for an example of signal line assignments to the Multichannel bus interface components.

4.7 CABLE TERMINATION REQUIREMENTS

All implementations of the Multichannel bus must have both pull-up and pull down termination. The termination must be located at the physical ends of the cable. The termination scheme for the Multichannel bus requires pull-up termination to +5 Vdc at one end of the cable and pull-down termination to ground at the other end of the cable. Figure 4-2 illustrates the termination scheme for the Multichannel bus. Any device attached to the Multichannel bus that is not located at the physical end of the cable must not provide any additional termination. Because the Multichannel bus does not include any +5 Vdc voltage lines, the pull-up termination must be provided by a Multichannel bus compatible device. The pull-down termination can be provided using a termination module that attaches to the end plug on the Multichannel bus with ground supplied through the Multichannel bus. The design guidelines section provides an example of a termination implementation that allows jumper selection of no termination, pull-up termination, or pull-down termination.

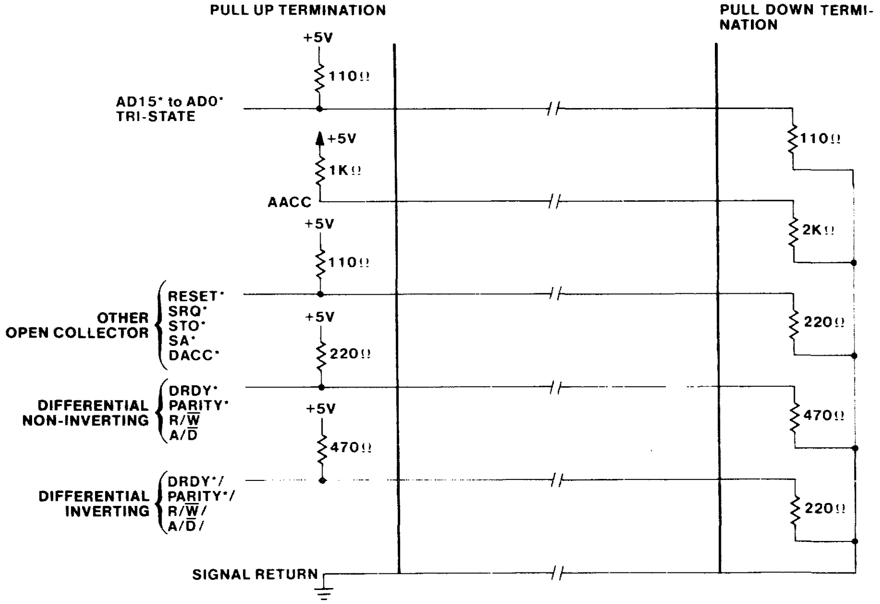


Figure 4-2 MULTICHANNEL™ Bus Termination Scheme

CHAPTER 5 DESIGN GUIDELINES AND SYSTEM APPLICATIONS

5.1 INTRODUCTION

Because the basic bus cycle of the the Multichannel bus is relatively simple, the physical implementation of the Multichannel bus interface is generally straight-forward. The order of events within a single bus cycle does not demand a high degree of decision making and the Multichannel bus interface can be directly implemented without an onboard microcontroller in a Basic device. For this type of Basic device, it would depend on the Supervisor or Controller to provide the message protocol. In general, Supervisors and Controllers would use an onboard controller to provide the required decision making to observe the Multichannel bus message protocol.

This section provides representative circuit examples of design solutions for specific signal line implementations. The circuit examples shown are not intended to provide a complete bus interface nor are the examples shown the most optimum solution. The circuits illustrate some of the considerations involved in the various signal line implementations.

5.2 DEVICE SELECT CIRCUIT

The device select circuit design example is useful for devices that make a direct implementation of the Multichannel bus. For devices that have an onboard microcontroller, this particular operation would typically be performed by the microcontroller software. The circuit example is designed to scan every address mode bus cycle for the device number assigned to the device. When found, the circuit generates a select signal to the control logic on the device. The circuit example does not show any circuitry for issuing the required Address Mode Accept signals to the address mode talker device. Figure 5-1 illustrates the device select circuit.

5.2.1 Circuit Function

The device select circuit compares the device number field of word 1 of the address mode parameter block to the number selected on the 4-bit switch SW1. If the numbers match, the circuit asserts Active High to signal the device that it is selected and change the operation of the device select circuit. Once selected, the circuit automatically switches the comparison value and will respond only to a device number of (0FH), the deselect device number. When the Active signal is non-asserted or Low, the switch elements in switch SW1 are defined as one when the switch is open and zero when the switch element is closed. However, when the Active signal is asserted to select the device, the source for the SW1 switch elements goes High. With the Active signal High, both open and closed switch elements input a High value to the exclusive OR gates providing a comparison value of 0FH.

The circuit example also illustrates the type of circuitry required to assure that the device select circuit remains synchronized to the message protocol of the address mode transfers.

5.2.2 Circuit Operation

The circuit is initially synchronized anytime the Multichannel bus Reset is asserted. When Reset is asserted, the synchronizing flip-flops, F1 and F2, and one-shot, OS1, are reset. The Active signal from F2 is Low and the First signal from F1 is High indicating the device is not selected and that it is waiting for the first word of an address mode message.

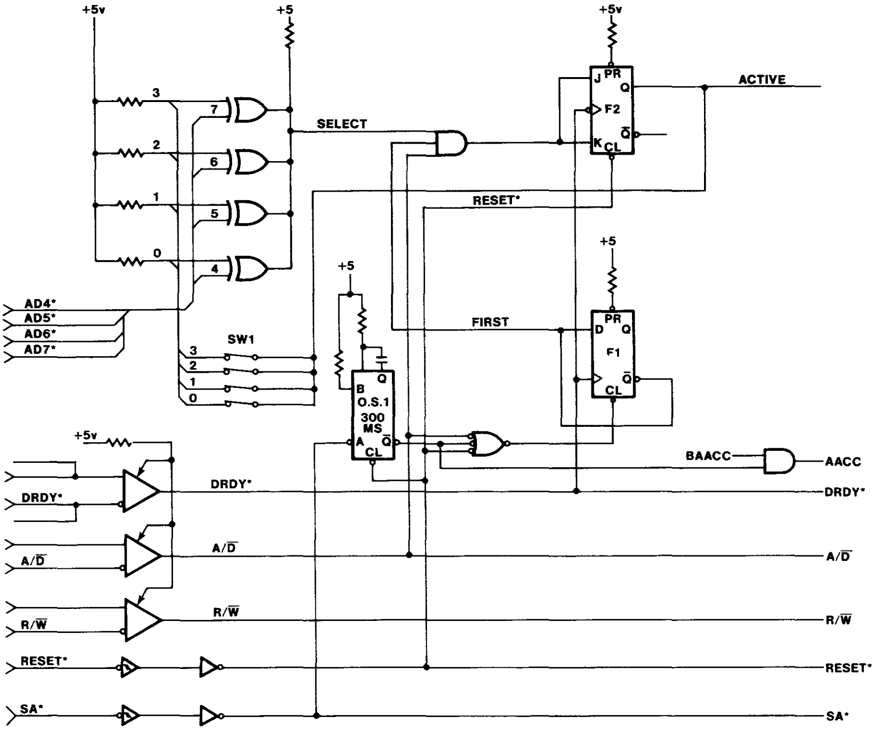


Figure 5-1 Device Select Circuit

Following Reset, the device select circuit is in the passive state and continually monitoring the bus lines. F2 is prepared for selection anytime A/D̄ is asserted (High) and the signal configuration on AD4* through AD7* matches the setting on SW1. At the falling edge of DRDY*, F2 sets to select the device. At the rising edge of the DRDY* signal, F1 sets making the First signal Low. With First Low, the inputs to F2 are held Low during the second address mode bus cycle assuring that a false deselect is not detected. At the rising edge of the DRDY* signal for the second address mode bus cycle, F1 resets making the First signal High in preparation for the next address mode message.

Additional synchronization of the device select circuit is provided each time there is a data message over the Multichannel bus. Each time A/D̄ is non-asserted (Low), F1 is held reset and First is High. Thus, regardless of the number of data mode transfers, the device select circuit is prepared to accept the next address mode bus cycle as the first word of an address message.

The one shot, OS1, provides synchronization anytime the Supervisor preempts control of the Multichannel bus by asserting Supervisor Active. Because the Supervisor can assert Supervisor Active anytime, including between the first and second transfer of an address mode message, this synchronization is required. At the falling edge of SA*, OS1 is triggered and the output of OS1 resets F1. The device select circuit is now prepared to accept the first word of the deselect address mode message that the Supervisor must output as a result of assuming control of the Multichannel bus. The 300 ns duration of OS1 allows the bus lines to stabilize before completing the first address mode bus cycle. During this time, F2 can be reset if the deselect device number is detected but the device is kept from asserting AACC until OS1 has timed out. This assures that DRDY*, if asserted, is held asserted until after OS1 times out.

5.3 PARITY CIRCUIT EXAMPLE

The parity circuit example illustrates the type of implementation that assures that both the signal timing and protocol requirements are met. Figure 5-2 illustrates a general parity generation and checking circuit. Figure 5-3 illustrates a minimum implementation of the Supervisor Take Over poll response where the only interrupt capability required is the parity error check.

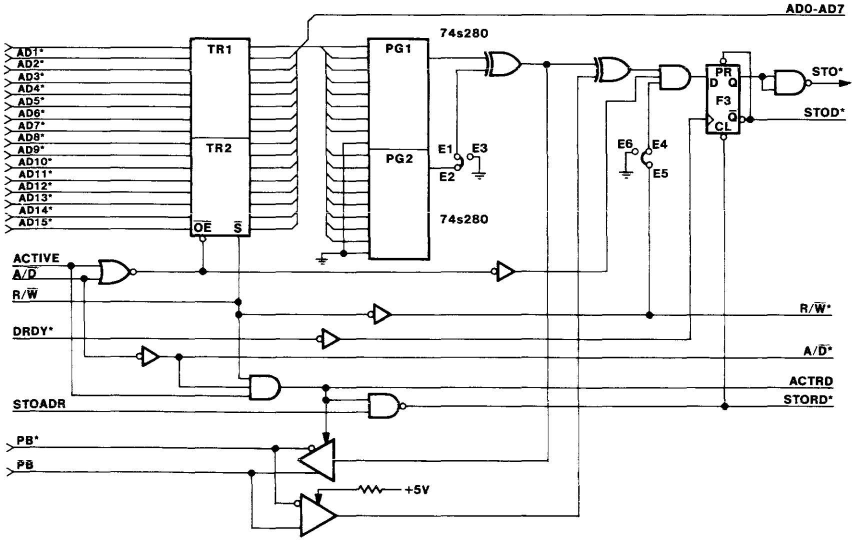


Figure 5-2 Parity Generation and Check Circuit

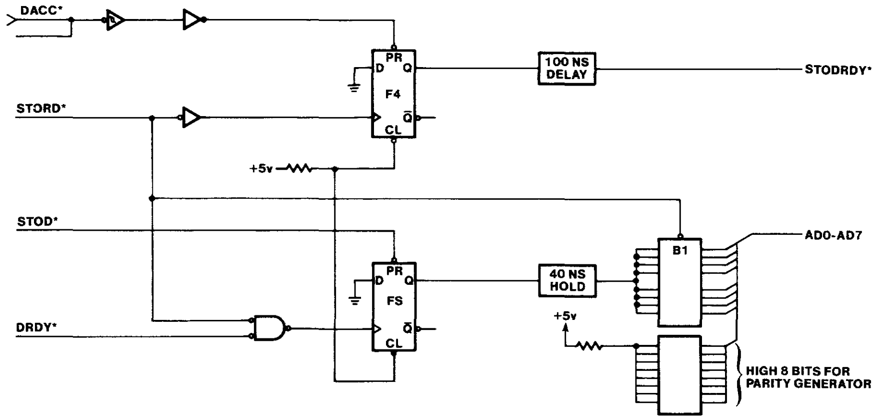


Figure 5-3 STO Poll Response Circuit

5.3.1 Parity Generation and Checking Circuit

The circuit example illustrated in Figure 5-2 is active anytime the device is the talker or listener for a bus cycle. The transceivers, TR1 and TR2, are enabled for all address mode bus cycles and any data mode bus cycles when Active is asserted by the device select circuit. For all other bus cycles, TR1 and TR2 are in the high-impedance state and the device is disconnected from the address/data lines.

When the device is a listener, the parity generator circuits develop a parity bit from the data received on the address/data lines. The parity generator circuits, PG1 and PG2, operate on the device side of the address/data transceivers. This parity bit is then compared with the state of the PB* differential line pair. The generation and checking of parity is performed prior to the talker asserting DRDY*. If the locally generated parity does not compare with the parity bit received from the talker, F3 will set at the leading edge of the DRDY* signal. With F3 set, the device asserts the STO* line to signal an error condition to the Supervisor. Operation continues uninterrupted and F3 remains set until the Supervisor services the STO interrupt by conduction an STO poll.

When the device is the talker, the parity generator circuits develop a parity bit from the data provided by the device. The parity bit developed is used to control the state of the PB* differential line pair.

The jumper set, E1, E2, and E3, allows the parity generator to be configured for either 8-bit or 16-bit operation. When jumpered as shown in Figure 5-2, the parity generator circuits are configured for 16-bit operation and the outputs of the PG1 and PG2 are exclusive ORed together to produce a single parity bit. When E1 is connected to E3, the parity generator output for AD8* through AD15* is ignored. The input to the exclusive OR gate is grounded and the exclusive OR gate output follows the input from PG1.

The jumper set, E4, E5, and E6, allows the device to operate in both parity checking and non-checking mode. When jumpered as shown in Figure 5-2, the check circuit is configured to assert the STO interrupt anytime a parity error is detected and the device is a selected listener for the bus cycle. When E4 is connected to E6, the D input to the check flip-flop is held low and the parity check circuit is disabled.

5.3.2 STO Poll Response Circuit

The circuit example illustrated in Figure 5-3 provides for a response anytime the Supervisor conducts an STO poll. As shown in the circuit illustration, the device responds to an STO poll with a 00H if the device is not asserting STO* and 0FFH if it is asserting STO*.

Flip-flop F4 is used to assert the DRDY* differential line pair. Flip-flop F4 and buffer B1 are used to provide the zero or non-zero response to the STO poll. F4 is normally set and STODRDY* signal is high. The signal STORD* is developed on Figure 5-3 and combines the holdover signal STOADR with the data mode portion of the STO poll message. STOADR is asserted when the device decodes its device number and a read command for register 0H. When STORD* is first asserted, the buffer B1 is enabled and F4 is reset. The output from B1 is immediately available to the parity generation circuits and the Multichannel bus address/data transceivers. The low output from F4 (STODRDY* asserted) is delayed 100 nanoseconds to assure the required minimum specified setup times. When the Supervisor acknowledges the DRDY* by asserting DACC*, F4 is preset to complete the handshake.

Flip-flop F5 provides the input to the buffers through a 40 nanosecond holdover circuit. F5 is preset anytime the parity check circuit detects a parity error. At the start of the STO poll when STORD* is asserted, the parity check flip-flop F3 (Figure 5-2) is reset removing the preset from F5. At the trailing edge of the internal data ready signal that asserts DRDY* to the Multichannel bus, F5 resets. The holdover assures the minimum timing specifications are met.

5.4 HANDSHAKE CIRCUIT EXAMPLE

The handshake circuit examples illustrates an implementation of the read (device is the talker) and write (device is a listener) data mode bus cycle interface timing. In both examples, part of the data mode handshake timing is directly provided by the interface circuit rather than by the device. This allows for improved throughput over the Multichannel bus because the handshake response is made concurrent with the device performing data manipulation. Figure 5-4 illustrates the read data-mode handshake circuit. Figure 5-5 illustrates the write data-mode handshake circuit.

5.4.1 Read Handshake Circuit (Data Mode)

The circuit example illustrated in Figure 5-4 is active anytime the device is the talker for a data mode bus cycle. When the device is the talker, it must assert DRDY* to signal the listener that data is available. It must also respond to DACC* asserted (the listener signaling that it received the data element) by non-asserting DRDY* in preparation to transferring the next data element.

The ACTRD signal was developed on Figure 5-2 and is asserted anytime the device is selected as the talker and the a data mode bus cycle is required. The device asserts ORDY when address/data line set-up requirements are met for the bus cycle. F6 is normally set and the clock input is low. When the clock input goes high, F6 resets to assert DRDY*. When the listener asserts DACC*, OS2 is triggered and the low output from OS1 presets F6 and non-asserts DRDY*. At the same time, OS2 asserts CMPLT to the device so that the device can prepare to send the next data element.

5.4.2 Write Handshake Circuit (Data Mode)

The circuit example illustrated in Figure 5-5 is active anytime the device is the listener for a data mode bus cycle. When the device is the listener, it must assert DACC* to signal the talker that the data element was received. It must also respond to DRDY* being non-asserted (the talker signaling that the bus cycle is completed) by non-asserting DACC* in preparation to receiving the next data element.

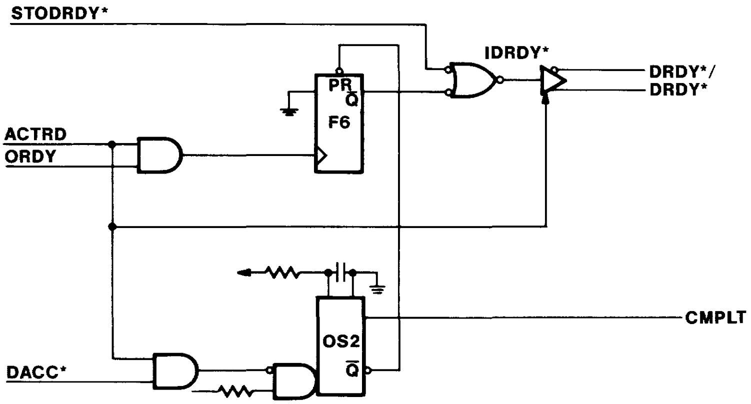


Figure 5-4 Read Handshake Circuit (Data Mode)

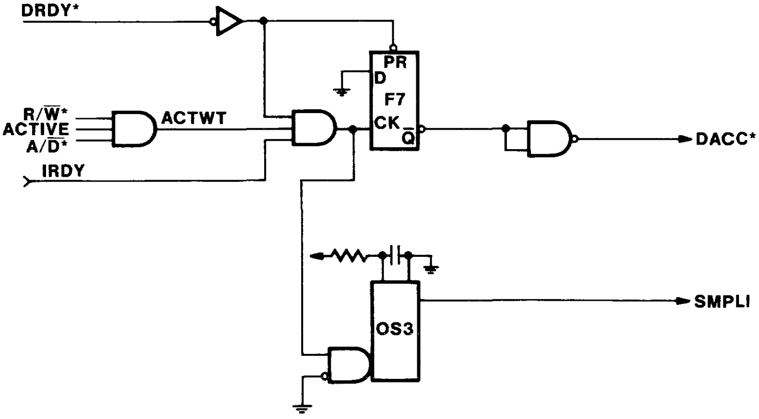


Figure 5-5 Write Handshake Circuit (Data Mode)

The ACTWT signal is asserted anytime the device is selected as the listener and a data mode bus cycle is required. The device asserts IRDY when it is ready to accept a data element from the address/data lines. F7 is normally set and the clock input is low. When the clock input goes high, F7 resets to assert DACC*. At the same time, OS3 is triggered to assert SMPLI to the device as a strobe for the device to sample the address/data lines. When the talker non-asserts DRDY*, F7 is preset to non-assert DACC*.

5.5 TERMINATION CIRCUIT EXAMPLE

The Multichannel bus must be terminated with pull-up termination at one end of the bus cable and pull-down termination at the other end. The termination must be located at the physical ends of the bus. Because the pull down termination is to ground and ground is available in the Multichannel bus, a passive pull-down termination module can be used. Because +5 Vdc is not present on any of the Multichannel bus lines, a passive pull-up termination module cannot be used and pull-up termination must be provided by a device attached to the Multichannel bus.

Several options are available. The Supervisor can be designated to always provide pull-up termination, all other devices do not provide termination, and a termination module used to provide pull-down termination. A second alternative is for all devices to provide pull-up termination that can be enabled or disabled as required and a termination module used to provide pull-down termination. A third alternative is for all devices to provide selectable pull-up/pull-down termination that can be enabled or disabled as required. Because the first two alternatives are subsets of the third alternative and the third alternative allows the most system design flexibility, an example of the third alternative (Figure 5-6) is presented here.

The device would provide sockets for the resistor packs and jumper stake pins at the E points shown in Figure 5-6. Because the jumpers are located on the voltage input to the resistor packs, the resistor packs must not be installed in the sockets when termination must be disabled. Otherwise, the floating common connection in the resistor pack will couple the Multichannel bus together. Thus to disable termination, remove all the jumpers and all the associated resistor packs.

The following procedure configures the circuit example for positive (pull-up) termination.

1. Install 16-pin DIP 220 ohm resistor packs (Beckman 898-1-R220 or equivalent) in sockets RP5 and RP6.

2. Install 6-pin SIP 220 ohm resistor packs (Beckman 763-1-R220 or equivalent) in sockets RP1, RP3, and RP4.

3. Install a 6-pin SIP 470 ohm resistor pack (Beckman 763-1-R470 or equivalent) in socket RP2.

4. Install the following jumpers:

   - E1 to E2
   - E4 to E5
   - E7 to E8

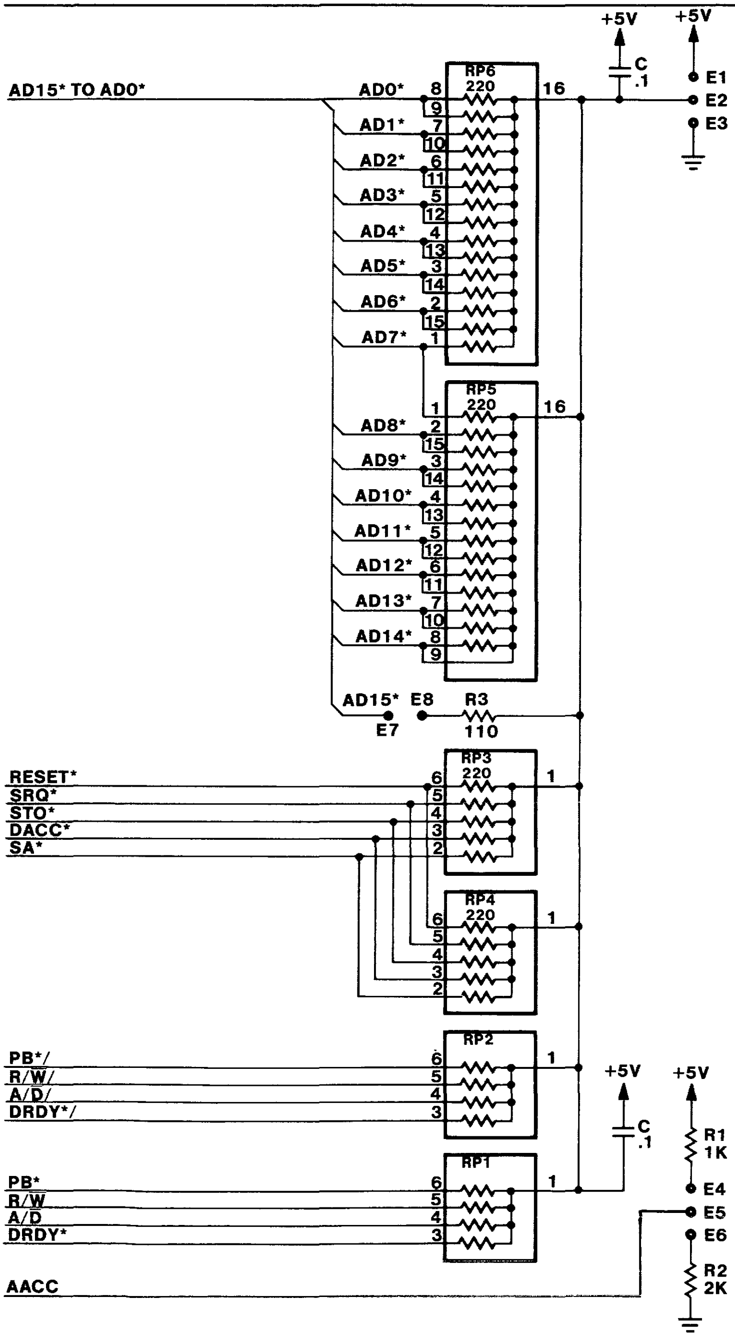


Figure 5-6 Termination Circuit Example

CHAPTER 6
LEVELS OF COMPLIANCE

6.1 INTRODUCTION

This section bounds the variability allowed within the Multichannel bus specification. The main purpose in bounding variability is to assure the maximum amount of upward compatibility. In most cases, mixing devices designed to different levels of compliance results in the more complex devices in the system operating at the level of the least complex device.

6.2 DATA PATH

The Multichannel bus allows for 8- and 16-bit data path devices. Devices that use 8-bit transfer elements but interface to all 16 address/data lines are classed as having a 16-bit data path. Typically, 8- and 16-bit data path devices cannot be mixed on the same implementation of the Multichannel bus. The transfer element size on a 16-bit data path device can be either 8-bits or 16-bits for memory or register; but, must be 16-bits for the address mode transfers. The transfer element size cannot be mixed within the data mode time of a message. That is, the address mode transfer elements at the start and end of the message could be 16-bits and the data mode transfer elements could be 8-bits, but the data mode transfer elements must not be a mixture of 8-bit and 16-bit transfer elements. A 16-bit data path device can restrict the data mode transfer elements to 16-bit only. A 16-bit data path device must transfer 8-bit transfer elements over the low order address/data lines, AD7* to AD0*.

The 8-bit data path devices are limited to two 8-bit address mode transfers which limits them to an address pointer range of 0 to 255. The actual address range of an 8-bit device can exceed the address pointer range provided that all data mode transfer blocks start within the address pointer range. The address pointer range can be extended using the Address Extension register (BH). When used, the Address Extension register must be loaded separately using a register data transfer before the memory data message is initiated.

6.3 SIGNAL LINE CONNECTIONS

In general, a Multichannel bus compatible device needs to connect to most of the bus lines. The Parity Bit differential line pair is classed as optional and is the only line that need not be implemented on all types of Multichannel bus compatible devices. The following is a list of additional exceptions to full Multichannel bus connection:

- A Supervisor (16-bit data path) must connect to all lines except the Parity Bit differential line pair.

- A Controller (16-bit data path) is allowed the same exception as the Supervisor and in addition need not connect to the Service Request line; however, it must connect to the Supervisor Take Over line and must support the STO Status (0H) and Device Command (3H) registers.

- A Basic device (16-bit data path) is allowed the same exception as the Supervisor and the Controller. In addition, the Basic device need not connect to the Supervisor Take Over line.

- Any Basic device that connects to and can drive the Supervisor Take Over line must support the STO Status register (0H).
- Any Controller or Basic device that connects to and can drive the Service Request line must support the SRQ Status (1H) and SRQ Mask (2H) registers.
- 8-bit data path devices need not connect to the high-order byte data lines (AD15* - AD8*).
- Very specialized devices (e.g. talker only or listener only) need not connect to bus lines that are redundant for the specialized operation.

6.4 DOCUMENTATION

The documentation (Hardware Reference Manual or equivalent) for a Multichannel bus compatible device should include specifications for the following items:

- type of device - Supervisor (SUP), Controller (CON), or Basic device (BD)
- data path width - 8-bit (D8) or 16-bit (D16)
- 16-bit word boundary restrictions, odd only (D16-O) or even only (D16-E)
- Parity Bit support - 8-bit (P8) or 16-bit (P16)
- Supervisor Take Over support (STO)
- Service Request support (SRQ)

6.5 COMPLIANCE MARKING

The compliance level of a Multichannel bus compatible board should be clearly indicated in the printed specifications. Capabilities that are mandatory for a given device functional level (Supervisor, Controller, or Basic device) need not be included in the compliance level marking. Omission of the marking for a capability denotes that the device does not support the capability. For example, a Supervisor with a 16-bit data path, 8-bit transfer capabilities, and parity would be marked as follows:

SUP D8 D16 P16

A 16-bit data path Controller with Parity Bit and Service Request support would be marked as follows:

CON D16 SRQ P16

A 16-bit data path Basic device without Supervisor Take Over, Service Request, or parity support and data transfers limited to even byte boundaries would be marked as follows:

BD D16E

APPENDIX A

A.1 INTRODUCTION

This appendix provides a summary description of the Multichannel bus signal lines. Refer to Section 2 for the full descriptions of the signal lines and the implementation parameters. Refer to Section 3 for the electrical specifications for the signal lines, and to Section 4 for the mechanical specifications. Table A-1 lists the Multichannel bus pin assignments.

Table A-1 Multichannel Bus Pin Assignments

LOWER ROW			UPPER ROW		
PIN	MNEMONIC	SIGNAL NAME	PIN	MNEMONIC	SIGNAL NAME
1	GND	Ground	2	AD0*	Address/Date Line 0
3	GND	Ground	4	AD1*	Address/Data Line 1
5	GND	Ground	6	AD2*	Address/Data Line 2
7	GND	Ground	8	AD3*	Address/Data Line 3
9	GND	Ground	10	AD4*	Address/Data Line 4
11	GND	Ground	12	AD5*	Address/Data Line 5
13	GND	Ground	14	AD6*	Address/Data Line 6
15	GND	Ground	16	AD7*	Address/Data Line 7
17	GND	Ground	18	AD8*	Address/Data Line 8
19	GND	Ground	20	AD9*	Address/Data Line 9
21	GND	Ground	22	AD10*	Address/Data Line 10
23	GND	Ground	24	AD11*	Address/Data Line 11
25	GND	Ground	26	AD12*	Address/Data Line 12
27	GND	Ground	28	AD13*	Address/Data Line 13
29	GND	Ground	30	AD14*	Address/Data Line 14
31	GND	Ground	32	AD15*	Address/Data Line 15
33	GND	Ground	34	RESET*	Reset
35	GND	Ground	36	AACC	Address Mode Accept
37	GND	Ground	38	SRQ*	Service Request
39	GND	Ground	40	STO*	Supervisor Take Over
41	GND	Ground	42	DACC*	Data Mode Accept
43	GND	Ground	44	SA*	Supervisor Active
45	PB*/	PARITY BIT (INV.)	46	PB*	PARITY BIT
47	R/$\overline{W}$/	Read-Not-Write (Inv.)	48	R/$\overline{W}$	Read-Not-Write
49	A/$\overline{D}$/	Address-Not-Data (Inv.)	50	A/$\overline{D}$	Address-Not-Data
51	DRDY*/	Data Ready (Inv.)	52	DRDY*	Data Ready
53	RES	Reserved	54	RES	Reserved
55	RES	Reserved	56	RES	Reserved
57	RES	Reserved	58	RES	Reserved
59	RES	Reserved	60	RES	Reserved

A.2 SIGNAL LINE SUMMARY

ADDRESS/DATA LINES (AD15* - AD0*)

The selected talker for a given bus cycle uses the 16 Address/Data lines to transfer the required information to the listener.

PARITY BIT (PB*, PB*/)

The Parity Bit differential line pair is used to verify the integrity of the information transferred during a bus cycle. Parity must be developed over the full data path of the Multichannel bus regardless of the transfer element size (8-bit byte or 16-bit word).

ADDRESS-NOT-DATA (A/$\overline{D}$, A/$\overline{D}$/)

The Address-Not-Data differential line pair is used by the master to identify the type of information transferred during a bus cycle. The master asserts the Address-Not-Data line pair to identify the transfer as address mode. When Address-Not-Data is non-asserted, the transfer is data mode.

READ-NOT-WRITE (R/$\overline{W}$, R/$\overline{W}$/)

The Read-Not-Write differential line pair is used by the master to identify the direction of information transfer during a bus cycle relative to the master. The master asserts the Read-Not-Write line pair to identify the transfer as a read operation. When Read-Not-Write is non-asserted, the transfer is write. All address information transfers must be write operations. Data information transfers can be either read or write.

DATA READY (DRDY*, DRDY*/)

The Data Ready differential line pair is asserted by the talker to signal the listener that information (data) is valid on the Address/Data lines.

ADDRESS MODE ACCEPT (AACC)

The Address Mode Accept line is used by all slave state devices to indicate receipt of the address information during an address mode bus cycle. The Address Mode Accept signal operates as a wired AND. After all devices have asserted the address accept line, the master detects the address accept and continues with the address mode bus cycle.

DATA MODE ACCEPT (DACC*)

The Data Mode Accept line is used by the selected listener during a data mode bus cycle to indicate receipt of the data information.

SUPERVISOR TAKE OVER (STO*)

The Supervisor Take Over interrupt is used by a slave to indicate task completion, parity errors, and other device errors.

SERVICE REQUEST (SRQ*)

The Service Request interrupt is used by a slave device to indicate device operational status and any other device specific conditions.

SUPERVISOR ACTIVE (SA*)

The supervisor asserts the Supervisor Active line any time it is in the master state. The supervisor non-asserts the Supervisor Active line as the last step in relinquishing control of the bus to a selected master. The supervisor can preempt control of the bus at any time by asserting the Supervisor Active line.

RESET (RESET*)

The supervisor asserts the Reset line to initialize all devices on the Multichannel bus to a known state.

intel®

CORRIGENDUM

August 1983

MULTIBUS® HANDBOOK

This IEEE Standard was approved April 29, 1983. IEEE Std 796 may be subject to Patent 4,257,095 held by Intel Corporation. IEEE takes no position with respect to patent validity. Intel Corporation has assured the IEEE that it is willing to grant a license under their patent on reasonable, nondiscriminatory terms and conditions to anyone wishing to obtain such a license. The Standards Office, and the licensing details are obtainable from the legal department of Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051, USA.

MULTIBUS® is a registered trademark of Intel Corporation. It is not a part of the title of IEEE Std 796-1983. Care should be taken to avoid use of the Intel trademark in specifying conformance with the IEEE Standard and in referring to products of other manufacturers.

When IEEE Std 796-1983 was approved as an IEEE Standard, the following additions and corrections were incorporated in the document:

Section 2.2.2.4

In paragraph 2, substitute the following lines:

(1) Transfer of even byte on DATO*-DAT7*
(2) Transfer of odd byte on DATO*-DAT7* (using byte function)

In Figure 6 make the following changes in the 8 bit master, 16 bit master and 16 bit memory blocks: change "LOW" to "EVEN". In key to circled numbers at the lower right of the figure, delete "LOW" and the parentheses around "EVEN" in 1, and delete "HIGH" and the parentheses around "ODD" in 2.

In paragraph 6, line 2: delete "(high)"; line 3: delete "high" and the parentheses around "odd".

In paragraph 7, line 2: delete "high" and the parentheses around "odd".

Section 3.1.2.1, paragraph 2: substitute "typ" for "max" within the parentheses on line 1 and within the equation.

Section 3.1.3, Table 1, column 1, Parameter: add a superior 2 following "Tolerance" and a superior 3 following "Ripple (Peak to Peak)". Add the following footnotes below the table:

[2] Includes line, load, temperature, and ripple effects.

[3] At 5 MHz bandwidth.

Table 1, in the three right hand columns of "Combined Line and Load Reg", change the values as indicated below:

column head:	+5	+12	-12
	4.9 - 5.2	11.8 - 12.5	12.5 - (-11.8)

Section 3.2: Add the following sentence at the end of paragraph 3: "All timing is measured at 1.5 V with loading capacitance of C and terminations specified in Table 3."

Table 2, page 3-7: add "(typ)" following t_{PD}in the left hand column.

Section 3.3: Delete the two sentences at the beginning of the section, and replace with the following:

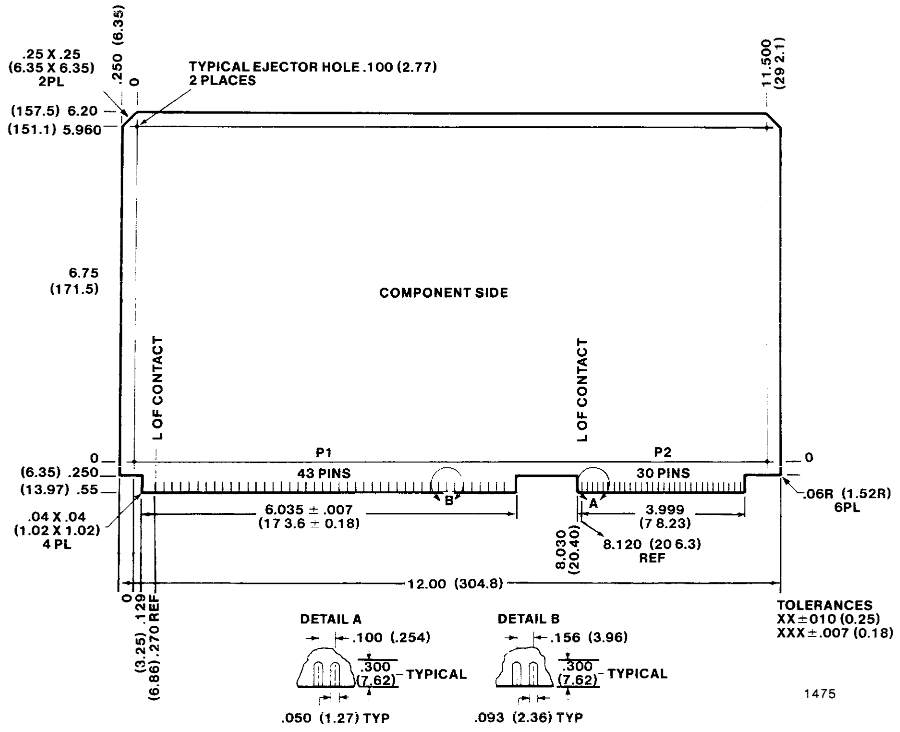
3.3 Receiver Modules, Driver Modules, and Terminations. Non-timing specifications, unique to each signal line or to groups of signal lines are presented in Table 3. The requirements for the signal line receivers, drivers, and bus terminations, and the locations of the receiver, driver, and termination for each signal are given.

Table 3: Delete minus signs before numbers in column headed I_{OH}
Max$_u$A

Section 4.2.1, line 1: Revise the beginning of the sentence to read, "The *P1 and P2* connectors on the printed circuit boards...."

Table 5: Add the following notes following the footnote at the bottom of the table:

Pins 1-40 are for "SPECIAL USE." Special uses are defined in categories.
Only category #1 is currently described:
Category #1 is unconstrained use.
Other categories are expected to include higher performance busses, I/O, interfaces, etc.
Pins 41-60 are intended for future address, data, and/or other P1-related signals.



Note that all dimensions are in inches with millimeters provided in parentheses for reference only. The inch dimensions govern.

Figure 33. Standard Printed Wiring Board Outline

NOTE: IEEE P969, I/O Bus Specification is a project presently under development within the Technical Committees of the Institute of Electrical and Electronics Engineers. The draft document reprinted in this handbook is not an official IEEE draft and should not have been identified with IEEE's acronym and project number.

Intle Corportion, 3065 Bowers Ave., Santa Clara, CA 95051; Tel. (408) 987-8080
Intel International, Brussels, Belgium; Tel (02) 661 07 11
Intel Japan k.k., Ibaraki-ken; Tel. 029747-8511

Printed in U.S.A./OMO/0883/15K/TPN/PR/MB MULTIBUS Single Board Computers