ALTOS

586T/986T

System Reference Manual

586T/986T
System Reference Manual

Federal Communications Commission Notice

Warning

This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions manual, may cause interference to radio communications.

It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment.

Operation of this equipment in a residential area is likely to cause interference in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE. NEW EDITIONS OF THIS DOCUMENT WILL INCORPORATE CHANGES AS THEY ARE PUBLISHED.

Acknowledgements

CP/M-86, MP/M, and MP/M-86 are registered trademarks of Digital Research, Inc.

Z8Ø is a trademark of Zilog, Inc.

Multibus is a trademark of Intel Corporation.

UNIX is a trademark of Bell Laboratories.

XENIX is a trademark of Microsoft Corporation and is a microcomputer implementation of the UNIX operating system.

About This Manual

This manual is intended for system programmers, engineers, technicians, and others who need to understand the internal operation of the 586T and 986T. It presents information about functions that are internally programmed and those that can be programmed by the user.

Those who need information about signal names, voltages, pin assignments, and specific circuit elements, should consult the Altos 586T/986T Maintenance Manual (P/N 690-15602-001).

Manual Conventions

Certain conventions are followed in the descriptions and diagrams used in this manual. Those which may not be obvious are listed below.

<u>Convention</u>	<u>Meaning</u>
*	The asterisk symbol, following a capitalized, bolded mnemonic indicates a "not" function <u>or</u> an active low signal. Example: **CSINT***.
Hexadecimal notation	A hexadecimal number is indicated by an "H" following the number. Example: 29H

Some terms are used interchangably and mean the same thing:

System Bus and Multibus
Counter-timer and programmable interval timer
Byte bus and 8-bit bus
System memory and main memory
I/O and peripheral

Contents

Appendices

ILLUSTRATIONS

TABLES

TABLES

Chapter 1
Introduction to the 586T/986T

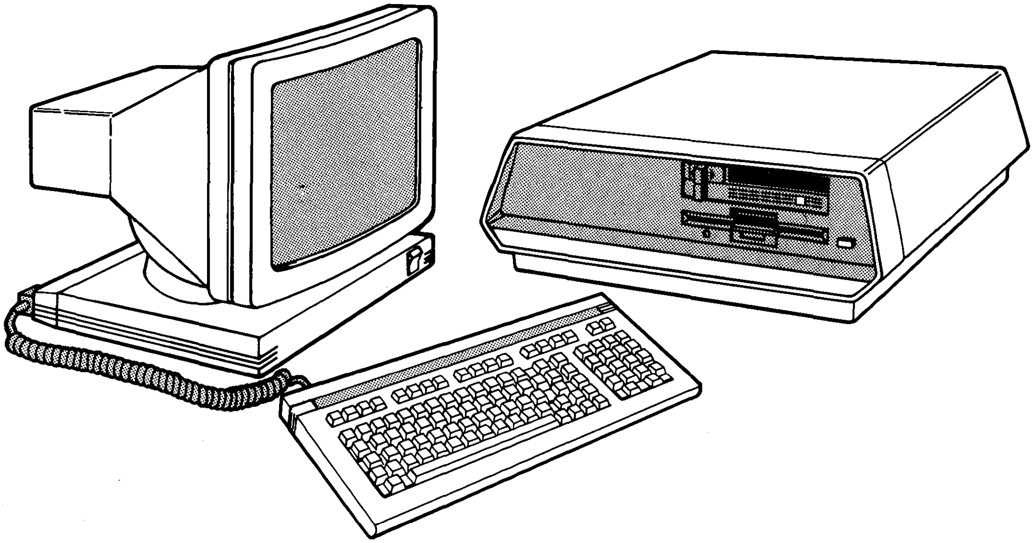


Figure 1-1. Altos 586T/986T Computer System.

Overview

The Altos 586T/986T is a powerful, multiuser, multitasking system designed to meet the needs of micro and mini-computer users. It supports industry-standard operating systems and languages, communications, and business application software. The 8Ø86-based, 16-bit architecture of the 586T/986T can perform larger, more comprehensive tasks for more users than an 8-bit system.

Hardware Features

The 586T/986T has the following hardware features:

- Multiple processor CPU board-

  8086 central processing unit operating at 10 MHz
  Z80A serial I/O processor

- 586T has 512 Kbytes of Main Memory
  986T has 1 Mbyte of Main Memory
  Both systems use byte parity error checking

- Intelligent Controller board for Hard Disk, Streaming Tape, and Floppy Disk units.

- Integral 40 or 80 Mbyte Winchester drive, 1 Mbyte Floppy Disk drive, and 60 Mbyte cartridge Streaming Tape unit.

- WorkNet Local Area Network port (RS422)

- Firmware for self-diagnostics at power-up.

- Time-of-day clock with battery back-up.

- Power fail detection circuitry.

Capabilities

The 586T/986T has these outstanding performance features:

- Multiple processing. Ten serial RS232 channels on the 986T support up to nine users, plus a serial printer. The 586T supports five users and a serial printer. The systems can handle hard disk, tape, and floppy disk operations concurrently.

- Memory Management. A proprietary memory management system provides logical-to-physical address translation and contiguous physical memory blocks for each task.

This reduces the need for swapping users or tasks out to disk. The system also provides write and access protection to each assigned block of memory.

o Hard Disk Control. The Hard Disk Controller can:

perform multiple sector read and writes;

cross over track and sector boundaries;

do automatic, transparent retries;

perform overlapped seeking when two drives are connected and accessed concurrently.

o Cartridge Tape Back-up. The 586T and 986T provide cartridge streaming tape back-up for the Hard Disk (up to 6Ø megabytes per DC6ØØA cartridge).

Options

Several configurations are possible by adding options to the standard systems.

o Memory Expansion. Memory on the 586T can be expanded to 1 Mbyte by adding a 512 Kbyte Memory Expansion board. (The 986T contains 1 Mbyte of memory as standard equipment).

o Hard Disk Expansion. The systems support hard disk expansion through two connectors on the rear panel, **H.D. EXPANSION** and **H.D. RADIAL#1**, to which data and control cables can be attached.

o Communications. A Communications printed circuit board supporting SNA or X.25 protocol and four serial (RS232) channels can be installed in lieu of the serial expander board on both the 586T and 986T systems.

Operating Systems

The 586T and 986T run the following operating systems:

- Concurrent CP/M-86
- XENIX version 3.0 RTS (Run Time)

Related Publications

The following publications are available from Altos:

Part Number	Title
15602	System Maintenance Manual
16177	Setting-Up Guide
16333	Illustrated Parts List
15494	Introduction to Concurrent CP/M-86
13499	Introduction to XENIX
14646	WorkNet

Specifications

Physical Characteristics

Width	16 7/8"
Height	6"
Depth	18"
Weight	36 lbs.

Environmental and Safety Standards

Meets FCC Class A requirements
UL and CSA approved

Ambient Temperature Range
50`- 90 degrees F (15 -32 degrees C)

Relative Humidity
non-condensing 20-80%

Power dissipation
300 Watts

Chapter 2
System Hardware

Introduction

This chapter summarizes the system's printed circuit board hardware. The system's I/O devices are discussed in Chapter 5.

Physical Description

The basic 586T/986T system is built around the following subsystems: a CPU printed circuit board, a Controller printed circuit board, a floppy disk drive, hard disk drive, and streaming tape unit. These components comprise the 586T system.

The 986T system includes two more printed circuit boards: a serial expander board which adds four ports to the system, and a 512-Kbyte expansion memory board (see the System Block Diagram, Figure 2-1).

Following is a brief description of the above subsystems.

Central Processing Unit (CPU) Board

The CPU board is the heart of the system; it contains the master processor for high level problem solving and efficient servicing of the Controller board Hard Disk, Streaming Tape, and Floppy Disk interrupts. A slave I/O processor, a Z8ØA, handles character interrupts from the serial I/O channels. There are six on-board RS232 ports which can connect to five ASCII terminals and a serial printer or any combination determined by the operating system.

The principal circuits are as follows:

- 8Ø86 master processor with its peripherals
- Interrupt Controller and System Timer
- 512 Kbytes of RAM memory with parity generation and parity error detection
- Memory Manager and Control circuit
- EPROM
- Bus Interface Controller
- Port Decoder
- I/O Processor including Z8ØA CPU, three Z8ØA SIO/Øs, two 8254 timers, and Z8ØA PIO

CPU board functions are discussed in detail in Chapter 3. The I/O Processor is discussed in detail in Appendix A.

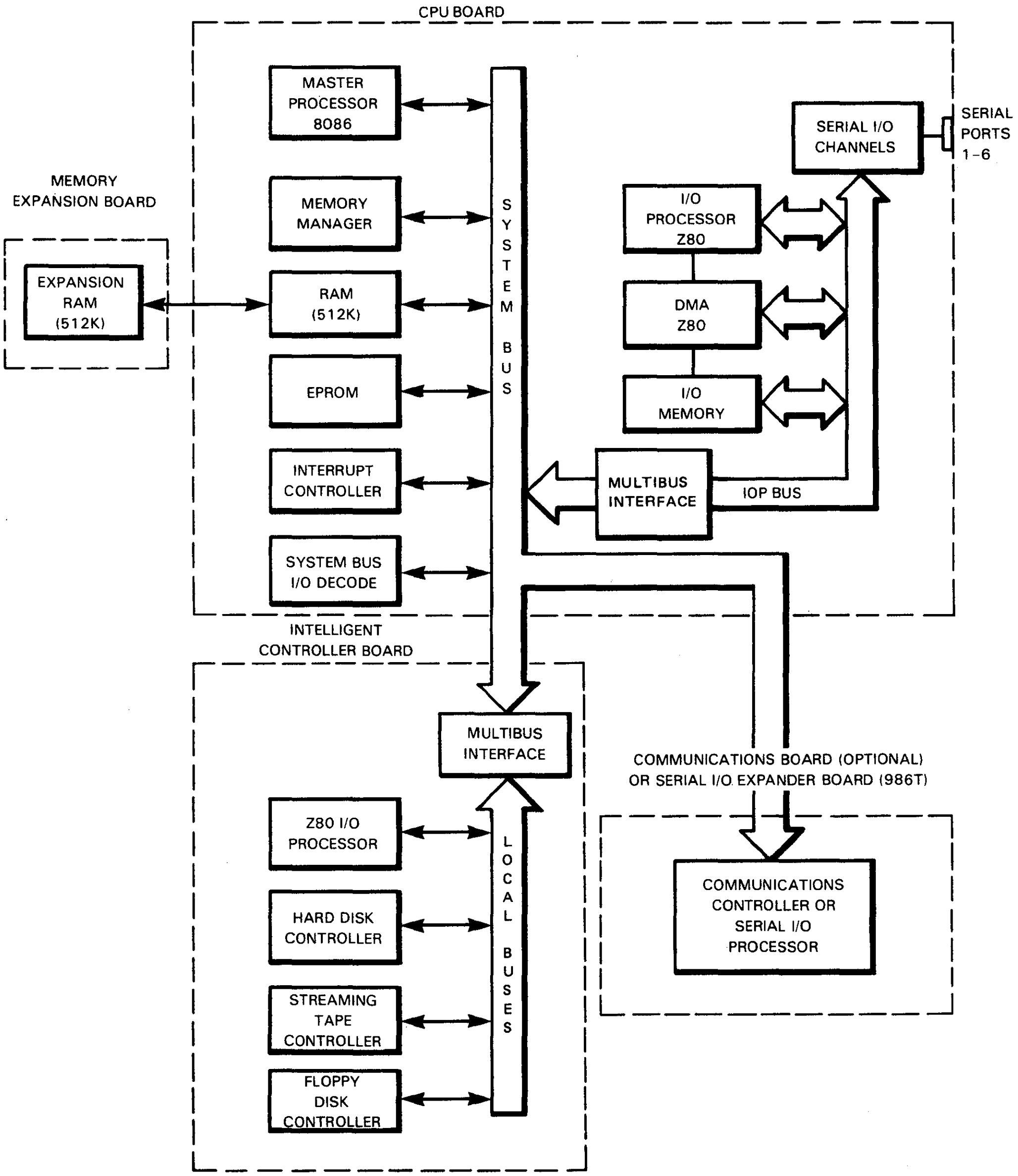


Figure 2-1. System Block Diagram.

Controller Board

The Controller board interfaces to the CPU board and provides channels and control for the system's hard disk(s), floppy disk, and streaming tape unit. These devices are locally controlled by an on-board Z8ØA processor which receives initial I/O commands from the host processor on the CPU board via command words (parameter blocks). When an I/O operation is completed, a status byte in the command word is updated and the command word is returned to the host.

The following major circuits are contained on the Controller board:

- Multibus Interface
- Z8ØA CPU
- 16Kbyte RAM
- 8Kbyte PROM
- AMD 9517 DMA Controller
- AMD 9519 Interrupt Controller
- WD2Ø1Ø Hard Disk Controller
- 2Kbyte Hard Disk Buffer
- NEC 765 Floppy Disk Controller
- Streaming Tape Controller

Controller board functions are described in Chapter 4.

Serial I/O Expander Board

A Serial I/O Expander board is standard on the 986T (optional on the 586T). It contains a stand-alone I/O processor identical in operation to the I/O Processor on the CPU board. When it is installed, it expands the number of serial channels and RS232 ports by four.

The following hardware is used:

- Z8ØA CPU
- Two Z8ØA SIO/Øs
- Two AMD 9513 system timing counters
- 8219 Bus Controller

The I/O Processor on the Expander board is discussed in detail in Appendix A.

512 Kbyte RAM Expansion Board

A RAM Expansion board increases system memory to 1 Mbyte of dynamic RAM. It is standard on the 986T, optional on the 586T.

Chapter 3
Central Processing Unit Functions

Introduction

The Central Processing Unit (CPU) board contains the system master processor together with its peripherals, and the character I/O processor and channels. Of interest to the systems engineer and programmer are the programmable devices, memory, and buses; how the system is initialized, its operation, and the addresses of ports and I/O devices.

Hardware Overview

8086 Master Processor and System Memory

The 8086 microcomputer chip is the system master processor. It carries out the instructions of the resident operating system and application programs. To achieve maximum throughput and speed, I/O tasks are downloaded by the 8086 to I/O processor subsystems.

The 8086 is a 16-bit microprocessor with 16 data lines and 20 address lines. The address lines provide one megabyte of memory space (2**20 = 1 megabyte). The 8086 assigns a separate 64K byte space for I/O devices and addresses these by the 16 lines, A0 - A15.

Memory Organization

The 8086 organizes memory in arrays of 8-bit bytes. A byte may be stored at an even or an odd address. The 8086 always fetches a word (two bytes) at a time from an even boundary (even-numbered address). Therefore, for maximum processing speed, a word should be stored on an even boundary so that it can be fetched in a single operation. A word stored at an odd boundary requires two fetches.

A word is always stored with the most significant byte in the higher memory location. For example, when 53D7H is stored at memory location 934H, it is stored as follows:

```
                        start of address 935H
                        |
                    D 7 5 3
                    |
start of address 934H
```

Memory Segmentation

Memory segmentation is a method of efficiently using available memory. Memory space is divided into blocks or segments, each with a base address. Segments can then be moved to provide maximum free space when needed. This is done by placing segments contiguously, or by moving them out to disk. This technique is also known as dynamic memory allocation.

The 8Ø86 divides memory space into 64K byte segments. The location of the segment depends on the value stored in the segment register. There are four types of segment registers, CS, DS, SS, and ES. To obtain more information about segment register operations refer to the **8Ø86 Family User's Manual** published by Intel.

Figure 3-1 illustrates the concept of memory segmentation. A segment is relocated in memory simply by changing the base address. However, the 16-bit base address only defines a 64K block. An actual physical address requires 2Ø bits.

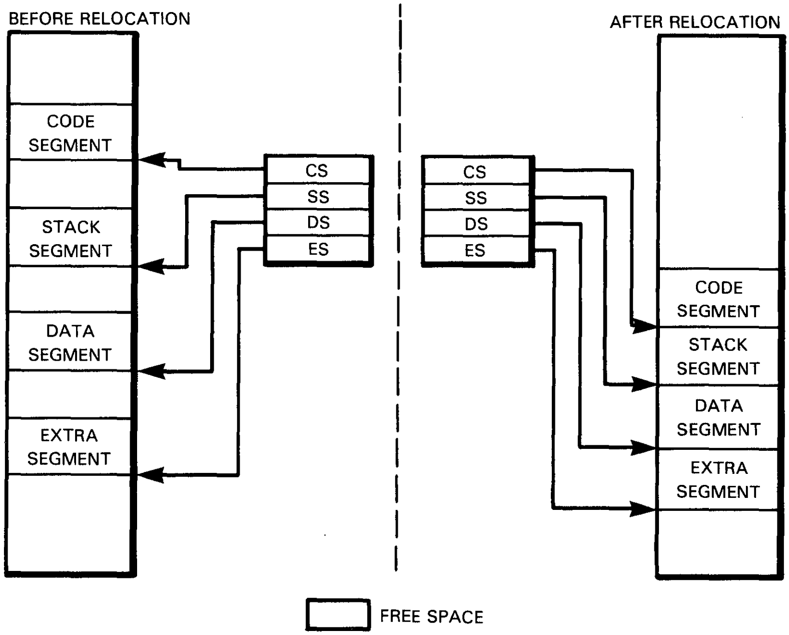


Figure 3-1. Dynamic Relocation

The following section describes how a physical address is generated.

8086 Address Generation

Application programs generally deal with logical, rather than physical addresses, since it is a function of the computer to provide physical addresses for memory locations and I/O devices. The 8Ø86 generates logical addresses for each address-value in a program. A logical address is then translated into a physical address (see Figure 3-2).

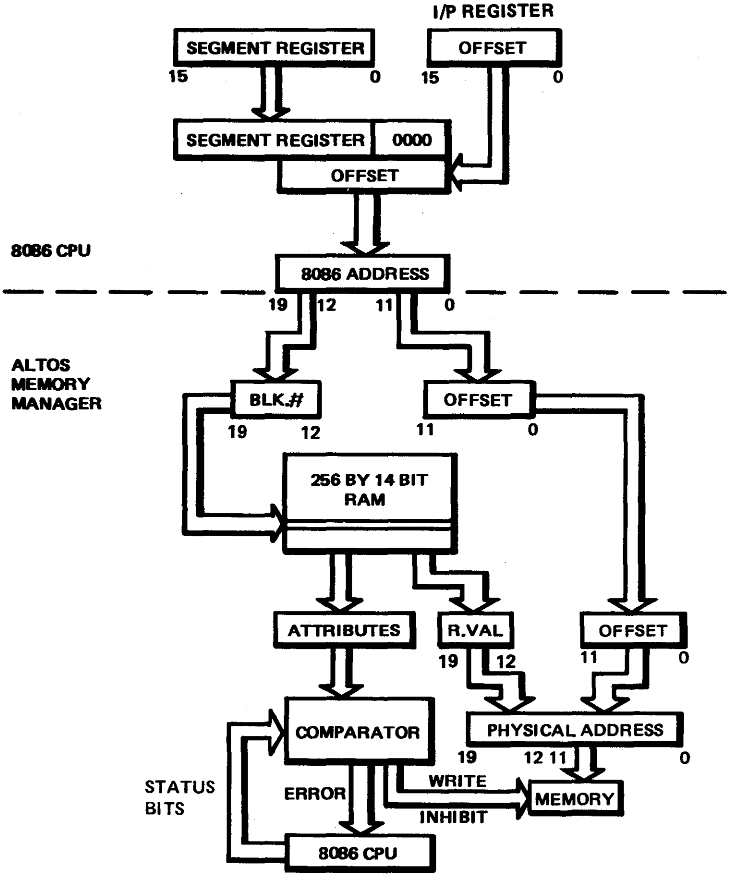


Figure 3-2. Memory Management Block Diagram

As shown in Figure 3-2, a 16-bit value is placed in the segment register representing the segment's current location. The register is then shifted left four positions and zeros are appended to the four least significant positions. A 16-bit offset value representing the address in the instruction (from the program) is added to this value. The resulting 2Ø-bit number is the 8Ø86 logical address.

The memory segmentation technique used by the 8Ø86 provides some limited access protection because an offset cannot go beyond the 64K boundary of the segment (2 to the power of 16 = 64K) unless the base value in the segment register is altered. However, the 8Ø86 does not protect against segment overlap. Further, any program may access a segment register at any time. The Altos Memory Management circuit overcomes these limitations (see the following section).

Altos Memory Manager

General Concepts

The Altos Memory Manager extends the 8Ø86 memory management facility. It supports

- Write protection
- Access protection
- Limit checking
- Non-contiguous memory allocation.

One megabyte of system memory is organized as 256 pages of 4-kilobytes each. Each page has a number of memory protection attributes and can be mapped into any 4 kilobyte block of physical memory. The attribute and mapping bits for all 256 pages of memory are stored in static RAM.

Memory protection, violation detection, and memory mapping all take place during a memory access. When memory is accessed, the mapping bits are used to map the page into the physical memory location. At the same time, the attribute bits are examined to test the validity of the access.

If the access violates any of the attributes, then an interrupt is generated and the address of the access is latched. The CPU can ascertain what the error was by reading the Violation Port (78) and the Error Address ports (6ØH to 6FH). See tables 3-8 and 3-9.

Memory Manager Operation

See Figure 3-2 for the following discussion. One megabyte of memory address space is divided into 256 pages of 4K (4Ø96) bytes. Each page is numbered from ØØ to FF (hexadecimal) and accompanied by six flag and control bits (attributes). This data is stored in a 256 x 14-bit RAM. The RAM is addressed by the eight most-significant bits of the 8Ø86 address. When a memory access is made, a page number (relocation value) is selected to replace the eight most-significant bits of the 8Ø86 logical address and form a 2Ø-bit physical address.

The Memory Management RAM may be accessed through I/O ports 2ØØH to 3FEH to examine the address and attributes of a particular page (see **Memory Management I/O Ports** and Table 3-6).

Memory User Types

The 8Ø86 memory segmentation scheme divides memory use into four types as shown in Figure 3-1. The Altos 586T/986T further identifies two types of memory users:

o 8Ø86 applications or operating system users, and

o Other bus masters.

Each page of memory can be protected from access (either read or write) by application programs. It can also be protected from write access by any of the memory users. For example, the Memory Manager has provisions for protecting each memory page against System Mode memory writes (see **User Mode and System Mode for an explanation of System Mode**), thus making the page read only. The same page can be marked to disallow access by an Application Mode program.

Finally, the Memory Manager implements the concept of "privileged" instructions. A privileged instruction can only be executed via a request to the operating system. This prevents one problem subprogram from issuing instructions to an I/O device being used by another problem subprogram. All I/O and interrupt disabling instructions are treated by the 586T/986T as privileged instructions. Applications programs must run with interrupts enabled and make I/O access requests through the operating system. Actual I/O accesses by an application program while it is running are inhibited. An attempt by an applications program to perform an I/O operation causes a System Call interrupt (see **Interrupt Controller**).

Memory Management Violation Detection

The memory management violation detection circuitry detects operations that violate the attributes of any particular page in memory. For instance, if a page in memory is flagged as a stack boundary page, and a push is performed within the 128-byte guard band at the bottom of the page, then a violation is detected and a nonmaskable interrupt is generated.

Other violations also cause nonmaskable interrupts. They are RAM parity errors, system bus timeout errors, and invalid instruction violations.

All of these violations are latched and can be read at the Violation Port (Port 78). The bits in the violation port are defined in Table 3-7. The violation port is a read-only port. Two bits have been added to the violation port that are not violation bits. They are simply status bits generated by jumpers for use by the system or by automatic testing equipment.

When a violation is detected the address is latched along with the USER MODE bit (see **User Mode and System Mode**). This gives the system software more information about the violation. This information can be read at ERROR ADDRESS PORT 1 and ERROR ADDRESS PORT 2. The bit definitions for these ports are listed in Tables 3-8 and 3-9.

After a violation has occurred, further violations will not produce nonmaskable interrupts until the current violation has been cleared. Violations can be cleared by performing an I/O instruction to the CLEAR VIOLATION PORT (Port 7Øh).

Normally, only one violation bit will be set in the violation port if an error occurs. However, if another violation occurs before the first violation has been cleared, then the new violation will be detected and latched. Thus, more than one violation bit could be set. No information is given about which violation occurred first. Only the address of the first violation is latched.

Under Xenix, the panic trap violation number is read from the Violation Port.

Interrupt Controller

The interrupt controller is an 8259A-2 device. Port addresses are listed in table 3-4. There are eight interrupt request inputs each of which can be masked on or off. Table 3-1 lists the interrupt request signals in order of decreasing priority.

Table 3-1. Interrupt Priority Levels

Bit	Interrupt Request Signal	Priority
IRØ	System Call	1
IR1	Timer Interrupt	2
IR2	Hard Disk Interrupt	3
IR3	Tape Interrupt	4
IR4	I/O Processor Interrupt	5
IR5	INT4	6
IR6	INT5	7
IR7	Floppy Disk Interrupt	8

System Counter Timer

The system counter timer is an 8254 device located on the system data bus. It is accessed by the 8Ø86 through I/O ports that are listed in Table 3-4 and provides the baud rate for serial port 6 and the system timer interrupt.

There are three counter timers in the 8254:

> Counter Timer Ø is used to generate the baud rate for serial port 6.
>
> Counter timer 1 is used as a prescaler for counter timer 2. In other words, the output of counter timer 1 is connected to the input of counter timer 2.
>
> Counter timer 2 output is the system timer interrupt which is connected to the 8259A-2 interrupt controller.

The counting rate for counter Ø and counter 1 is 5 MHz and the counter rate for counter 2 is determined by the output of counter 1.

I/O Processor

The I/O processor services the system's serial I/O ports. Detailed operation of the I/O Processor is provided in Appendix A. The following is a summary of the hardware. The CPU board contains a Z8ØA CPU, Z8ØA DMA, three Z8ØA SIO's, two 8254 counter timers (or programmable interval timers), 8K x 8 EPROM, 2K x 8 static RAM, 58176A Calendar Clock, 8219 bus arbiter and interface logic.

Z80A Processor

The I/O processor communicates with the host CPU (8Ø86) via calls to system memory locations and CPU interrupts. When the Z8ØA is initialized by an interrupt (**IOP Channel Attention**) from the 8Ø86, it addresses and then executes a control program residing in local EPROM memory (see **I/O Processor Memory**). The Z8ØA also employs a small amount (2K) of static RAM for temporary storage and buffering.

I/O Processor Memory

The Z8ØA has 64 KB of memory space divided in two halves. The upper 32 KB is mapped to system memory, the lower 32 KB to local EPROM and static RAM. The Z8ØA can access any part of the 1 MB of system memory space, but must do so in blocks of 32 KB. This is accomplished by pre-loading a block number register. When the Z80A accesses system memory, it places the block number on Multibus address lines 15 to 19. The Z80A supplies the remainder of the address on its 16 address lines.

Table 3-2 defines the Z80A memory space.

Table 3-2. Z80A Memory Map

Memory Type	Z80A Address	Range
EPROM	00h to 1FFFH	(8K)
STATIC RAM	2000 to 27FFH	(2K)
SYSTEM MEMORY	8000h to FFFFH	(32KB blocks)

Z80 DMA Controller

The Z80A DMA Controller performs dual port data transfer from any of the synchronous serial channels to local memory or system memory. It has the ability to terminate a data transfer as the result of a pattern match.

Z80 I/O Addressing

The Z80A has a 256 byte I/O address space. The I/O addresses for the various peripherals are shown in Table 3-10. Use only addresses shown in the table since not all of the I/O addresses are uniquely decoded.

Calendar Clock

This CMOS device keeps date and time even when the system power is off. When the system power is off, the clock is powered by a replaceable lithium battery, which has a normal life of two to five years.

System Interfaces

System Bus

The Altos 586T/986T system bus is electrically similar to the Intel Multibus. Signals and signal timing are the same. A standard multibus board can be connected to the 586T or 986T system.

There are, however, some differences between the 586T/986T system bus and the Multibus, as listed below.

- A **PARITY ERROR*** signal has been added. This active low signal indicates that a system memory parity error was detected during the current bus cycle. The parity error signal will be valid at least 25 nanoseconds before **XACK*** goes active.
- The **INIT*** signal is being driven with a low power Schottky part instead of an open collector driver.
- Two interrupt request signals, **INT4** and **INT5** are available for use by an external Multibus board.
- The signal **AACK*** has been added. This is an advanced bus acknowledge signal. When memory is accessed, this active-low signal will be asserted at least 120 nanoseconds before data is valid on the bus.

CPU-Controller Interface

The CPU board interfaces with the Controller board via the system bus and the Multibus. An 8289 Bus Arbiter exercises control over the system bus to determine which bus master, 8086, Z80A, or Controller board CPU has control. When the Controller board CPU has control, data is transferred through data transceivers by Multibus control logic on the Controller board. Address information is translated directly from the system bus to the Multibus.

Serial I/O Interfaces

RS232C

The serial I/O interface consists of three SIOs and two 8254 counter timers. Each SIO supports two serial output ports for a total of six serial I/O ports. Four additional serial I/O ports can be added via a serial expander board. The counter timers are multi-timing elements for the SIOs under software control. The Z8ØA SIOs are general purpose, dual channel, parallel-to-serial converter/controller devices. The serial channels are normally configured for RS232C asynchronous operation. However, channels 1 and 5 can be strapped (via jumpers) for RS232C synchronous operation (see **APPENDIX** B).

RS422 (WorkNet)

Channel 3 can be strapped for operation as an 8ØØ kilobit networking port (RS422) (see WorkNet manual, P/N 69Ø-14646-ØØ2).

System Initialization and Operation

8086 Reset and Initialization

When a power-on or reset condition occurs, the 8Ø86 begins execution at physical address FFFFØH in system memory. A jump instruction at that location points to the beginning of EPROM memory at FEØØØH. The 8Ø86 executes the code in EPROM, which initializes the system and loads the operating system.

Certain memory locations are reserved by the 8Ø86 for its operations (see Figure 3-3). Locations from ØH to 3FFH are reserved for interrupt pointers. Locations FFFFØH to FFFFFH (top of memory) are reserved for the bootstrap program.

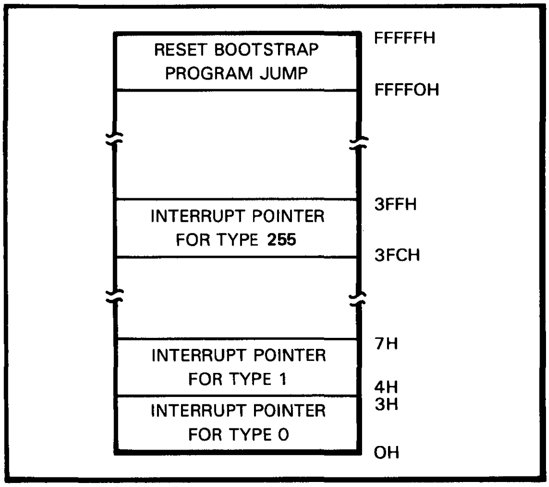

RESET BOOTSTRAP PROGRAM JUMP
FFFFFH
FFFFOH
INTERRUPT POINTER FOR TYPE 255
3FFH
3FCH
INTERRUPT POINTER FOR TYPE 1
7H
4H
INTERRUPT POINTER FOR TYPE 0
3H
OH

Figure 3-3. Reserved Memory Locations

Minimum and Maximum Modes

The 8Ø86 is strapped for maximum mode operation, which means that an 8288 bus controller is used to generate bus timing and control signals compatible with the Multibus architecture.

User Mode and System Mode

The ALTOS 586T and 986T accomplish system memory protection and system configuration protection with the help of a USER MODE bit. The system is in user mode when the bit is set. Application programs should always run in user mode. Conversely, when the bit is not set, the system is said to be in SYSTEM MODE. Operating systems run in system mode.

The system is initialized in system mode and stays in system mode until the User Mode bit is set. All instructions are allowed in system mode. In user mode, I/O instructions are not allowed.

To enter user mode the operating system sets a User Mode Request bit (bit Ø) in the Control Bits Port (58H). The system then enters user mode when interrupts are enabled. Interrupts must be enabled so that user mode system calls may be processed. Whenever a non-maskable interrupt, maskable interrupt acknowledge, or system reset occurs, the system jumps out of user mode.

All I/O operations are inhibited when the system is in user mode. This prevents application programs from changing the memory management ports or reinitializing the peripherals. Interrupts may not be disabled when in user mode. If they are, then an Invalid Instruction violation is detected and the memory manager produces a non-maskable interrupt which resets the user mode bit.

The normal exit procedure from user mode is through a System Call interrupt (see Table 3-1). A System Call interrupt is generated whenever an I/O instruction is performed while in user mode.

Note that bus masters other than the 8Ø86 may not perform I/O instructions to any port in the range ØØØØH to Ø3FFH.

Control Bits

Three system control bits may be written through the Control Bits Port (58H). They are:

- Request User Mode
- Enable Non-Maskable Interrupt
- Warm Start Bit

The Request User Mode bit is set as described under **User Mode and System Mode.**

The Enable Non-Maskable Interrupt bit enables non-maskable interrupts when set to 1. It is disabled at power-up or reset and is intended primarily for diagnostic purposes.

The Warm Start bit is accessible for use by system software, but is not actually used. It indicates whether the system was initialized by a power-up or by the reset switch. Power-up always clears this bit, but reset sets the bit. If the bit is set to a 1, the system start was a warm start (reset).

Table 3-3 shows bit definition.

Table 3-3. Control Bits Port Definition (Port 58H)

BIT	BIT NAME	TRUE
Ø	Request User Mode	1
1	Not Used	1
* 2	Enable Non-maskable Interrupt	1
3	Not Used	1
4	Not Used	1
5	Not Used	1
6	Not Used	1
7	Not Used	1
8	Warm Start Bit, Not Used	1
9	Not Used	1
1Ø	Not Used	1
11	Not used	1
12	Not Used	1
13	Not Used	1
14	Not Used	1
15	Not Used	1

* This bit is set to Ø at power-up.

Addressing

I/O Port Decoder

The 8Ø86 can support both 8-bit and 16-bit I/O for a total of 65,536 addressable ports. Not all address combinations are used in this application. For eight bit I/O operations, the 8Ø86 uses half of the 16-bit data bus for even I/O addresses and the other half for odd I/O addresses. Even addresses cause data to be moved over Data Bus bits Ø through 7.

The block diagram for the 586T/986T I/O port decoder is shown in Figure 3-4. The decoder has two parts: The memory manager port decoder and the general I/O port decoder.

Table 3-4 shows the hex addresses, port size and functions for the I/O ports. Note that port sizes are shown as Ø, 8, or 16 bits, where:

Ø means the port is 8 or 16-bit access, but does not use the data bus.

8 means 8-bit access, using the data bus.

16 means 16-bit access, using the data bus.

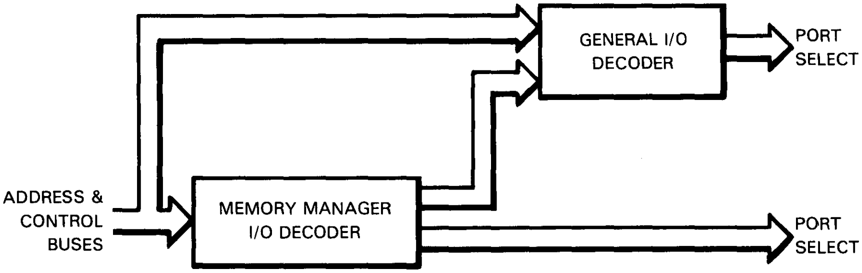


Figure 3-4. I/O Port Decoder Block Diagram

Table 3-4. 8086 I/O Port Addresses

NOTE: Ports 0000H through 003FH are not decoded

Hex Address	**Port Size** (in bits)	**Function and Comments**
0040h to 0047	0	Clear System Call
0048h to 004FH	0	Channel attention 0 (reserved for future bus master channel attentions)
0050h to 0057H	0	Z80A I/O Processor Chan att.
0058h to 005FH	16	Control Bits Port - Write Only. See Table 3-3.
0060h to 0067H	16	Error Address 2 - Read Only. See Table 3-9.
0068h to 006FH	16	Error Address 1 - Read Only. See Table 3-8.
0070h to 0077H	0	Clear Violation Port
0078h to 007FH	16	Violation Port - Read Only. See Table 3-7.
* * * INTERRUPT CONTROLLER PORTS * * *		
0080	8	ICW2, ICW3, ICW4, or OCW1
0081		Decoded but not used
0082	8	ICW1, OCW2, or OCW3
0083		Decoded but not used
0084H to 00FFH		Do not use these ports!!! They map into ports 80-83.
* * * 8254 SYSTEM TIMER PORTS * * *		
0100H		Decoded but not used
0101H	8	Control Word Register - Write Only
0102H		Decoded but not used

Table 3-4 (Cont'd). 8086 I/O Port Address Definitions

Hex Address	Port Size (in bits)	Function and Comments
0103H	8	Counter 2
0104h		Decoded but not used
0105H	8	Counter 1
0106H		Decoded but not used
0107	8	Counter 0
0108H to 01FFH		Do not use these ports!!! They map into the system timer ports (100-107).
* * *	MEMORY MANAGEMENT PORTS * * *	
0200H to 03FFH	16	Memory Management Ports Even addresses only. See Tables 3-5 and 3-6.
0400H to FFFFH		Reserved for system bus I/O.

Memory Management I/O Port Definition

The following table shows the bit definition of the Memory Management ports. A detailed explanation of each bit follows on the next page. Table 3-6 is a listing of the port addresses.

Table 3-5. Memory Management I/O Port Definition

	Map and Attribute Bit Definition	
Bit	**Bit Name**	**True**
0	Physical Address 12	1
1	Physical Address 13	1
2	Physical Address 14	1
3	Physical Address 15	1
4	Physical Address 16	1
5	Physical Address 17	1
6	Physical Address 18	1
7	Physical Address 19	1
8	Not Used, Undefined	
9	Not Used, undefined	
10	Not Used, Undefined	
11	Allow I/O Processor Write	1
12	Allow System Write	1
13	Stack Boundary Page	1
14	Allow User Access	1
15	Allow User Write	1

Following is a detailed description of each bit:

Bits

0 through 7 define the physical address of the page in memory. Suppose that port 268H (port for page 34H) has bits 0 through 7 set to 57H. This would reposition logical addresses 34000H-34FFFH into physical memory locations 57000H-57FFFH. The port addresses for each page in memory are listed in Table 3-1.

8, 9 and 10 are undefined.

11 is the Allow I/O Processor Write bit; when this bit is set to a 1, it allows any bus master other than the 8086 to write to that page in memory.

12 is the Allow System Write bit. When this bit is set to a 1, the 8086 in SYSTEM MODE(not USER MODE)is allowed to perform a write to that page in memory.

13 is the Stack Boundary Page bit. When set to a 1, this bit indicates to the violation detection circuitry that the stack may not grow below that page in memory.

14 is the Allow User Access bit. When this bit is set to a 1, the 8Ø86 in user mode can perform reads or writes to that page in memory.

15 is the Allow User Write bit. When this bit is set to 1, the 8Ø86 in user mode can perform memory writes to that page in memory.

NOTE

For the 8Ø86 to write a page in USER MODE, both bits 14 and 15 should be set for that page.

If a read or write to a memory page is attempted and it conflicts with any of the attribute bits, a memory violation is generated. See **Memory Management Violation Detection.**

Memory Management I/O Port Addresses

The 256 by 14 bit RAM portion of the Memory Manager is accessed through 256 I/O ports having even numbered addresses from 2ØØH through 3FEH. The memory page number and the logical address range for each port is listed in Table 3-6.

Table 3-6. Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
00	00000-00FFF	0200
01	01000-01FFF	0202
02	02000-02FFF	0204
03	03000-03FFF	0206
04	04000-04FFF	0208
05	05000-05FFF	020A
06	06000-06FFF	020C
07	07000-07FFF	020E
08	08000-08FFF	0210
09	09000-09FFF	0212
0A	0A000-0AFFF	0214
0B	0B000-0BFFF	0216
0C	0C000-0CFFF	0218
0D	0D000-0DFFF	021A
0E	0E000-0EFFF	021C
0F	0F000-0FFFF	021E
10	10000-10FFF	0220
11	11000-11FFF	0222
12	12000-12FFF	0224
13	13000-13FFF	0226
14	14000-14FFF	0228
15	15000-15FFF	022A
16	16000-16FFF	022C
17	17000-17FFF	022E
18	18000-18FFF	0230
19	19000-19FFF	0232
1A	1A000-1AFFF	0234
1B	1B000-1BFFF	0236
1C	1C000-1CFFF	0238
1D	1D000-1DFFF	023A
1E	1E000-1EFFF	023C
1F	1F000-1FFFF	023E

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
20	20000-20FFF	0240
21	21000-21FFF	0242
22	22000-22FFF	0244
23	23000-23FFF	0246
24	24000-24FFF	0248
25	25000-25FFF	024A
26	26000-26FFF	024C
27	27000-27FFF	024E
28	28000-28FFF	0250
29	29000-29FFF	0252
2A	2A000-2AFFF	0254
2B	2B000-2BFFF	0256
2C	2C000-2CFFF	0258
2D	2D000-2DFFF	025A
2E	2E000-2EFFF	025C
2F	2F000-2FFFF	025E
30	30000-30FFF	0260
31	31000-31FFF	0262
32	32000-32FFF	0264
33	33000-33FFF	0266
34	34000-34FFF	0268
35	35000-35FFF	026A
36	36000-36FFF	026C
37	37000-37FFF	026E
38	38000-38FFF	0270
39	39000-39FFF	0272
3A	3A000-3AFFF	0274
3B	3B000-3BFFF	0276
3C	3C000-3CFFF	0278
3D	3D000-3DFFF	027A
3E	3E000-3EFFF	027C
3F	3F000-3FFFF	027E

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
40	40000-40FFF	0280
41	41000-41FFF	0282
42	42000-42FFF	0284
43	43000-43FFF	0286
44	44000-44FFF	0288
45	45000-45FFF	028A
46	46000-46FFF	028C
47	47000-47FFF	028E
48	48000-48FFF	0290
49	49000-49FFF	0292
4A	4A000-4AFFF	0294
4B	4B000-4BFFF	0296
4C	4C000-4CFFF	0298
4D	4D000-4DFFF	029A
4E	4E000-4EFFF	029C
4F	4F000-4FFFF	029E
50	50000-50FFF	02A0
51	51000-51FFF	02A2
52	52000-52FFF	02A4
53	53000-53FFF	02A6
54	54000-54FFF	02A8
55	55000-55FFF	02AA
56	56000-56FFF	02AC
57	57000-57FFF	02AE
58	58000-58FFF	02B0
59	59000-59FFF	02B2
5A	5A000-5AFFF	02B4
5B	5B000-5BFFF	02B6
5C	5C000-5CFFF	02B8
5D	5D000-5DFFF	02BA
5E	5E000-5EFFF	02BC
5F	5F000-5FFFF	02BE

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
6Ø	6ØØØØ-6ØFFF	Ø2CØ
61	61ØØØ-61FFF	Ø2C2
62	62ØØØ-62FFF	Ø2C4
63	63ØØØ-63FFF	Ø2C6
64	64ØØØ-64FFF	Ø2C8
65	65ØØØ-65FFF	Ø2CA
66	66ØØØ-66FFF	Ø2CC
67	67ØØØ-67FFF	Ø2CE
68	68ØØØ-68FFF	Ø2DØ
69	69ØØØ-69FFF	Ø2D2
6A	6AØØØ-6AFFF	Ø2D4
6B	6BØØØ-6BFFF	Ø2D6
6C	6CØØØ-6CFFF	Ø2D8
6D	6DØØØ-6DFFF	Ø2DA
6E	6EØØØ-6EFFF	Ø2DC
6F	6FØØØ-6FFFF	Ø2DE
7Ø	7ØØØØ-7ØFFF	Ø2EØ
71	71ØØØ-71FFF	Ø2E2
72	72ØØØ-72FFF	Ø2E4
73	73ØØØ-73FFF	Ø2E6
74	74ØØØ-74FFF	Ø2E8
75	75ØØØ-75FFF	Ø2EA
76	76ØØØ-76FFF	Ø2EC
77	77ØØØ-77FFF	Ø2EE
78	78ØØØ-78FFF	Ø2FØ
79	79ØØØ-79FFF	Ø2F2
7A	7AØØØ-7AFFF	Ø2F4
7B	7BØØØ-7BFFF	Ø2F6
7C	7CØØØ-7CFFF	Ø2F8
7D	7DØØØ-7DFFF	Ø2FA
7E	7EØØØ-7EFFF	Ø2FC
7F	7FØØØ-7FFFF	Ø2FE

Table 3-1 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
80	80000-80FFF	0300
81	81000-81FFF	0302
82	82000-82FFF	0304
83	83000-83FFF	0306
84	84000-84FFF	0308
85	85000-85FFF	030A
86	86000-86FFF	030C
87	87000-87FFF	030E
88	88000-88FFF	0310
89	89000-89FFF	0312
8A	8A000-8AFFF	0314
8B	8B000-8BFFF	0316
8C	8C000-8CFFF	0318
8D	8D000-8DFFF	031A
8E	8E000-8EFFF	031C
8F	8F000-8FFFF	031E
90	90000-90FFF	0320
91	91000-91FFF	0322
92	92000-92FFF	0324
93	93000-93FFF	0326
94	94000-94FFF	0328
95	95000-95FFF	032A
96	96000-96FFF	032C
97	97000-97FFF	032E
98	98000-98FFF	0330
99	99000-99FFF	0332
9A	9A000-9AFFF	0334
9B	9B000-9BFFF	0336
9C	9C000-9CFFF	0338
9D	9D000-9DFFF	033A
9E	9E000-9EFFF	033C
9F	9F000-9FFFF	033E

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
A0	A0000-A0FFF	0340
A1	A1000-A1FFF	0342
A2	A2000-A2FFF	0344
A3	A3000-A3FFF	0346
A4	A4000-A4FFF	0348
A5	A5000-A5FFF	034A
A6	A6000-A6FFF	034C
A7	A7000-A7FFF	034E
A8	A8000-A8FFF	0350
A9	A9000-A9FFF	0352
AA	AA000-AAFFF	0354
AB	AB000-ABFFF	0356
AC	AC000-ACFFF	0358
AD	AD000-ADFFF	035A
AE	AE000-AEFFF	035C
AF	AF000-AFFFF	035E
B0	B0000-B0FFF	0360
B1	B1000-B1FFF	0362
B2	B2000-B2FFF	0364
B3	B3000-B3FFF	0366
B4	B4000-B4FFF	0368
B5	B5000-B5FFF	036A
B6	B6000-B6FFF	036C
B7	B7000-B7FFF	036E
B8	B8000-B8FFF	0370
B9	B9000-B9FFF	0372
BA	BA000-BAFFF	0374
BB	BB000-BBFFF	0376
BC	BC000-BCFFF	0378
BD	BD000-BDFFF	037A
BE	BE000-BEFFF	037C
BF	BF000-BFFFF	037E

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
CØ	CØØØØ-CØFFF	Ø38Ø
C1	C1ØØØ-C1FFF	Ø382
C2	C2ØØØ-C2FFF	Ø384
C3	C3ØØØ-C3FFF	Ø386
C4	C4ØØØ-C4FFF	Ø388
C5	C5ØØØ-C5FFF	Ø38A
C6	C6ØØØ-C6FFF	Ø38C
C7	C7ØØØ-C7FFF	Ø38E
C8	C8ØØØ-C8FFF	Ø39Ø
C9	C9ØØØ-C9FFF	Ø392
CA	CAØØØ-CAFFF	Ø394
CB	CBØØØ-CBFFF	Ø396
CC	CCØØØ-CCFFF	Ø398
CD	CDØØØ-CDFFF	Ø39A
CE	CEØØØ-CEFFF	Ø39C
CF	CFØØØ-CFFFF	Ø39E
DØ	DØØØØ-DØFFF	Ø3AØ
D1	D1ØØØ-D1FFF	Ø3A2
D2	D2ØØØ-D2FFF	Ø3A4
D3	D3ØØØ-D3FFF	Ø3A6
D4	D4ØØØ-D4FFF	Ø3A8
D5	D5ØØØ-D5FFF	Ø3AA
D6	D6ØØØ-D6FFF	Ø3AC
D7	D7ØØØ-D7FFF	Ø3AE
D8	D8ØØØ-D8FFF	Ø3BØ
D9	D9ØØØ-D9FFF	Ø3B2
DA	DAØØØ-DAFFF	Ø3B4
DB	DBØØØ-DBFFF	Ø3B6
DC	DCØØØ-DCFFF	Ø3B8
DD	DDØØØ-DDFFF	Ø3BA
DE	DEØØØ-DEFFF	Ø3BC
DF	DFØØØ-DFFFF	Ø3BE

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
E0	E0000-E0FFF	03C0
E1	E1000-E1FFF	03C2
E2	E2000-E2FFF	03C4
E3	E3000-E3FFF	03C6
E4	E4000-E4FFF	03C8
E5	E5000-E5FFF	03CA
E6	E6000-E6FFF	03CC
E7	E7000-E7FFF	03CE
E8	E8000-E8FFF	03D0
E9	E9000-E9FFF	03D2
EA	EA000-EAFFF	03D4
EB	EB000-EBFFF	03D6
EC	EC000-ECFFF	03D8
ED	ED000-EDFFF	03DA
EE	EE000-EEFFF	03DC
EF	EF000-EFFFF	03DE
F0	F0000-F0FFF	03E0
F1	F1000-F1FFF	03E2
F2	F2000-F2FFF	03E4
F3	F3000-F3FFF	03E6
F4	F4000-F4FFF	03E8
F5	F5000-F5FFF	03EA
F6	F6000-F6FFF	03EC
F7	F7000-F7FFF	03EE
F8	F8000-F8FFF	03F0
F9	F9000-F9FFF	03F2
FA	FA000-FAFFF	03F4
FB	FB000-FBFFF	03F6

Table 3-6 (Cont'd). Memory Management I/O Port Addresses

BLOCK NUMBER	LOGICAL ADDRESS RANGE	ATTRIBUTE AND MAPPING PORT ADDRESS
FC	FC000-FCFFF	03F8
FD	FD000-FDFFF	03FA
FE	FE000-FEFFF	03FC
FF	FF000-FFFFF	03FE

* All reads from memory locations FE000-FFFFF access the EPROM. There is no way to access those locations in RAM.

Table 3-7. Violation Port (78)

Violation Port Bit Definition		
Bit	Bit Name	Detected
0	Invalid Instruction	1
1	RAM Parity ERROR	1
2	Not Used, Undefined	
3	End of Stack Warning	1
4	System Write Violation	1
5	Not Used, Undefined	
6	Not Used, Undefined	
7	User Mode Write Violation	1
8	System Bus Timeout Error (no activity on bus for 10 ms)	1
9	Not Used, Undefined	
10	I/O Processor Write Violation	1
11	User Mode Access Violation	1
12	Status Bit 0* -system config. jumper	
13	Status Bit 1* -system config. jumper	
14	Not Used, Undefined	
15	Manual Switch NMI (jumper to halt program for software development)	1

* A0 indicates that the jumper is in place and a 1 indicates that the jumper is not in place.

Table 3-8. Error Address Port 1

BIT	BIT NAME	TRUE
Ø	MEMORY ERROR ADDRESS Ø	1
1	MEMORY ERROR ADDRESS 1	1
2	MEMORY ERROR ADDRESS 2	1
3	MEMORY ERROR ADDRESS 3	1
4	MEMORY ERROR ADDRESS 4	1
5	MEMORY ERROR ADDRESS 5	1
6	MEMORY ERROR ADDRESS 6	1
7	MEMORY ERROR ADDRESS 7	1
8	MEMORY ERROR ADDRESS 8	1
9	MEMORY ERROR ADDRESS 9	1
1Ø	MEMORY ERROR ADDRESS 1Ø	1
11	MEMORY ERROR ADDRESS 11	1
12	MEMORY ERROR ADDRESS 12	1
13	MEMORY ERROR ADDRESS 13	1
14	MEMORY ERROR ADDRESS 14	1
15	MEMORY ERROR ADDRESS 15	1

Table 3-9. Error Address Port 2

BIT	BIT NAME	TRUE
Ø	Not Used	1
1	Not Used	1
2	Not Used	1
3	Not Used	1
4	Not Used	1
5	Not Used	1
6	Not Used	1
7	Not Used	1
8	USER MODE	1
9	WARM START	1
1Ø	Not Used, Undefined	1
11	ENABLE NMI	1
12	MEMORY ERROR ADDRESS BIT 16	1
13	MEMORY ERROR ADDRESS 17	1
14	MEMORY ERROR ADDRESS 18	1
15	MEMORY ERROR ADDRESS 19	1

Z80 I/O Addresses

The following table lists Z8ØA addresses. The Z8ØA has a 256 byte I/O address space. Only the addresses shown in the table should be used because not all of the I/O addresses are uniquely decoded.

Table 3-1Ø. Z8ØA I/O Addresses

Device	Hex Address	Function
Address Latch	ØØH	System memory block number (bits Ø thru 4)
Counter Timer Ø	2ØH	Counter Ø provides baud rate for port 3
" " "	21H	Counter 1 provides baud rate for port 4
" " "	22H	Counter 2 provides baud rate for port 1
" " "	23H	Control byte for Counter timer Ø
Counter Timer 1	24H	Counter Ø provides baud rate for port 2
" " "	25H	Counter 1 provides baud rate for port 5
" " "	26H	Counter 2 timer interrupt
" " "	27H	Control byte for Counter Timer 1
SIO Ø	28H	Channel A data for serial port 3
" "	29H	Channel A control for serial port 3
" "	2AH	Channel B data for serial port 4
" "	2BH	Channel B control for serial port 4
SIO 1	2CH	Channel A data for serial port 1
" "	2DH	Channel A control for serial port 1

Table 3-1Ø (Cont'd). Z8ØA I/O Addresses

Device	Hex Address	Function
" "	2EH	Channel B data for serial port 2
" "	2FH	Channel B control for serial port 2
SIO 2	3ØH	Channel A data for serial port 5
" "	31H	Channel A control for serial port 5
" "	32H	Channel B data for serial port 6
" "	33H	Channel B control for serial port 6
PIO	34H	Data port A
"	35H	Command port A
"	36H	Data port B
"	37H	Command port B
DMA Controller	3CH	All read and write registers
" "	4ØH	Clear carrier sense and parity error bit
Calendar Clock Chip	8ØH	Counter - thousandths of seconds
" "	81H	Counter - hundredths and tenths of seconds
" "	82H	Counter - seconds

Table 3-10 (Cont'd). Z80A I/O Addresses

Device	Hex Address	Function
" "	83H	Counter - minutes
" "	84H	Counter - hours
" "	85H	Counter - Day of Week
" "	86H	Counter - Day of Month
" "	87H	Counter - Months
" "	88H	Latches - Thousandths of seconds
" "	89H	Latches - Hundredths and tenths of seconds
" "	8AH	Latches - Seconds
" "	8BH	Latches - Minutes
" "	8CH	Latches - Hours
" "	8DH	Latches - Day of the Week
" "	8EH	Latches - Day of the Month
" "	8FH	Latches - Months
" "	90H	Interrupt Status Register
" "	91H	Interrupt Control Register
" "	92H	Counter Reset
" "	93H	Latch Reset
" "	94H	Status Bit

Table 3-1Ø (Cont'd). Z8ØA I/O Addresses

Device	Hex Address	Function
" "	95H	"GO" Command
" "	96H	Standby Interrupt
" "	9FH	Test Mode

Note - The baud rate for port 6 is provided by the 8254 which is connected to the 8Ø86. See I/O **Processor** section.

Chapter 4
Controller Board Functions

Hardware Overview

The function of the Controller board is to provide fast data transfer between the host CPU and the system peripheral devices: hard disk, floppy disk, and streaming tape unit. All three devices can operate concurrently. Communication between the Controller board and the CPU board takes place over the Multibus (see Figure 4-1).

Many of the subsystems on the Controller are programmable and depend on the operating system or the Controller board's resident program (Channel Control Program) for their parameters. This provides the systems programmer with many options in designing a system. Not all of the subsystem programmable features are listed here. Consult the manufacturer's data manual for more information about each device. Following is a summary of the Controller board hardware.

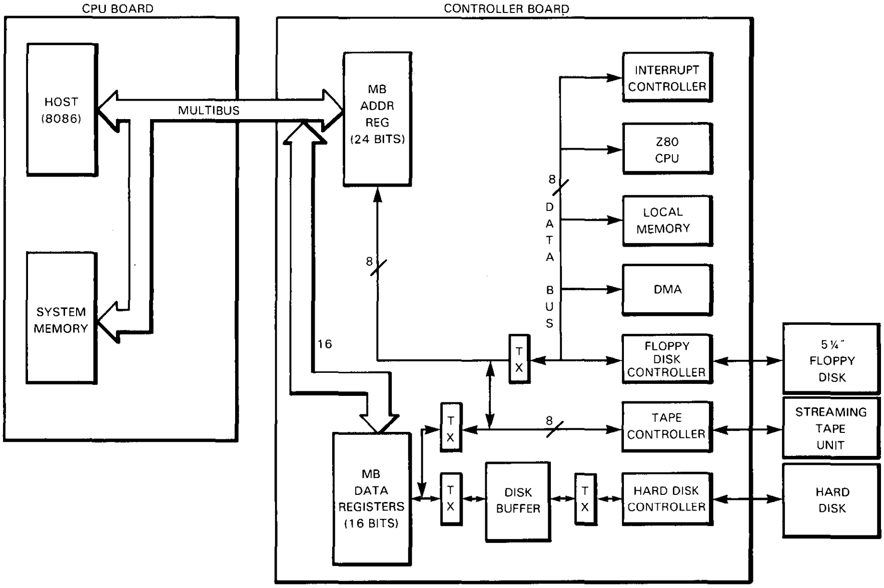


Figure 4-1. Simplified Block Diagram

Z-80A Processor

The Z80A processor, running at 4 MHZ, operates as a slave to the host CPU. When the host wishes to perform an I/O operation, it can request the services of the hard disk, floppy disk, streaming tape, or all three, concurrently. It can then continue processing without waiting for the response.

Every I/O request from the host to the Z80A is sent via the local Interrupt Controller. If the Z80A is "busy", the interrupt remains pending in the Interrupt Controller. When the Z80A acknowledges the interrupt, it receives information from the Interrupt Controller about the type of interrupt. It then must obtain further information about the I/O operation from a parameter block stored in the host memory.

Since the Z80A cannot access host memory, the parameter block must be moved to local memory via a memory-to-memory transfer operation (see **Memory Write**). The Z80A reads the parameter block and then executes the operation using routines from the Channel Control Program, previously stored in local memory. (see **Controller Initialization**).

The Z80A and Channel Control Program also service interrupts from local I/O devices, such as the hard disk, streaming tape, floppy disk, and DMA controller, when an I/O operation is completed. The Z80A reads the end-of-job status of each device and reports the results to the host processor.

A non-maskable interrupt is sent to the Z80A when a power failure or Multibus parity error occurs. When a power failure is sensed, the write gate to the Hard Disk is immediately dropped so that the Hard Disk is unable to write.

DMA Controller

The DMA (Direct Memory Access) Controller handles data transfers to and from local memory and an external device. The external device may be the Floppy Disk Controller buffer, the Tape Controller buffer, or the Multibus Interface registers. Internal registers in the DMA Controller chip (AM9517) can be programmed by the Z80A to transfer a single word or a block of data at a time. (see **Controller Sequences** for details on each DMA operation).

NOTE

Hard disk data transfers do not use Local DMA control.

Local Memory

Up to 64K of local memory is addressable by the Z8ØA. The first 8K of memory is allocated to programmable read-only memory (PROM) which stores the power-up diagnostic code and the system boot-up code. Dynamic random access memory (DRAM) is located from 32K to 48K and contains the Channel Control Program used by the Z8ØA for I/O processing and for temporary storage (see **Controller Initialization**). Table 4.1 is a memory map.

Table 4-1. Local Memory Map

ØK - 8K	PROM
8K - 32K	Not Used
32K - 48K	DRAM
48K - 64K	Not Used

Data Buses

The Local Data Bus is eight bits wide to conform to the Z8ØA protocol. It allows data transfer between the Z8ØA and the Interrupt Controller, Local Memory, DMA controller, and Floppy Disk Controller. Because the Local Data Bus is isolated from the Buffered Data Bus and the Byte Bus by transceivers (marked TX in Figure 4-1) operations on one bus do not interfere with another bus so long as the transceivers are in an inactive state. A transceiver becomes active when data is to be moved onto or off a data bus.

Interrupt Controller

All I/O operations begin and end with the generation of an interrupt. Interrupts are received by the Interrupt Controller (AM9519), which stores the interrupt and determines a priority. The interrupt lines are hardwired to the Interrupt Controller as shown in Figure 4-2. Normally, a disk interrupt (level Ø) has the highest priority and the programmable timer (level 7) has the lowest. However, priorities can be changed by programming a mask in the Interrupt Controller.

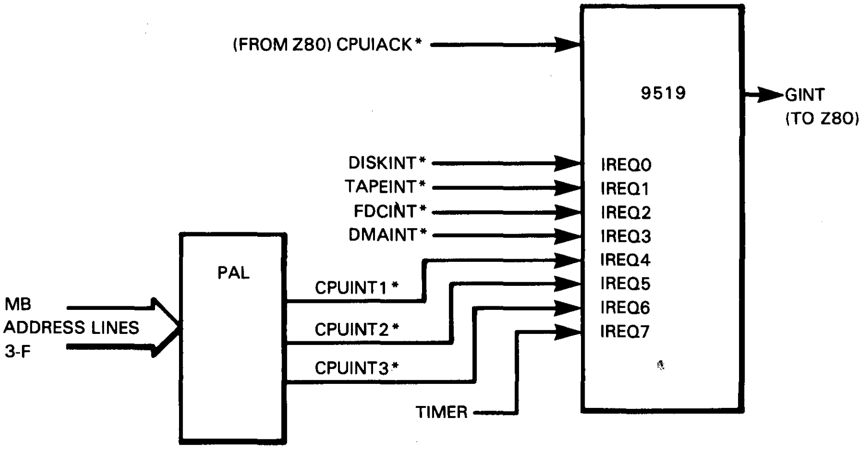


Figure 4-2. Interrupt Controller

Hard Disk Controller

The Hard Disk Controller (WD2Ø1Ø) combines into a single chip all of the standard circuitry needed to interface a host processor to ST5Ø6 type Winchester disk drives. The Channel Control Program implements the following capabilities of the controller:

- Multiple sector read and writes
- Ability to cross over track and sector boundaries

- o Automatic, transparent retries
- o Overlapped seeking when two or three drives are connected and accessed concurrently.
- o ECC checking.

Programming

The WD2Ø1Ø performs the basic commands issued by the host to operate the disk drive such as Restore, Seek, Read Sector, Write Sector, Scan ID, etc. Before sending the command, the Z8ØA must program the WD2Ø1Ø's internal registers. In addition, an external Select-Drive-Head (SDH) register (I/O address 192, decimal) must be loaded with head and drive select information. See **ADDRESSING** for the bit assignments of the external SDH register.

Sector Format

The format of each sector on the disk is determined by the disk controller's Format command. A diagram of the sector format is shown in Chapter 5.

Tape Controller

The Tape Controller is a self-contained electronic package with sufficient intelligence to carry out tape read, write, and movement commands sent to it by a host computer. It communicates on one side with the host using the industry-standard QIC-Ø2 interface, and on the other side with the streaming tape unit using the QIC-36 interface.

The QIC-Ø2 interface is asynchronous, employing handshaking and a minimum number of control lines (see Figure 4-3). QIC-36 is also known as the Basic Streaming Tape Interface (BSTI). Drives using this standard are designed to operate in a streaming manner and record data at 1Ø,ØØØ flux transitions per inch. QIC-36 is discussed more fully in the 586T/986T Maintenance Manual (P/N 69Ø-156Ø2-ØØ1).

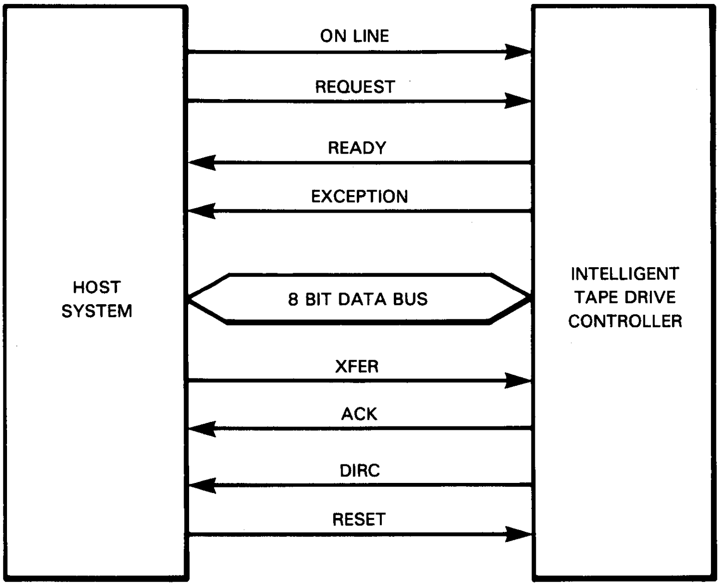


Figure 4-3. QIC-Ø2 Interface

Programming

Seven commands, consisting of eight bits each, can be programmed into the controller Command register by the host. The three most significant bits, 7,6, and 5 define the type of command. Bits 4,3,2,1 and Ø contain the command bits.

Table 4-2. Tape Command Bit Definition

Command	Bits 7,6,5	Bits 4,3,2,1,Ø
Select (drive)	ØØØ	MMMMM (ØØØØ1 selects drive Ø)
Position	ØØ1	ØØMMM (ØØØ1Ø is erase tape; ØØØØ1 rewinds to BOT)
Write Data	Ø1Ø	ØØØØØ
Write File Mark	Ø11	ØØØØØ
Read Data	1ØØ	ØØØØØ
Read File Mark	1Ø1	ØØØØØ
Read Status	11Ø	ØØØØØ

M= Ø or 1

(See Figure 4-3.) The controller accepts a command when the READY line is true. One command, the Read Status command, is accepted when READY is not true, provided EXCEPTION is also true. This allows the host to read the Tape Controller's status when there is an error condition.

In the 986T design configuration, ONLINE is always true, and therefore, need not be considered when programming. Tape commands are discussed in detail in Chapter 5.

General Operation

When the host (Z8ØA) is ready to write data on the tape, it begins transferring the data in 512 byte blocks via the DMA Controller. (The Tape Controller has three 512 byte buffers for temporary storage.) When the first of the three buffers is full, the controller starts tape motion and begins writing to the tape, while, at the same time, accepting data for the second and third buffers. The DMA transfer rate is approximately 2ØØ Kbytes/sec.

The data is written on the tape in bit serial format, one track at a time (see Chapter 5 for tape format).

The controller writes a gap and sync mark preceding each 512 byte block of user data. After writing the data block, the controller records a one-byte block address, which indicates the number of the block. The address is incremented by one after each block of data. Following the block address, the controller writes the CRC character. The process is repeated for each block of data to be written.

The above method of writing a continuous stream of data is referred to as "streaming tape mode". For every 512 bytes of data stored, 531.5 to 551 bytes are written to the tape. Because conventional inter-record gaps are not used, tape utilization is about 97 percent. See Chapter 5 for more information about the streaming tape unit.

Error Checking

A Cyclic Redundancy Check (CRC) is performed when writing to tape or reading from tape. A CRC error in the write mode causes a block to be rewritten until no CRC error occurs or until the limit of 16 consecutive same block re-writes occurs. A CRC error in the read mode causes a read re-try until no CRC error occurs or the limit of 16 read re-tries have been made.

Floppy Disk Controller

The Floppy Disk Controller (FDC) is an LSI chip, the uPD765, which contains the circuitry for interfacing a processor (the Z8ØA) to one or more (up to four) floppy disk drives. In this application, it supports a single double-sided, double-density (MFM) 5-1/4 inch drive. The FDC is fed by an external digital data separator.

Handshaking signals are provided to interface the FDC to the DMA Controller (9517). Thus the processor need only load the command into the FDC, after which the data transfer occurs under control of the FDC and the DMA Controller. The FDC is capable of multisector transfers. The DMA transfer rate is approximately 32 Kbits/sec.

Fifteen commands can be executed by the FDC, including the basic read, write, scan, format, seek, recalibrate, sense status, and their variations. Details of each command are found in the manufacturer's data handbook.

Programming

Two internal registers in the FDC may be accessed by the Z80A: a status register and a data register. The 8-bit status register may be read at any time. The data register actually consists of a stack of 8-bit registers, only one of which can be latched to the bus at one time.

The values for track stepping rate, head load time, and head unload time, are set by the Controller board's firmware during Controller initialization.

Sector Format

The data record length used in this application is 512 bytes. For more information about sector format see Chapter 5.

CPU-Controller Interface

Multibus Interface

Multibus is a protocol for transferring information between systems which are similar or disparate. Its structure consists of address lines, data lines, and control lines. The Multibus specifications determine the timing of the signals on the Multibus, and the order in which they occur. This permits the Multibus interface to connect to any system which has a compatible Multibus interface.

All communication between the 586T/986T CPU board and the controller board takes place over the Multibus interface. Two 50-pin cables carry the Multibus signals between J1/J2 on the CPU board and J1/J2 on the controller board (see System Block Diagram).

Host-to-Controller Board Communication Protocol

Handshaking between the host processor(8Ø86) and the Z8ØA is via six interrupt signals; three from the host to the controller board and three from the controller board to the host. Following are the interrupt signal definitions:

Host to Z8ØA:

CPUINT1* requests a hard disk or memory-to-memory operation
CPUINT2* requests a floppy disk operation
CPUINT3* requests a tape operation.

Multibus addresses to receive the above interrupts:

Hex	Decimal	Interrupt
7ØØØ - 7ØØ7	28672 - 28679	CPUINT1*
7ØØ8 - 7ØØF	2868Ø - 28687	CPUINT2*
7Ø1Ø - 7Ø17	28688 - 28695	CPUINT3*

NOTE

Since Multibus address lines Ø - 2 are not decoded at the Interrupt Controller, the lower three bits Ø - 7 do not have an effect.

Z8ØA to Host:

INTOUT1* indicates the end of a hard disk or memory-to-memory operation.
INTOUT2* indicates the end of a tape operation
INTOUT3* indicates the end of a floppy disk operation.

To generate a Multibus interrupt the Z8ØA performs an I/O write to address 6ØH. The bit definitions are as follows:

Bit	Name	Host Level
Ø - 4	Not used	
5	INTOUT1*	Level 2
6	INTOUT2*	Level 3
7	INTOUT3*	Level 7

Parameter Blocks

Before the host generates an interrupt, it must set up a parameter block in main memory and also a pointer with which the controller board Channel Control Program can access the parameter block. The parameter block contains information about the operation to be performed.

The pointer for a disk or memory-to-memory parameter block is located at physical address 4ØØH. For the floppy disk and tape parameter blocks the pointers are at 4Ø4H and 4Ø8H, respectively. The following tables define the bytes for each parameter block.

Table 4-3. Hard Disk Interface Parameter Block

Byte	Content	Description
Ø	read, write, format command	configuration command
1	status	status
2	cylinder-low	last cylinder-low
3	cylinder-high	last cylinder-high
4	head and drive	last head and drive
5	starting sector no.	last sector no./track
6	byte count-low	
7	byte count-high	
8	system memory address-offset low	
9	system memory address-offset high	
1Ø	system memory address-segment low	
11	system memory address-segment high	
12	reserved	
13	number of sectors processed before error	
14	reserved	
15	job done	

Total = 16 bytes

```
Byte Ø (Command Byte):    1 = read
                          2 = write
                          4 = configuration
                          5 = host parameter block address change
```

If the command is 5, then a new address must be supplied to the four address bytes in the parameter block.

```
Byte 1 (Status byte): 7 6 5 4 3 2 1 Ø
                      x x x x x x x x (binary)
                      | | | | | |   |
                      | | | | | |   marginal sector
                      | | | | | bad sector
                      | | | | record not found
                      | | | ecc error
                      | | not ready
                      | write fault
                      operation aborted
      status byte = Ø means no errors
```

NOTE: The status byte above is reported to the screen by the operating system when there is a hard disk failure. However, if the operating system is not loaded, or the system fails to boot, then status is reported by the system monitor as it is received from the Hard Disk controller chip, and not as above. See Hard Disk Status Byte, Chapter 5.

Table 4-4. Streaming Tape Interface Parameter Block

Byte	From the host	To the host
0	command	
1	status	status
2	transfer count- low	
3	transfer count- high	
4	system memory address- offset low	
5	system memory address- offset high	
6	system memory address- segment low	
7	system memory address- segment high	

total 8 bytes

Byte 0 (command byte):

1 = read
2 = write
3 = retension
4 = select
5 = erase
6 = rewind to BOT
7 = read file mark
8 = write file mark
9 = read status
10 = parameter block address change

If command = 10, then a new address must be supplied to the parameter block in bytes 4 - 7.

```
Byte 1 (Status byte): x x x x x x x x (binary)
                      | | | | | | | |
                      | | | | | | | cartridge not in place
                      | | | | | | write protected
                      | | | | | end of media
                      | | | | data errors
                      | | | BIE not located
                      | | file mark detected
                      | no data detected
                      eight or more retries
```

Note: The status byte above is reported by the operating system if there is a tape failure during normal operation. If the operating system is not loaded, tape status is reported differently. See Chapter 5.

Table 4-5. Floppy Disk Interface Parameter Block

Byte	Content
Ø	command
1	status
2	drive
3	cylinder
4	surface(head)
5	start sector
6	system memory address- offset low
7	system memory address- offset high
8	system memory address- segment low
9	system memory address- segment high
1Ø	transfer count- low
11	transfer count- high

total 12 bytes

```
Byte Ø:        1 = read sector          3 = format a track
(Command)      2 = write sector         4 = parameter block
                                            address change
```

If command = 4, then a new adress must be supplied to the four address bytes in the parameter block.

```
Byte 1:        x x x x x x x x (binary)
(Status)       | | | | | |   |
               | | | | | |   job is done
               | | | | | lost data
               | | | | crc error
               | | | record not found
               | | write fault
               | write protect
               not ready
```

Note: The status byte above is used by the operating system during normal operation. If the operating system has not loaded, status is reported to the system monitor. See Chapter 5.

Controller Initialization

Power-up

During power-up of the 586T/986T system, control is switched to the PROM device on the controller board. In the first phase of the initialization, diagnostic code in the PROM is used to check out systems on the controller board. If a faulty component is detected, the failure is reported to the on-board LED's.

The system continues to operate as long as a catastrophic failure does not occur. Refer to **SYSTEM FIRMWARE**, Chapter 6, for a discussion of the diagnostic tests.

Channel Control Program

After a successful power-up, and completion of the diagnostic tests, the host CPU starts the boot-up process. The boot-up code written into a PROM in local memory is loads a portion of the operating system into main memory from the operating system resident device (hard disk or floppy).

During the final phase of the boot-up process, a portion of the operating system comprising the Channel Control Program (CCP) is loaded into local RAM memory. The CCP contains the drivers for the hard disk, floppy disk, and streaming tape subsystems.

After the controller board has been initialized; that is, the diagnostic tests have run and the channel control program has been loaded into local memory, all controller input/output operations are then governed by the Channel Control Program and the Z8ØA I/O processor. However, an I/O operation is always started by the CPU board host processor.

Controller Sequences

The following paragraphs describe the controller sequences for data transfer operations. They are:

- Write Main Memory, Read from Hard Disk (Disk Read)
- Read Main Memory, Write to Hard Disk (Disk Write)
- Read Main Memory, Write to Local Memory (Memory Write)
- Write Main Memory, Read from Local Memory (Memory Read)
- Tape - Local Memory Transfers
- Floppy Disk - Local Memory Transfers

NOTE

Data cannot be transferred between local memory and the Hard Disk.

Disk Read

1. The host requests a disk read operation by sending **CPUINT1*** to the Interrupt Controller (see Figure 4-2).
2. The Z8ØA checks the channel busy bit (**CBUSY**, Input Port, bit 3) to see if the previous disk operation has completed; if it has not, it branches to another routine.
3. The Z8ØA I/O processor, operating under the control of the CCP (Channel Control Program), acknowledges the interrupt and initiates a memory-to-memory transfer (see **Memory Write**). A parameter block of 16 bytes is copied from a predefined location in main memory to local memory.

4. The Z80A extracts information from the parameter block, such as cylinder number, head number, sector number, and number of bytes to be read, and initializes the Hard Disk Controller (WD2010). The parameter block also supplies an address for the Multibus Interface Address Buffers. The address is the starting location in main memory to which the data is to be moved.

5. The Z80A issues a Read Sector command to the Hard Disk Controller. Data is read from the disk a sector at a time (512 bytes), then converted from serial to parallel in the WD2010, and moved to the disk buffers.

6. The signals **DKTXFR** and **MBREAD** (Data Transfer Port) are asserted by the Z80A to start the Disk Buffer Control sequence and the Data Transfer Control sequence (see Controller Block Diagram). Data is moved from the disk buffers via the Multibus to main memory.

7. When the Hard Disk Controller has read the number of sectors commanded by the Read Sector command, it generates **DISKINT*** (see Figure 4-2).

8. The Z80A receives **GINT** from the Interrupt Controller and waits until **CBUSY** (off) has been received from the Data Transfer Control logic. The disk buffers and the Multibus control circuits are reset.

9. After reset, the Z80A reads the status of the Hard Disk Controller.

10. Using the status information received above, the Z80A composes a new parameter block which is returned to main memory. The Z80A then generates **INTOUT1*** to the host processor. The host processor reads the parameter block which indicates whether or not the disk read was successful.

Disk Write

A disk write operation is similar to a disk read, except that data moves in the opposite direction. The signals **DKTXFR** and **MBREAD*** (see Data Transfer

Port) are issued by the Z8ØA to start the Multibus and Disk Buffer control sequences. The Z8ØA issues a Write Sector instead of a Read Sector command to the Hard Disk Controller. Data moves from the disk buffers to the Hard Disk Controller, is converted from parallel to serial, and is written onto the disk a sector at a time.

When the number of sectors programmed has been written, the Hard Disk Controller generates a disk interrupt. Since the disk interrupt is not issued until the last sector has been written, the disk buffers and the Multibus control logic are not reset until that time. Therefore, an additional sector beyond the last one is stored in the disk buffers, but that data is not used.

Memory Write

To transfer data from main memory to local memory, the host issues a **CPUINT1*** (Figure 4-2). The sequence is the same as for a disk read operation up to step 4. The Z8ØA programs the DMA Controller (9517) with address information and number of bytes to be transferred and loads the Multibus Address Registers with the starting address in main memory. The signals **MBREAD*** and **MMTXFR** (Data Transfer Port) are issued to start the Multibus control logic.

The Data Transfer Control logic enables the DMA Controller. Data in word form (16 bits) is moved from main memory to the Multibus interface registers and then is transferred in byte form (8 bits) over the local data buses to local memory. Each DMA cycle moves two bytes. The DMA signals the end of the transfer by generating an interrupt **DMAINT***. At this time, the Multibus control logic has fetched an extra word because it is not reset until the DMA issues an interrupt. The extra word is not used.

Memory Read

A local memory to main memory transfer uses the same procedure as outlined above, with some slight differences. The Multibus control sequence is started with the signals **MMTXFR** and **MBREAD.**

Tape-Local Memory Transfers

The streaming tape unit is used in two ways- to back up an entire disk image, or to back up a file. The operations of a tape read or write are quite complex, and involve several sequences. For a disk backup, data is first moved from the disk to main memory (see **Disk Read**). Then it is moved to local memory (see **Memory Write**). Finally, it is transferred to tape from local memory.

Since the disk reads data faster than the tape unit can write it, data buffering is employed. Two 5-block buffers are maintained in local memory (Each block = 512 bytes, equivalent to a sector of data). Data from the disk is stored in one buffer, while data is retrieved from the other buffer to be written to the Tape Controller RAM (buffers).

Conveniently, the tape records data in 512 byte blocks. Since the streaming tape unit does not start and stop, the buffers are used in a ping-pong fashion, similar to the disk buffers, to maintain a constant flow of data to the tape buffers. The DMA transfer rate is approximately 200 Kbytes/sec.

The following describes the sequence of events for transferring data between local memory and tape.

1. The host requests a tape operation by sending **CPUINT3*** to the Interrupt Controller.
2. After acknowledging the interrupt in the usual fashion (see **Interrupt Controller**), the Z80A retrieves a parameter block as described under **Disk Read.**
3. The Z80A initializes the DMA controller with address and block size information.
4. For a tape write operation, the Z80A issues an enable signal to the Tape Command register allowing it to be loaded by the Z80A.
5. When the Tape Controller sends tape READY (the signal is **TRDY** at the Input Port), the Z80A sends the tape Write command according to the

QIC-Ø2 protocol. The communication lines between the Tape Controller and the Z8ØA are through the Input port and the Tape port.

NOTE

QIC-Ø2 is an industry standard interface for tape drives. See Figure 4-3.

For a tape read operation, the sequence is similar, except that the Z8ØA loads a Read command into the Tape Command register.

The tape controller starts DMA action.

6. The **TDMA** bit is set in the Tape Port, requesting DMA service.

7. The Tape Controller starts the tape moving, and data transfer is enabled between the Tape Controller RAM and the tape unit.

8. For a write operation, DMA action is started by setting the **TWR** bit in the Tape Port.

9. Handshaking takes place between the Tape Controller and the host (Z8ØA) for each byte that is transferred. At the end of a block (512 bytes) the Tape Controller sets **TRDY** true, indicating to the host that the Tape Controller is ready for the next block.

1Ø. The host can terminate a write sequence by issuing a Write File Mark command. The Controller resets **TRDY**, writes a file mark, and rewinds the tape to BOT (beginning of tape).

11. After the tape is rewound to BOT, the Controller sets **TRDY** again, indicating that it is ready to accept the next command.

Floppy Disk-Local Memory Transfers

Data is transferred between the Floppy Disk Controller (FDC) registers and local memory in much the same way as for tape read or write.

The host requests an FDC operation by issuing **CPUINT2*** to the local Interrupt Controller. The Z80A retrieves a Floppy Disk parameter as described under **Disk Read.** The DMA controller is initialized with address and block size information.

Before a read or write command (or any command involving disk movement) is issued to the Floppy Disk drive, the Floppy drive motor must be turned on. The motor is commanded by the **MOTORON** signal from the Floppy Port (see Chapter 5 for Floppy drive considerations).

Addressing

Addressing is a method by which a computer can select or enable one or more of the peripheral devices under its control. A peripheral device always includes a chip or a group of chips.

The Z80A uses all of its address lines (16) to access local memory which gives it the ability to address up to 64K bytes (2**16 = 64K).

I/O Address Decoders and Ports

Fewer address lines are needed to address I/O devices. Typically, the number of devices addressed is 256 or less. This requires, at most, eight address lines. Decoder chips, that decode the address, cause a particular device to be enabled.

After being selected, devices often need control information. Control information is placed on the data bus and gated to the correct device(s) using a port (a port is generally an entry or an exit to a bus).

The following table shows address assignments for the signals output by the I/O decoders for both read and write operations.

Table 4-6. I/O Decoder Addresses

Address					
(Decimal)	(Hex)	BIOW*	BIOR*	Signal	Selects
0 - 7	0 - 7	x	x	CSDISK*	Hard Disk Controller
16	10	x	x	CSFDC*	Floppy Disk Controller
32	20	0	1	FPORT*	Floppy Port
48	30	0	1	TCOMMAND*	Tape Command Register
48	30	1	0	TAPE*	Tape Status
64	40	0	1	TPORTCS*	Tape Port
80	50	0	1	DTPORTCS*	Data Transfer Port
96	60	1	0	ENINPUTP*	Input Port
96	60	0	1	LDMBINT*	Load Multibus Int.
112	70	0	1	LDCTR1*	MB Addr. Counter 1
128	80	0	1	LDCTR2*	MB Addr. Counter 2
144	90	0	1	LDCTR3*	MB Addr. Counter 3
160	A0	x	x	CSINT*	Int. Controller
176-191	B0-BF	x	x	CSDMA*	DMA Controller
192	C0	0	1	WRSDH*	Write SDH register

x = don't care

NOTE: Since address lines 0,1 and 2 are not used, bits 0-7 are don't cares.

Bit Assignments

Table 4-7. SDH Register (21F)

Bit	Assignment
0	HEADSEL 0
1	HEADSEL 1
2	HEADSEL 2
3	HEADSEL 3
4	DRIVESEL 0
5	DRIVESEL 1
6	not used
7	not used

The signals HEADSEL 0-3 are decoded by the disk drive to select up to 16 heads.

HEADSEL 3 can be jumpered on the Controller board to be a "low current" signal.

DRIVESEL Ø-1 signals are decoded to generate drive select signals for three drives.

Table 4-7. Input Port

Bit	Signal	Explanation
Ø	TEXCEPT	Tape Exception
1	TRDY	Tape Ready
2	TDIR	Tape Direction
3	CBUSY	Channel Busy
4	DRIVESELD	A hard disk drive 1,2, or 3 has been selected.
5	MBPERR	Multibus Parity Error
6	POWERFAIL	Indicates a Power Supply Interruption
7	TSELD	Tape unit has been selected

Table 4-8. Floppy Port

Bit	Signal	Explanation
Ø	PLCT	
1	SOFTREADY	
2	FTC	
3	QDEN	
4	MOTORON	Floppy motor on
5	SOFTBDRQ	
6	Clock	Diagnostic LED #1 (LED on = 1)
7	Data	Diagnostic LED #2 (LED on = 1)

Table 4-9. Tape Port

Bit	Signal	Explanation
Ø	TREQ	Tape Request
1	TDMA	Configure for tape DMA
2	TWR	Start tape DMA request
3	ENRDYINT	Enable Ready Interrupt
#4	TRESET*	Tape Reset
5	THD	Tape Threshold
6	not used	
7	not used	

Used for software reset of the Tape Controller. TRESET* must be active for 25 microseconds or longer. TRESET* is not needed for normal operation.

Table 4-1Ø. Data Transfer Port

Bit	Signal	Explanation
Ø	INITMBSQR*	Reset Multibus Sequencer
1	INITBFR*	Reset Disk Buffer
2	DKTXFR	Multibus Disk Transfer
3	MMTXFR	Memory-to-Memory Transfer
4	MBREAD	Multibus Read
#5	DRESET*	Reset H.D. Controller (25 usec min)
6	RESETERR	Reset MB Parity Error and Power Fail latches (This bit must be turned off after being turned on)
7	ONEFET	

After a hard disk reset, an additional 25 microsecond wait period is required before the WD2Ø1Ø can be programmed.

Chapter 5
Storage Devices

Streaming Tape Unit

Description

The tape unit is a 1/4-inch streaming cartridge tape drive packaged in a 5 1/4-inch footprint that is compatible with the 5 1/4-inch floppy disk footprint. Its primary function is backup for the Winchester disk drive.

Features of the tape unit are as follows:

- Space-saving, light-weight assembly
- High-capacity, low-cost media
- High data transfer rate
- High data integrity

The tape drive is connected to the Controller board by a single 5Ø conductor ribbon cable. For experimental purposes the drive may be moved a maximum of 3 meters away from the controller.

Performance Summary

As shown in Table 5-1, when an industry-standard 1/4-inch magnetic tape cartridge is loaded into the tape drive (see Figure 5-1), 45 or 6Ø Mbytes of data can be stored or backed-up in one 1/4-inch tape cartridge.

The tape drive has a data transfer rate of 9Ø Kbytes per second. It can back up 45 Mbytes in 9 minutes, 6Ø Mbytes in 12 minutes.

Other possible uses for the tape unit are transaction logging, data collection, data exchange, and program loading.

Table 5-1. Performance Specifications Tape

No. of Tracks	9
No. of Channels *	2
Capacity DC 300XL	45 Mbytes
Capacity DC 600A	60 Mbytes
Backup Time DC 300XL	9 min.
Backup Time DC 600A	12 min.
Recording Mode	NRZI
Recording Data Density	8000 bpi
Encoding Method	4-to-5 RLL **
Flux Density	10,000 ftpi
Track Capacity DC 300XL	5.0 Mbytes
Track Capacity DC 600A	6.6 Mbytes
Data Transfer Rate	90 KBytes/sec
Tape Speed	90 ips
Start/Stop Time	300 ms

* Channel is defined as one write head gap followed by one read head gap.

** RLL is defined as "Run Length Limited".

Data Integrity

The tape drive uses AC erase. This eliminates the problem of peak shift caused by DC erase, which biases the media. This method allows recording at extremely high density. The brushless DC drive motor speed is tightly regulated by a digital servo system.

A Cyclic Redundancy Check (CRC) is performed when writing to tape or reading from tape (see Error Checking in Chapter 4).

A technique that ensures reliable data recovery is the use of a wide band phase-locked-loop which follows instantaneous speed variations. A 4-to-5 bit run-length limited coding technique is used to maintain precise synchronization of the data detection system (see Table 5-2).

The 4-to-5 code translates 4 bit nybbles of data to 5-bit nybbles of encoded data. By design there are never more than two consecutive zeros in the data stream, regardless of how the five-bit-encoded nybbles are strung together.

Table 5-2. 4-to-5 Run Length Limited Code

HEX	BINARY		HEX	BINARY
00	0000	=	19	11001
01	0001	=	1B	11011
02	0010	=	12	10010
03	0011	=	13	10011
04	0100	=	1D	11101
05	0101	=	15	10101
06	0110	=	16	10110
07	0111	=	17	10111
08	1000	=	1A	11010
09	1001	=	09	01001
0A	1010	=	0A	01010
0B	1011	=	0B	01011
0C	1100	=	1E	11110
0D	1101	=	0D	01101
0E	1110	=	0E	01110
0F	1111	=	0F	01111
PREAMBLE		=	1F	11111
POSTAMBLE		=	1F	11111
DATA BLOCK MARKER		=	07	00111
FILE MARK		=	05	00101

Storage Media

The tape drive uses the DC 300XL (450 feet of tape) or the DC 600A (600 feet of tape) 1/4-inch tape cartridge. Different write currents are required for the two cartridges due to the difference in oxide coating thickness and coercivity. The tape controller determines the cartridge type by measuring the distance between the BOT (Beginning of Tape) and the load point holes (3 feet for the DC 300XL and 4 feet for the DC 600A). It then selects the appropriate write current for the cartridge which is inserted.

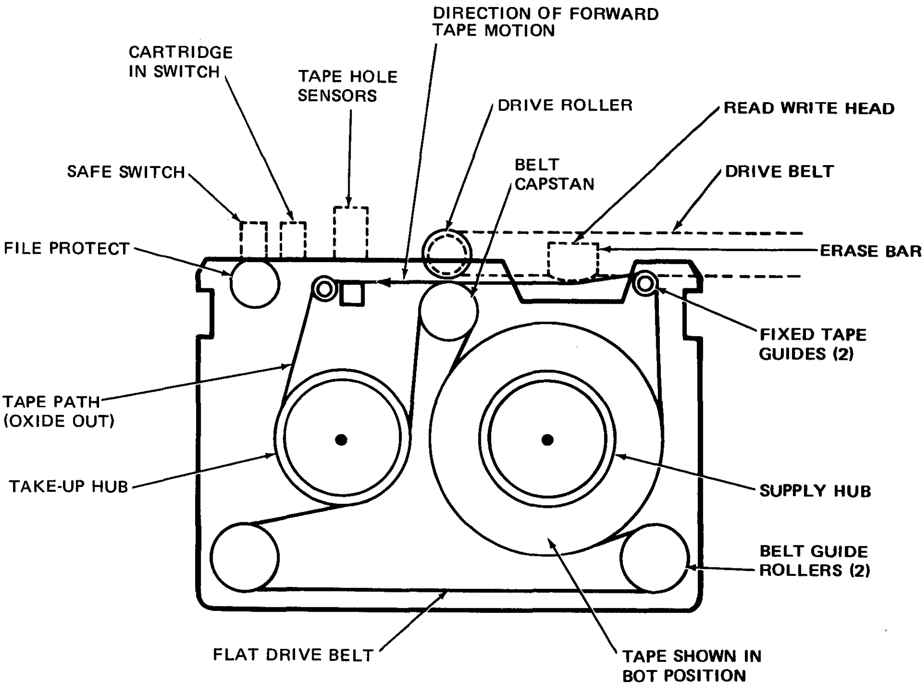


Figure 5-1. 1/4-inch Streaming Tape Cartridge

Tape Motion

The tape drive records data in a serpentine pattern (see Figure 5-2). The pattern is created by writing track 0 with the lower pair of heads enabled while moving from BOT to EOT. An erase bar which precedes these heads does a full tape width erase on the first pass. When the end of tape is reached, the lower pair of heads is disabled and the upper pair of heads is enabled. The capstan motor is reversed and as the tape moves from EOT to BOT, track 1 is written. When the beginning of tape is reached, tape motion is stopped, the head assembly is stepped to the next track pair location and the process is repeated until nine tracks are written.

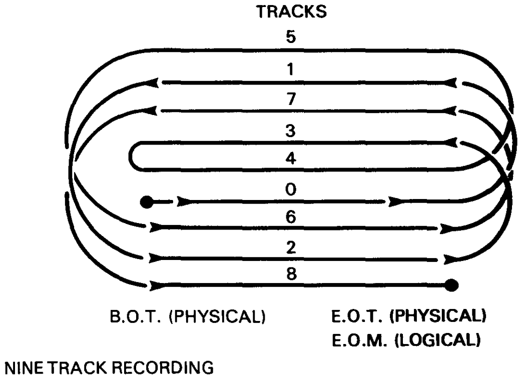


Figure 5-2. Serpentine Recording

Tape Commands

Drive and Track Selection

A **SELECT** command from the controller enables the tape drive to communicate with the controller. When enabled, the LED on the front panel of the tape drive is on during read, write, or position operations and when stopped between files. The front panel LED is off when a cartridge is not inserted in the drive.

An alternate method of selecting and enabling the drive is to use the **SELECT DRIVE, LOCK CARTRIDGE** command. When this command is used, **EXCEPTION** is asserted if the cartridge is removed while the front panel LED is on.

Motion Commands

There are three types of motion commands:

BOT command- rewinds the tape at high speed to BOT (Beginning of Tape).

ERASE command- the tape is rewound to BOT. With the erase bar on, the tape is moved forward to EOT (End of Tape). The entire tape is erased and then rewound to EOT.

RETENSION command- rewinds the tape to BOT at high speed, moves it forward to EOT, and then rewinds it back to BOT.

Write Command

When a **WRITE** command is received from the controller, the tape drive ensures that the tape is up to speed before writing to the tape. Data is written immediately after the load point or after a file mark if a file has already been written.

Each formatted block of data (512 bytes) is written immediately after the preceding block. A 12 to 3Ø byte preamble is written at the beginning of each formatted block and a Ø.5 to 2´byte postamble is written at the end of each formatted block. See **Data Format.**

Write File Mark Command

A file mark may be used to identify the end of recorded data or a division between groups of data. The command may be given to conclude writing or to create a division in the data being written without stopping the tape.

Read Command

The **READ** command instructs the drive to read data. Data is transferred from the drive to the host with asynchronous handshakes.

Read File Mark

The **READ FILE MARK** command allows the user to seek to the end of a file. The controller reads the tape searching for the file mark. Data is not transferred to the host. When a file mark is found, tape motion is stopped, and the host is informed that a file mark was found.

Read Status Command

The **READ STATUS** command is used to transfer status of the drive to the host.

Status Bytes

Six bytes of status information are maintained by the tape controller and can be requested by a **READ STATUS** command. The host must perform a read status operation when there is an exception (error) condition. However, the host may request status at any time.

Table 5-3. Status Bytes

Bit	Byte Ø
Ø	File mark detected
1	Bad block not located
2	Unrecoverable data error
3	End of media
4	Write protected cartridge
5	Unselected drive
6	Cartridge not in place
7	Status byte Ø bits (active)

Bit	Byte 1
Ø	Power-on/reset occurred
1	End of recorded media
2	Bus parity error
3	Beginning of media
4	Marginal block detected (Eight or more read retries for one block)
5	No data detected
6	Illegal command
7	Status byte 1 bits (active)

Bytes 2 (MSB) and 3 (LSB) contain a binary count of the number of recoverable errors that occurred during the last read or write operation.

Bytes 4 (MSB) and 5 (LSB) contain a binary count of the number of underruns that occurred during the last read or write operation.

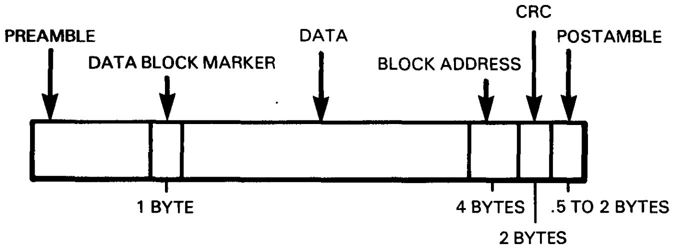
Underruns

To write readable data on the tape, the tape must be in constant motion. Therefore, the flow of data from the host must be sufficient to keep the tape controller's buffers full. If data flow from the host is interrupted, an underrun will occur. The tape drive will not stop, but may, at intervals, write a duplicate of the preceding block.

If data transfer is seriously interrupted, the drive may write a duplicate block and then reverse tape motion until data transfer resumes. At that point, the drive searches for the end of the last block and begins writing.

Data Format

The tape controller on the 586T/986T is configured to write and read a QIC-24 recording pattern, which is an industry standard for 1/4-inch magnetic tape. Each formatted block contains 531.5 to 551 bytes of bit serial data (see Figure 5-3).



12 TO 30	BYTES	PREAMBLE	4	BYTE	BLOCK ADDRESS
1	BYTE	DATA BLOCK MARKER	2	BYTES	CRC
512	BYTES	DATA	0.5 TO 2	BYTES	POSTAMBLE

Figure 5-3. QIC-24 1/4-inch Streaming Tape Format

Hard Disk

Description

The Hard Disk Controller (WD2010) interfaces to ST506 type drives. The WD2010's internal circuitry is capable of handling up to 8 heads per drive, but an external SDH register has been added to allow 16 heads to be accommodated. The SDH register also contains two drive select lines which are decoded to generate drive select signals for three drives.

The Controller can control up to three drives; one internal, and two external. The drives can be of different types so long as the seek rates are within the parameters of the WD2010 chip.

NOTE

Using or attaching a disk drive not installed by Altos may void the system warranty.

Currently, the drive capacities used are 40 and 80 Mbytes. Each disk drive's track 0 contains its specific configuration information (number of heads, cylinders, etc.) which is written on the drive at the factory when the drive is installed. This information is used by the operating system to program the Hard Disk Controller and stays with the drive as long as it is not destroyed. The Format program for normal customer use does not write on track 0.

WARNING: Formatting destroys data on all other tracks!

Performance Summary

The following table summarizes the drive specification limits. These vary, of course, depending on the model of drive(s) installed.

Table 5-4. Performance Specifications (Hard Disk)

No. of Heads	1 to 16 (drive dependent)
No. of Cylinders	up to 2048 (drive dependent)
Sectors per track	16 (programmed by format utility)
Bytes per Sector	512 (programmed by format utility)
Unformatted Capacity	Up to 160 MBytes
Formatted Capacity	Determined by formula: = No. of heads x no. of cylinders x 512 bytes/sector x 16 (no. of sectors/track)
Data Transfer Rate	5 MBits/second

Operational Characteristics

Operational characteristics are typical for Winchester drives.

Sector Format

The following figure shows the format that is written on the disk.

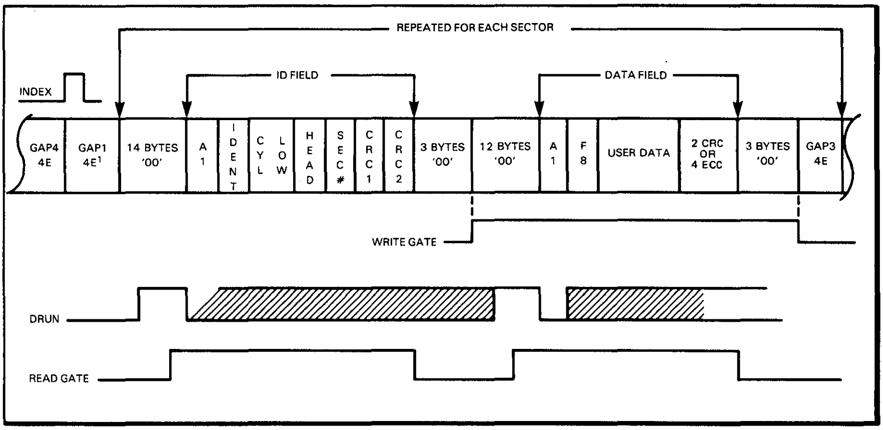


Figure 5-4. Hard Disk Sector Format

Floppy Disk Drive

Description

The Floppy Disk Drive is a half-height, 5 1/4-inch, double density, double-sided Mitsubishi 4853 or equivalent. It has a capacity of 1 Mbyte unformatted, or 720 Kbytes formatted.

Performance Summary

The following table summarizes the specifications of the Mitsubishi disk drive.

Table 5-5. Performance Specifications (Floppy Drive)

No. of heads	2
No. of tracks (each side)	80
Sectors per track	Programmed by operating system
Bytes per sector	Programmed by operating system
Data transfer rate	250 KB/sec.
* Seek settle time	18 msec max.
* Head switching time	100 usec min.
* Write gate delay	see note below

* See notes below

Operational Characteristics

* The firmware driver for the Floppy Disk controller meets the requirements for seek settle time, head switching time, and write gate delay.

Seek Settle Time- defined as the delay time required between the completion of a **SEEK** operation and the start of a write operation (**WRITE DATA or FORMAT**) to allow the heads enough time to settle on the track.

Head Switching Time. When floppy sides are switched, a delay time is needed to allow electrical transients at the heads to settle.

Write Gate Delay. On the Mitsubishi drive, a tunnel erase head trims the recorded track during a write. The erase head trails the write head by 1.Ø msec., so it is necessary to wait this amount of time at the completion of a **WRITE DATA** or **FORMAT** operation before starting any other operation on the drive.

The controller's **MOTORON** line (Floppy Port, Chapter 4) permits direct software control of the motor. From a stopped condition, the motor may need up to 25Ø msec. to get up to speed, and up to 5ØØ msec. to go **READY.** The firmware turns the motor off if the drive is not accessed for 1 to 2 seconds.

Normally, a **RECALIBRATE** command can be used to retract a read/write head to the track Ø position. However, if a command is issued to a drive when it is not ready, the command is aborted. The drive will not go ready without a diskette in place and the motor turned on. To allow a drive to be recalibrated without a diskette in place, a software ready, (**SOFTREADY**), can be set which makes the controller think the drive is ready. The heads may then be moved.

Sector Format

The format written by the controller's **FORMAT** command conforms to the IBM System 34 Double Density format (MFM). The variables in the format are number of bytes/sector, number of sectors/track, gap length, and data pattern, which are supplied by the host from the operating system program.

Chapter 6
System Firmware

586T/986T Monitor Program

Overview

The system, or resident, monitor is a collection of programs, that remain permanently in memory (on board EPROM on the CPU board) and provide initial system validation, as well as overall coordination and control of the operating system.

The functions of the Monitor are to provide download capability and a disk boot facility. This permits the user to load operating systems and programs into the memory. The Monitor does the following:

First, it performs an initial system validation (Power-up test) that exercises all of the major components in the system. This test establishes a hardware base from which the programmer can work.

Second, the monitor drivers perform the tasks of checking port status, inputting data and handling basic error recovery and reporting. The serial I/O service routine preprocesses data so that all users' programs are handled efficiently.

Physical Location

The monitor is fixed at locations FCØØØH to FFFFØH and 4ØØH to FFFH (Intel interrupt vectors are located at ØØØH to 3FFH). The object programs reside in the upper portion of the monitor, starting with the highest address, and working down (see Figure 6-1). These programs are written in EPROM and cannot be relocated, overlayed, or written over. The hardware reserved locations are allocated for specific hardware routines.

```
FFFFF(H) ---------------------------------------------- 1024K
                   Hardware Reserved Locations
FFFF0(H) ----------------------------------------------
                     Monitor Programs-EPROM
FC000(H) ---------------------------------------------- 1008K


                           User Area


1000(H)  ---------------------------------------------- 4.0K
                     Altos Monitor Work Area
0400(H)  ---------------------------------------------- 1.0K
               3FE(H)       CS
                       -----------         255
                            IP          Interrupt
                       -----------         Type
                       ~         ~       Numbers
                       ~         ~
                       -----------
                            CS               5
                       -----------
                            IP
               014(H)  -----------
                            CS               4
                       -----------
                            IP
               010(H)  -----------
                            CS               3
                       -----------
                            IP
               00C(H)  -----------
                            CS               2
                       -----------
                            IP
               008(H)  -----------
                            CS               1
                       -----------
                            IP
               004(H)  -----------
                            CS
                       -----------           0
                            IP
             Intel/586T/986T Interrupt Vector Table
00000(H) ---------------------------------------------- 0K
```

Figure 6-1. Memory Map Altos 586/986T

Monitor Start Up

The Monitor is executed whenever the system is powered up or reset. The power-up sequence starts with a series of tests that validate the system. These tests are as follows:

1. Firmware EPROM Checksum- A word is assigned as a checksum value. This test does an add on all words, including the checksum, and verifies that the checksum is zero. The purpose of the test is to assure, with a high probability, that the information inside the EPROM is correct.

2. Map RAM Data Bus Ripple- This test ripples a "one" bit across the data bus at location Ø of the map RAM. This helps to verify the integrity of the data bus.

3. Map RAM Address Bus Ripple- this test writes a background pattern (ØØØØH), starting at location Ø, while rippling a one across the address bus. The test, for all locations:

   a. reads and verifies all are Ø's

   b. writes FFFFH into the same location

   c. reads and verifies all are 1's

4. Map RAM Content March- writes a background pattern (ØØØØH) into all locations. Starting at location Ø, in ascending order and for each location, the test:

   a. reads and verifies all are Ø's

   b. writes FFFFH into same location

   c. reads and verifies all are 1's (ie. FFFFH)

   This test checks the integrity of the Map Ram address bus.

5. Main RAM Data Bus Ripple- algorithm same as #2 above. All bits in main RAM are rippled.

6. Main RAM Address Bus Ripple- the main RAM is divided into 4 banks of 64K words. Each bank is tested in exactly the same way. Starting at location Ø, it writes a background pattern (ØØØØH) while rippling a one across the address bus to all locations. The test:

   a. reads and verifies that the pattern is ØØØØH

   b. writes the complement (FFFFH)

   c. reads and verifies the complement

7. Main RAM Content March. Similar to test 4.

8. 8254 Timer- The test initializes all three counters to zero and starts the counters. It then stops the counters and verifies that all counters contain values other than zero.

9. 8259 Interrupt Controller- The timer is set up to issue an interupt. This test makes sure that an interrupt occurs.

A. Checks for a parity error in main RAM.

B. Checks for an error from the Controller board. A failure of any test in the Controller board diagnostic returns the message **FAILED POWER-UP TEST B.**

A system reset simply reinitializes the I/O ports and resets the memory map leaving the user's RAM intact.

Monitor Sign-on

Once these functions have been performed, the screen scrolls and the following message appears in approximately 4 seconds:

```
PASSED POWER-UP TEST
ALTOS COMPUTER SYSTEMS-586/986
Monitor Version X.X
Press any key to interrupt boot
```

If no key is typed within 4 seconds, a default hard disk boot is performed. If any key is typed, the following message appears on the screen:

```
Enter [1] to boot from Hard Disk
Enter [2] to boot from Floppy Disk
Enter [3] to enter Monitor

Enter option:
```

Entering a 3 gets you into the Monitor commands and the Monitor prompt appears on the screen:

```
<A,B,D,G,I,K,L,M,O,R,S,X>
```

The Monitor prompt displays all of the commands that are now available. The following sections describe each of the commands in detail.

Monitor Commands

The Monitor Commands are one character command names followed by option dependent operands. The operands are address or word values. The word values are limited to four characters. Address values are usually two-word values. Any byte operand values are also limited to a maximum of two characters. All operand numerical inputs must be hexidecimal. All letter inputs must be UPPER-CASE. Any inputs other than hexidecimal causes the system to error, sound the console bell and place an asterisk on the screen. The same error occurs if there is a checksum failure during the processing of a command (see Read Intel Format Hex Data). The prompt then returns and requests further inputs from the user.

The Monitor Commands are as follows:

A	ALTER MEMORY
B	BREAKPOINT

D	DISPLAY MEMORY
G	GO TO
I	PORT INPUT
K	DISK I/O
L	LOAD BOOT
M	MOVE MEMORY
O	PORT OUTPUT
R	REGISTER
S	SINGLE STEP INSTRUCTION
X	READ HEX (DOWN LOAD)

Address Values

An address value is a pair of word values that are used to load the segment register and base registers. The word values must, in all cases, be seperated by a colon. If the segment value is not specified in any command, it defaults to zero.

```
<address>=(<segment address><:>)<displacement address>
```

i.e.:

```
G FF11:ØØAC
```

sets the CS register to FF11, the IP register to ØØAC.

```
G F114
```

sets the CS register to ØØØØ, the IP register to F114.

The 586T and 986T implement straight-forward direct memory addressing by adding a 16-bit displacement (two object code bytes) to the Data Segment register. This 16-bit address displacement, when stored in program memory, has the low-order byte preceding the high-order byte. The base segment address must be supplied by the Data Segment register when addressing data memory.

Word Values

Word values are only four hexidecimal digits. (The result is an arithmetic expression consisting of + and - and hexidecimal numbers). Leading zeros are not required, even if the first valid character is an alphabetic character between "A" and "F".

i.e.:

```
D FF11:ØØAC+ØØFB,25-1A
```

displays 11 decimal bytes of memory starting at FF2B7.

Byte Values

Byte values are any two valid hexidecimal digits. If less than two digits are entered, the higher order nibble defaults to zero. Leading zeros are not required for byte values.

i.e.:

```
D FFFØ,4
```

displays four bytes of memory at location ØFFFØ.

Program Down Load

The Program Down Load Command is used to transfer files from a remote computer system to the host system. The file to be moved must have been changed to Intel Hex Format by a suitable conversion program before the transfer begins.

Read Intel Format Hex Data

The remote computer is connected to one of the serial ports on the rear connector panel. Note that Channel Ø (labelled **PORT 1**) is the usual input for the console keyboard and would not normally be used for this purpose. When the Down Load command (X) is given, the Monitor responds with:

```
Channel no. (Ø-4):
```

The user must enter the channel number desired (Ø-4). When the number is entered, the Monitor responds with:

```
Ready -
```

The channel selected is read continuously for Intel format hexidecimal data. The data is then placed in the memory stream based on the control information within the data. If a checksum failure is detected during an input operation, the console bell sounds, and an error indicator (asterisk) is displayed. If no errors are detected, the data stream is read until the ending record is read from the port. This, then, reverts control to the user console. If the data stream contained a start address record, the CS and IP register save areas are updated with that address. This causes control to be passed to the loaded program from any user specified, subsequent Go command.

Load Bootstrap Command

```
L<space>(<drive>)<CR>
```

NOTE

The space in the command is given by the Monitor.

The first sector, of the user specified disk, is read into memory. The header record, in the first sector, contains the base paragraph address for the command to be loaded. The base address is then extracted from the header record. The bootstrap program is then read into that location. Stream control, of the Load Bootstrap Command, is then passed to the first byte of the loaded program. If the drive is not ready, the Monitor is reentered and an error is displayed.

```
Drive = 0-3, Floppy Disks 0-3
        4-5, Hard Disks 0,1
```

Boot Format

The bootstrap format, for each system offered, is as follows:

Floppy Disk

CP/M-86, MP/M-86

```
512 bytes/sector and 9 sectors/track

4th byte, 1st sector  = low byte of Load Seg. Add.
5th byte, 1st sector  = high byte of Load Seg. Add.
10th byte, 1st sector  =0

129th byte, 1st sector to end of 2nd track = data

1st track is cyclinder 0, head 0
2nd track is cyclinder 0, head 1
```

OASIS

```
256 bytes/sector and 16 sectors/track

4th byte, 1st sector  =  low byte of Load Seg. Add.
5th byte, 1st sector  =  high byte of Load Seg. Add.
10th byte, 1st sector = 1
11th byte, 1st sector, to end of track = data
```

UNIX

```
512 bytes/sector and 9 sectors/track

4th byte, 1st sector =  low byte of Load Seg. Add.
5th byte, 1st sector =  high byte of Load Seg. Add.
10th byte, 1st sector = 2

1st byte, 1st sector to end of 3rd sector = data
```

Hard Disk

512 bytes/sector and 16 sectors/track

CP/M-86, MP/M-86

4th byte, 1st sector = low byte of Load Seg. Add.
5th byte, 1st sector = high byte of Load Seg. Add.
1Øth byte, 1st sector = Ø

129th byte, 1st sector to end of 1st track = data

OASIS

4th byte, 1st sector = low byte of Load Seg. Add.
5th byte, 1st sector = high byte of Load Seg. Add.
1Øth byte, 1st sector = 1

11th byte, 1st sector to end of 1st track = data

UNIX

4th byte, 1st sector = low byte of Load Seg. Add.
5th byte, 1st sector = high byte of Load Seg. Add.
1Øth byte, 1st sector = 2

1st byte, 1st sector to end of 3rd sector = data

NOTE

The Monitor starts execution at CS=Load Segment, IP=Ø immediately after system boot.

System Commands

Perform Disk I/O

```
K<space><IOPB address><CR>
```

The I/O operation, described in the IOPB (input/ output parameter block) at its user specified address, is performed and updated with the status of the operation. No user notification is returned. Success or failure must be determined by inspecting the IOPB manually through the use of the Display Memory Command. This operation allows the user to see if the correct parameters were placed into the IOPB. (see Chapter 4 for format of the IOPB).

Byte or Word I/O Command

```
I<space><B>or<W><space><portno><space><or><CR>
```

This command reads the word or byte of data from the specified port and displays the result to the user.

```
O<space><B>or<W><space><portno><space><data><CR>
```

This command writes the word or byte of data, specified by the user, to the selected I/O port.

Go To Address Command

```
G<space>(<starting address>)<CR>
```

The command transfers program control by setting the CS and IP registers to a newly specified value. If no starting address is supplied, control is transfered to the current settings of the CS and IP registers as defined in the register save area. (see R command).

Ram Based Debugger

A debugger is a development tool for editing object programs. It allows the user to single step a code segment and to control execution by means of a breakpoint.

Single-stepping allows the user to examine register or memory locations during execution.

A breakpoint allows the user to control execution by placing a software interrupt in the object code at locations specified by the user. The breakpoint transfers control to the debugger and allows the user to replace the original object code at any location and to view the current status. It is possible to define locations to jump to and track a system reset or monitor call outside of the PROM based monitor.

Single-stepping an Instruction

```
S<space>(<starting address>)<CR>
```

The Single-step command transfers control to the target address, if any, and executes a single instruction at that location. If no address is supplied, the system executes the next instruction pointed to by the CS:IP pair. The Monitor receives control following execution of the instruction, and displays the CS:IP register pair (see R command).

Set Breakpoint Command

```
B<space><CR>
          or
B<space>(<address>)<CR>
          or
B<space><-breakpoint number><CR>
```

There are three options with this command:

1. If no operands are specified, the contents of the Breakpoint Table are displayed on the terminal. The Breakpoint table consists of a breakpoint number and the address of the location to be breakpointed during the execution of a Go command.

2. If an address is specified, it is set into the Breakpoint Table in the first available slot (assuming there is space available in the table). (The Breakpoint Table has a maximum of eight slots or breakpoints pending at any one time.) The targeted memory locations are not altered until the user specifies a Go instruction. The Breakpoint instruction (INT 3)

is then inserted into the program. This states that any subsequent addressing of that location will cause a program interrupt and reentry to the Monitor.

3. The third option available is to remove a breakpoint instruction from the table. This may be done if the particular breakpoint has no further applicability or to free up a slot in the breakpoint table. Enter a minus sign followed immediately by the breakpoint location to be removed from the table. The number may be obtained from the display (see option 1 of this command).

NOTE

Setting a TRAP flag in the register save area will also cause Breakpoints to occur. In this case, a Breakpoint will occur following the execution of each instruction.

Display and Alter Register Contents

```
R<space><CR>
            or
R<space>(<register name>)
```

If no operands are specified, the monitor displays the contents of all of the registers, including the flag bytes. This display represents: (1) the values of the registers at the last breakpoint instruction and; (2) the values that will be reloaded into the registers prior to control being passed to the Go or Step Target instruction areas.

If an operand is specified, it must be the name of one of the system registers (i.e. AX for AX register, FL for Flag, etc.). The Monitor displays the current value of the register and prompts for a new value to be supplied. If only a carriage return is entered in response to this request, the register will not be changed.

Alter Memory Command

```
A<space><starting address><CR>
                and
A<space>or<,>
```

The Monitor displays the address and the current contents of the address, and prompts for new data. If the content of the address is satisfactory and no new data is to be entered, press A<space> or <CR>. New data entries must be entered in hexidecimal followed by A<space> or <CR>. Entering new data into the address simply slides the old data out. The Monitor continues to display characters until terminated by a non-hex character. A data byte comparison is done, following the data substitute operation, ensuring that the substitute operation was successful. If the substitute operation was not successful, an error message is displayed on the screen.

Display Memory Command

```
D<space><starting address><space><number of bytes><CR>
```

Starting address = address of first byte to be displayed

Number of bytes = hex number of bytes to be displayed

The Monitor automatically formats the data into groups of sixteen bytes. Each line displays the address of the first byte displayed on the line. Each line is broken into groups of four bytes with the ASCII data displayed to the right of each group. If you wish to interrupt a long display operation, simply press any key and the display will terminate.

Move Memory-to-Memory Command

```
M<space><from address><space><to address><space><length><CR>
```

This command moves the amount of memory specified to the destination address specified. If the "from address" is omitted, a zero is assumed.

Monitor System Calls

The Monitor provides low level drivers to drive Altos 586T/986T hardware devices from external programs coded in both high level and low level languages. The drivers check port status and handle data input, basic error recovery and error notification.

The interface to each driver routine is an accepted protocol which is not changed. Using fixed protocols isolates user programs from unexpected changes in the low level hardware, allowing some degree of device independence.

The interface chosen for the Altos system provides the user with a known entry point to the Monitor service routines and then expects parameters to be passed in registers BX and CX.

The user should call the Monitor at location FE00:0000 with a Far call (consult the 8086 instruction list) in order to set the CS register correctly. Following the Monitor Processing of the request, a Far return is executed to return control to the caller. The DS and ES registers are saved over any call to the Monitor for service, however no other registers are saved. Any register data that is required over a Monitor call should be saved prior to the call instruction being executed.

Register BX on entry to the Monitor must contain one of the following codes. An invalid code causes the Monitor to re-display the prompt. Register CX generally contains a parameter, or a parameter address. Byte values are returned from the Monitor in register AL, word values in AX. Other registers used are DX and ES.

00 = Return control to Monitor
01 = Return status of I/O channel port selected by CX
02 = Return character from I/O channel selected by CX
03 = Write character in DL to I/O channel selected by CX
04 = Set channel attributes (in DX) for I/O channel selected by CX
05 = Return I/O channel attributes in AX for I/O channel selected by CX
06 = Write CRLF to I/O channel selected by CX
07 = Write string pointed to by ES:DX to I/O channel in CX
08 = Perform disk I/O from IOPB pointed to by ES:CX
09 = Reserved
10 = Return default ops console in AL and top of Memory pointer in ES:DX
11 = Return boot code in AL
12 = Cold boot from disk selected by CX
13 = Reserved
14 = Reserved
15 = Reinitialize Monitor

MONITOR CALL
(Monitor Code 00)

Returns control back to the Monitor and issues the prompt for Monitor commands discussed earlier in this document.

CONSTAT
(Monitor Code 01)

The Monitor selects the register CL indexed I/O port and tests to see if it has a character waiting to be read. If there is a read pending, 0FF(H) is returned in register AL; if no character read is pending, 000(H) is returned. No validity checks are performed on the console index, thus an invalid index causes unpredictable results.

CONIN (Monitor Code 02)

The Monitor selects the register CL indexed I/O channel port and reads a character from that location. If no character is available, the Monitor waits until a character is available before returning control to the caller.

CONOUT (Monitor Code 03)

The Monitor selects the register CL indexed I/O channel port. The character supplied in register DL is written to the console. If the channel is unavailable, the Monitor waits until it is able to output the character.

GETTATTRIB (Monitor Code 04)

Return I/O channel attributes in AX register for I/O channel index in CL register.

SETTATTRIB (Monitor Code 05)

Sets I/O channel attributes from DX register for I/O channel index in CL and reinitializes that channel.

NOTE

Channel Indexes vary from Ø to 5. Channel Ø is the operators console. Channels 1-5 are the serial ports of the corresponding numbers. Only channels Ø and 1 are initialized by the monitor. The system timer should be initialized before the Setattrib can be instituted to channel 5. Channels 2 through 5 should be initialized before issuing any system calls to those channels.

CRLF (Monitor Code 06)

The Monitor selects the register CL indexed I/O channel and writes a carriage return and line feed to that device. If the console is busy, the Monitor waits until it is available, writes the sequence and then returns to the caller.

WRITEBUF
(Monitor Code 07)

The Monitor selects the register CL indexed I/O channel and writes the string of characters, pointed to by ES:DX, to that console. The string of characters is delimited by a byte of ØØØH as a terminator. Control returns immediately after handing off the buffer to the intelligent serial channel.

DISKIO (Monitor Code 08)

The Monitor checks the IOPB address in register ES:CX to determine the disk drive address. If the address is less than 4 (base 1Ø), then the floppy disk I/O routine is called (see Figure 6-2 for IOPB Format) to process the request. If the drive number is equal to, or greater than 4 and less than 6 (base 1Ø) then the hard disk I/O routine is called to process the request.

The Monitor seeks to the selected track, selects the head and sector indicated in the IOPB, and issues the command from the IOPB to the device.

If the operation returns a non zero status, the Monitor retries the operation for the number of times specified in the IOPB. If the operation is not successful after the number of retries specified, control is returned to the caller with the status field containing the final error status.

Following a successful read or write of a sector the count field of the IOPB is reviewed to see if all sectors specified have been read and written. The count must not exceed the boundary of one track because no seek is performed after the first seek to the track and head indicated in the IOPB. A full track may be read, however, by seeking to the first sector on a track and then supplying a count large enough to include all other sectors on the track.

```
000(H)    --------------------------------------------
                    (For Monitor Use Only)
004(H)    --------------------------------------------
           * Command Opcode      I       Drive
006(H)    --------------------------------------------
                         Track Number
008(H)    --------------------------------------------
                 Head            I         Sector
00A(H)    --------------------------------------------
               Sector Count     I * Return Code Status
00C(H)    --------------------------------------------
                 Status Mask    I     Retries
00E(H)    --------------------------------------------
                     DMA Offset Address
010(H)    --------------------------------------------
                     DMA Segment Address
012(H)    --------------------------------------------
                         Sector Length
014(H)    --------------------------------------------
                    (For Monitor Use Only)
016(H)    --------------------------------------------
                    (For Monitor Use Only)
018(H)    --------------------------------------------
                    (For Monitor Use Only)
01A(H)    --------------------------------------------
```

Figure 6-2. IOPB Mapping for Altos 586T and 986T

CONDEF (Monitor Code 10)

Returns default operators console index being used by the Monitor in Al and top of memory in ES:DX.

BOOTCODE (Monitor Code 11)

Returns the Monitor boot code in AL (01H = Hard Disk, 02H = Floppy Disk).

DISKBOOT (Monitor Code 12)

Boots system from disk selected by CX.

SYSINIT
(Monitor Code 15)

Reinitializes the Monitor. This command is used to bring the Monitor functions back to life after the monitor RAMS have been written over. This is the same as a hardware reset.

Monitor Control Blocks

```
MONITOR RAM LOCATIONS           00400-00FFF

     ccb-Z80 Channel Control Block

00400     .blkb  8
          .blkb  8

     scb - Z80 System Configuration Block

00410     01          System Operating Command
          00          Reserved
          00          ccb address
          04
          00
          00


MONITOR PROM LOCATIONS          FE000-FFFFF

FFFF0          Power up and Reset Entry Address

FFEC8-FFFEF   Z80 Channel Control Program- controls
              hard disk, floppy disk, and streaming
              tape operations.
```

Controller Board Firmware

Overview

Controller Board firmware is contained in an PROM addressed by the on-board Z8Ø I/O Processor. Its functions are:

1. To verify the operation of the major components on the Controller Board at power-up.

2. To supply the basic drivers to the operating system and hardware diagnostic programs to boot from either the Floppy Disk or Hard Disk.

3. To provide the capability to download code from the system memory to the Controller Board local memory.

Power-Up Diagnostics

The Controller Board's firmware diagnostic program is executed at power-up. Any failure of a major component is reported to the on-board LEDs. The LED outputs are defined in the System Maintenance Manual.

The following tests are run:

1. PROM Checksum. Verifies the PROM by doing a checksum of the bytes in the PROM. The sum should be the same as the preset value.

2. Local RAM Data Bus Ripple Test. Checks the integrity of the local RAM data bus. A pattern of Øs is written to location 8ØØØH and then a '1' is rippled across the data bus to ensure adjacent bits are not stuck.

3. Local RAM Address Bus Ripple Test. Checks the integrity of the local RAM address bus. Starting with location 8ØØØ, a background pattern of Øs is written to each succeeding location while rippling a '1' bit across the address bus.

Returning to location zero, the test then verifies each location written and writes a complement pattern. The complement pattern is then verified.

4. RAM Content March. The memory is filled with a background pattern of 55H and each location is verified. The complement, AAH, is then written back to the each location and verified. This process continues for the entire 16K bytes.

5. Interrupt Controller. Tests the ability of the Interrupt Controller chip to generate and respond to interrupts.

6. DMA Controller. Channels Ø and 1 of the local DMA controller are each tested with a memory-to-memory block transfer. A pattern is written in the destination area and verified sometime later. The terminal count and interrupt signal are also verified.

7. Floppy Disk Controller. To verify the floppy disk interface, the firmware issues a Specify command to set the initial values of the three interval timers. A Sense Drive Status command is used to obtain the status of the drive. If status is correct, the interface is verified.

8. Timer Test. A pulse is generated to the Timer input on the Interrupt Controller to cause an interrupt 16.125 mS after time Ø. This test checks if the interrupt occurred at the specified time.

9. Hard Disk Controller. The firmware writes and reads to 5 out of 7 registers in the Hard Disk Controller. If no error is detected, the interface is verified.

A. Streaming Tape Controller. The interface is verified by issuing a software reset and checking that the reset/power-on bit, (bit Ø), is set in status byte 1 (see **Status Bytes** in Chapter 5).

If any of the Controller board tests fail, a message is displayed on the screen:

FAILED POWER-UP TEST B

A pair of LEDs on the Controller board (see schematics, sheet 1) indicate which circuit on the Controller board failed.

During normal operation (no failure), both LEDs are on and remain on in a steady state. If a fault occurs during the diagnostic test, LED #1, the "clock" indicator, starts to cycle on and off.

Cycles last for 16 "clocks" and then start over. The test continues until power-off or reset occurs.

LED #2, the "data" indicator, only turns on if a bit is set for that position. The code sequence is 1, 2, 4, 8, 10, 20 ... 8000 (see Table 6-1). For example, if the data indicator turns on during the fifth clock, the failure indicated is a DMA checksum (see Table 6-2).

Table 6-1. Controller Board LED Indicators

	LED 1 (Clock)	**LED 2 (Data)**	**Duration**
No error or Z8Z80A inoperative	ON	ON	Steady state
Start of error	OFF	OFF	1 to 2 seconds
Clock 1	ON	*ON/OFF	0.5 seconds
Clock 2	OFF	*ON/OFF	0.5 seconds
⋮	⋮	⋮	⋮
Clock 16**	OFF	*ON/OFF	0.5 seconds

* ON = 1, OFF = 0
** Return to start of error

Table 6-2. Diagnostic Test Error Equates

```
romcheckerr     =    0001h    ; ROM checksum failure
datcheckerr     =    0002h    ; data bus error
addcheckerr     =    0004h    ; address bus failure
ramcheckerr     =    0008h    ; RAM failure
dmacheckerr     =    0010h    ; DMA checksum failure
intcheckerr     =    0020h    ; interrupt chip failure
flopcheckerr    =    0040h    ; floppy chip failure
hardcheckerr    =    0080h    ; hard disk failure
tapecheckerr    =    0100h    ; tape controller failure
timercheckerr   =    0200h    ; timer-counter error
linecheckerr    =    0400h    ; line to interrupt chip fails
spare3          =    0800h    ; spare error code
spare4          =    1000h    ;
spare5          =    2000h    ;
spare6          =    4000h    ;
spare7          =    8000h    ;
```

Appendix A
586T/986T Serial Controller

INTRODUCTION

The Serial Controller on the 586T/986T CPU board is a Z80A processor together with its peripheral chips and six serial ports. The I/O processor on the Serial Expander board (986T) is a similar circuit that contains four serial ports and operates in an identical fashion.

Communication between the host processor and the Serial Controller is accomplished by means of registers. Initially, a single 24-bit register (called the "Initialization Register") exists at location 1FFFC (1000:FFFC). This register contains a pointer to a group of registers. The group of registers may exist anywhere in memory (assuming they are addressable by an external Multi-Bus host). When the Serial Controller is interrupted, the initialization register is read. From this point, initialization is accomplished according to the parameters in the register groups. The initialization register will be read only after the controller is interrupted, allowing the reserved locations to be reused by the host unless reinitialization is required.

Multibyte registers containing 16-bit counts or 24-bit addresses are always stored in reverse order, low byte first, high byte last.

REGISTER DEFINITION

Initialization Register

The Initialization Register is a 24-bit register residing at location 1FFFC (hex). It contains the address of the system, channel, and floppy disk register array.

System Registers

Firmware Version Register

The firmware version register is normally a read-only register containing the level of the firmware in the controller. There is only one case where the host will write into this location. Before system initialization, the host should store a zero into this location (and the System Command register). When initialization is complete, the actual firmware version number will be stored in this location. This location may be tested to identify an uninitialized controller (or initialization in progress). The version numbers will always be in the range of (1-63).(Ø-7) where the version number occupies the five most-significant bits. The sub-version number occupies the three least-significant bits.

```
|v|v|v|v|v| |s|s|s|
```

System Command Register

The system command register is an 8-bit register used to pass commands to all channels simultaneously. Handshaking is accomplished by setting bit seven of the command. The host should ensure that the last command has been executed (bit 7 = Ø) before issuing a new command. Commands are as follows:

Ø	disable controller
1	enable controller
2	disable interrupts
3	enable interrupts
4	reset interrupt
5-127	unused

System Status Register

The system status register is an 8-bit register used to pass status to the host concerning channel-independent information. The status bits are defined as follows:

bit Ø	Controller enabled
bit 1	Interrupts enabled
bit 2	Interrupt pending (not qualified by "Interrupts enabled")
bit 3	Bus error
bits 4-7	not used - always Ø

Interrupt Vector Register

The Interrupt Vector Register is a 16-bit register containing data to allow quick response to an interrupting condition. The register is logically divided into four fields.

bits Ø-2	interrupting modem channel number
bit 3	modem interrupt
bits 4-6	interrupting receiver channel number
bit 7	receive interrupt
bits 8-1Ø	interrupting transmitter channel number
bit 11	transmit interrupt
bits 12-14	other interrupting device number
bit 15	other device interrupt

Device types are as follows:

Ø	unassigned
1	network receive packet available
2	network transmitter ready
3	unassigned
4	floppy disk idle
5	unassigned

New Command Register

The new command register is an 8-bit register used to indicate the existence of a new command. This register should be incremented any time a command is written into the system command register, the floppy disk command register, the time-of-day register, or any of the six channel command registers.

Communication Channel Registers

Channel Parameter Register

The channel parameter register is a 16-bit register containing the following information:

Async mode

bit 0	parity enable		
bit 1	parity even		
bits 2-3	=0		
	=1	1	stop bit
	=2	1.5	stop bits
	=3	2	stop bits
bits 4-5	=0	5	bits/character
	=1	7	bits/character
	=2	6	bits/character
	=3	8	bits/character
bit 6	unused		
bit 7	ring buffer receiver enable/-TTY		
bits 8-11	=0	selectable rate	
	=1	75	bps
	=2	110	bps
	=3	134.5	bps
	=4	150	bps
	=5	300	bps
	=6	600	bps
	=7	1200	bps
	=8	1800	bps
	=9	2000	bps
	=10	2400	bps
	=11	3600	bps
	=12	4800	bps
	=13	7200	bps
	=14	9600	bps
	=15	19.2	kbps

bit	12	unused
bit	13	unused
bit	14	CTS control
bit	15	DSR control

Channel Status Register

The channel status register is a 16-bit register containing the following information:

bit	Ø	transmitter empty
bit	1	Data Terminal Ready
bit	2	break in progress
bit	3	Request To Send
bit	4	parity error
bit	5	receive data lost (SIO overrun)
bit	6	framing error
bit	7	parity OR overrun OR framing error
bit	8	receive character(s) ready
bit	9	unused - always Ø
bit	1Ø	unused - always Ø
bit	11	unused - always Ø
bit	12	transmitter ready/-active
bit	13	unused - always Ø
bit	14	unused - always Ø
bit	15	unused - always Ø

Channel Command Register

The channel command register is an 8-bit register used to pass commands to the controller. Bit seven is used to indicate the existence of a valid command. The host should test bit seven to ensure that the last command has been executed. The controller will clear bit seven after reading the command. At this time, the four least significant bits will also be cleared, while bits four through six will retain their values. Bits four through six contain the logic level corresponding to the interrupt enable status. The commands available are as follows:

Ø	no operation
1	initialize channel
2	start transmitter

3	acknowledge receiver
4	abort transmitter
5	reserved
6	"Break" control
7	network control
8	change parameters
9	reset error conditions
1Ø	reset modem interrupt request
11	execute from MultiBus

bit 7	command valid
bit 6	transmit interrupt enable
bit 5	receive interrupt enable
bit 4	modem interrupt enable

Transmit Data Buffer Address Register

The transmit data buffer address register is a 24-bit register containing the address of the data to be transmitted. After tansmit termination, this register will contain the address of the last character transmitted plus one.

Transmit Data Buffer Length Register

The transmit data buffer length register is a 16-bit register containing the number of bytes to be transmitted. After transmit termination, this register will contain the number of characters in the buffer NOT transmitted. (Note: this value will be Ø unless the transmission was aborted.)

Receive Data Buffer Address Register

The receive data buffer address register is a 24-bit register containing the address of the receive buffer when receiving in ring buffer receive mode. In Networking mode, this register points to the Network Control Block.

Receive Data Buffer Length Register

The receive data buffer length register is a 16-bit register containing the length of the receive data buffer. The receive buffer length may be in the range of 2-32768 bytes. In Networking mode, this register contains the number of receive buffers.

Receive Buffer Input Pointer Register

The receive buffer input pointer register is a 16-bit register containing a pointer (relative to the receive data buffer address) to the first empty location in the receive buffer. As characters are received, the controller will store the character at the specified location, and increment this pointer (modulo= buffer length). Bit 15 is the "counter invalid" bit. When this bit is set, the contents of the counter are in the process of being updated. This bit may be ignored if the buffer is =< 255 bytes in length. In Networking mode, this register contains the value of the first empty entry in the Network Control Block.

Receive Buffer Output Pointer Register

The receive buffer output pointer register is a 16-bit register containing a pointer (relative to the receive data buffer address) to the first full location in the receive buffer. As characters are removed from the buffer, the host will increment this pointer modulo buffer length. When the input pointer is equal to the output pointer, the buffer is empty. Bit 15 is the "counter invalid" bit. When this bit is set, the contents of the counter are in the process of being updated. This bit may be ignored if the buffer is =< 255 bytes in length. In Networking mode, this register contains the value of the first full Network Control Block.

TTY Receive Register

The TTY receive register is an 8-bit register used to receive data in applications not requiring multiple-byte ring-buffering. In TTY receive mode, the byte is simply stored in the TTY receive register.

Selectable Rate Register

The selectable rate register is a 16-bit register used to transfer a 16-bit value directly to the bit rate generator. When the rate is specified through this register, the SIO controller is programmed for clock/16 mode. The value to store in this location can be calculated by the formula:

(5-86) value = 312500/desired bit rate

In Networking mode, this register is used to store the network address.

Expansion Register

Because some computers/programming languages cannot (or cannot easily) generate/support multiple occurrence arrays containing an odd number of bytes, an 8-bit register is included at the end of each channel register array. This register completes the array of an even number of bytes which may or may not be located on a word boundary. The register is reserved for the purpose of future expansion.

Time of Day Registers

Control Register — 1 byte

value = 1 (81)	update time	(controller -> memory)
value = 2 (82)	set time	(memory -> controller)

Time Registers — 7 bytes

seconds	1 byte
minutes	1 byte
hours	1 byte
day	1 byte

```
     date                                  1 byte
     month                                 1 byte
     year                                  1 byte

insure command bit 7 = 0

if setting new time:

     write new time into time registers

issue command (81 or 82)

increment "NewCommand" register

if reading present time:

     wait for command bit 7 = 0

     read time from time registers
```

PROGRAMMING

Overview

Programming the Altos Intelligent Peripheral Controller can best be described in nine sections:

Controller Activation
Host/Controller Communication
Controller Initialization
Channel Initialization
Channel Transmit Operation
Channel TTY Receive Operation
Channel Ring Bufferred Receive Operation
Modem Control/Status Operation
Execute from Multi-Bus

Controller Activation

After a power-up or controller hardware reset, the controller first initializes all hardware devices, clears internal RAM, and enters a tight loop. To activate the controller, the host computer must first generate and initialize the system and channel register arrays. (The system register array followed by the eight channel register arrays form a single contiguous block that will hereafter be called the Controller Control Block or CCB.)

The next step is to store the address of the Controller Control Block at Multi-Bus location 1FFFC hex.

Values must exist at location 1FFFC in the following order:

```
     low  mid  high xxxx
```

The first two locations must contain the value ØØ hex.

The last step is to generate a "Channel Attention" signal to the controller. This operation will start the controller. When activation is complete, the controller will store the firmware version number (guaranteed not to be zero) in the first location of the Controller Control Block. At this point, the controller is ready to accept commands.

Host/Controller Communication

Commands to the controller are controlled by ten registers within the CCB. The process that reads the command registers is generally in a quiet (not asleep) state. During this time, it will read the "New Command Register." If it finds that this register has not changed, it will exit without checking the system or channel command registers.

When the "New Command Register" has changed, the process will individually test the command registers, and, if there is a command pending, activate the appropriate process to service the request.

This mechanism is very efficient in the general case (when no commands are pending), but does demand the proper sequencing of events. Any time the host issues a command, it must alter the value in the "New Command Register." (The recommended procedure is to increment the value after each command.)

The controller will acknowledge receipt of a command by clearing bit seven of the command byte. Although there is generally no reason to wait for the acknowledgment after issuing a command (see note below), it is essential that bit seven be tested before issuing a new command. The procedure then becomes the following:

```
Test if last command still pending
<exit or loop if true>
Set new parameters (if required for command)
Issue new command
Increment "New Command Register"
Exit
```

Note

> Channel status for a given channel is not valid while a command is pending.

Controller Initialization

After activation, the controller is in an idle state waiting for the system command to enable the controller. Issue the command. At this time, controller interrupts may be enabled.

Channel Initialization

After selecting the desired operating parameters (i.e., parity, word length, etc.), this information must be stored into the channel parameter register. The channel may now be initialized with the "Initialize Channel" command. The channel is now ready to transmit or receive. If any of the interrupt enable bits were set with the command, and their associate conditions are true, an interrupt would be generated at this point. At any time, a No-Op command may be issued that will only affect the interrupt enable bits.

Channel Transmit Operation

To transmit a block of data, the procedure is as follows:

```
Test if last command still pending
<exit or loop if true>
Test if transmitter ready
<exit or loop if false>
Set address and byte count
Issue transmit command
Increment "New Command Register"
Exit
```

Channel TTY Receive Operation

To receive a character in TTY mode, the procedure is as follows:

```
Test if last command still pending
<exit or loop if true>
Test if receive character available
<exit or loop if false>
Read status (test for errors)
If Error
Issue Reset Error command
Increment "New Command Register"
Exit
Else
Read character from TTY register
Issue Receiver Ack command
Increment "New Command Register"
Exit
```

Channel Ring Buffered Receive Operation

To receive a character in Ring Bufferred Receive mode, the procedure is as follows:

```
Test if last command still pending
<exit or loop if true>
Test if receive character available
<exit or loop if false>
Read status (test for errors)
If Error
Post status
```

```
Issue Reset Error command
Increment "New Command Register"
Exit
Else
Transfer Characters
Issue Receiver Ack command
Increment "New Command Register"
Exit
```

Transfer Characters:

```
While (Input Pointer) >< (Output Pointer) Do
{   TTYInputRoutine <- ((BufferAddress)+(OutputPointer));
(OutputPointer) <- (OutputPointer)+1;
If (OutputPointer) = (BufferLength) Then
(OutputPointer) <- Ø;
}
```

Modem Status Operation

```
Test if last command still pending
<exit or loop if true>
Test if modem status changed
<exit or loop if false>
Read status (test for errors)
Issue Reset Modem Interrupt command
Increment "New Command Register"
Exit
```

Execute from Multi-Bus

Appendix B
CPU Board Jumpers

The following pages detail jumper connections on the CPU board for various configurations.

586T/986T CPU Jumper Specifications

Jumper Label	Schematic Page No.	Function and Comments
E1	15	This is a "don't care" since the circuit is not used.
E2	15	This is a "don't care", since the circuit is not used.
E3	5	A jumper should always be in place at E3. This jumper has been provided for automatic testing only.
E4	3	This jumper should be left open.
E5	10	A jumper should normally be in place between pins 2 & 4. Jumpering between pins 3 & 4 allows the use of an 8K x 8 static RAM.
E6	10	A jumper should always be in place between pins 2 & 3.
E7		Not used.
E8	10	A jumper should always be in place at E8. This jumper has been provided for automatic testing only.
E9	1	A jumper should always be in place at E9. This jumper is provided for automatic testing only.
E10	9	Jumpers can be installed between pins 1 & 3 and 2 & 4. These jumpers are available for software configuration.
E11	7	Jumpered for 2764 EPROMs.
E12	14	This jumper block is used for software configuration only. Jumpers can be installed between pins 1 & 2, 3 & 4, 5 & 6 and 7 & 8.
E13	15	This is a "don't care", since the circuit is not used.

Jumper Label	Schematic Page No.	Function and Comments
E14	15	This is a "don't care", since the circuit is not used.
E15	15	This jumper is not used.
E16	15	This jumper is a don't care, since the circuit is not used.
E17	14	E17 provides jumpering options for the REQ TO SEND and CLEAR TO SEND signals on serial port 6. The normal configuration is to jumper pins 4 & 6 and 1 & 3. This permanently asserts the modem CLEAR TO SEND pin and connects the SIO RTS/ pin to the SIO CTS/ pin.
E18	14	E18 provides a modem option for serial port 5. The pinning for E18 is the same as E17.
E19	14	E19 provides the ability to connect serial ports to a synchronous modem. For asynchronous use jumpers should be in place between 3 & 5 and 4 & 6. For synchronous modems that provide clock signals jumpers should be placed between pins 1 & 3 and 2 & 4. Some modems may require a clock signal generated by the computer. In this case a jumper should be placed between pins 7 & 8.
E2Ø	12	This jumper selects modem options for serial port 4. Jumpering is the same as it is for E17.
E21	12	E21 provides modem jumpering for serial port 3 and jumpering for WORKNET. The normal pinning should have jumpers between pins 1 & 2 and 4 & 6. For WORKNET the jumpers should be between 1 & 3 and 2 & 4.

Jumper Label	Schematic Page No.	Function and Comments
E22	12	The jumpering should be between the following pins: 3 & 5, 7 & 8, 11 & 13, 15 & 16.
E23	12	E23 provides jumpering for either asynchronous operation or WORKNET operation. For asynchronous operation jumpers should be in place between pins 1 & 3 and 2 & 4. For WORKNET jumpers should be in place between pins 3 & 5 and 4 & 6.
E24	12	E24 provides a modem option for serial port 3 and jumpering for WORKNET. For normal asynchronous operation jumpers should be between 2 & 4 and 3 & 5. Alternately pins 1 & 3 and 4 & 6 could be jumped which connect the SIO RTSA/ to CLEAR TO SEND and REQ TO SEND to the SIO CTSA/. For WORKNET jumpers should be between 1 & 3 and 2 & 4.
E25	12	This jumper provides for asynchronous operation or WORKNET operation. For asynchronous operation a jumper should be between 1 & 2. For WORKNET operation the jumper should be between 2 & 4.
E26	13	E26 provides a modem option for serial port 2. The jumpering is the same as it is for E17.
E27	13	E27 provides the ability to connect serial port 1 to a synchronous modem. The jumpering is the same as it is for E19.
E28	13	E28 provides a modem option for serial port 1. The jumpering is identical to E17.

Jumper Label	Schematic Page No.	Function and Comments
E29	12	E29 was provided for development purposes only and should not be connected.
E3Ø	3	E3Ø provides line termination for the CCLK signal. This jumper should be installed only when expansion boards are installed but don't provide for CCLK termination.
E31	3	E31 provides line termination for the BCLK signal. This jumper should be installed only if other boards in the system do not provide BCLK termination.
E32	15	Not used.
E33	12	E33 is a jumper for WORKNET and should have a jumper between pins 2 and 4.
E34	12	E33 is a jumper option for WORKNET. For normal WORKNET operation jumpers should be installed between pins 1 & 2 and 3 & 4.

Appendix C
Controller Board Jumpers

The following pages detail jumper connections on the Controller Board for various configurations.

586T/986T Jumper Specifications

Jumper Label	Schematic Page no.	Function and Comments
E1	8	Normally open; used to terminate the BCLK* signal which is normally not required.
E2	3	Always in place; for auto testing.
E3	4	Normally open; used to test streaming tape.
E4	6	No jumper; this is a test point.
E5	6	No jumper; this is a test point.
E6	6	Normally open; used in testing the streaming tape controller.
E7	2	Normally in place between 3 and 4. Positions 1 and 2 disable write pre-compensation.
E8	2	Always in place; for auto testing.
E9	2	Normally in place between 3 and 4 connecting J3-2 to HEADSEL3*. A jumper is placed between 1 and 2 for older drives requiring LOW CURRENT.
E1Ø	2	Always in place; for auto testing.
E11 or E	6	No jumper; a test point for the tape controller.
E12	11	Normally jumpered from 1 to 2 for correct system operation. A jumper is placed from 3 to 4 for diagnostic testing of the disk buffer.

Index

586T/986T System Reference Manual

READER COMMENT FORM

Altos Computer Systems
2641 Orchard Parkway
San Jose, CA 95134

This document has been prepared for use with your Altos Computer System. Should you find any errors or problems in the manual, or have any suggestions for improvement, please return this form to the ALTOS PUBLICATIONS DEPARTMENT. Do include page numbers or section numbers, where applicable.

____

____

____

____

____

____

____

____

____

____

____

____

System Model Number____________________

Serial Number________________

Document Title____

Revision Number________________________Date__________________

Name____

Company Name____

Address____

____

____

ALTOS

Printed in U.S.A.
P/N 690-15813-002

2641 Orchard Park Way, San Jose, California 95134
(408) 946-6700., Telex 470642 ALTO UI

April 1985