iSBC 215™
WINCHESTER
DISK CONTROLLER
HARDWARE REFERENCE MANUAL

Order Number: 121593-002

REV.	REVISION HISTORY	PRINT DATE
-001	Original Issue	1/81
-002	Manual updated for minor corrections.	9/81

Additional copies of this manual or other Intel literature may be obtained from:

Literature Department
Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

The information in this document is subject to change without notice.

Intel Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel Corporation assumes no responsibility for any errors that may appear in this document. Intel Corporation makes no commitment to update nor to keep current the information contained in this document.

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's software license, or as defined in ASPR 7-104.9(a)(9).

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Intel Corporation.

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

BXP	Intelevision	Micromap
CREDIT	Intellec	Multibus
i	iRMX	Multimodule
ICE	iSBC	Plug-A-Bubble
iCS	iSBX	PROMPT
im	Library Manager	Promware
INSITE	MCS	RMX/80
Intel	Megachassis	System 2000
Intel	Micromainframe	UPI
		μScope

and the combination of ICE, iCS, iRMX, iSBC, iSBX, MCS, or RMX and a numerical suffix.

PREFACE

This manual provides information regarding the installation, programming, operation, and servicing of the iSBC 215™ Winchester Disk Controller.

Related documents include:

- *The 8086 Family User's Manual,* Order No. 9800722
- *Intel MULTIBUS™ Specifications,* Order No. 9800683
- *Intel 8080/8085 Assembly Language Reference Manual,* Order No. 9800301
- *MCS-86™ MACRO Assembly Language Reference Manual,* Order No. 900640
- *MCS-86/85™ Family User's Manual,* Order No. 121506
- *8089 Assembler User's Guide,* Order No. 9800938
- *iSBX™ Bus Specification,* Order No. 142686
- *iSBX 218™ Flexible Disk Controller Hardware Reference Manual,* Order No. 121583

CONTENTS

CONTENTS (Continued)

TABLES

ILLUSTRATIONS

CHAPTER 1
GENERAL INFORMATION

1-1. INTRODUCTION

The Intel iSBC 215™ Winchester Disk Controller allows up to four Winchester technology disk drives (see Table 1-2 for disk specifications) to be interfaced with any Intel Multibus™ interface compatible computer system. It supports drives that use either open loop head positioning (Shugart SA600, SA1000 and SA4000, Quantum Q2000 or Fujitsu 2300, RMS 500, CDC Finch or Memorex 101) or closed loop head positioning (Pertec D8000 or Priam 3350 and 3450). It's design is based on the Intel 8089 I/O Processor, which allows Direct Memory Access (DMA) transfers, error detection and correction, and data management. The controller can operate in a multiprocessor environment and is fully compatible with all Intel 8-bit and 16-bit computers. The number of tracks per surface, sectors per track, bytes per sector and alternate tracks per surface are software selectable for each drive unit. (In addition, the Memorex, 14″ Shugart and Priam drives require that the sector size be set internally as shown in Chapter 2.) The single board assembly also features automatic error recovery and retry, transparent data error correction and multiple sector transfers. Seek operations on multiple drives can be overlapped with a read/write operation on another drive. The iSBC 215 controller is fully compatible with Intel 8086 CPU 20-bit addressing.

A typical multiple drive system using four Winchester disk drives and the iSBC 215 controller is shown in Figure 1-1. The controller also provides two Intel iSBX™ Bus connectors, J3 and J4, which allow other storage devices such as floppy disk drives or magnetic tape cartridge drives to be interfaced with Multibus interface compatible systems. The Intel iSBX 218™ Flexible Disk Controller, for example, attaches to one iSBX™ Connector, J4, allowing the controller to be interfaced with up to four double-density floppy disk drives. Figure 1-2 shows a typical multiple drive system using four 5¼″ or 8″ floppy disk drives, the iSBC

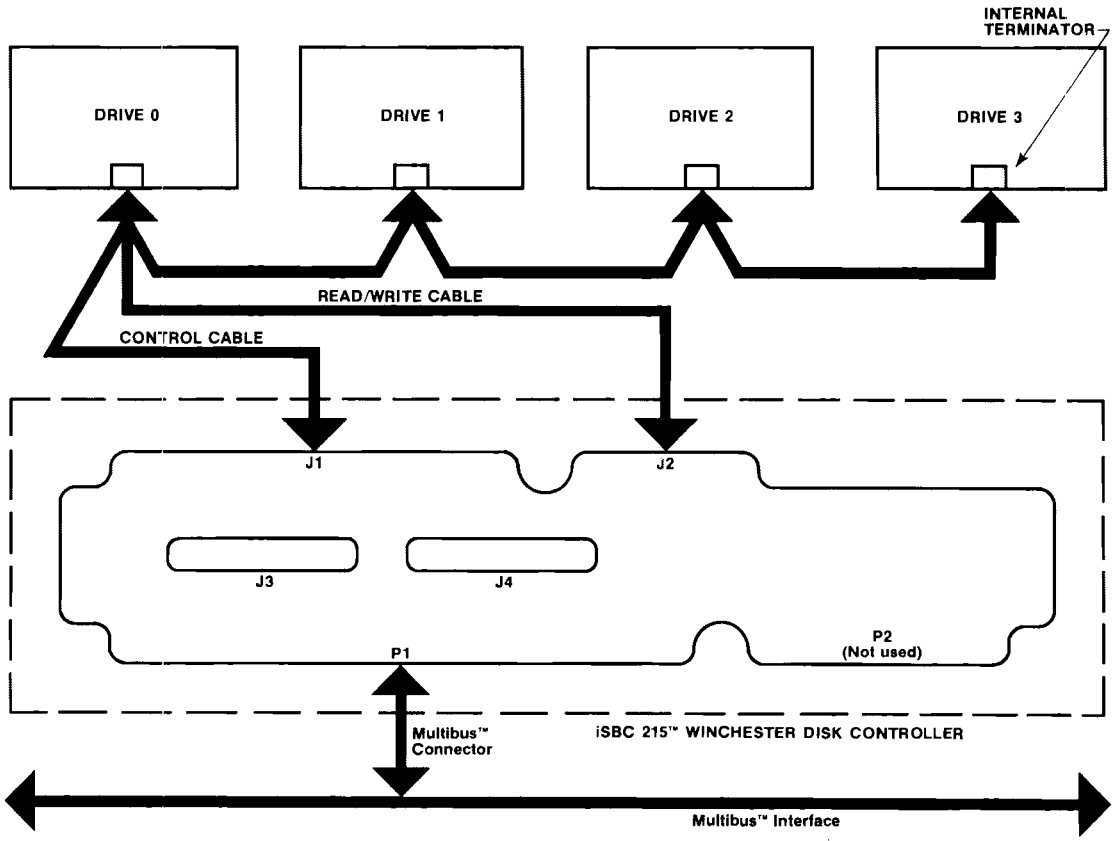


Figure 1-1. Typical Multiple Drive System Using Winchester Disk Drives

215 controller and the iSBX 218 Flexible Disk Controller. It should be noted that the controller can interface concurrently with Winchester disk drives through connectors J1 and J2, and with other storage devices through the iSBX™ Connectors, J3 and J4.

1-2. DESCRIPTION

The iSBC 215 Winchester Disk Controller is a single board assembly. It may reside in any Intel backplane or in a custom-designed configuration that is physically and electrically compatible with the Intel Multibus interface.

The host Central Processing Unit (CPU) communicates with the Disk controller via four blocks of information in host memory. Once the controller is initialized, a CPU I/O write to the controller Wake-Up Address initiates disk activities. The controller accesses the four blocks in the host memory to determine the specific operation to be performed, fetches the required parameters and completes the specified operation without further CPU intervention.

The controller board generates all drive, control and data signals and receives the drive status and data signals required to perform the entire disk drive interfacing task. During a disk read operation, the controller accepts serial data from the disk, interprets synchronizing bit patterns, verifies validity of the data, performs a serial-to-parallel data conversion, and passes parallel data or error condition indications to host memory. During a disk write operation, the controller performs parallel-to-serial data conversion and transmits serial write data and the write clock to the drive. As part of the disk format and write function, the controller appends an Error Checking Code (ECC) at the end of each ID and data field. Using this ECC, the controller hardware can detect errors of up to 32 bits in length; controller firmware can correct errors of up to 11 bits in length (see Figure 1-3).

The Intel 8089 I/O Processor provides optimum performance with minimum CPU overhead. An Intel 8288 Bus Controller and 8289 Bus Arbiter control access to the Multibus interface. Intel 2732 EPROMs provide on-board storage of the controller I/O control program and a resident diagnostic exerciser, and 2114 Static RAMs provide local memory for data buffering and for temporary storage of read/write parameters.

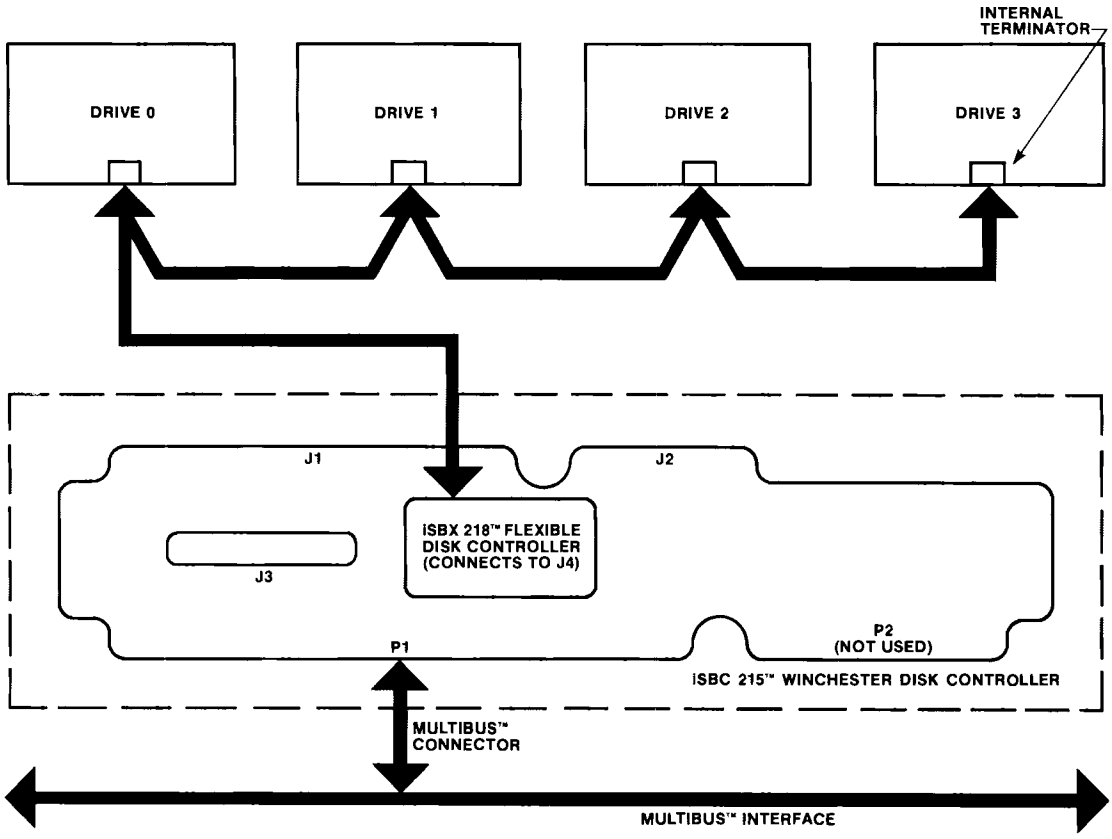


Figure 1-2 Typical Multiple Drive System Using Flexible Disk Drives and iSBX 218™ Flexible Disk Controller

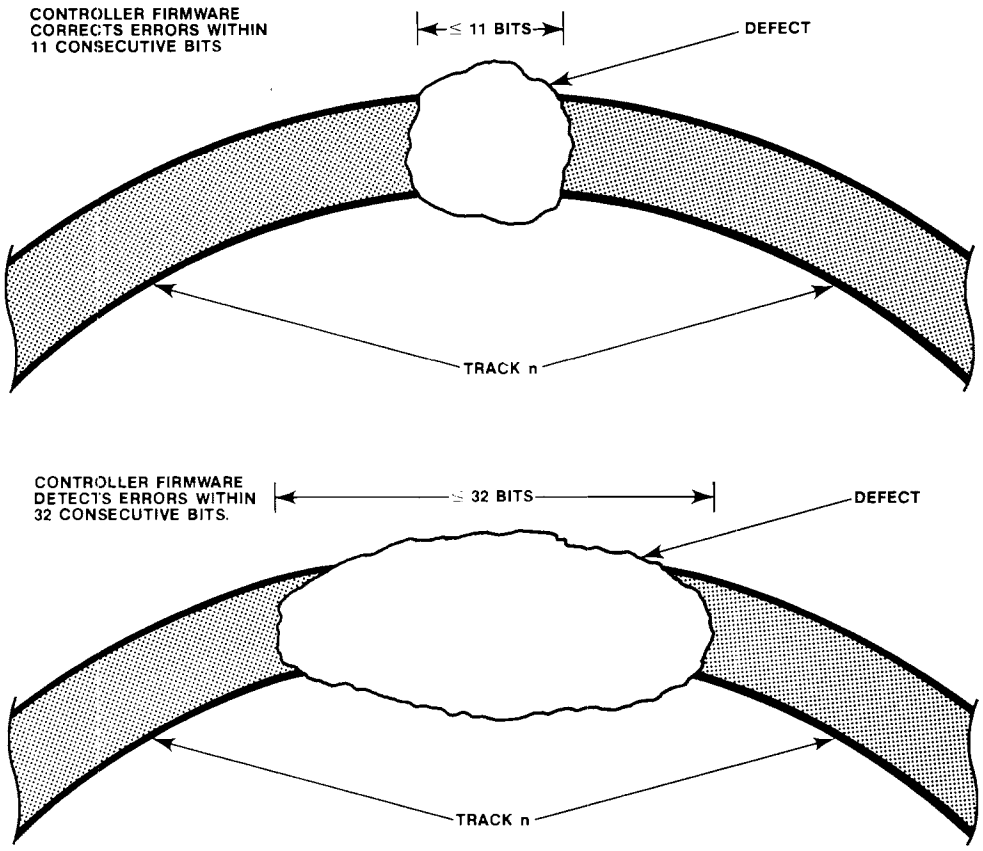


Figure 1-3. Automatic Error Checking and Correction

1-3. SPECIFICATIONS

Table 1-1 lists the physical and performance specifications of the iSBC 215 Winchester Disk Controller; Table 1-2 lists typical characteristics of the Winchester disk drives that are compatible with the iSBC 215 controller.

Table 1-1. Board Specifications

COMPATIBILITY	
CPU:	Any Intel mainframe or any Multibus™ interface compatible CPU. The controller can operate with either 16- or 20-bit addresses and with either 8- or 16-bit data bus widths.
Disk Drive:	Winchester disk drives (see Table 1-2); both open-loop and closed-loop head positioning types. Two versions of controller firmware (located in ROMs U87 and U88) are available, one for use with open-loop type drives and one for closed-loop drives. Flexible disk drives through on-board iSBX™ Connector (see iSBX 218™ Flexible Disk Controller specifications)

Table 1-1. Board Specifications (Continued)

DATA ORGANIZATION AND CAPACITY

Bytes per Sector and Sectors per Track:

Bytes/ Sector	SECTORS [1]					
	5¼" Rotating Memory Systems	14" Shugart	Fujitsu 2300/ Memorex	Pertec	8" Priam	14" Priam
128	54	96	64	69	70	104
256	31	57	38	42	42	62
512	17	31	21	24	23	34
1024	9	16	11	12	12	18

Formatted Disk Capacity:

Bytes Sector	FORMATTED CAPACITY/DRIVE[2]			
	5¼" Rotating Memory Systems	Control Data Corp	8" Shugart/ Quantum	14" Shugart
128	8.40 MBytes	29.25 MBytes	7.08 MBytes	19.86 MBytes
256	9.65 MBytes	28.03 MBytes	8.12 MBytes	23.58 MBytes
512	10.58 MBytes	24.98 MBytes	8.91 MBytes	25.65 MBytes
1024	11.21 MBytes	19.50 MBytes	9.43 MBytes	26.48 MBytes

Bytes Sector	FORMATTED CAPACITY/DRIVE[2] (Cont.)			
	Fujitsu/ Memorex	Pertec	8" Priam	14" Priam
128	7.99 MBytes	12.35 MBytes	23.29 MBytes	22.40 MBytes
256	9.49 MBytes	15.03 MBytes	27.94 MBytes	26.71 MBytes
512	10.49 MBytes	17.17 MBytes	30.62 MBytes	29.29 MBytes
1024	10.98 MBytes	17.18 MBytes	31.95 MBytes	31.02 MBytes

Drives per Controller:

Winchester Disk Drives — Up to four 8" Shugart, Quantum, Pertec or Priam drives through connectors J1 and J2 (see Table 1-2); up to two Memorex drives or 14" Shugart drives.

Flexible Disk Drives — Up to four 5¼" or 8" drives through the iSBX 218 Flexible Disk Controller connected to the iSBC 215™ board's iSBX™ connector, J4.

Error Detecting and Correction: The controller hardware can detect errors of up to 32 bits in length; controller firmware can correct errors of up to 11 bits in length (see figure 1-3).

CONTROLLER CHARACTERISTICS

Mounting: Occupies a card slot in iSBC 604/614 Modular Cardcage/Backplane or equivalent Multibus™ backplane connector.

Physical Characteristics:
- Width: 17.2 cm (6.8 inches)
- Length: 30.5 cm (12.0 inches)
- Height: 1.3 cm (0.5 inches)
- Weight: 0.54 kg (19 ounces)

Power Requirements: +5 Volts ±5% @ 3.25 amperes maximum;
−5 Volts ±5% @ 0.15 amperes maximum.

NOTE

Jumper and on-board voltage regulator allow −5 Volts or −12 Volts from Multibus™ connector to be used as voltage source for −5 Volt.

Environmental:
- Temperature: 0°C to +55°C, operating (+32°F to +131°F).
  −55°C to +85°C, non-operating (−67°F to +185°F).
- Humidity: Up to 90%, non-condensing.

[1]Maximum allowable for corresponding selection of Bytes per Sector.

[2]Applies to the following drive models: 5¼" RMS 512, Control Data Corp 9410-32, 8" Shugart SA1004, Quantum Q2010, 14" Shugart SA4008, Memorex 101, Fujitsu 2301, Pertec D8000, 8" Priam 3450 and 14" Pram 3350.

Table 1-2. Winchester Disk Drive Characteristics

	Rotating Memory Systems 512[1]	Control Data Corp 9410-32[2]	Shugart SA1004[1]/ Quantum Q2010	Fujitsu 2301[1] Memorex 101	Pertec D8020[2]	Priam 3450[2]
Capacity (Unformatted)	12.7 MBytes	31.89 MBytes	10.6 MBytes	11.7 MBytes	20.13 MBytes	34.94 MBytes
Read/Write Surfaces	8	4	4/2[3]	4	3	5
Tracks/Surface	153	595	256	244	466	520
Bytes/Track	10.4 KBytes	13.4 KBytes	10.4 KBytes	12 KBytes	14.4 KBytes	13.4 KBytes
Transfer Rate	625 KBytes/sec	806 KBytes/sec	524 KBytes/sec	593 KBytes/sec	864 KBytes/sec	806 KBytes/sec
Average Access Time	70 msec	50 msec	70 msec	70 msec	50 msec	50 msec
Rotational Latency	8.33 msec	8.33 msec	9.6 msec	10.1 msec	8.34 msec	8.3 msec
Track to Track	3 ms	10 ms	19 msec	20 msec	12 msec	10 msec

[1]Open loop step positioner.
[2]Closed loop servo voice coil technology.
[3]Quantum Q2010 has 2

CHAPTER 2
PREPARATION FOR USE

2-1. INTRODUCTION

This chapter provides information for use in preparing and installing the iSBC 215 Winchester Disk Controller. Included are instructions for unpacking and inspection, installation, setting switches, installing jumpers, and interfacing the controller board with the Multibus connector and disk drives.

2-2. UNPACKING AND INSPECTION

On receipt of the iSBC 215 controller from the carrier, immediately inspect the shipping carton for evidence of damage. If the shipping carton is damaged or water-stained, request that the carrier's agent be present when the carton is opened; if the carrier's agent is not present at the time of opening, keep the carton and packing materials for subsequent agent inspection.

For repairs or replacement of an Intel product damaged during shipment, contact Intel Technical Support Center (refer to Chapter 5) to obtain a Return Authorization Number and further instructions. A copy of the Purchase Order should be submitted to the carrier with the claim.

Carefully unpack the shipping carton and verify that the following items are included:

- iSBC 215 Winchester Disk Controller Printed Wired Assembly
- iSBC 215 Winchester Disk Controller Schematic Diagram

2-3. BOARD INSTALLATION CONSIDERATIONS

The iSBC 215 controller can be installed in any Intel cardcage/backplane or any user-designed backplane that is compatible with the Multibus interface and meets the controller's power and Multibus connector dimensional requirements. The controller occupies one backplane slot.

When installing the controller in a serial priority environment (e.g., within any of the Intel system chassis), wiring modifications are required to support serial priority; a daisy-chain technique, see Figure 2-1, establishes priority, The priority input (BPRN/) of the highest priority master is tied to ground. The priority output (BPRO/) of the highest priority master is then connected to the priority

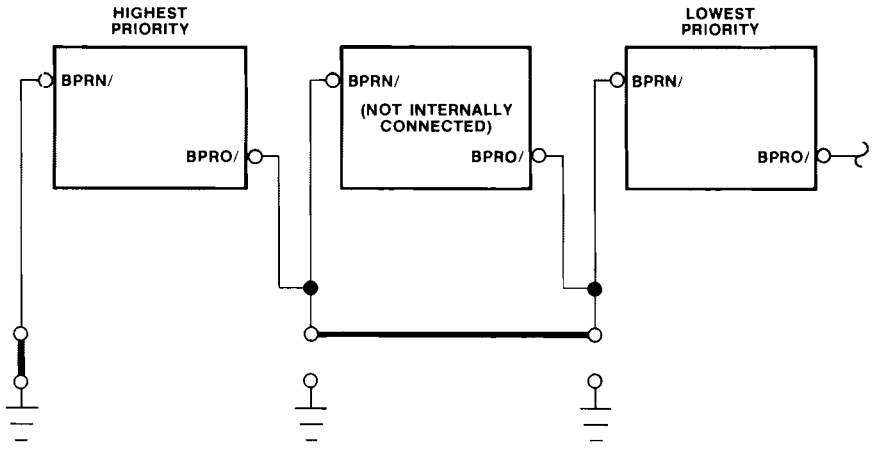


Figure 2-1. Serial Priority Resolution

input (BPRN/) of the next lowest priority master, and so on. ("/" following the signal name indicates an active low). This technique can accommodate a limited number of masters due to gate delays through the daisy-chain.

2-4. POWER REQUIREMENT

The board requires a +5 Volt ±5% power supply at a maximum current of 3.25 amperes, supplied through the Multibus connector. When interfacing with 8" Shugart/Quantum drives, an additional -5 Volt ±5% source at 150 milliamperes maximum is required. This -5-Volt supply can be obtained directly from the Multibus connector or from an on-board regulator that uses either the -10 or -12-Volt source from the Multibus connector (refer to Paragraph 2-14). When interfacing with an iSBX Bus through J3 or J4, additional voltage sources of +12 Volts, -12 Volts or both may be required, also supplied through the Multibus connector. (See individual iSBX Board specifications for tolerances and current requirements of these supplies.) Before installing the controller in a system chassis, make certain that the associated power supplies can supply the additional current that the controller board requires.

2-5. COOLING REQUIREMENT

When the controller is installed in a high temperature environment, make certain the ambient operating temperature does not exceed +55°C.

2-6. MULTIBUS™ CONNECTOR

The controller communicates with the CPU and other boards via the Multibus interface. Table 2-1 lists the Multibus connector pin assignments; Table 2-2 describes the controller Multibus interface signals. Figure 2-2 provides a diagram of the controller/Multibus interface timing signals and a table of the timing requirements. Table 2-3 gives current requirements and other characteristics related to the controller/Multibus interface.

The controller is connected to the Multibus interface through connector P1, an 86-pin, double-sided, printed circuit edge connector with 3.96 mm (0.156 in) contact centers. Connector P2 is not used.

2-7. SWITCH/JUMPER CONFIGURATIONS

A number of switches and jumpers (see Table 2-4) are provided on the controller board that allow the user to conveniently set the controller for the system environment in which it is to operate (8-bit or 16-bit system data bus, 8-bit or 16-bit I/O addressing, etc.) and for the type of drive to which it is to be interfaced (Shugart/Quantum, Memorex, etc., or iSBX board). Figure 5-1 shows the location of these switches and jumpers on the board. They should be set, as described in the following paragraphs, prior to installing the board in a cardcage or backplane.

Table 2-1. Multibus™ Connector P1 Pin Assignment

	P1 (Component Side)			P1 (Circuit Side)		
	Pin	Mnemonic*	Description	Pin	Mnemonic*	Description
Power Supplies	1	GND	Signal GND	2	GND	Signal GND
	3	+5V	+5Vdc	4	+5V	+5Vdc
	5	+5V	+5Vdc	6	+5v	+5Vdc
	7	+12V	+12Vdc	8	+12V	+12Vdc
	9	−5V	−5Vdc	10	−5V	−5Vdc
	11	GND	Signal GND	12	GND	Signal GND
Bus Controls	13	BCLK/	Bus Clock	14	INIT/	Initialize
	15	BPRN/	Bus Pri. In	16	BPRO/	Bus Pri. Out
	17	BUSY/	Bus Busy	18	BREQ/	Bus Request
	19	MRDC/	Mem Read Cmd	20	MWTC/	Mem Write Cmd
	21	IORC/	I/O Read Cmd	22	IOWC/	I/O Write Cmd
	23	XACK/	XFER Acknowledge	24	INH1/	Inhibit 1 disable RAM
Bus Controls and Address	25		Reserved	26	INH2/	Inhibit 2 disable PROM or ROM
	27	BHEN/	Byte High Enable	28	ADR10/	
	29	CBRQ/	Common Bus Request	30	ADR11/	Address
	31	CCLK/	Constant Clk	32	ADR12/	Bus
	33	INTA/	Intr Achknowledge	34	ADR13	

Table 2-1. Multibus™ Connector P1 Pin Assignment (Continued)

	P1 (Component Side)			P1 (Circuit Side)		
	Pin	Mnemonic*	Description	Pin	Mnemonic*	Description
Interrupts	35	INT6/	Parallel Interrupt Requests	36	INT7/	Parallel Interrupt Requests
	37	INT4/		38	INT5/	
	39	INT2/		40	INT3/	
	41	INT0/		42	INT1/	
Address	43	ADRE/	Address Bus	44	ADRF/	Address Bus
	45	ADRC/		46	ADRD/	
	47	ADRA/		48	ADRB/	
	49	ADR8/		50	ADR9/	
	51	ADR6/		52	ADR7/	
	53	ADR4/		54	ADR5/	
	55	ADR2/		56	ADR3/	
	57	ADR0/		58	ADR1/	
Data	59	DATE/	Data Bus	60	DATF/	Data Bus
	61	DATC/		62	DATD/	
	63	DATA/		64	DATB/	
	65	DAT8/		66	DAT9/	
	67	DAT6/		68	DAT7/	
	69	DAT4/		70	DAT5/	
	71	DAT2/		72	DAT3/	
	73	DAT0/		74	DAT1/	
Power Supplies	75	GND	Signal GND	76	GND	Signal GND
	77		Reserved	78		Reserved
	79	−12V	−12Vdc	80	−12V	−12Vdc
	81	+5V	+5Vdc	82	+5V	+5Vdc
	83	+5V	+5Vdc	84	+5V	+5Vdc
	85	GND	Signal GND	86	GND	Signal GND

* "/" following the signal name indicates an active low.

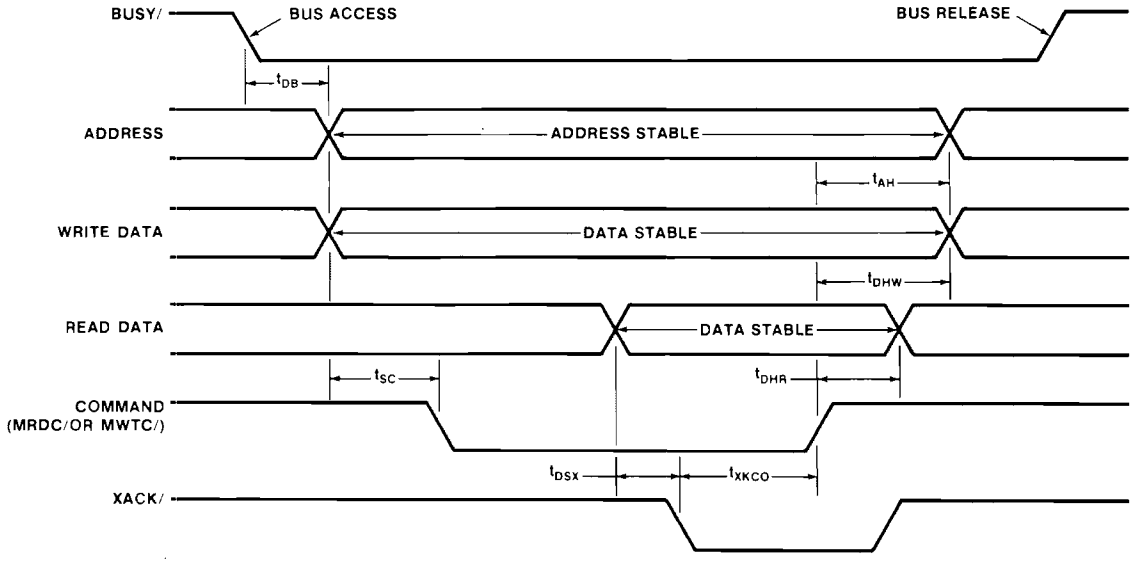
Table 2-2. iSBC 215™ Controller/Multibus™ Interface P1 Signal Descriptions

Signal	Functional Description
ADR0/, ADRF/ ADR10/-ADR13/	*Address.* These 20 lines transmit the address of the memory location or I/O port to be accessed. For memory access, ADR0/ (when active) enables the even byte bank (DAT0/-DAT7/) on the Multibus™ connector; i.e., ADR0/ is active for all even addresses. ADR13/ is the most significant address bit.
BCLK/	*Bus Clock.* Used to synchronize the bus contention logic on all bus masters.
BHEN/	*Byte High Enable.* When active low, enables the odd byte bank (DAT8/-DATF/) onto the Multibus™ connector.
BPRN/	*Bus Priority In.* When low indicates to a particular bus master that no higher priority bus master is requesting use of the bus. BPRN/ is synchronized with BCLK/.
BPRO/	*Bus Priority Out.* In serial (daisy chain) priority resolution schemes, BPRO/ must be connected to the BPRN/ input of the bus master with the next lower bus priority.
BREQ/	*Bus Request.* In parallel priority resolution schemes, BREQ/ indicates that a particular bus master requires control of the bus for one or more data transfers. BREQ/ is synchronized with BCLK/.
BUSY/	*Bus Busy.* Indicates that the bus is in use and prevents all other bus masters from gaining control of the bus. BUSY/ is synchronized with BCLK/.
CBRQ/	*Common Bus Request.* Indicates that a bus master wishes control of the bus but does not presently have control. As soon as control of the bus is obtained, the requesting bus controller raises the CBRQ/ signal.
DAT0/-DATF/	*Data.* These 16 bidirectional data lines transmit and receive data to and from the addressed memory location or I/O port. DATF/ is the most-significant bit. For data byte operations, DAT0/-DAT7 is the even byte and DAT8-DATF/ is the odd byte.

Table 2-2. iSBC 215™ Controller/Multibus™ Interface P1 Signal Descriptions (Continued)

Signal	Functional Description
INIT/	*Initialize.* Reset the entire system to a known internal state.
INT0/-INT7/	*Interrupt Request.* These eight lines transmit interrupt requests to the appropriate interrupt handler. INT0/ has the highest priority.
IOWC/	*I/O Write Command.* Indicates that the address of an I/O port is on the Multibus™ connector address lines and that the contents on the Multibus™ connector data lines are to be accepted by the addressed port.
MRDC/	*Memory Read Command.* Indicates that the address of a memory location is on the Multibus™ connector address lines and that the contents of that location are to be read (placed) on the Multibus™ connector data lines.
MWTC/	*Memory Write Command.* Indicates that the address of a memory location is on the Multibus™ connector address lines and that the contents on the Multibus™ connector data lines are to be written into that location.
XACK/	*Transfer Acknowledge.* Indicates that the address memory location has completed the specified read or write operation. That is, data has been placed onto or accepted from the Multibus™ connector data lines.

Master Command Access Timing



Slave Command Timing

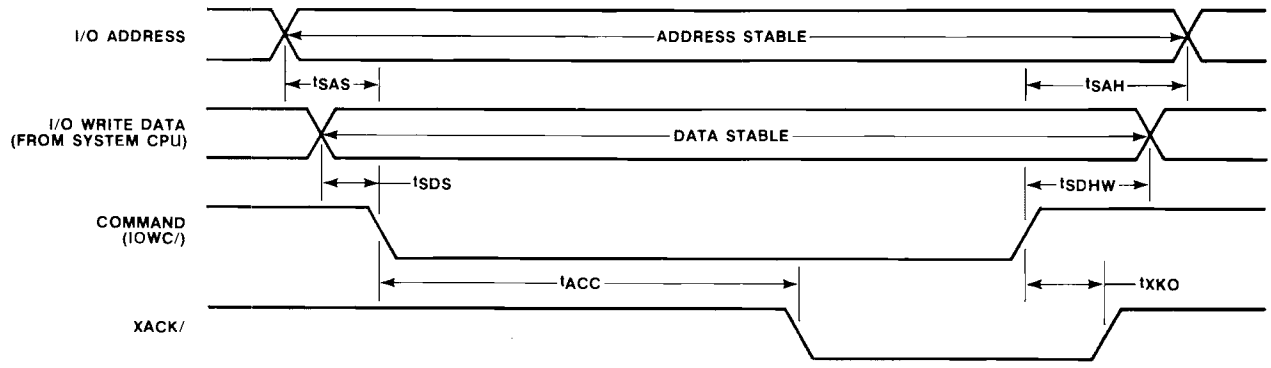
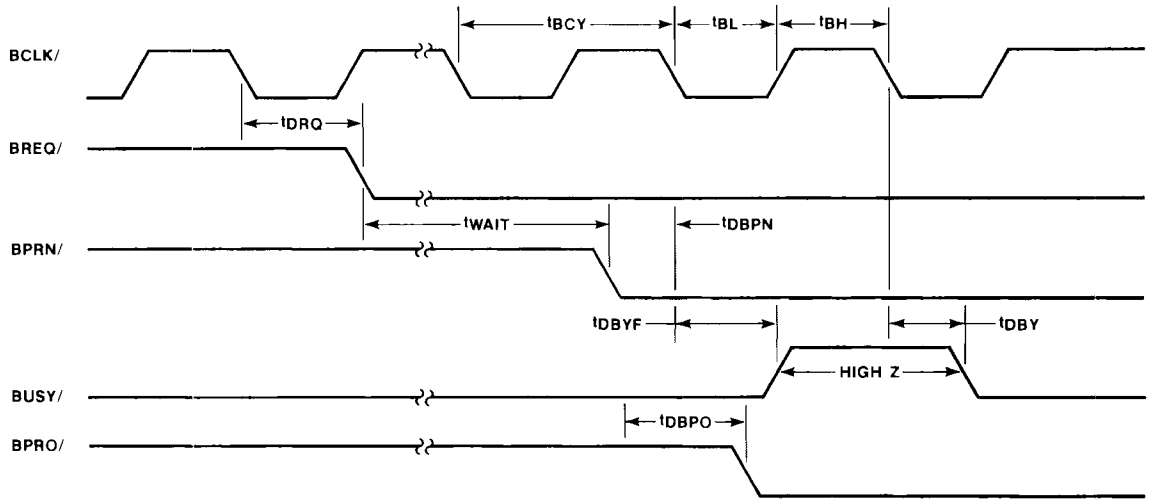


Figure 2-2. Master Command Access Timing

Bus Exchange Timing



Parameter	Time in Nanoseconds		Description
	Minimum	Maximum	
t_{SAS}	50		Address Setup Time to I/O Command
t_{SDS}	0		Data Setup Time to I/O Command
t_{SAH}	15		Address Hold Time from I/O Command
t_{SDHW}	30		Data Hold Time from I/O Command
t_{ACC}		8000	I/O Access Time
t_{XKO}	100		XACK/Hold Time from I/O Command
t_{BCY}	125		Bus Clock Cycle Time
t_{BL}	65		Bus Clock Low
t_{BH}	35		Bus Clock High
t_{DRQ}		35	Bus Request Delay
t_{DBY}		60	Bus Busy Turn On Delay
t_{DBYF}		35	Bus Busy Turn Off Delay
t_{DBPN}	15		Priority Input Setup Time
t_{DBPO}		25	BPRO/Serial Delay from BPRN/
t_{WAIT}		∞	Requesting Master Bus Access Time
t_{DB}	50		Busy to Address/Data Delay
t_{SC}	50		Address/Data Setup to Command
t_{XKCO}		750	XACK/ to Command Turn Off
t_{AH}	50		Address Hold Time
t_{DHW}	50		Data Hold Time
t_{DHR}	0		Read Data Hold Time
t_{DSX}	0		Data Setup Time Before XACK/

Figure 2-2. Master Command Access Timing (Continued)

Table 2-3. iSBC 215™ Controller/Multibus™ Interface Signal Characteristics

Bus Signals	Driver 1, 3					Receiver 2, 3			
	Location	Type	I_{OL} Min_{ma}	I_{OH} $Min_{\mu a}$	C_O Min_{pf}	Location	I_{IL} Max_{ma}	I_{IH} $Max_{\mu a}$	C_I Max_{pf}
DAT0/-DATF/ (16 lines)	Masters	TRI	32	−5000	300	Masters and Slaves	−0.5	125	18
ADR0/-ADR13/, BHEN/ (21 lines)	Masters	TRI	32	−5000	300	Slaves	−0.8	90	18
MRDC/, MWTC/	Masters	TRI	32	−5000	300	Slaves	−0.7	50	18
IOWC/						Slaves	−0.4	20	5
XACK/	Slaves	TRI	48	−2000	300	Masters	−1.2	60	18
BCLK/						Master	−0.5	60	18
BREQ/	Each Master	TTL	10	−400	60				
BPRO/	Each Master	TTL	10	−400	60				
BPRN/						Master	−0.5	60	18
BUSY/, CBRQ	All Masters	O.C.	20	-	250	All Masters	−0.5	60	18
INIT/						All	−0.5	60	18
INT0/-INT7/ (8 lines)	Slaves	O.C.	40	-	300				

Notes:

1. Driver Requirements:
   - I_{OH} = High Output Current Drive
   - I_{OL} = Low Output Current Drive
   - C_O = Capacitance Drive Capability
   - TRI = 3-State Drive
   - O.C. = Open Collector Driver
   - TTL = Totem-pole Driver
2. Receiver Requirements:
   - I_{IH} = High Input Current Load
   - I_{IL} = Low Input Current Load
   - C_I = Cap Active Load
3. Low and High Voltage Requirements:

   Receiver:

   $0 \leqq V_{IL} \leqq 0.8V$

   $2.0V \leqq V_{IH} \leqq 5.5V$

   Driver:

   $0 \leqq V_{OL} \leqq 0.5V$

   $2.4V \leqq V_{OH} \leqq 5.5V$

2-8. WAKE-UP ADDRESS SELECTION

The controller communicates with the host CPU through four I/O communications blocks located in the host memory. When the controller is to receive instructions, it goes to the beginning address of the first I/O communication block. This address is called the wake-up address (WUA). The WUA may be at any address in host memory. Sixteen WUA switches (S1-1 through S1-8 and S2-3 through S2-10, see Figure 5-1) are provided on the controller board that allow the user to set the controller for the selected wake-up address. The function of each switch is shown in the table in Figure 5-1. Any switch set to ON represents a logical 1.

The controller multiplies the settings of the WUA switches by 2^4 (shifts the number four places to the

left) to create a 20-bit WUA. Note that due to this shift, the four least-significant bits of the selected WUA must be zeros. When accessing host memory, the controller transmits the entire 20-bit WUA through the Multibus interface. If the host memory uses 16-bit addressing, the four most significant bits of the 20-bit WUA must be zero. This is accomplished by setting the four most significant bits of the WUA switches (S1-1 through S1-4) to zero.

Table 2-4. Configuration Jumpers and Switches

Function	Pin or Switch
Wake-Up Address	S1-1 through S1-8 S2-3 through S2-10
8-Bit or 16-Bit System Data Bus Capability	S2-1
8-Bit or 16-Bit Host Processor I/O Port Addressing	S2-2
Interrupt Priority Level	W19-C to W19-0 through W19-7
Any Request	W18
Common Bus Request	W23
Voltage Selection	W20- and W21
Winchester Drive Manufacturer Selection	W1, W2, W5, W6 through W10 W13 through W17, W22
iSBX Bus Control	W3, W4, W11 and W12, W24

2-9. WAKE-UP I/O PORT ADDRESS SELECTION

The host processor communicates with the controller through an I/O port. The WUA switches also set the address of this I/O port. For a host processor with 8-bit I/O port addressing, bits 0 through 7 of the unshifted WUA determine the wake-up I/O port address; for a host processor with 16-bit I/O port addressing, bits 0 through F determine the address.

I/O Address Selection switch S2-2 on the controller board (see Figure 5-1) determines the type of I/O port addressing the host processor uses: ON for 16-bit addressing; OFF for 8-bit addressing.

2-10. SYSTEM DATA BUS SELECTION

System data bus selection switch S2-1 on the controller board (see Figure 5-1) sets the controller for the type of system data bus with which the controller is to interface: ON for 16-bit bus, OFF for an 8-bit bus. This switch allows the controller to use its 16-bit data transfer mode to access the system bus (if the system memory supports 16-bit accesses), even though the host processor only supports 8-bit accesses.

2-11. INTERRUPT PRIORITY LEVEL

The controller's internal interrupt request signal can be assigned to any of eight interrupt priority levels (INT0/ to INT7/) on the Multibus connector. To select the interrupt request priority level, place a jumper link as shown in Table 2-5 and Figure 5-1.

Table 2-5. Interrupt Priority Level Selection

Priority Level Selected	Wire Wrap	
	From Pin	To Pin
0	W19-C	W19-0
1	W19-C	W19-1
2	W19-C	W19-2
3	W19-C	W19-3
r	W19-C	W19-4
5	W19-C	W19-5
6	W19-C	W19-6
7	W19-C	W19-7

2-12. ANY REQUEST SELECTION

The *any request* function allows the controller to be set to relinquish control of the Multibus interface following a request from:

1. A higher priority device only (jumper between pins W18-1 and W18-2 on the controller board).
2. Any device, lower or higher priority, (jumper between pins W18-1 and W18-3).

Figure 5-1 shows the location of the selection pins.

2-13. COMMON BUS REQUEST

The common bus request function allows the controller to take advantage of higher bus transfer rates by arbitrating for the use of the bus only when other bus controllers have access requests pending. The controller will:

1. Arbitrate for the bus on every access, (jumper between pins W23-1 and W23-2 on the controller). This mode is used when other bus controllers do not implement common bus request.
2. Arbitrate for the bus to acquire the bus for the first access and rearbitrate only when another bus controller requests use of the bus.

2-14. WINCHESTER DRIVE INTERFACE

The iSBC 215 Winchester Disk Controller has been designed to communicate with any of four unique

Table 2-6. 8″ Winchester Drive Manufacturer Selection

Jumper No.	MANUFACTURER												Function
	5¼″ RMS		8″ Shugart/ Quantum		Memorex/ 14″ Shugart Fujitsu 2300		Pertec		Priam		CDC		
	From	To	From	To	From	To	From	To	From	To	From	To	
W1	1	3	1	3	1	3	1	2	1	2	1	3	Open/Closed Head Positioning
W2	—	—	—	—	1	2	—	—	1	2	1	2	Vendor Select
W5	1	2	1	3	1	2	1	2	1	2	1	3	RD —
W6	1	2	1	3	1	2	1	2	1	2	1	3	RD + } Level
W7	1	2	1	3	1	2	1	2	1	2	1	3	RDCL + } Select
W8	1	2	1	3	1	2	1	2	1	2	1	3	RDCL - }
W9	—	—	1	2	—	—	—	—	—	—	1	2	Shugart Tri-State Select
W10	1	2	1	2	—	—	1	2	1	2	1	2	Radial Select
W13	1	2	1	2	1	3	1	2	1	3	1	3	Hard/Soft Sectoring
W14	1	2	1	2	1	3	1	3	1	3	1	3	Shugart AM Control
W15	—	—	—	—	1	2	1	2	1	2	1	2	Shugart GAP Control
W16	1	2	1	2	1	3	1	2	1	3	1	3	Hard/Soft Sectoring
W17	1	2	1	2	1	2	1	2	—	—	1	2	INDEX Select
W22	1	2	1	2	1	2	1	3	1	2	1	2	Pertec RD Clock Select

NOTE
— means not installed
The iSBX bus control jumpers, W3, W4, W11 and W12, are factory wired for the configuration required when the iSBX Bus is not being used. See Paragraph 2-17 and Table 2-9 for a description of the use of these jumpers.

Winchester technology disk drive interfaces: 8″ Shugart/Quantum, Memorex/14″ Shugart, Pertec and Priam.[1] The Shugart, Quantum and Memorex drives use a stepper motor for head positioning (called open-loop head positioning); the Pertec and Priam drives use a linear positioner coupled with a servo surface on one disk for position feedback (closed-loop head positioning).

[1] *The manufacturer's models with which the controller interfaces are: 8″ Shugart (Models SA1002 and SA1004), Quantum (Models Q2010, Q2020, Q2030 and Q2040), Memorex (Models 101 and 102), 14″ Shugart (Models SA4004 and SA4008), Pertec (Model D8000), Rotating Memory Systems (Models 506 and 512) and Control Data Corporation (Models 9410 24 and 32), Priam (Models 570, 1070, 2050, 3350 and 3450).*

The controller can control up to four 8″ Shugart, Quantum, Pertec or Priam drives, or up to two Memorex or 14″ Shugart drives. It cannot control drives of different manufacturers concurrently.

The jumpers listed in Table 2-6 allow the controller to be set for the selected drive type. In addition, two versions of the controller firmware (located in ROMs U87 and U88) are available, one for use with open-loop type drives and one for closed-loop drives. Boards configured for use with open-loop drives come from the factory with open-loop firmware installed and with jumpers preset for 8″ Shugart/ Quantum drives; boards configured for closed-loop drives come with closed-loop firmware and with jumpers preset for Pertec drives. Converting the controller from the 8″ Shugart/Quantum interface to a Memorex/14″ Shugart interface or from Pertec to Priam merely requires changing the connections of some of the jumpers as shown in Table 2-6 and Figure 5-1. Converting the controller from an open-loop interface to a closed-loop interface, and vice versa, requires the ROMs to be changed in addition to changing jumpers.

Interface cables must also be constructed and installed according to the type of drive being used as described in Paragraph 2-15.

2-15. -5-VOLT SELECTION (8″ SHUGART/ QUANTUM CDC DRIVES ONLY)

Figure 5-1 shows the location of the Voltage Selection pins for the -5 Volt power supply. Install jumpers as described in Table 2-7 to select -5 volts either from the Multibus connector or from the on-board regulator and to select the voltage source for the regulator.

2-16. CABLING REQUIREMENTS

Interface cables between the controller and the disk drives must be fabricated according to the type of drive being used and the number of drives. Figures 2-3 through 2-7 show the connector pin assignments for the controller and for each type of drive. A 50-pin mass-terminated socket connector 3M 3425/6050 or

equivalent, is recommended for mating with J1 of the controller board. A 40-pin 3M 3417-6040 or equivalent connector is recommended for mating with J2. The mass-terminated sockets are easily attached to flat ribbon cable using the jig that the connector manufacturer supplies. The Control Cable that connects to J1 requires a 50-conductor ribbon cable; the Read/Write cable that connects to J2 requires one or two 20-conductor ribbon cables, depending on the drive configuration (refer to Paragraph 2-16). Cable length for the control cable cannot exceed a total length of 10 feet; total length for any Read/Write cable must not exceed 10 feet. See the respective service manual for the type of connectors required for the cable end that connects to the drives.

Each of the cables shown in Figures 2-3 through 2-7 require a number of wire cross-overs "scrambling" between the controller connectors and the drives. It is suggested that the scrambling be done at the drive interface connector.

NOTE

The cabling and drive interconnecting information given in Paragraphs 2-15 and 2-16 and in Figures 2-3 through 2-6, reflect the specifications at the time this manual was printed. Before proceeding with construction of interconnecting cables, check the drive's hardware reference manual for current pin assignments and interface requirements.

2-17. DRIVE INSTALLATION

The requirements for connecting the controller to the disk drive or drives varies between drive types. The following discussion and Figure 2-10 describes the specific interconnection requirements for each drive type.

Shugart SA1000 or Quantum Q2000. When connecting the controller to a single 8" Shugart/ Quantum drive, a Shugart SA1200 Data Separator and three interconnecting cables are required (see Figure 2-10. One control cable and one NRZ read/ write cable are required to interface the controller with the drive and data separator, respectively. A separate MFM read/write cable is then required to transmit read/write information between the data separator and the drive.

When controlling multiple drives, Drive 0 (which is called the master and is equipped with the data separator) allows control and read/write data to be routed to and from up to three additional slave drives. The control cable for multiple drive configurations is daisy-chained from the master to the slave drives. Physically, the cable consists of a ribbon cable with an in-line connector for each drive. One MFM read/write cable is required from each slave drive to the master drive.

Memorex 101 and 102 or Shugart SA4000. The controller can drive one or two Memorex/14" Shugart drives. When connecting the controller to a single drive, both a control and a read/write cable are required. When controlling two drives, a single cable, such as the control cable described for the Shugart/Quantum drives, is required that daisy-chains the control information to both drives as shown in Figure 2-10. A split (bifurcated) cable is required to route NRZ read/write data to and from the two drives.

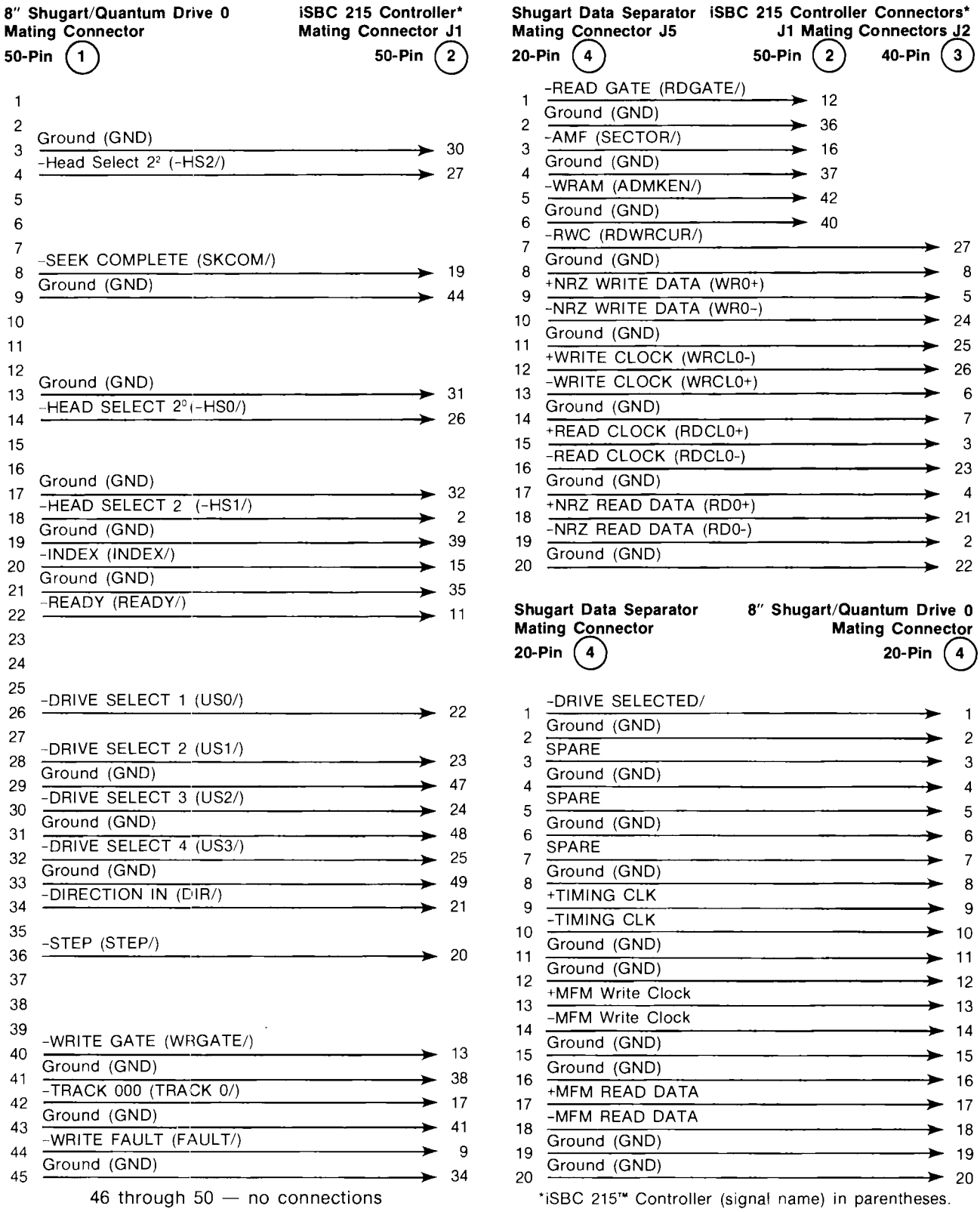
Pertec D8000 and Priam 570, 1070, 2050 and 3450. The connector on the Pertec and Priam drives transmit both control and read/write data. When connecting the controller to a single drive, a bifurcated (split) cable that combines the control lines and the read/write lines from the controller is requires as shown in Figure 2-10. When controlling multiple drives, a cable such as the control cable described for the Shugart drives is required that daisy-chains the control and read/write information between the four drives.

RMS 500. When connecting the controller to a single RMS drive, an RMS Data Separator and three interconnecting cables are required. See Figure 2-8 similar to Shugart SA1000 and Quantum Q2000 above.

Table 2-7. -5-Volt Selection

Jumper	From	To	Function
W21	1	2	Select -5 volts from Multibus™ connector
	1	3	Select -5 volts from regulator (requires jumper to be set on W20)
W20	1	2	Select -10 volts from Multibus™ connector as source for -5 Volt regulator
	1	3	Select -12 volts from Multibus™ connector as source for -5 Volt regulator

8" Shugart/Quantum Drive Cable Wiring Diagrams



*iSBC 215™ Controller (signal name) in parentheses.

Figure 2-3. 8" Shugart/Quantum Drive Interconnecting Cable Requirements

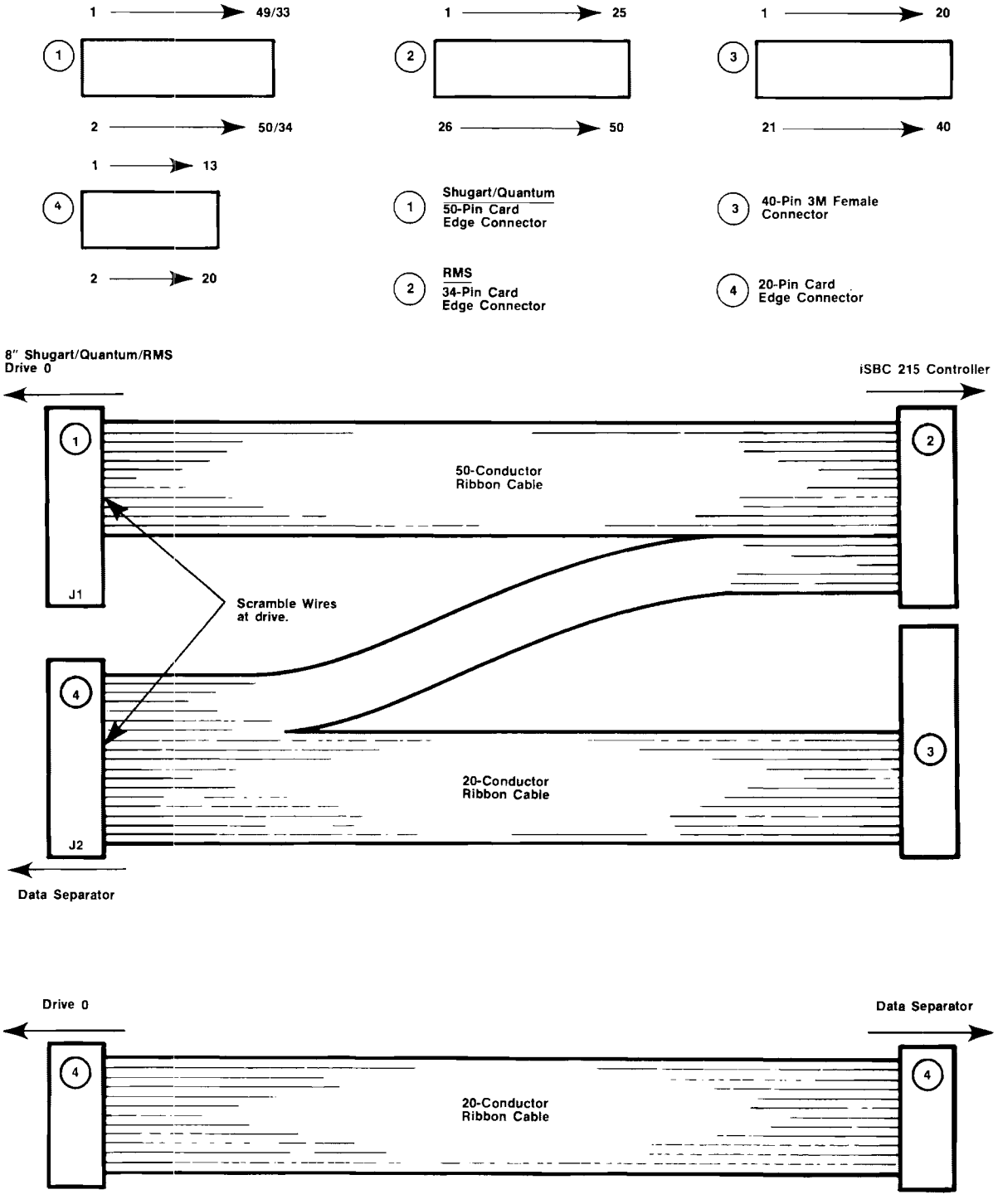
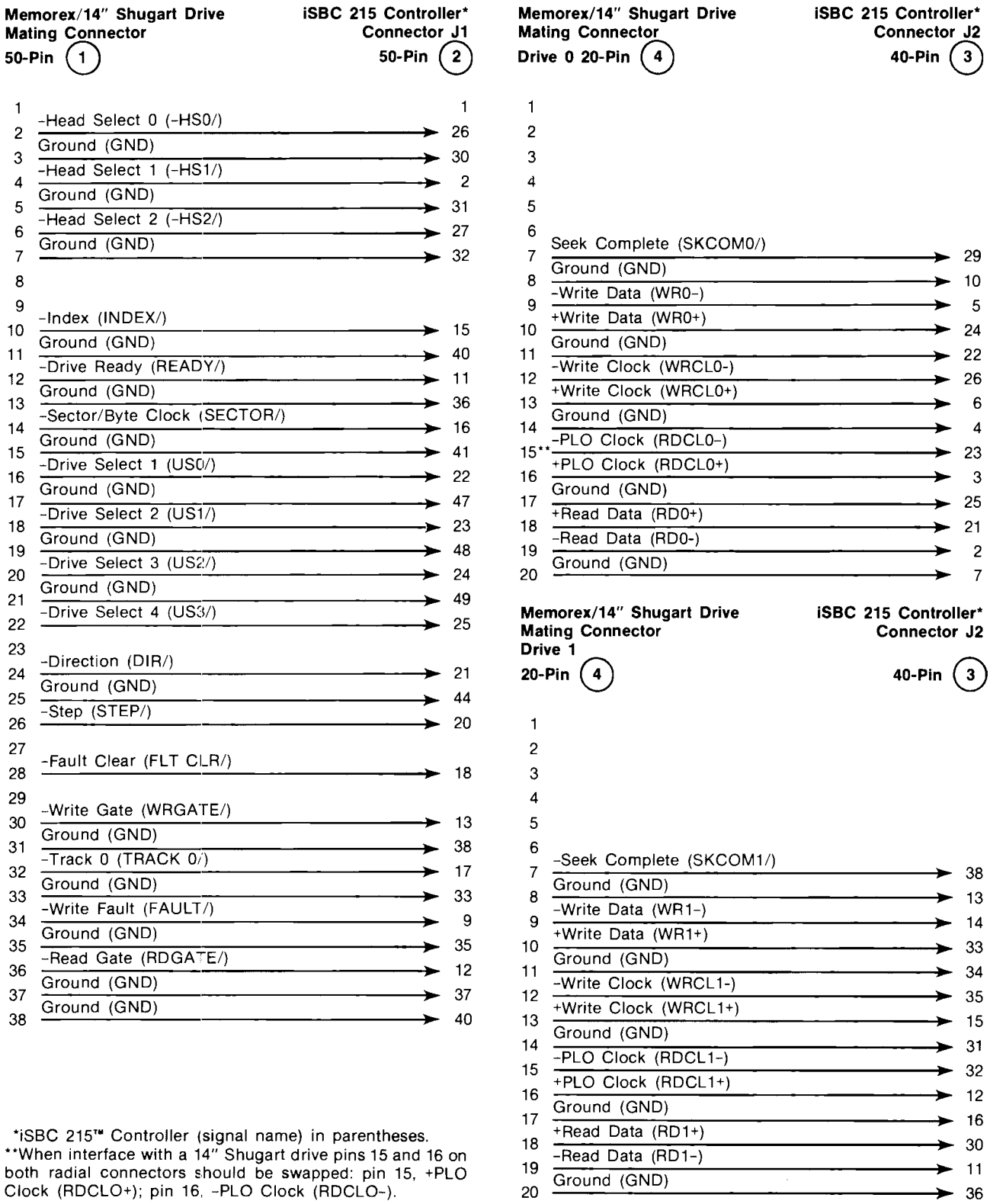


Figure 2-3. 8" Shugart/Quantum Drive Interconnecting Cable Requirements (Continued)

Fujitsu 2300/Memorex/14" Shugart Drive Cable Wiring Diagram



*iSBC 215™ Controller (signal name) in parentheses.
**When interface with a 14" Shugart drive pins 15 and 16 on both radial connectors should be swapped: pin 15, +PLO Clock (RDCLO+); pin 16, -PLO Clock (RDCLO-).

Figure 2-4. Fujitsu 2300/Memorex/14" Shugart Drive Interconnecting Cable Requirements

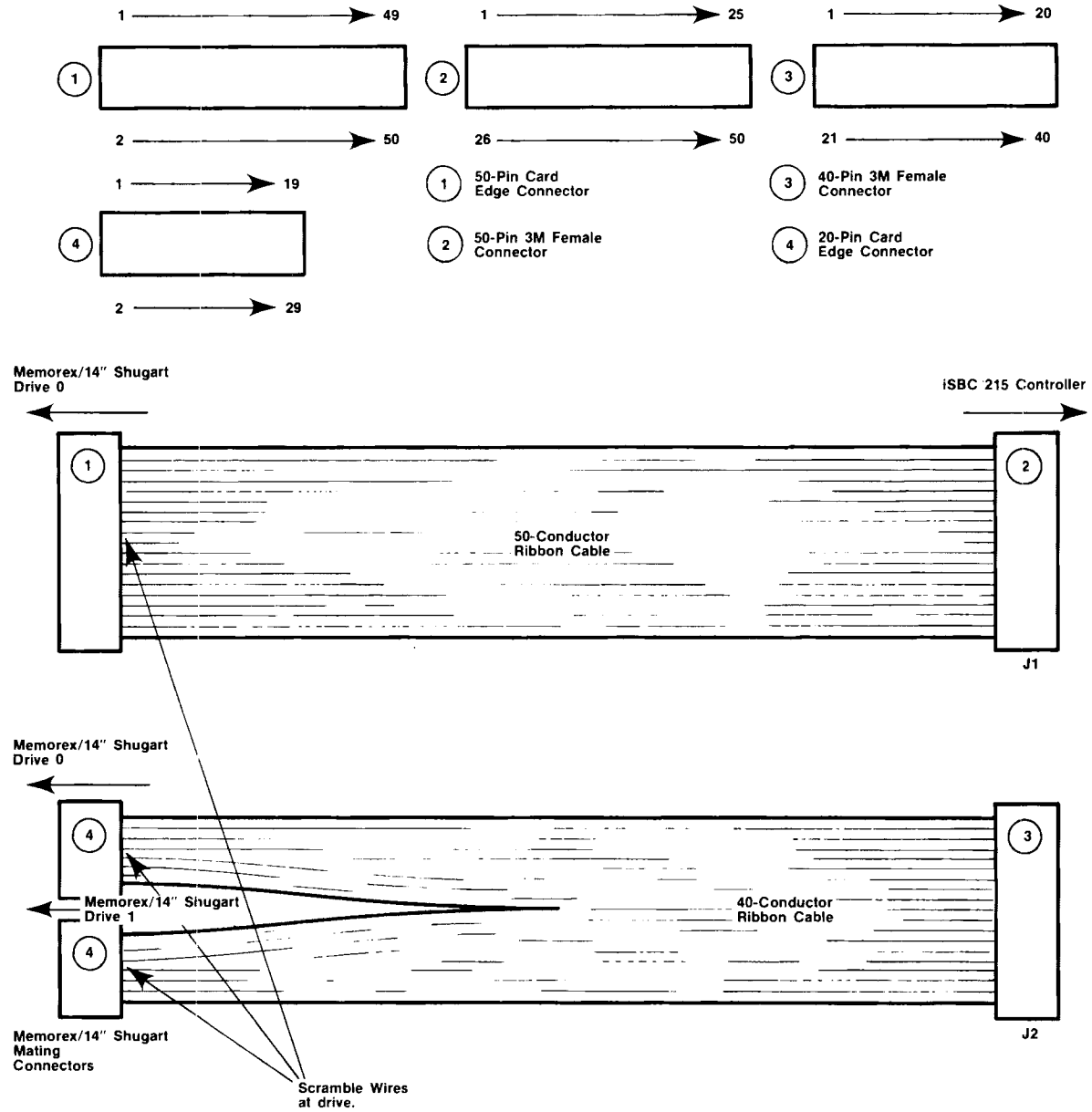


Figure 2-4. Fujitsu 2300/Memorex/14" Shugart Drive Interconnecting Cable Requirements (Continued)

Pertec Drive Cable Wiring Diagram

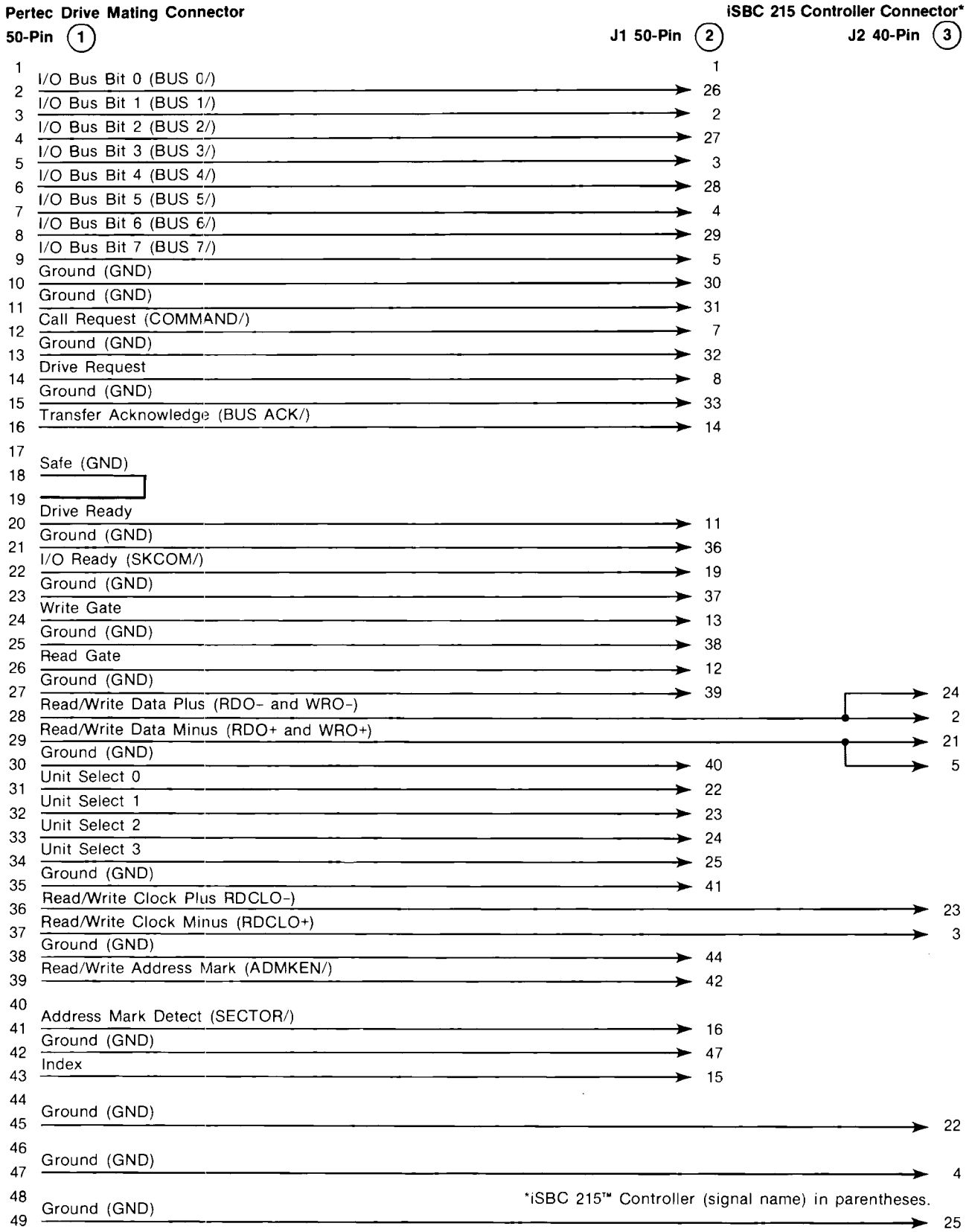


Figure 2-5. Pertec Drive Interconnecting Cable Requirements

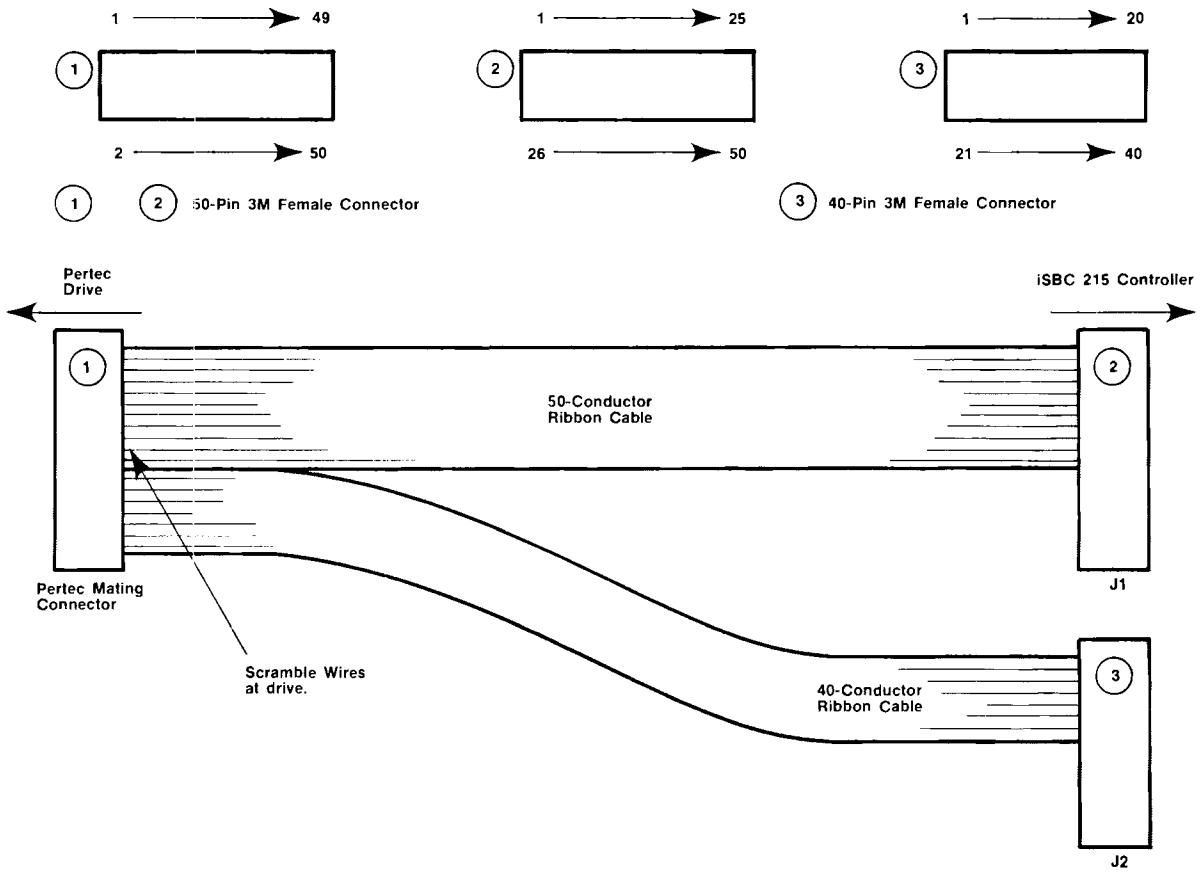


Figure 2-5. Pertec Drive Interconnecting Cable Requirements (Continued)

Priam Drive Cable Wiring Diagram

Priam Drive Mating Connector 50-Pin (1)		J1 50-Pin (2)		iSBC 215 Controller Connector J2 40-Pin (3)
1		1		
2	+DBUS 0 (BUS 0/)	26		
3	+DBUS 1 (BUS 1/)	2		
4	+DBUS 2 (BUS 2/)	27		
5	+DBUS 3 (BUS 3/)	3		
6	+DBUS 4 (BUS 4/)	28		
7	+DBUS 5 (BUS 5/)	4		
8	+DBUS 6 (BUS 6/)	29		
9	+DBUS 7 (BUS 7/)	5		
10	Ground (GND)	30		
11	-READ GATE	12		
12	Ground (GND)	31		
13				
14	Ground (GND)	32		
15	-WRITE GATE	13		
16	Ground (GND)	33		
17	-RD	7		
18	-WR	20		
19	+AD1	45		
20	+AD0	21		
21	Ground (GND)	35		
22	-DRIVE SELECT 1	22		
23	-DRIVE SELECT 2	23		
24	-DRIVE SELECT 3	24		
25	-DRIVE SELECT 4	25		
26	Ground (GND)	38		
27	Ground (GND)	39		
28				
29	-HEAD SELECT 3	43		
30	-HEAD SELECT 2	18		
31	-HEAD SELECT 1	10		
32	Ground (GND)	41		
33	-INDEX	15		
34	Ground (GND)	19		
35	-READY	11		
36	Ground (GND)	44		
37	-SECTOR MARK	16		
38	Ground (GND)	47		
39	+WRITE DATA			5
40	-WRITE DATA			24
41	Ground (GND)			22
42	+WRITE CLOCK			26
43	-WRITE CLOCK			6
44	Ground (GND)			4
45	+READ/REFERENCE CLOCK			23
46	-READ/REFERENCE CLOCK			3
47	Ground (GND)			25
48	+READ DATA			21
49	-READ DATA			2
50	Ground (GND)			7

*iSBC 215™ Controller (signal name) in parentheses.

Figure 2-6. Priam Drive Interconnecting Cable Requirements

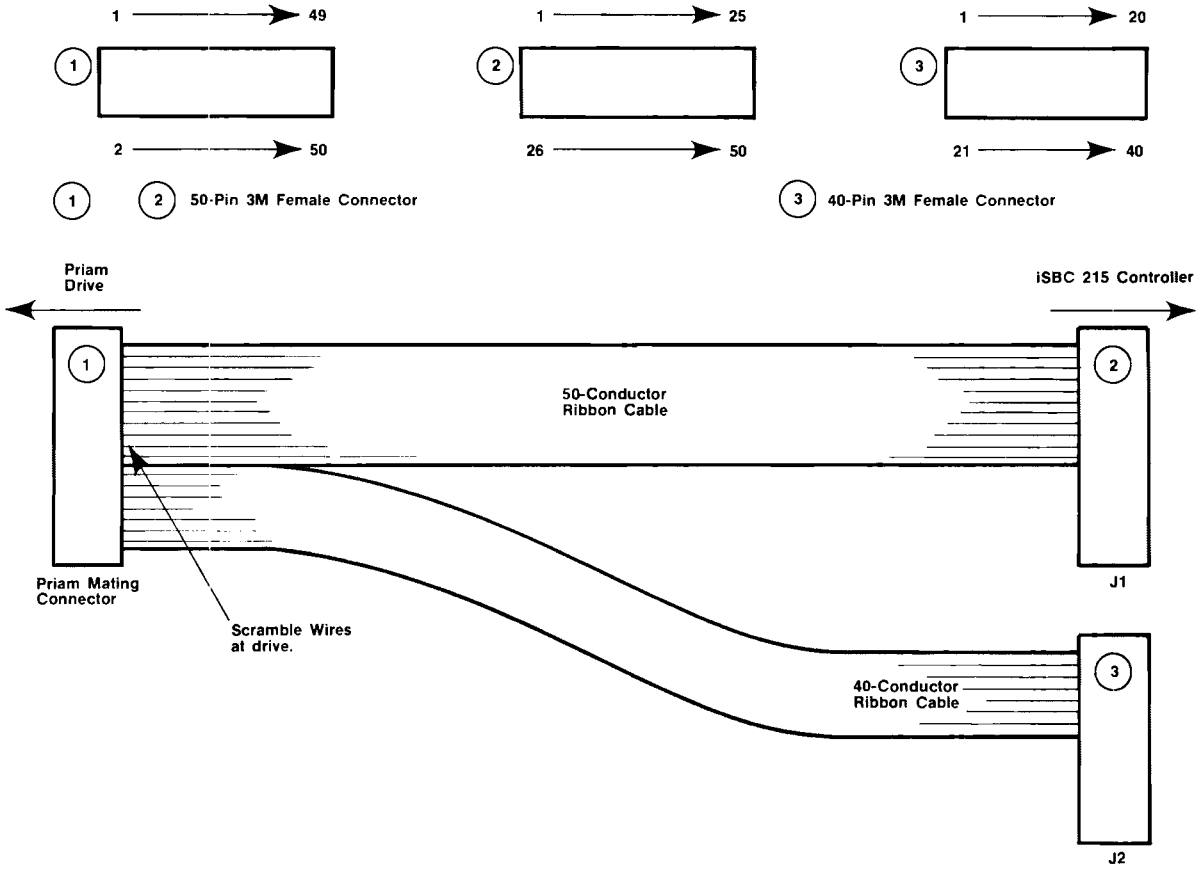
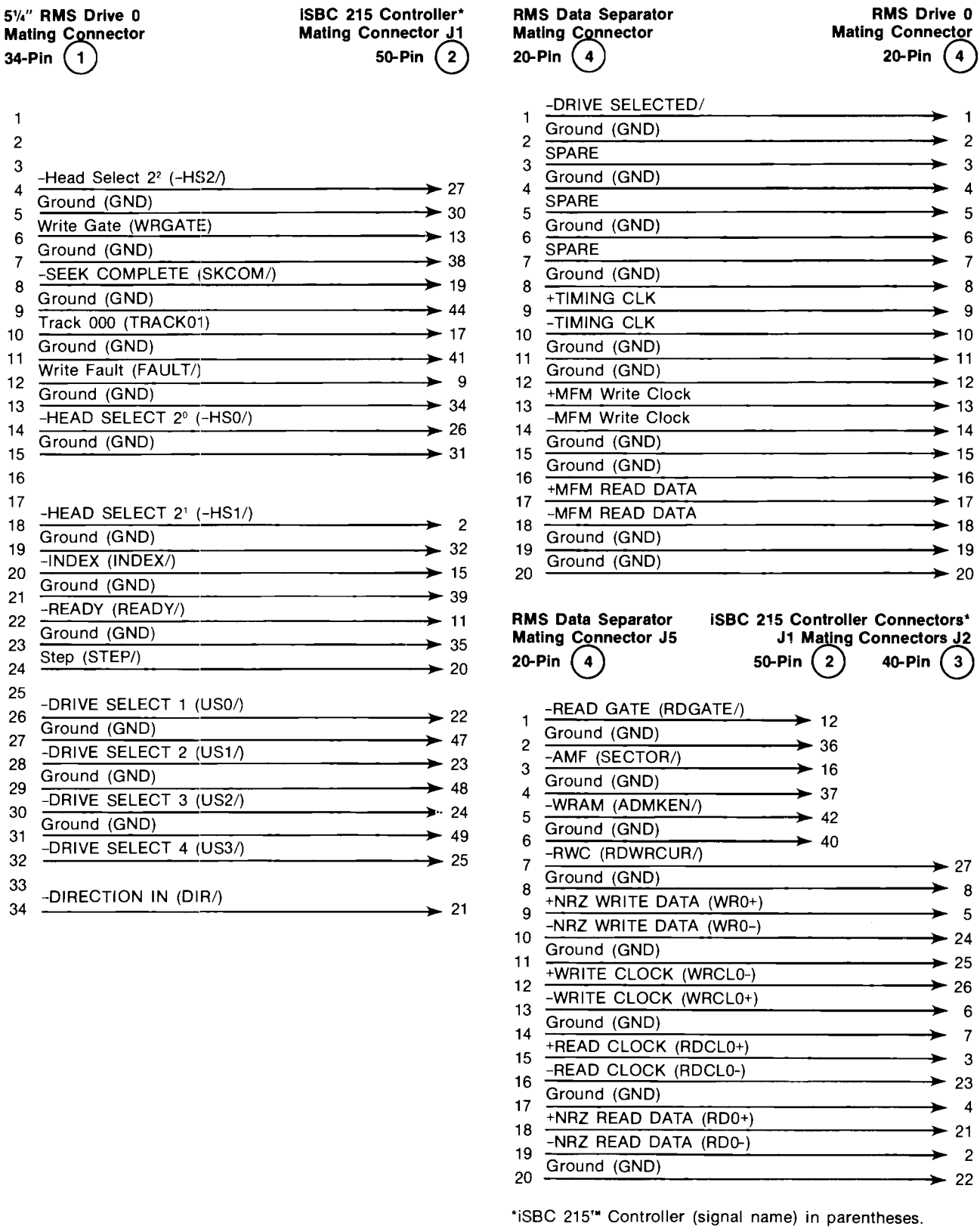


Figure 2-6. Priam Drive Interconnecting Cable Requirements (Continued)

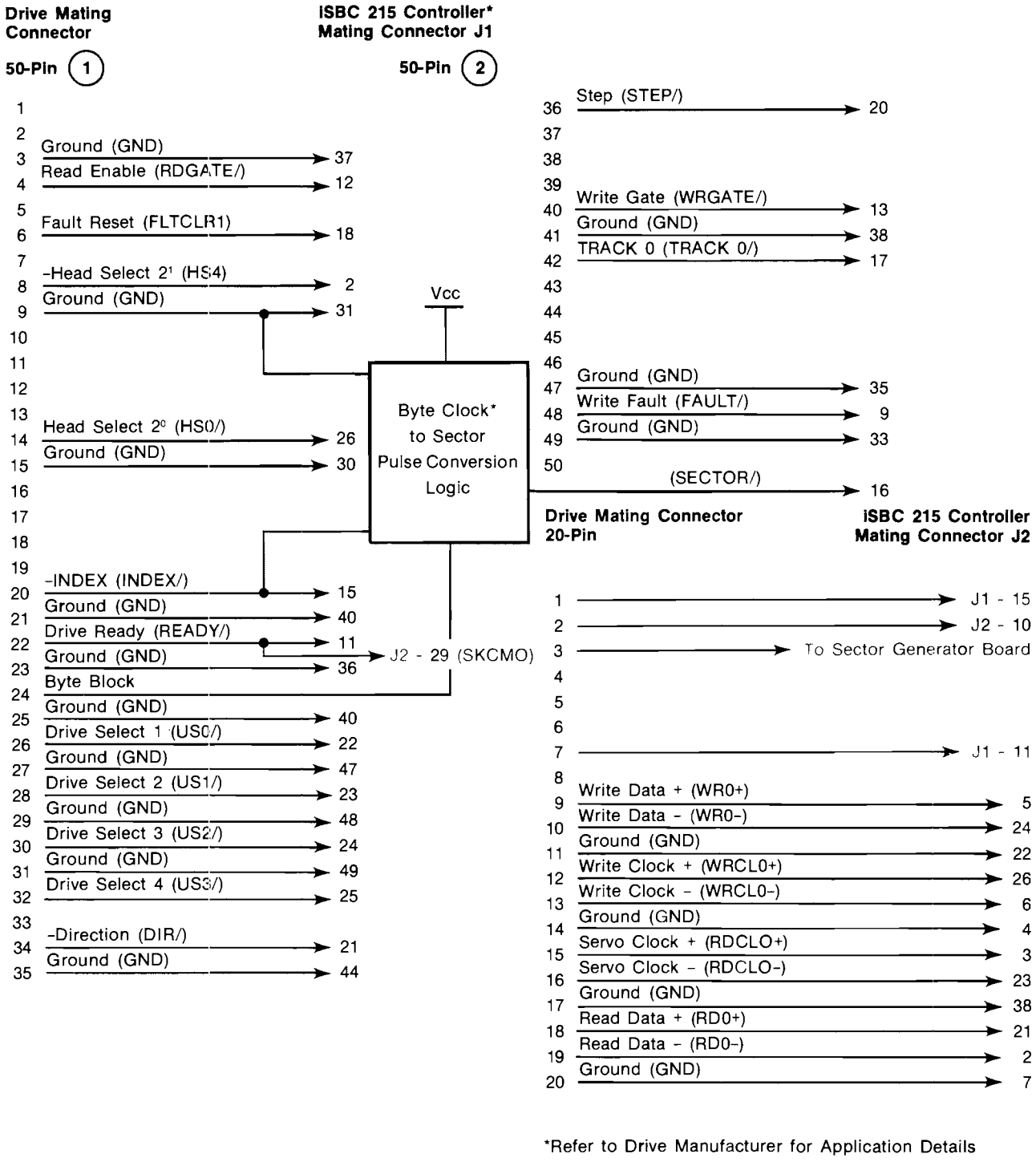
5¼" RMS Drive Cable Wiring Diagrams



*iSBC 215™ Controller (signal name) in parentheses.

Figure 2-7. 5¼" RMS Drive Interconnecting Cable Requirements

Control Data Corp Drive Cable Wiring Diagrams



*Refer to Drive Manufacturer for Application Details

Figure 2-8. Control Data Corporation Drive Interconnecting Cable Requirements

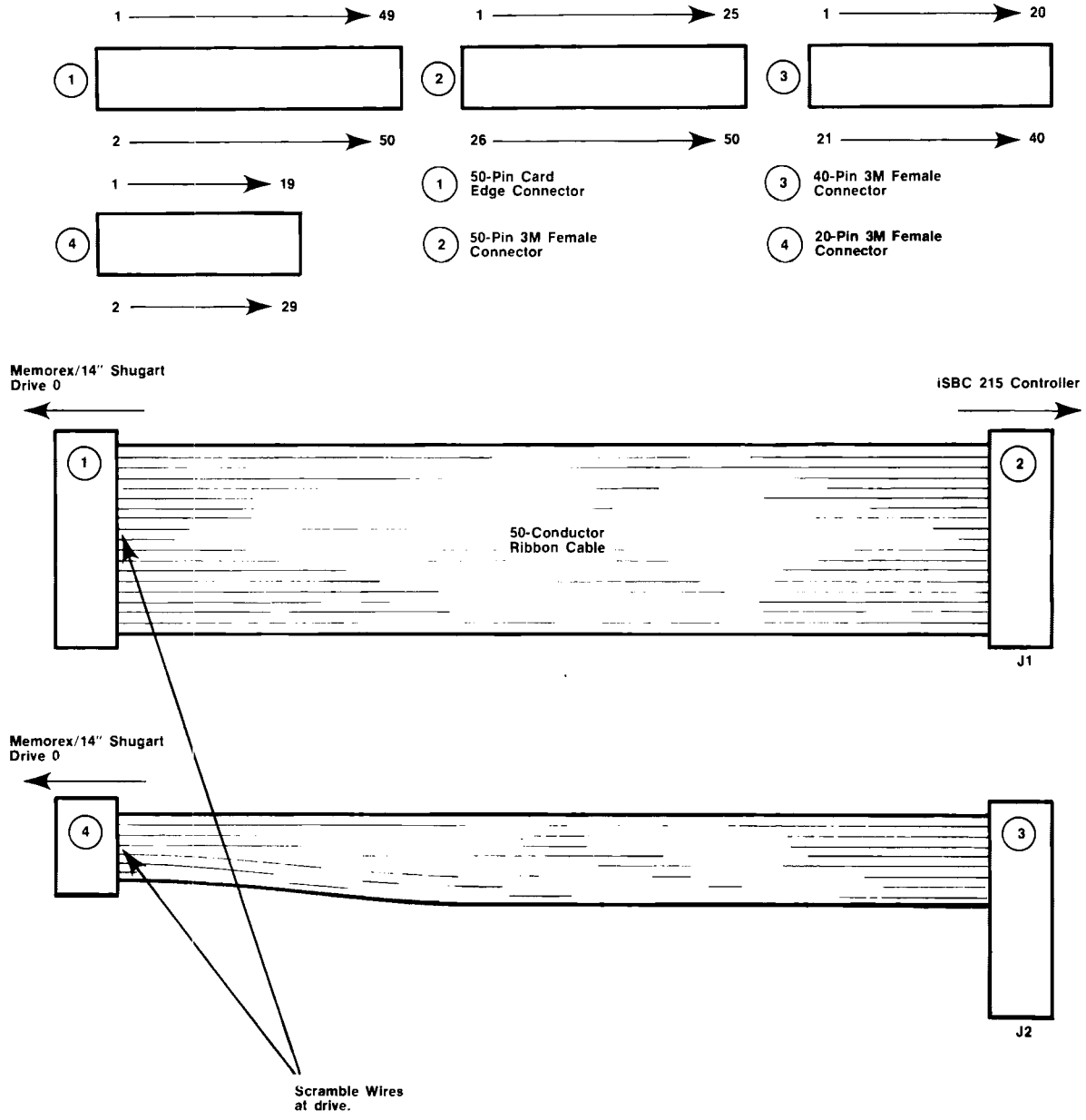
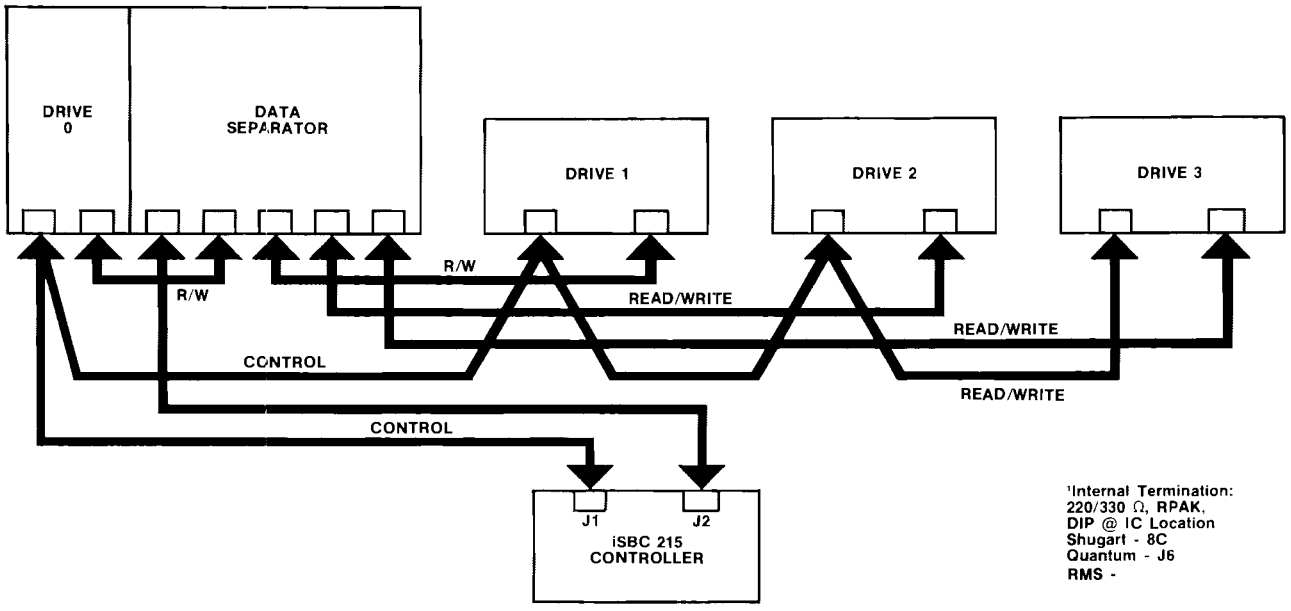


Figure 2-9. Control Data Corporation Drive Interconnecting Cable Requirements

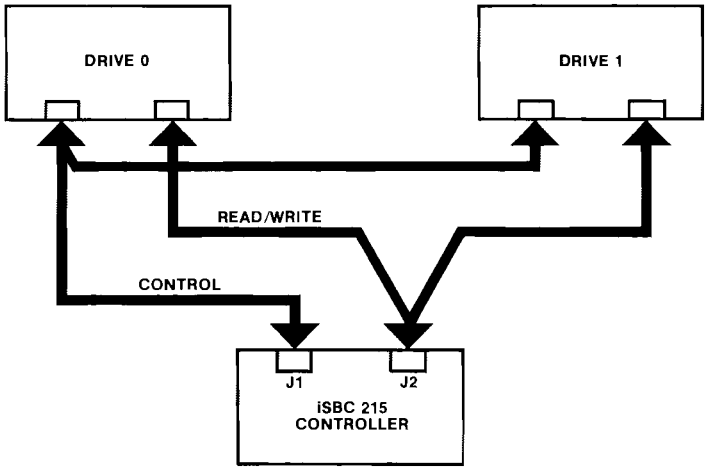


INTERFACE WITH 8" SHUGART/QUANTUM OR 5¼" RMS DRIVES

[1]Internal Termination:
220/330 Ω, RPAK,
DIP @ IC Location
Shugart - 8C
Quantum - J6
RMS -

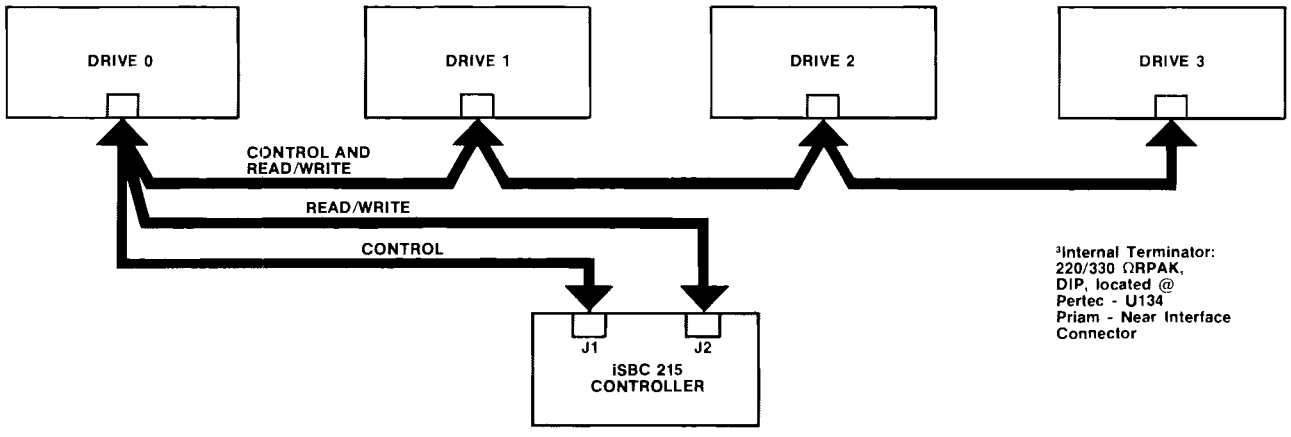
NOTE

Termination locations may change. Consult manufacturer's hardware reference manual for drive.



INTERFACE WITH MEMOREX/14" SHUGART DRIVES

[2]Internal Termination:
220/330 Ω, RPAK,
DIP @ IC Location
Memorex - 18D
Shugart - 3H



INTERFACE WITH PERTEC AND PRIAM DRIVES

[3]Internal Terminator:
220/330 ΩRPAK,
DIP, located @
Pertec - U134
Priam - Near Interface Connector

Figure 2-10. Controller to Drive Interfacing

Table 2-8. J3 and J4 Pin Assignments

Pin	Mnemonic	Description	Pin	Mnemonic	Description
43	MD8	MDATA Bit 8	44	MD9	MDATA Bit 9
41	MDA	MDATA Bit A	42	MDB	MDATA Bit B
39	MDC	MDATA Bit C	40	MDD	MDATA Bit D
37	MDE	MDATA Bit 3	38	MDF	MDATA Bit F
35	GND	Signal Ground	36	+5V	+5 Volts
33	MD0	MDATA Bit 0	34	MDRQT	M DMA Request
31	MD1	MDATA Bit 1	32	MDACK/	M DMA Acknowledge*
29	MD2	MDATA Bit 2	30	OPT0	Option 0
27	MD3	MDATA Bit 3	28	OPT1	Option 1
25	MD4	MDATA Bit 4	26	RDMA	Terminate DMA
23	MD5	MDATA Bit 5	24		Reserved
21	MD6	MDATA Bit 6	22	MCS0/	M Chip Select 0
19	MD7	MDATA Bit 7	20	MCS1/	M Chip Select 1
17	CND	Signal Gnd	18	+5V	+5 Volts
15	IORD/	I/O Read Cmd	16	MWAIT/	M Wait
13	IOWRT/	I/O Write Cmd	14	MINTR0	M Interrupt 0
11	MA0	M Address 0	12	MINTR1	M Interrupt 1
9	MA1	M Address 1	10		Reserved
7	MA2	M Address 2	8	MPST/	iSBX Multimodule Board Present
5	RESET	Reset	6	MCLK	M Clock
3	GND	Signal Gnd	4	+5V	+5 Volts
1	+12V	+12 Volts	2	-12V	-12 Volts

All undefined pins are reserved for future use. *The iSBC 215 does not drive this signal.

2-18. iSBX MULTIMODULE™ INTERFACE

Controller board connectors J3 and J4 have each been designed to interface with Intel iSBX I/O controllers or other I/O modules designed to meet the Intel iSBX Bus Specifications. The Intel iSBX 218 Flexible Disk Controller connects to the J4 connector and provides an interface between the iSBC 215 controller board and up to four 5¼" or 8" double density flexible (floppy) disk drives. The iSBX 218 controller interfaces directly with the iSBC 215 software as described in Chapter 3. Instructions for installing the iSBX 218 controller on iSBC 215 board are given in Paragraph 2-18.

I/O modules that interface the iSBC 215 controller with other storage devices such as magnetic tape cartridge drives or bubble memories can also be designed and connected to J3, J4 or both, (see Table 2-8). The device select function of the iSBC 215 software allows the controller to be interfaced with up to 256 devices through an iSBX connector, J3 and J4. Note that DMA Acknowledge Pin 32 is not connected on the iSBC 215. A more detailed description of the iSBX Bus is given in the *Intel iSBX Bus Specification* manual, Order No. 142686.

The iSBX bus control pins, W3, W4, W11, W12 and W24 (see Table 2-9), control the *External Terminate,*

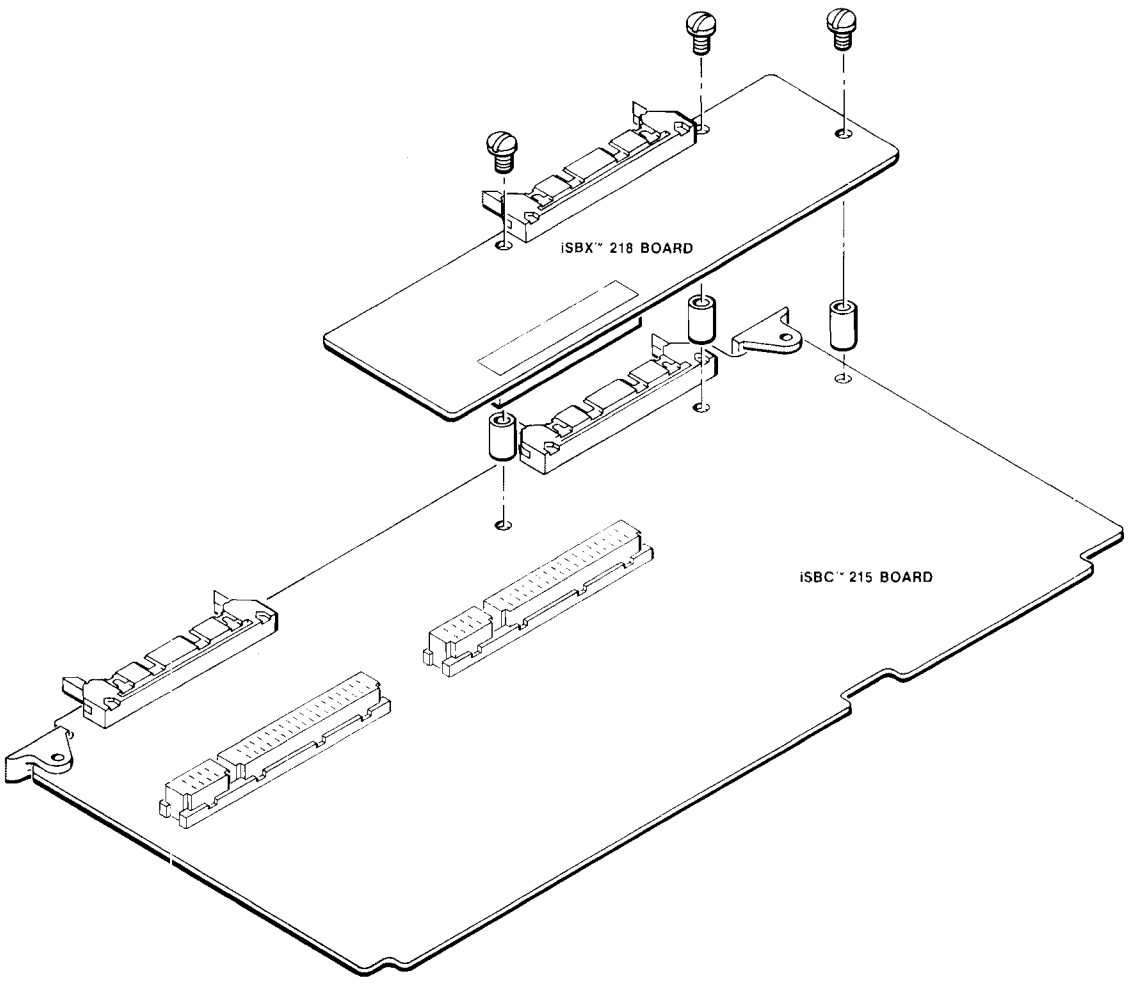


Figure 2-11. Installing the iSBX 218™ Board on the iSBC 215™ Controller Board

and *DMA* request lines on the iSBX bus. (See Figure 5-1 for the location of these pins on the controller board.) The asterisks in Table 2-9 indicate the required jumper configuration for these pins when the iSBX bus is not to be used. Information on the use of these pins for user designed iSBX bus interfaces is given in Paragraph 3-32.

Instructions for writing controller-to-drive interface software for I/O modules designed to the iSBX Bus Specifications are given at the end of Chapter 3.

2-19. iSBX 218™ BOARD INSTALLATION

The iSBX 218 board connects to J4. Six screws and three threaded spacers secure the Multimodule board to the controller board as shown in Figure 2-9. Before installing the iSBX 218 board, install a jumper wire between pins W12-1 and W12-3 and between pins W4-1 and W4-2 on the iSBX 215 board. A single cable that transmits both control and read/write information is required to connect the iSBX 218 controller to the flexible disk drives as shown in Figure 1-2. Refer to the *iSBX 218™ Flexible Disk Controller Hardware Reference Manual*, Intel Order No. 121583, for further installation details and operating information.

Table 2-9. iSBX™ Bus Control Jumper Pins

Pins	Pin Connection	Function
W3	1-2*	External Terminate (J3) terminated on controller board.
	—	External Terminate (J3) driven by iSBX I/O Controller
W4	1-2*	External Terminate (J4) terminated on controller board
	—	External Terminate (J4) driven by iSBX I/O controller
W11	1-2	OP00 (J3) driven
	1-3	OP01 (J4) driven
	—*	OP00 and OP01 receiving
W12	1-2	OP10 (J3) driven
	1-3	OP11 (J4) driven
	—*	OP10 and OP11 receiving
W24	1-2	The iSBX I/O controller on J4 uses DMA request and the iSBX i/O controller on J3 does not use DMA request or is not installed.
	1-3	The iSBX I/O controller on J3 uses DMA request and the iSBX I/O controller on J4 does not use DMA request or is not installed.
	—*	Either both iSBX I/O controllers are not installed or both use the DMA request or neither use the DMA request.

*Required configuration when either the external terminate function or when the iSBX™ Bus is not being used (factory wired).

2-20. POWER UP/DOWN CONSIDERATIONS

If power is applied to, or removed from, the system while a drive is READY, a spurious disk write operation could occur. To prevent this from happening always ensure that the drives are not spinning when system power to the controller is switched on or off.

2-21. DIAGNOSTIC CHECK

A PROM-resident self-diagnostic may be used to verify the controller operation. Instructions for execution of the diagnostic are given in Chapter 3.

CHAPTER 3
PROGRAMMING INFORMATION

3-1. INTRODUCTION

This chapter describes the programming conventions that must be followed to initiate and monitor the transfer of data between the host memory and a disk drive (or the iSBX connector). Included in this section are a discussion of: disk organization, track sectoring format, disk controller communications protocol, interrupt handling, the use of disk control functions, and special instructions for programming I/O transfers through the iSBX interface.

3-2. PROGRAMMING OPTIONS

The iSBC 215 Winchester Disk Controller has been designed to interface with Winchester technology disk drives as specified in Chapters 1 and 2. The board also has two iSBX connectors that allow it to communicate with other I/O devices through an iSBX I/O Controller such as the iSBX 218 Flexible Disk Controller.

The iSBC 215 controller contains a ROM resident I/O transfer program, designed to control data transfers between the controller and Winchester drives as well as between the controller and flexible disk drives connected to the iSBX 218 controller. Paragraphs 3-5 through 3-30 provide instructions for using the iSBC 215 controller firmware.

In addition, the iSBC 215 controller can also execute programs that the user has written in 8089 assembler code to control other I/O devices through the iSBX bus on the board. Instructions for writing and using these programs are provided in Paragraphs 3-31 and 3-32.

3-3. DISK ORGANIZATION

In the following discussion, a head is assumed to be associated with a single disk surface. Each surface can have up to 4096 tracks (circular data paths numbered 0 through 4095). The set of tracks on multiple recording surfaces at a given head position or location is referred to as a "cylinder" (see Figure 3-1). A drive that has 4096 tracks per surface thus has 4096 cylinders.

Each track is divided into equal-sized sectors. Each of these sectors includes a sector identification block with error checking information and a data block, also with error checking information. The iSBC 215 controller allows the user to select the size of the data block; the size of the data block then determines the maximum number of sectors permitted per track (as shown in Table 1-1).

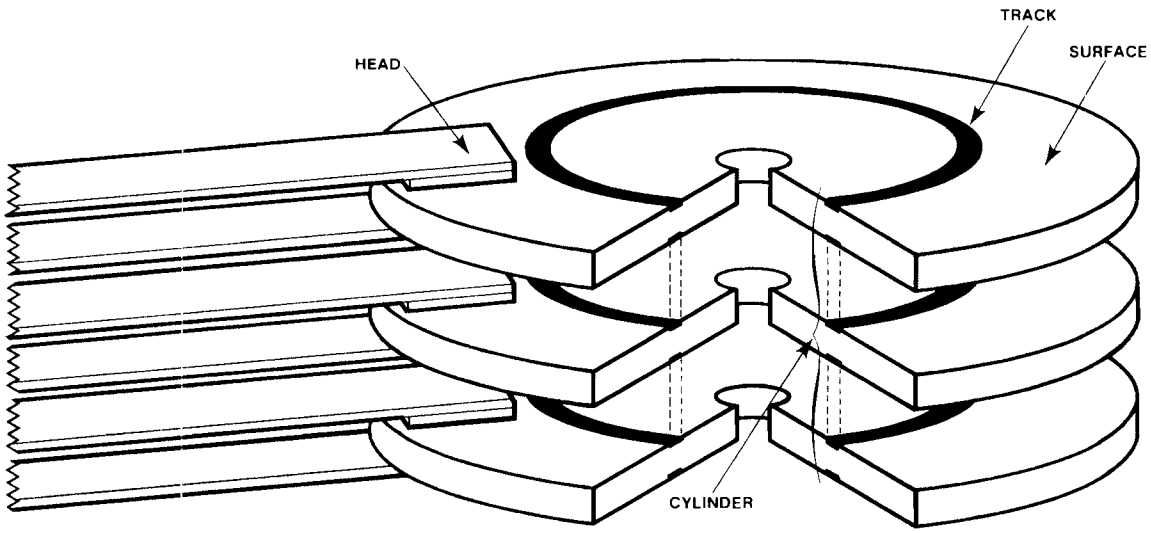


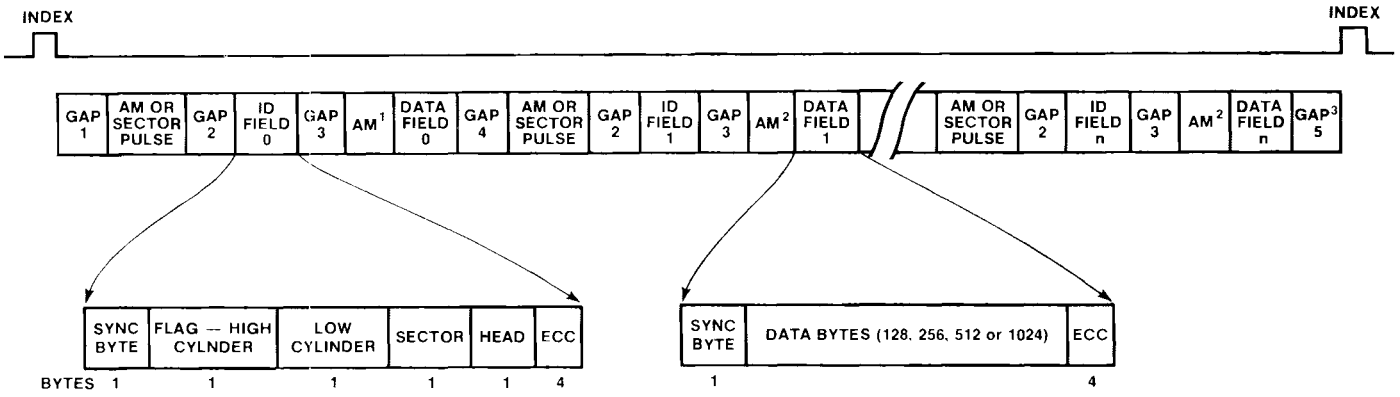
Figure 3-1. Disk Drive Organization and Terminology

3-4. TRACK SECTORING FORMAT

The controller generates the format of the sector identification block, the data block and the error checking fields of each sector of the disk, one track at a time. Figure 3-2 shows how the controller organizes this information for 8″ Winchester drives. Refer to Paragraph 3-14 and 3-15 for further information on track formatting. Refer to the *iSBX 218™ Flexible Disk Controller Hardware Reference Manual* for information on flexible disk track formatting.

3-5. CONTROLLER I/O COMMUNICATIONS BLOCKS

The host processor and the disk controller use four blocks of host memory and one host I/O port to exchange instructions and status. The I/O communications blocks are titled: Wake-Up Block, Channel Control Block, Controller Invocation Block and I/O Parameter Block. Sixty-eight bytes of host memory must be dedicated to the I/O communications blocks.



GAP AND FIELD SIZES IN BYTES				
FIELD	8″ SHUGART/RMS/ QUANTUM	FUJITSU 2300/ MEMOREX/ 14″ SHUGART CDC 9410 - 32	PERTEC	PRIAM
GAP 1	11	0	11	0
ADDRESS MARK OR SECTOR PULSE	1	0[2]	3	0[2]
GAP 2	0	12	14	12
ID FIELD	9	9	9	9
GAP 3	12	14	20	20
ADDRESS MARK (Beginning of Data Field)	1[1]	0	0	0
DATA FIELD Bytes/Sector				
128	133	133	133	133
256	261	261	261	261
512	517	517	517	517
1024	1029	1029	1029	1029
GAP 4	17	8	22	8

[1]8″ Shugart/Quantum drives only.
[2]Sector Pulse
[3]GAP 5 is of indeterminate length. It is residual unused space.

Figure 3-2. Sector Data Format

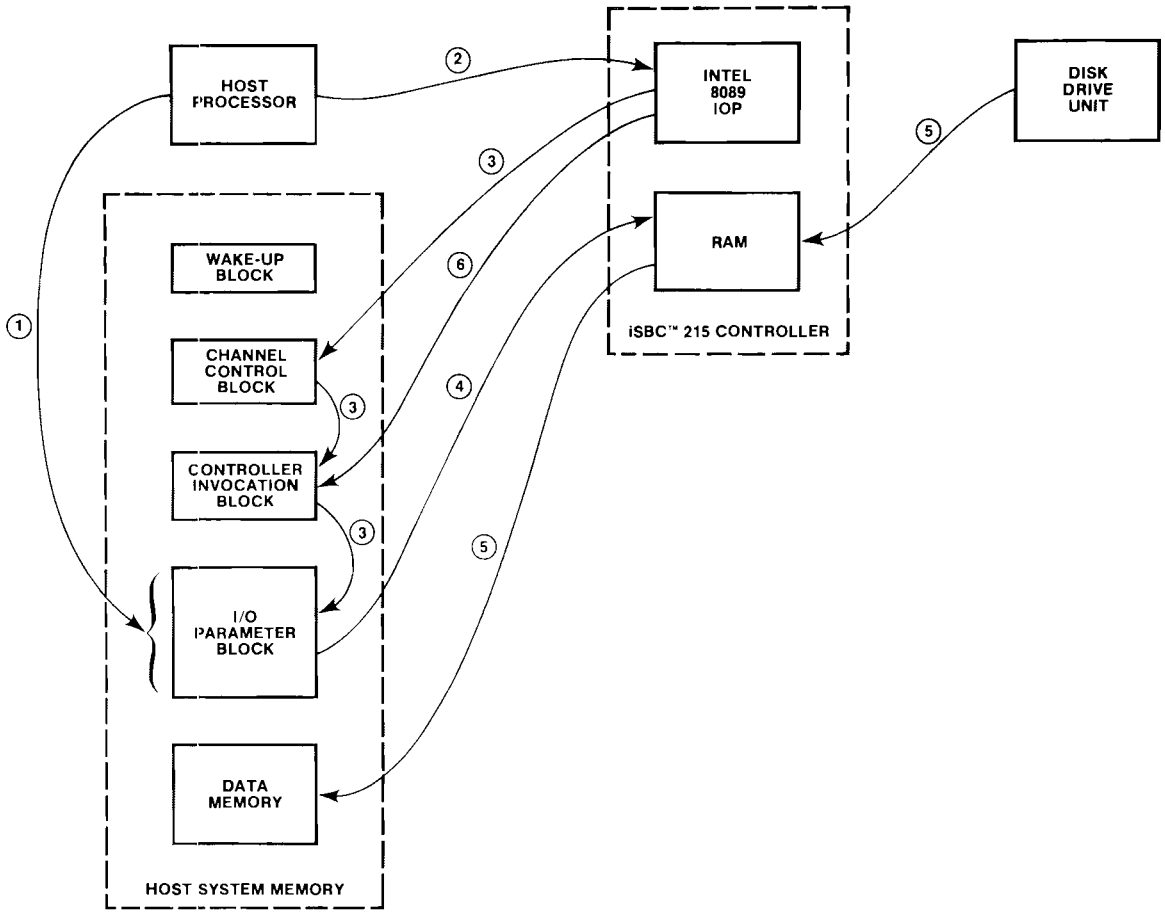


Figure 3-3. Host CPU-Disk Controller Interaction Through the I/O Communications Block

NOTE

Following the initialization of the controller, the Wake-Up Block, Channel Control Block and Controller Invocation Block must be maintained at their assigned locations. The location of the I/O Parameter Block can be changed providing that the I/O Parameter Block Pointer in the Controller Invocation Block is changed to correspond to the new location.

The controller uses these blocks to perform three basic functions: initialize the controller, check and transmit status, and obtain user selected disk access functions and parameters. In addition to these I/O communications blocks, certain controller functions (such as track formatting) also require data/parameter buffers in host memory. Dedicated locations in host memory, however, are not required for these buffers. One I/O port in the host processor's addressable I/O space is also required. The host uses this port, called the Wake-Up I/O Port, to initiate controller activity.

The sequence in which the controller accesses these blocks varies with the type of operation being performed, but for general data transfers (reads or writes), the blocks are accessed as follows:

1. The host loads the I/O Parameter block in system memory with a command and parameters for the function the controller is to perform (for example read data). See Figure 3-3.
2. The host then transmits a wake-up command (01H) to wake-up I/O port, signaling the controller to go to I/O communications blocks for instructions.
3. The controller goes to the Channel Control Block and links its way through the Controller Invocation Block to the I/O Parameter Block. (The Wake-Up Block is used only during controller initialization and by 8089 firmware.)

4. At the I/O Parameter Block, the controller reads the command and parameter data into its RAM and begins the data transfer function.
5. The controller reads data from the selected drive into its RAM, then performs a DMA transfer of the data from RAM into system memory.
6. When the data transfer is complete, the controller posts the status in the Controller Invocation Block, sends an interrupt to the host and awaits further instructions.

These I/O communications blocks are accessed in a similar manner when performing a write function.

A detailed description of these blocks and the data required in each is provided in Paragraphs 3-7 through 3-11. Refer to Paragraphs 2-7 through 2-10 for a discussion of selecting the wake-up address, wake-up I/O port address and 8-bit or 16-bit host.

3-6. HOST CPU-CONTROLLER-DISK DRIVE INTERACTION

Figure 4-2 shows a simplified block diagram of the major hardware sections of the host CPU, host memory, controller and disk drives. The host system memory contains all the controller I/O communications blocks, as well as the data buffers. The host initiates controller activity through the wake-up I/O port, which it addresses through the Multibus interface. The Intel 8089 I/O processor (IOP) handles all communications between the host CPU, host memory and disk drives, once the host has initiated controller activity. Controller operations software is contained in on-board PROM. RAM on the controller board facilitates intermediate data storage between the host and the disk drive. The iSBX bus provides a second I/O transfer path between the controller and an I/O controller such as the iSBX 218 Flexible Disk Controller.

3-7. WAKE-UP I/O PORT

To invoke controller activity, the host CPU transmits a wake-up command byte to the controller through the wake-up I/O port. Three wake-up commands are allowed:

00H	CLEAR INTERRUPT — Controller to host interrupt is reset; controller reset is cleared.
01H	START OPERATION — Instructs controller to start the operation that the elements of the I/O parameter block define.
02H	RESET CONTROLLER — Performs hardware reset of controller. A clear interrupt (00H) must be initiated following this command. (Each time the controller is reset, the communications link between the controller and the host must be re-established through the Initializing function.)
03H through FFH	Reserved.

The sixteen wake-up address switches on the controller board determine the address of the wake-up I/O port as described in Paragraph 2-9.

3-8. WAKE-UP BLOCK

The Wake-Up Block is the first of the I/O communications blocks (see Figure 3-4). It is used to establish a link between the controller and the I/O communications blocks in host system memory.

3-9. CHANNEL CONTROL BLOCK

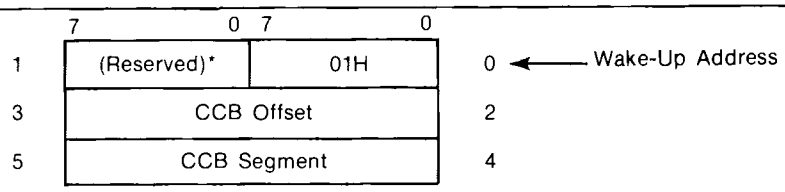
The controller uses the Channel Control Block to indicate the status of the internal processor (the Intel 8089 I/O Processor) and to invoke processor program operations. The Channel Control Block requires 16 bytes (see Figure 3-5). Except for the BUSY 1 flag (byte 1) and the Controller Invocation Block address (bytes 2 through 5), the information contained in this block is used to invoke controller operations that are transparent to the host.

3-10. CONTROLLER INVOCATION BLOCK

The controller uses the Controller Invocation Block (CIB) to post status to the host CPU and to locate the starting address for the controller's on-board disk interface program. The status semaphore byte (byte 3) has a special purpose. The host uses this byte to indicate to the controller whether it has read the current contents of the status byte and is ready for a status update. The Controller Invocation Block requires 16 bytes (see Figure 3-6).

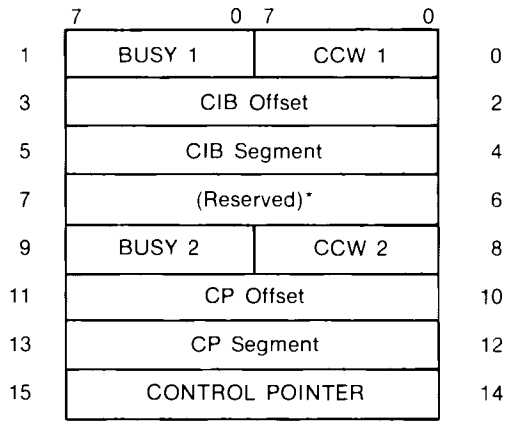
3-11. I/O PARAMETER BLOCK

The I/O Parameter Block (IOPB) contains the controller operating commands, which define the function the controller is to perform (read, write, etc.), and the parameters of the function (memory address, disk head and cylinder, etc.). The I/O Parameter Block requires 30 bytes of host memory space. Figure 3-7 describes the function of each byte.



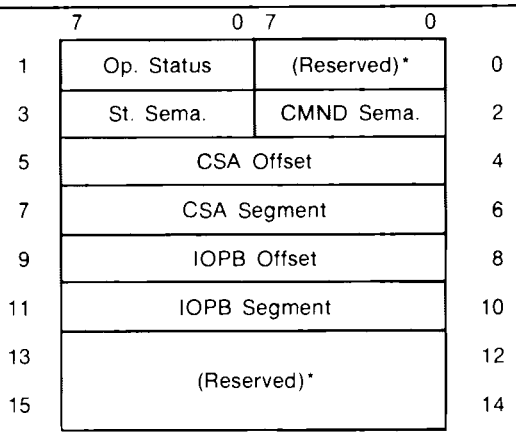
Byte	Function
0	SYSTEM OPERATION COMMAND — Must be set to 01H.
1	Reserved.
2 through 5	CHANNEL CONTROL BLOCK (CCB) ADDRESS — Address of first byte of Channel Control Block. (Address = Offset + Segment X 2^4).

Figure 3-4. Wake-Up Block



Byte	Function
0	CHANNEL CONTROL WORD 1 — Indicates location of Intel 8089 I/O Processor control store program: 01H — Controller local memory (ROM) 03H — Host system memory. (Used only when executing user written I/O program from host memory. (Refer to paragraph 3-32.)
1	BUSY 1 FLAG — Indicates whether controller is busy or idle. 00H — Idle FFH — Busy
2 through 5	CONTROLLER INVOCATION BLOCK (CIB) ADDRESS — Address of fifth byte of Controller Invocation Block.
6 and 7	Reserved.
8	CHANNEL CONTROL WORD 2 — Must contain 01H.
9	BUSY 2 WORD — Not meaningful to host CPU.
10 through 13	CONTROL POINTER ADDRESS — Address must point to the Control Pointer in the next sequential word.
14 and 15	CONTROL POINTER — Must be set to 0004H.

Figure 3-5. Channel Control Block



* Set to all zeros.

Byte	Function
0	Reserved.
1	CONTROLLER OPERATION STATUS — Bits 0 through 2 indicate event completed. Bit 3 indicates the device that completed the event. Bits 4 and 5 indicate drive associated with event. Bit 6 indicates error: soft (recoverable), 0, or hard, 1. Bit 7 indicates a summary error that can be checked through the transfer error status function (refer to paragraph 3-28).
2	COMMAND SEMAPHORE — Controller does not use this byte. It is provided for use as a multiprocessor interlock.
3	STATUS SEMAPHORE — Controller posts status only when this byte is 00H; when new status has been posted, controller sets byte to FFH. When host CPU has read status, it sets this byte to 00H.
4 through 7	CONTROL STORE PROGRAM ADDRESS — Starting address of controllers on-board disk interface program. Set to 0000H.
12 through 15 Reserved	I/O PARAMETER BLOCK ADDRESS — Address of first byte of I/O parameter block.

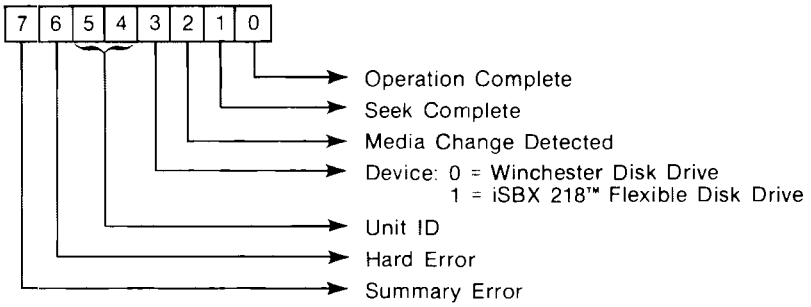


Figure 3-6. Controller Invocation Block

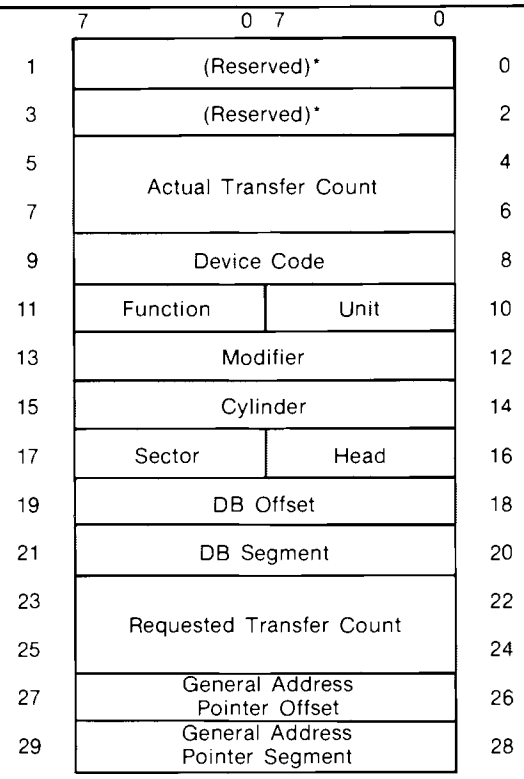
3-12. TYPICAL CONTROLLER OPERATIONS

The following section describes how to set up the I/O communications blocks in the host memory, how to initialize the controller and how to perform the various data transfer operations. It is assumed that the controller board has been properly installed as described in Chapter 2.

3-13. INITIALIZING THE CONTROLLER

The controller must be initialized before any data transfer activities between the host system memory and the disk drives can be initiated. Initialization of the controller involves:

1. Establishing a link between the 8089 and the I/O communications blocks in host system memory.



* Set to all zeros.

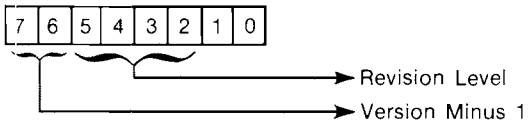
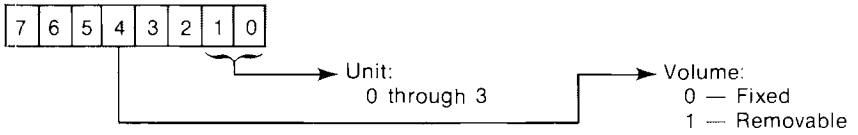
Byte	Name and Function
0 through 3	Reserved.
4 through 7	ACTUAL TRANSFER COUNT — Count of bytes actually transferred between the system and the disk or controller. Four-byte binary number, least significant bits in first byte. Controller writes count to IOPB following termination or completion of an operation. If count does not match requested transfer count, operation was prematurely terminated; check status. When performing the track formatting function, a count of 6 is set in the Actual Transfer Count word. When performing the status transfer function, the count is set to 12. When initializing drive 0, this word has a special function. It is used to dis-- play the controller firmware's version number as shown below: 
8 and 9	DEVICE CODE — Code for type of device being accessed. 0000H — 8" Winchester Drive 0001H — iSBX 218 8" Flexible Disk 0003H — iSBX 218 5¼" Flexible Disk
10	UNIT — Code for drive unit being accessed: bits 0 and 1 address unit code; bits 2 through 7 are reserved. 
11	FUNCTION — Code for operation to be performed. Refer to following discussion of typical controller operations for a detailed discussion of these operations: 00H INITIALIZE 01H TRANSFER STATUS 02H FORMAT

Figure 3-7. I/O Parameter Block Description

	03H	READ SECTOR ID
	04H	READ DATA
	05H	READ TO BUFFER AND VERIFY
	06H	WRITE DATA
	07H	WRITE BUFFER DATA
	08H	INITIATE TRACK SEEK
	09H - 0BH	Reserved
	0CH	iSBX EXECUTE
	0DH	iSBX TRANSFER
	0EH	BUFFER I/O
	0FH	DIAGNOSTIC
12 and 13	MODIFIER — Code to modify function codes.	
	Bit 0	Suppresses interrupt on command completion when set to 1.
	Bit 1	Automatic retries for error recovery are inhibited when set to 1.
	Bit 2	Allows READ DATA, READ TO BUFFER AND VERIFY, WRITE DATA and WRITE BUFFER DATA functions to be modified to read or write deleted data, respectively, through the iSBX 218™ I/O controller: 0 = Normal Data; 1 = Deleted Data.
	Bits 3 through 15	Reserved.
14 and 15	CYLINDER — Binary number specifying logical cylinder code; bit 0 is least significant bit of number.	
16	HEAD — Binary number specifying logical head code; bit 0 is least significant bit of number.	
17	SECTOR — Binary number specifying logical sector code; bit 0 is least significant bit of number.	
18 through 21	DATA BUFFER ADDRESS — Address of first byte in host system memory data (parameter) buffer.	
22 through 25	REQUESTED TRANSFER COUNT — Count of bytes requested to be transferred between the system and the disk or controller. Four-byte binary number, least significant bits in first byte. See description of ACTUAL TRANSFER COUNT, bytes 4 through 7 in IOPB.	
26 through 29	GENERAL ADDRESS POINTER — General purpose address pointer.	

Figure 3-7. I/O Parameter Block Description (Continued)

2. Reading the parameters that describe the disk drives with which the controller is to interface into the controller's RAM buffer, using the Initialize function (FUNCTION = 00H).

This initialization must be performed following a:

1. Power-on event.
2. Controller reset (02H written to the wake-up I/O port).

After the controller has been initialized, any of the data transfer functions described in Paragraphs 3-14 through 3-25 can be performed in any sequence. (Refer to Paragraphs 4-12 through 4-15 for a detailed explanation of controller initialization.)

The following procedure gives the sequence in which the controller initializing activities must be performed. Prior to initializing the controller, check that the system data bus switch (S2-1), the host system I/O address switch (S2-2), the wake-up address switches (S1-1 through S1-8 and S2-3 through S2-10), and the interrupt level jumper have been set as described in the procedure titled Switch/Jumper Configurations in Chapter 2.

NOTE

When the system is first powered-on, the Pertec or Priam drives will not spin until each has received an initialize command. For each drive, the initialize command thus cannot be completed until the drive has reached its operating speed and entered the ready state. This spin-up time varies from approximately 20 seconds for the Priam drives to 90 seconds for the Pertec drives.

The Shugart and Memorex drives spin-up as soon as power is applied. If an initialize command is issued to a unit that has not yet reached operating speed, a not ready error is posted.

To initialize the controller, the host CPU must perform the following steps:

1. **Establish addresses for the four I/O communications blocks in host memory:**

Wake-Up Block	6 Bytes
Channel Control Block	16 Bytes
Controller Invocation Block	16 Bytes
I/O Parameter Block	30 Bytes

Remember that the address of the first byte of the Wake-Up Block must be equal to the wake-up

address set in the controller's wake-up address switches times 2^4. For example, if the switches are set to 0673H, the address of byte 0 of the Wake-Up Block is:

06730H	20-Bit Addressing
6730H	16-Bit Addressing

2. **Set up the shaded bytes in the Wake-Up Block (see Figure 3-8).**
3. **Set BUSY 1 flag (Optional).** Set the BUSY 1 flag (byte 1 of the Channel Control Block) to non-zero (FFH). This allows the host to monitor the BUSY 1 flag to find out when the initialization procedure is complete.
4. **Reset the controller.** Host writes a 02H to the wake-up I/O port.
5. **Clear the reset.** Host writes a 00H to the wake-up I/O port.
6. **Establish the host-controller communications link.** Write a 01H to the wake-up I/O port. The controller goes to the Wake-Up Block in host memory and records the address of the Channel Control Block, then goes to the Channel Control Block and clears the BUSY 1 FLAG. On all subsequent 01H commands to the wake-up I/O port, the controller will go to the Channel Control Block.
7. **Set up the shaded bytes in the Channel Control Block as shown in Figure 3-8.**
8. **Set up the shaded bytes in the Controller Invocation block as shown in Figure 3-8.** Be sure the STATUS SEMAPHORE, byte 3, is set to 00H.
9. **Set up the shaded bytes in the I/O Parameter Block as shown in Figure 3-8.** Be sure the UNIT, byte 10, is set for the correct unit number and the FUNCTION, byte 11, is set for the Initialize function (FUNCTION = 00H). Initialize unit 0 first.
10. **Establish parameter buffer.** Set up a disk drive parameter data buffer with the parameters for the drive to be initialized as shown in Figure 3-8. Be sure the data buffer address in the I/O Parameter Block points to the first address of this data buffer.
11. **Start initialize function.** Poll the BUSY 1 flag (Byte 1 of the CCB) and write a 01H to the wake-up I/O port when the flag is zero. The controller goes to the Channel Control Block, then links its way through the Controller Invocation Block and I/O Parameter Block and reads the disk drive parameters for the unit specified.
12. **Respond to and process the resulting interrupt or status or both.**
13. **Reset I/O Parameter Block.** Set the UNIT, byte 10, for the next unit to be initialized and set the data buffer address, byte 18 through 21, for the beginning address of the unit's disk parameters.
14. **Repeat steps 9 through 12 for each drive unit.** Note that the initialization procedure *MUST BE PERFORMED FOR ALL FOUR DRIVE UNITS*, starting with unit 0, even if one or more of the drives do not exist. Initialize all unattached drives with all zeros.
15. **Initialize flexible disk drive units.** If an iSBX 218 controller is installed on the iSBX 215 controller board, repeat steps 9 through 14 for all four flexible disk drive units.

NOTE

The Winchester disk drive units must be initialized before initializing the flexible disk drive units.

The controller is now initialized. This procedure need not be repeated except after a power-on or a controller reset. For all subsequent disk activities, the host communicates with the controller through the Channel Control Block, the Controller Invocation Block and the I/O Parameter Block.

3-14. TRACK FORMATTING

The Format Track function (FUNCTION = 02H) writes the gaps, sector headers and data fields (see Figure 3-2) on a track — one track per command. A track can be designated as a normal, assigned alternate or defective track. A defective track always points to an assigned alternate track. Refer to the discussion of alternate and defective track handling in Paragraph 3-15.

Use the following procedure to format a track.

1. **Set up the I/O Parameter Block as shown in Figure 3-9.**
2. **Set up a 6-byte data buffer for the type of track to be formatted as shown in Figure 3-9.** A track can be designated as a data track, assigned alternate track or defective track. The user pattern is repeated throughout the data field of every sector. In the case of a defective track, the user pattern is a pointer to the alternate track. If the alternate track is defective, it can not be used to point to another alternate. An interleave factor of 1 corresponds to consecutive sectors.

3. **Initiate the format operation.** Write a 01H to the wake-up I/O port.

4. **Respond to and process the resulting interrupt or status or both.**

NOTE

Always format the last track on head 0 as a data track. This track should then be reserved for use by the on-board diagnostic.

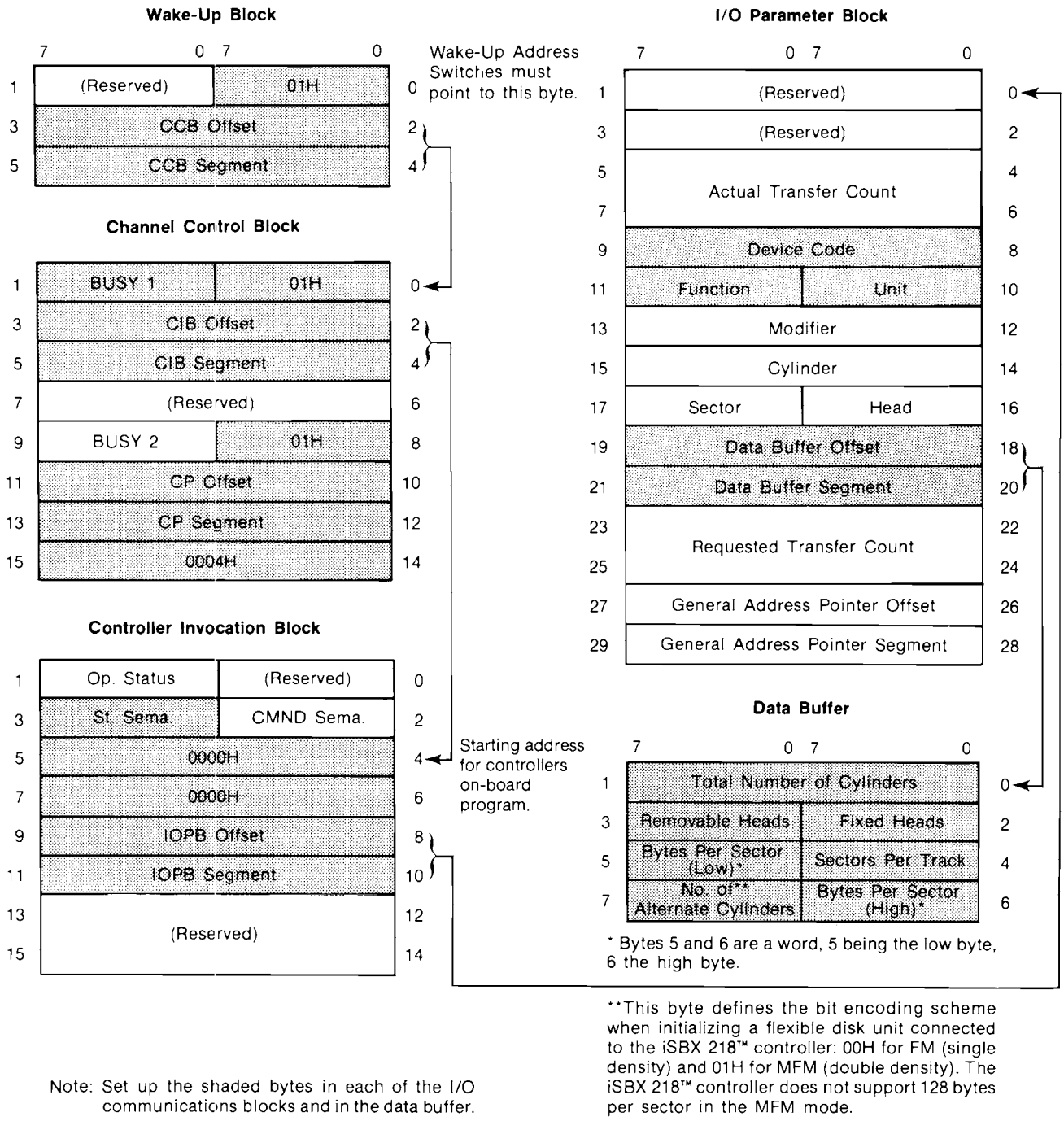


Figure 3-8. I/O Communications Blocks Linking

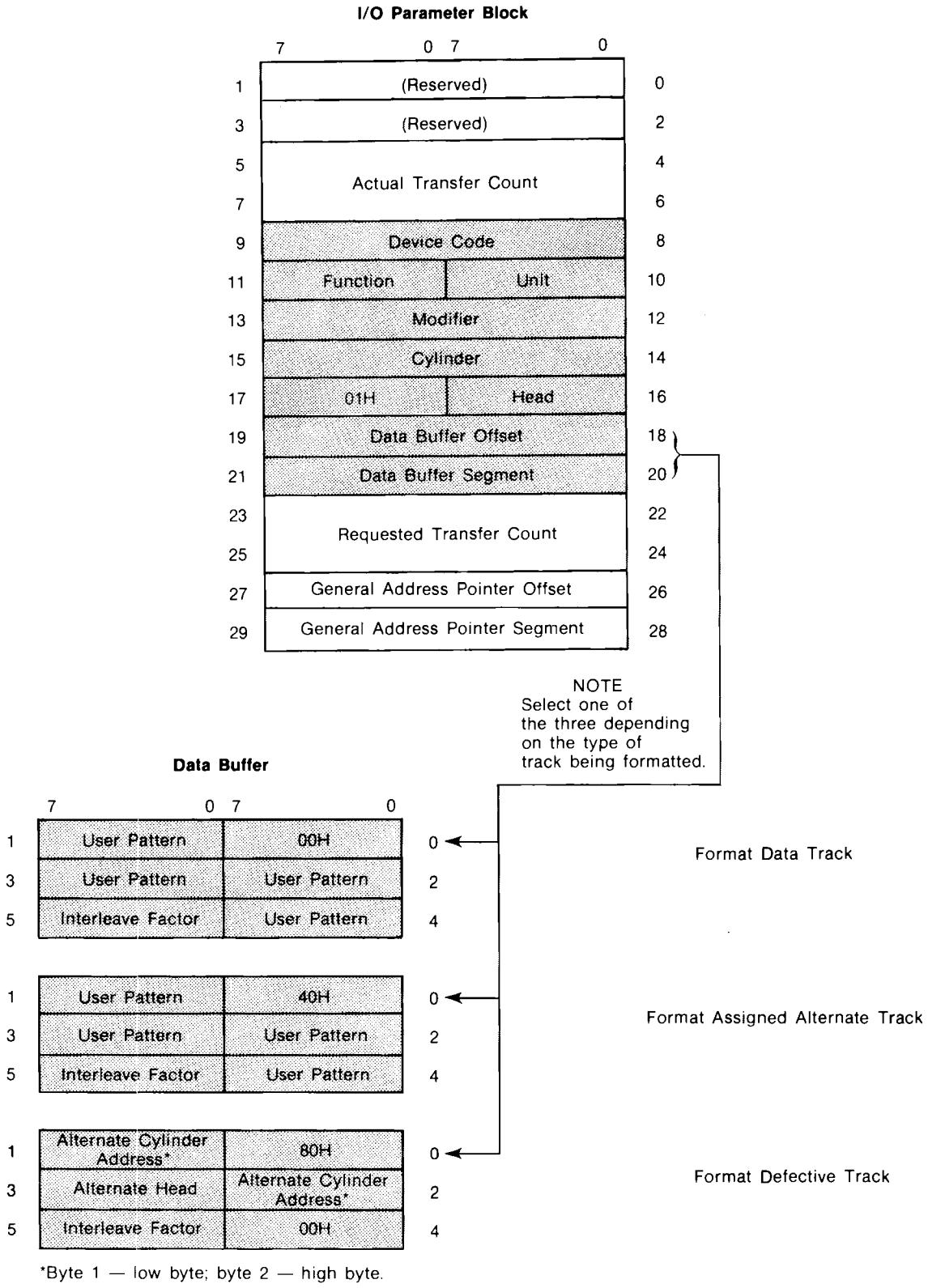


Figure 3-9. Track Formatting

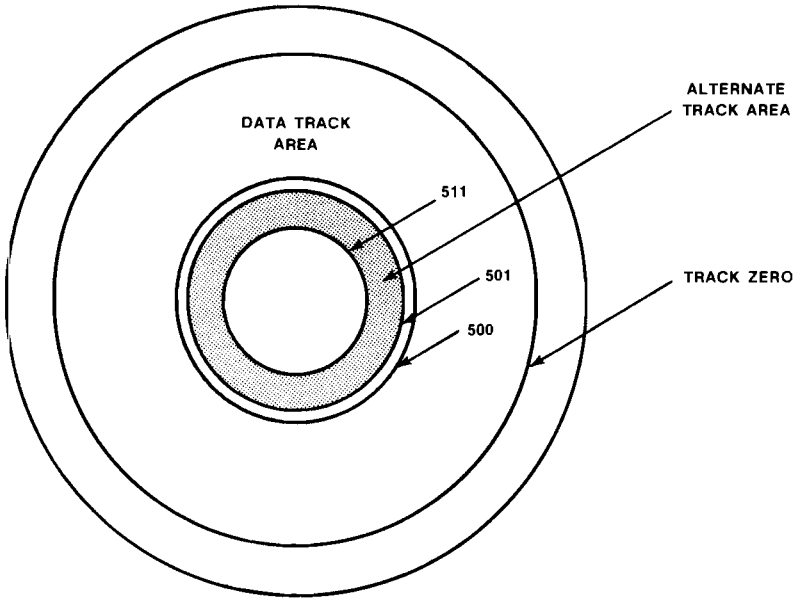


Figure 3-10. Alternate Track Formatting

3-15. ALTERNATE AND DEFECTIVE TRACK HANDLING

It is suggested that each disk surface be divided into two areas (see Figure 3-10), the data track area and the alternate track area. The user assigns the number of tracks in the alternate track area, typically 1 - 2% of the total number of available tracks on the surface. If a disk surface has 512 tracks, tracks 0 through 500 would constitute the data track area and tracks 501 through 510 would constitute the alternate track area. **The last track at Head 0 must be reserved for the diagnostic program.**

When a track within the data track area is deemed defective, the host reformats the track, giving it a defective track code and entering the address of the next available alternate track in the data fields. The alternate track that is selected must be formatted as an assigned alternate track.

When the controller accesses a track that has been previously marked defective, it will automatically invoke a seek to the assigned alternate track and use the alternate as if it were in the data track area. This operation is automatic and is invisible to the user, except for the added time required to complete the operation.

3-16. DATA TRANSFER AND VERIFICATION

Nine data transfer and verification command functions are allowed, selected through the FUNCTION byte in the I/O Parameter Block: Read Sector ID, Read Data, Read Data to Buffer and Verify, Write Data, Write Data from Buffer, Initiate Track Seek, Execute iSBX I/O Program, I/O Transfer through iSBX Bus, and Buffer I/O.

NOTE

All data transfers between the host system memory and a disk drive unit are buffered through the controller's on-board RAM buffer. During a write, the controller performs a DMA transfer of a one-sector block of data from the host system memory to the RAM buffer. It then transfers the sector serially from the RAM buffer to the disk in two byte increments. When reading from the disk, the controller performs a serial transfer of a sector of data from the disk to the RAM buffer in two byte increments. When the entire sector has been read into the RAM and all error checking has been completed, the controller then performs a DMA transfer of the one-sector block from the RAM to host system memory.

The controller contains a burst error checking code (ECC) computing circuit that creates an error checking code for each sector ID and each data block written into disk memory. When reading data from the disk, the controller verifies the sector ID and the information in the data blocks using these error checking codes. If errors are detected that can be corrected (occur within an eleven-bit burst or less), they are corrected and the remainder of the operation is completed. If the error cannot be corrected, the sector is re-read. If after 3 retries the errors remain uncorrectable, the operation is terminated and a Hard Error is indicated in the operation status byte (byte 1) of the Controller Invocation Block. To obtain detailed information on the nature of the error, perform the Transfer Error Status function (refer to Paragraph 3-28).

Each of the data transfer and verification functions is described in detail in the following paragraphs. To use any one of these functions, the host CPU must perform the following steps:

1. **Set up the I/O parameter block as shown in the paragraph describing the function.**
2. **Initiate the operation.** Write a 01H to the wake-up I/O port.
3. **Respond to and process the resulting interrupt or status or both.**

3-17. READ SECTOR ID

The Read Sector ID function (FUNCTION = 03H) searches for the first error free sector ID on the selected track and writes the contents of the sector ID field into a 5-byte data buffer in host memory (see Figure 3-11). An implied seek, head select or volume change, *is not performed.* The Read Sector ID is performed on the cylinder, volume and head that the previous function selected. One use of this function is to search the alternate track area for tracks that have not been assigned as alternates.

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-11, and reserve a 5-byte data buffer in host system memory.

3-18. READ DATA

The Read Data function (FUNCTION = 04H) reads data from the disk into host system memory. It begins reading with the first byte of the selected

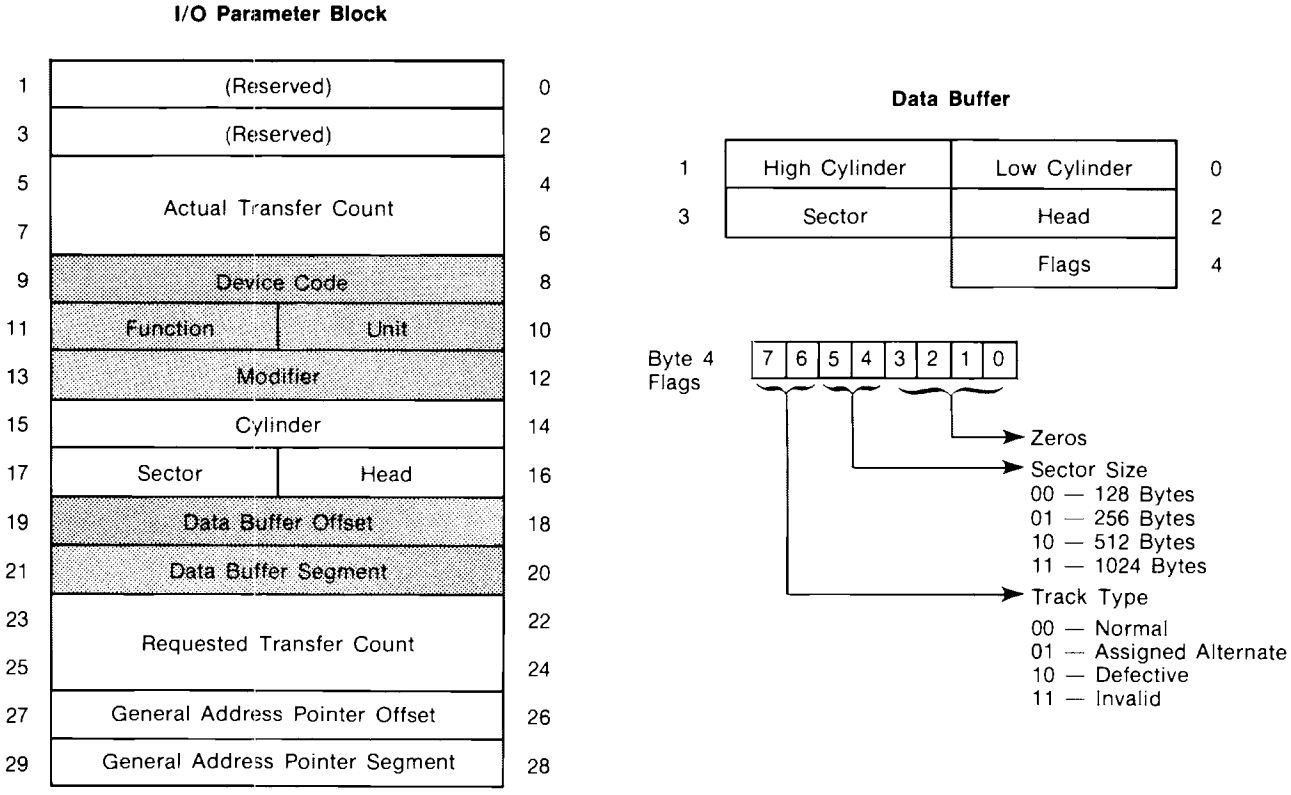


Figure 3-11. Read Sector ID

sector and ends reading when the requested byte count is reached, end of media is reached or a hard failure is detected. If multi-sector data transfers are requested the controller automatically seeks to the next sector, the next head and the next cylinder, in that order. Automatic head increments are supported only within the volume, fixed or removable, but not between volumes, for example, fixed across to removable. The last sector, head and track address in the data track area defines the end of media. An implied seek is invoked if the current head position is different from the specified track identification. The DATA BUFFER address set in the I/O parameter block is the address in host system memory where the first data byte read from the disk is to be transferred. Since the data being transmitted from the disk drive is buffered in the controller's RAM, data overruns cannot occur. To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-12.

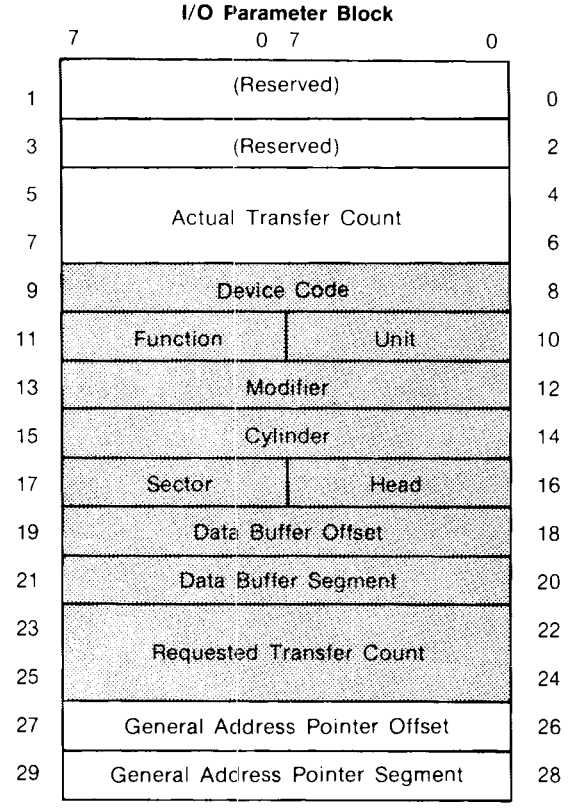


Figure 3-12. Read Data

3-19. READ DATA INTO CONTROLLER BUFFER AND VERIFY

The Read Data into Controller Buffer and Verify function (FUNCTION = 05H) reads data from the disk into the controller on-board RAM and checks the ECCs to verify the sector ID and data fields for all sectors affected. It begins reading with the first byte of the selected sector and ends reading when the requested byte count is reached, end of media is reached or a hard failure is detected. The multi-sector data verification is supported through the auto-sector, auto-head, auto-cylinder protocol described for Read Data function. End of media and implied seek are also supported as described for the Read Data functions.

The Read Data into Controller Buffer and Verify function has three applications:

1. Allows data to be verified after it has been written from host system memory to the disk.
2. Allows data to be transferred from one disk location to another by coupling this function with the Write Data from Controller Buffer function.
3. Allows data to be transferred from an Winchester disk to a device connected to the iSBX bus. To perform this operation, the Read to Buffer and Verify command is coupled with either the iSBX Execute command or the Write Buffer Data command (iSBX 218 controller is specified to receive the data).

To perform the Read Data into Controller Buffer and Verify function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-13.

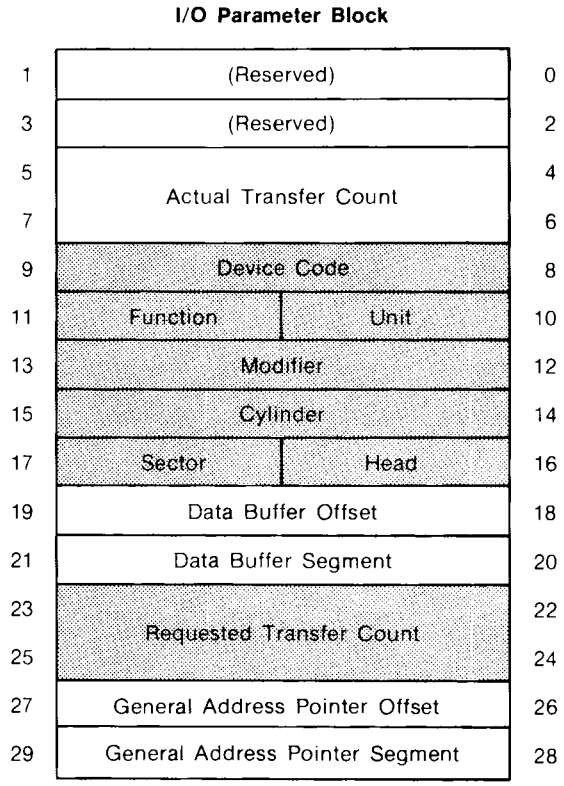


Figure 3-13. Read Data into Controller Buffer and Verify

3-20. WRITE DATA

The Write Data function (FUNCTION = 06H) writes data from host system memory onto the disk. It begins reading from the specified host data buffer address and writes to the first byte of the selected sector. It ends writing when the requested byte count is reached, end of media occurs or a hard failure is detected. When writing to more than one sector, the sector selection is automatic as described for the Read Data function. Auto-head increments and implied seek are also supported as described for the Read Data function. If writing ends in the midst of a sector, the remaining area of the sector is filled with zeros.

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-14.

I/O Parameter Block

1	(Reserved)	0
3	(Reserved)	2
5	Actual Transfer Count	4
7		6
9	Device Code	8
11	Function / Unit	10
13	Modifier	12
15	Cylinder	14
17	Sector / Head	16
19	Data Buffer Offset	18
21	Data Buffer Segment	20
23	Requested Transfer Count	22
25		24
27	General Address Pointer Offset	26
29	General Address Pointer Segment	28

Figure 3-14. Write Data

3-21. WRITE DATA FROM CONTROLLER BUFFER TO DISK

The Write Data from Controller Buffer to Disk (FUNCTION = 07H) writes data from the controller on-board RAM onto the disk. It begins reading from the first address of the controller's data buffer (4010H) and writes to the first byte of the selected disk sector. It ends writing when the requested byte count is reached, end of media occurs or a hard failure is detected. When writing to more than one sector, the sector selection is automatic as described for the Read Data function and the data in the buffer is repeated for each sector written. Auto-head increments, implied seek and end of media are also supported as is described for the Read Data function. If writing ends in the midst of a sector, the remaining area of the sector is filled with zeros.

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-15.

I/O Parameter Block

1	(Reserved)	0
3	(Reserved)	2
5	Actual Transfer Count	4
7		6
9	Device Code	8
11	Function / Unit	10
13	Modifier	12
15	Cylinder	14
17	Sector / Head	16
19	Data Buffer Offset	18
21	Data Buffer Segment	20
23	Requested Transfer Count	22
25		24
27	General Address Pointer Offset	26
29	General Address Pointer Segment	28

Figure 3-15. Write Data From Controller Buffer to Disk

3-22. INITIATE TRACK SEEK

The Initiate Track Seek function (FUNCTION = 08H) positions the read/write head on a specified track, if the head is not already on that track. When issued sequentially to several drives, this command allows multiple disk drives to perform concurrent (overlapping) seeks. If a seek to a cylinder beyond the end of media, including alternates, is initiated, the drive automatically performs a rezero operation

and posts invalid address error. If an operation complete interrupt is enabled, it is invoked when the seek command has been initiated and a seek complete interrupt (which is always enabled) is invoked when the seek is completed. The operation complete interrupt allows a function to be initiated on a second drive while the seek is being performed on the first drive.

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-16.

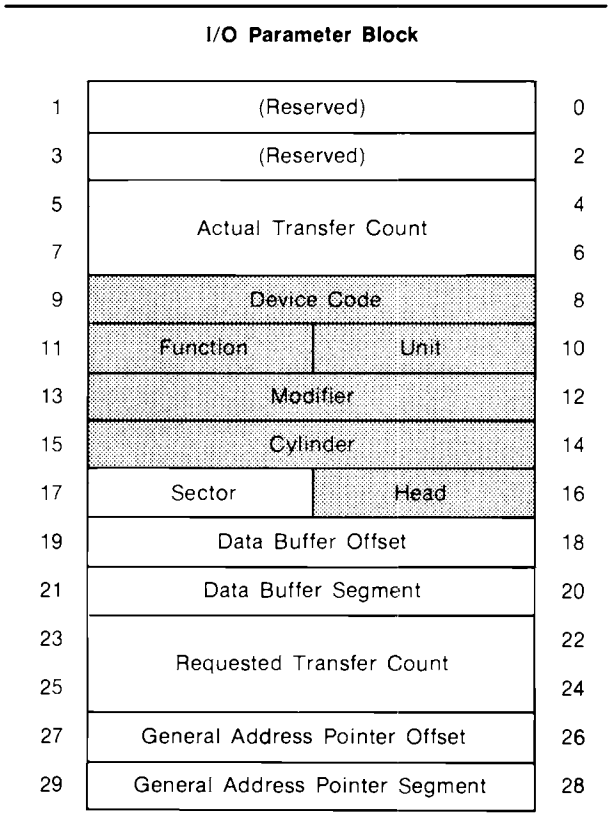


Figure 3-16. Initiate Track Seek

3-23. EXECUTE iSBX™ I/O PROGRAM

The Execute iSBX I/O Program function (FUNCTION = 0CH) transfers program control to a program stored in the controller on-board RAM memory. This program must be coded in 8089 assembler code. It is loaded into RAM using the Buffer I/O function (FUNCTION I/O = 0EH). Program control is transferred to the RAM address specified in the General Address Pointer, bytes 26 through 29 in the I/O parameter block. Upon completion of the program, the program must exit to ROM location 00C5H. The programs, which this function activates, are written to perform I/O transfers to peripheral devices through the iSBX bus (refer to Paragraphs 3-31 and 3-32 for more information concerning the use of this function).

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-17. The outlined bytes are optional. Their use depends on the requirements of the user written I/O program.

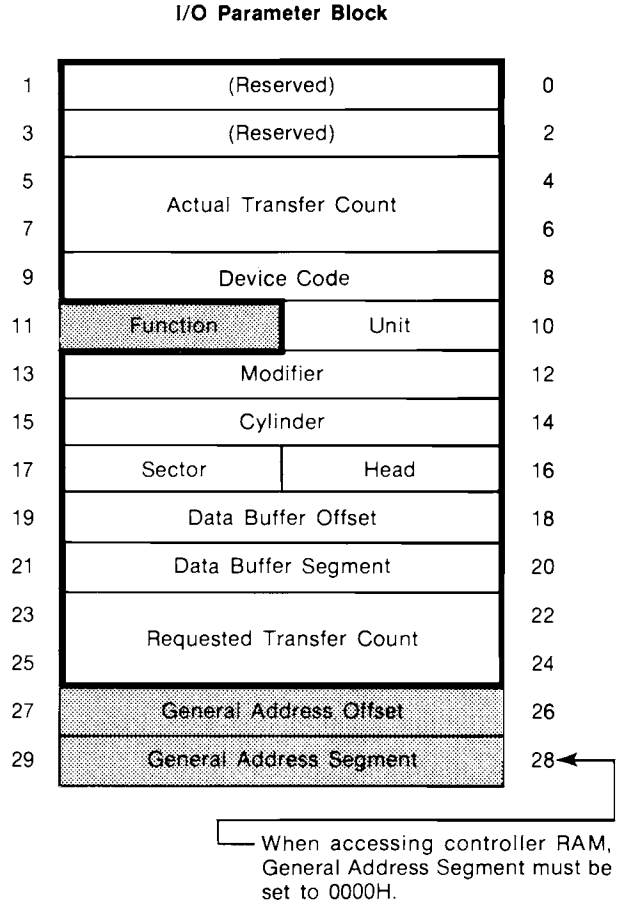


Figure 3-17. Execute iSBX™ Interface I/O Program

3-24. I/O TRANSFER THROUGH iSBX™ BUS

The I/O Transfer Through iSBX Bus function (FUNCTION = 0DH) transfers a block of data between host system memory and the iSBX bus ports. The beginning address in host system memory and the number of bytes to be transferred is specified in the respective locations in the I/O parameter block. The iSBX bus port address, width of the port (8 bit or 16 bit), direction of transfer and mode of transfer are specified in the cylinder and head locations of the I/O parameter block (Refer to Paragraphs 3-31 through 3-32 for more information concerning the use of this function.)

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-18.

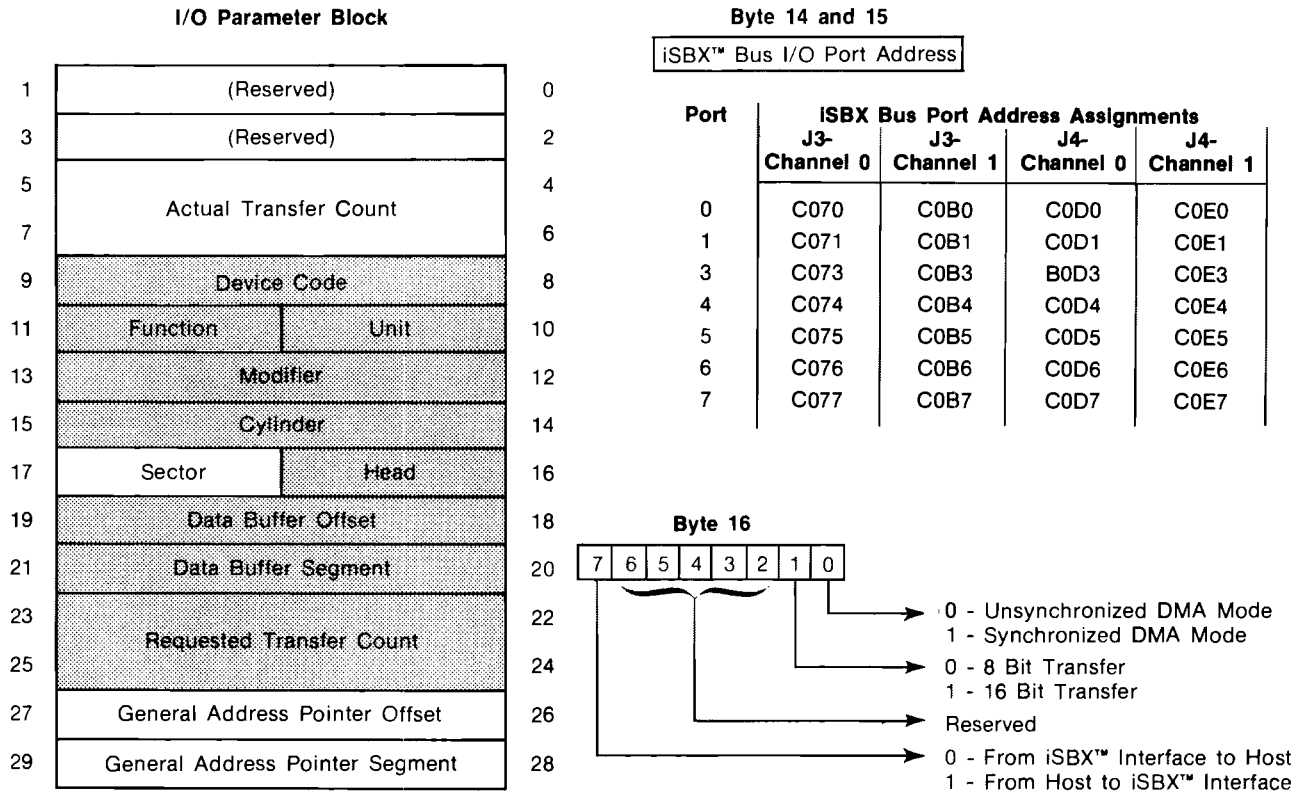


Figure 3-18. I/O Transfers Through iSBX™ Interface

3-25. BUFFER I/O

The Buffer I/O function (FUNCTION = 0EH) transfers data between the host system memory and controller on-board RAM. Beginning addresses in the host system memory and controller buffer memory are specified. Data transfer begins at these addresses and ends when the requested byte count is reached. Since the controller has only 64K bytes of local memory address space, the most significant bytes of the REQUESTED TRANSFER COUNT (bytes 24 and 25) are ignored.

CAUTION

Data transfers from the host system memory to the controller-buffer must be written to addresses within the range of 4000H to 4600H.

The beginning address in controller memory and the direction of data transfer are specified in the CYLINDER and HEAD fields, respectively:

Bytes 14 and 15	Starting controller memory address:
Bytes 14 and 15	Starting controller memory address: Byte 15 — High Byte Byte 14 — Low Byte
Byte 16	Direction of data transfer: 00H — From controller to host FFH — From host to controller

The Buffer I/O function has three applications. Its primary purpose is for use with the diagnostic program. It also allows memory-to-memory transfers with a minimum of host overhead. In addition, it allows down-loading of user written, I/O transfer control programs from system memory to controller memory. Such programs allow 8089 control of I/O transfers through the iSBX bus as discussed in Paragraph 3-23.

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-19.

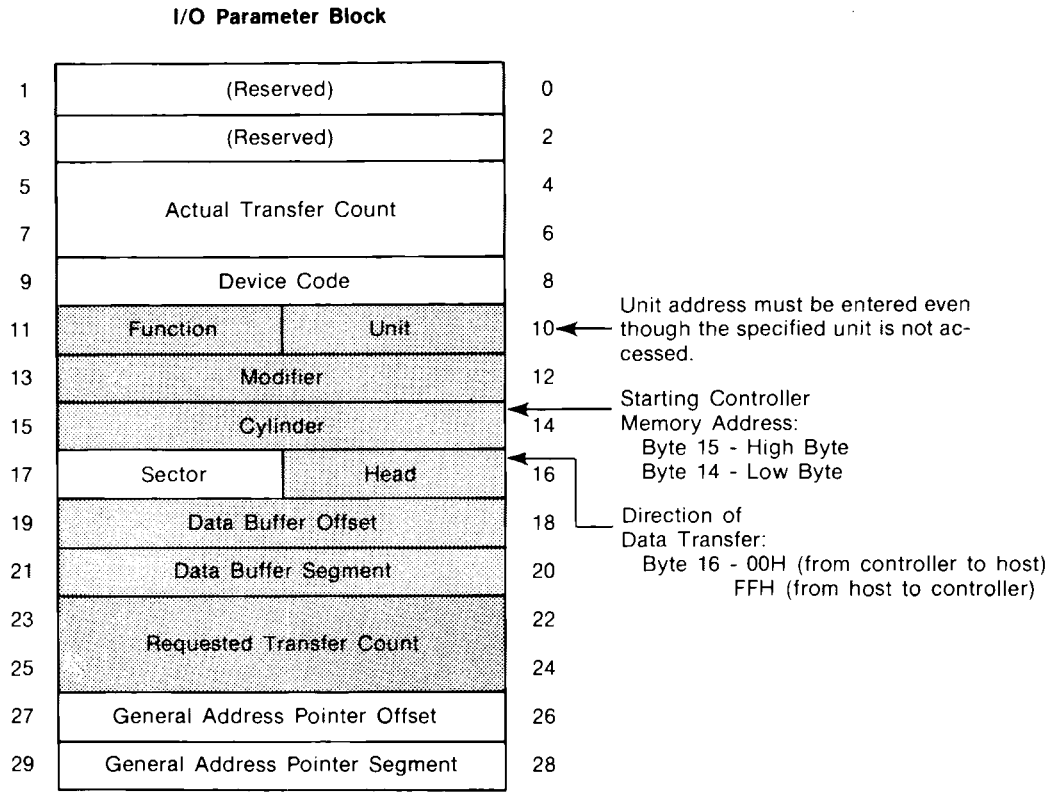


Figure 3-19. Buffer I/O

3-26. DIAGNOSTIC

The diagnostic function (FUNCTION = 0FH) causes the controller to perform a go/no-go self-diagnostic test that verifies internal data and status electronics and checks position and read/write electronics in the disk units. The diagnostic test program is contained in the controller's on-board PROM.

The diagnostic track is always located on a drive unit's last (highest number) track of head 0. When allocating memory space for the disk unit, this track must be dedicated to the diagnostic program. When initiating the diagnostic program, the head and cylinder are selected automatically, the user selects the drive unit. The diagnostic test is divided into three parts. The upper byte of the MODIFIER field (byte 13) determines the part of the diagnostic test that is executed:

Byte 13	Function Executed
00H	Controller seeks the designated diagnostic track, performs a read ID and verifies the track position. It then writes and reads sector 0 with a 55AAH data pattern and verifies that the data read matches the data written.
01H	Controller performs a ROM checksum test to verify the contents of ROM.
02H to FFH	Controller recalibrates the drive.

Any errors in the reading or writing are posted in the error status registers.

To perform this function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-20.

3-27. POSTING STATUS

When the controller has completed an operation (read data, seek track, etc.), it posts the operation status in byte 1, the OPERATION STATUS byte, of the controller invocation block, using the following procedure:

I/O Parameter Block

1	(Reserved)	0
3	(Reserved)	2
5	Actual Transfer Count	4
7		6
9	Device Code	8
11	Function / Unit	10
13	Modifier	12
15	Cylinder	14
17	Sector / Head	16
19	Data Buffer Offset	18
21	Data Buffer Segment	20
23	Requested Transfer Count	22
25		24
27	General Address Pointer Offset	26
29	General Address Pointer Segment	28

Figure 3-20. Diagnostic

1. The controller checks the STATUS SEMAPHORE byte (byte 3 of the controller invocation block) for 00H.
2. If the STATUS SEMAPHORE byte is non-zero, it indicates that the host CPU has not checked the OPERATION STATUS byte for the last status posted. When the host CPU does check the operation status, it sets the STATUS SEMAPHORE byte to 00H and clears the interrupt.
3. When the controller reads 00H in the STATUS SEMAPHORE byte, it posts the current status in the OPERATING STATUS byte, sets the STATUS SEMAPHORE byte back to non-zero and sets an interrupt if enabled (see MODIFIER, bytes 12 and 13, in Figure 3-7).
4. The host CPU in turn, either polls the STATUS SEMAPHORE byte periodically for a non-zero or is interrupted, indicating that new status is present.

The status posted includes: operation complete, seek complete, media change detected and errors detected. If an error was detected, the unit on which the error occurred and an indication of whether the error was a hard error or a summary error is posted (see Figure 3-6). A more detailed description of the error is recorded in the error status buffer in the controller memory. To examine this error status the user transfers the information in the error status buffer from the controller to host system memory using the transfer error status function (FUNCTION = 01H) described in Paragraph 3-28.

It should be noted that error status information is not cumulative. The error status buffers are cleared at the beginning of each new command operation, except the Transfer Error Status Command.

3-28. TRANSFER ERROR STATUS

The Transfer Error Status function (FUNCTION = 01H) transfers error status from the 12-byte error status buffer in the controller memory to a data buffer in the host system memory. The user can then examine the status bits to determine the cause of the error. Table 3-1 shows the information stored in each byte of the error status buffer. Table 3-2 describes which kind of errors are indicated by the setting of the hard (unretrievable) error and soft (retrievable) error bits in bytes 0 through 2. To perform the Transfer Error Status function, set up the shaded bytes in the I/O parameter block as shown in Figure 3-21.

I/O Parameter Block

1	(Reserved)	0
3	(Reserved)	2
5	Actual Transfer Count	4
7		6
9	Device Code	8
11	Function / Unit	10
13	Modifier	12
15	Cylinder	14
17	Sector / Head	16
19	Data Buffer Offset	18
21	Data Buffer Segment	20
23	Requested Transfer Count	22
25		24
27	General Address Pointer Offset	26
29	General Address Pointer Segment	28

Figure 3-21. Transfer Error Status

Table 3-1. Error Status Buffer

Byte	Function
0 and 1	HARD ERROR STATUS — See Table 3-2.
2	SOFT ERROR STATUS — See Table 3-2.
3 and 4	DESIRED CYLINDER
5	DESIRED HEAD AND VOLUME
6	DESIRED SECTOR
7 and 8	ACTUAL CYLINDER AND FLAGS*
9	ACTUAL HEAD AND VOLUME
10	ACTUAL SECTOR
11	NUMBER OF RETRIES ATTEMPTED

*Flags located in bits 4 through 7 of byte 8.

3-29. INTERRUPTS

The controller normally posts interrupts to the host on three conditions:

1. Command complete
2. Seek complete
3. Media change (change disk pack)

The interrupt on command complete can be disabled by entering a one in bit 0 of the Modifier word in the I/O parameter block (bytes 12 and 13). The seek complete and media change interrupts can not be disabled. To clear an interrupt, the host writes a 00H to the Wake-Up I/O port.

Pins on the controller board allow the interrupt priority level of the controller to be set from 0 to 7. Refer to the discussion of interrupt priority level selection in Chapter 2.

3-30. CONTROLLING DATA TRANSFER THROUGH THE iSBX™ BUS

Two iSBX connectors, J3 and J4, are provided on the iSBC 215 board, which allow access to the controller's iSBX bus. The iSBX bus is an Intel standard I/O interface (refer to the *Intel iSBX™ Bus Specification,* Manual Order No. 142686 for detailed information on this standard). It provides 16 data lines and three address lines, providing a total of eight 16-bit I/O ports per connector. Using both J3 and J4, the iSBC 215 controller can thus communicate through the iSBX bus with up to 16 separate peripheral ports.

The iSBX 218 Flexible Disk Controller connects to iSBX connector J4 and allows communication with up to four flexible disk drives. In addition, users can design I/O controller devices that interface with the iSBX bus and use the 8089 to control data transfer.

Two methods are available to control the transfer of data between the iSBC 215 controller and a device connected to the iSBX interface:

1. Commands from the iSBC 215 controller ROM based I/O program.
2. User written I/O program.

The iSBX 218 Flexible Disk Controller uses the ROM based I/O program to control data transfers to and from the flexible disk drives, as described in Paragraphs 3-5 through 3-29. The following paragraphs describe how data can be transferred between the iSBC 215 controller and a user designed I/O controller connected to the iSBX bus, using either the ROM based I/O program or a user written I/O program.

3-31. I/O TRANSFERS USING iSBC 215™ CONTROLLER RESIDENT FIRMWARE

As has been described at the beginning of this chapter, the controller has a ROM based I/O transfer program that is designed to control Winchester drives through the on-board drive interface or flexible disk drives through an iSBX 218 board, which has been attached to iSBX connectors J4. The iSBX TRANSFER command in this program can also be used for general data transfer between the host system memory and a user designed I/O controller, which has been connected to the iSBX bus.

The iSBX TRANSFER command allows the transfer of data between the host memory and the iSBX bus in the same manner as with the WRITE DATA or READ DATA commands. In this case, however, the user must provide the necessary interface hardware between the iSBX connector(s) and the I/O device with which the controller is to communicate. This interface can be very simple, involving data buffers and limited handshaking capability, or as sophisticated as the disk drive interface circuitry used in the iSBX 218 and iSBC 215 controllers. The complexity of the interface will depend on the type of I/O device being interfaced with and the desired data transfer rate.

3-32. DATA TRANSFER USING USER WRITTEN I/O TRANSFER PROGRAMS

A second method of initiating and controlling data transfer between the host and the iSBX interface is through a user designed program written in 8089 assembler code. This method is more difficult to implement, but also more flexible. Such programs can be executed either from host memory or from the iSBC 215 controller on-board RAM.

Table 3-2. Bit Functions in Hard and Soft Error Bytes

Byte	Bit	Function
0	0 through 2	Reserved for future use.
	3	RAM ERROR — Controller RAM error was detected.
	4	ROM ERROR — Controller ROM error was detected.
	5	SEEK IN PROGRESS — Indicates a seek was already in progress for a unit when another disk operation was requested.
	6	ILLEGAL FORMAT TYPE — Both alternate track and defective alternate track flag set indicating an attempt to create an alternate track for a defective alternate track, which is not allowed, or an attempt to access an unassigned alternate track.
	7	END OF MEDIA — End of media was encountered before requested transfer count expired.
1	8	ILLEGAL SECTOR SIZE — Sector size read from the sector ID field conflicts with sector size information that controller specified in initialization command.
	9	DIAGNOSTIC FAULT — Micro-diagnostic fault detected.
	A	NO INDEX — Controller did not detect index pulse.
	B	INVALID COMMAND — Invalid function code detected.
	C	SECTOR NOT FOUND — Desired sector could not be located on selected track.
	D	INVALID ADDRESS — Invalid address was requested.
	E	SELECTED UNIT NOT READY — Selected unit is not ready, not connected, or not responding to unit connect request.
	F	WRITE PROTECTION FAULT — An attempt has been made to write to a write protected unit.
2	0 through 2	Reserved for future use.
	3	DATA FIELD ECC ERROR — Error has been detected in the data field of a sector. If bit 6 in Controller-Invocation status byte (byte 1) is set, error is hard and uncorrectable. If bit 6 is not set, error is soft and correctable.
	4	ID FIELD ECC ERROR — Error has been detected in the ID field of a sector. If bit 6 in Controller-Invocation status byte (byte 1) is set, error is hard and uncorrectable. If bit 6 is not set, error is soft and correctable.
	5	DRIVE FAULT — Hardware fault detected in selected drive unit. Fault characterized by: read/write fault, positioner fault, power fault or speed fault.
	6	CYLINDER ADDRESS MISCOMPARE — ID field contains a cylinder address different from the expected cylinder address.
	7	SEEK ERROR — Hardware seek error was detected.

Executing the program from host memory is inherently slower than executing the program from on-board RAM, because it requires constant access of the Multibus interface. This method, however, allows the size of the program to be virtually unlimited. The procedure for executing a program from host memory is much the same as for executing a program stored in controller local memory:

1. I/O communications blocks are established in host system memory.
2. The Wake-Up Address switches in the controller are set for the address of the first byte of the wake-up block.
3. The host initiates program execution with 01H written to the wake-up I/O port.

There are two important differences in the set up of the I/O communications blocks when executing I/O programs from host system memory.

1. Byte 0 of the channel control block must be set to 03H to indicate to the controller that the I/O program is located in host memory.
2. The controller invocation block becomes the I/O parameter block. Refer to the *8086 Family User's Manual*, Manual Order No. 9800722 for detailed information on setting up an I/O parameter block when the I/O program is to be executed from host system memory.

Executing the program from on-board RAM presents space limitations, but allows data transfers to be performed at the 8089's full program execution speed. To overcome some of the limited RAM space problems, the program can be divided into shorter routines, which are stored in the host memory and read into RAM as needed. Separate routines might thus be written for disk formatting, checking status, writing and reading. The iSBX EXECUTE com-

mand, allows an I/O transfer routine or program that is stored in iSBC 215 controller RAM to be started from a host program. When writing an I/O transfer program, the following software and hardware considerations should be noted.

I/O PORT ADDRESSING

The eight iSBX bus ports reside in the controller's memory mapped I/O space, with each I/O port being given two addresses: one to connect it to connector J3 and another for J4. Table 3-3 shows these addresses. To access any of these ports for a data transfer, the 8089 merely executes a write or a read to the address of the selected port.

Table 3-3. iSBX™ Bus I/O Port Addresses

Port	ISBX Bus Port Address Assignments			
	J3- Channel 0	J3- Channel 1	J4- Channel 0	J4- Channel 1
0	C070	C0B0	C0D0	C0E0
1	C071	C0B1	C0D1	C0E1
3	C073	C0B3	B0D3	C0E3
4	C074	C0B4	C0D4	C0E4
5	C075	C0B5	C0D5	C0E5
6	C076	C0B6	C0D6	C0E6
7	C077	C0B7	C0D7	C0E7

RAM SPACE ALLOCATION

The controller RAM is used for a variety of purposes, and as such, only a portion of it is available for storage of an iSBX bus I/O program and its parameters. The available RAM space is shown in Table 3-4. Note that enough space has been reserved in the data buffer to store an entire 1024 byte disk sector of data. If the sectors are to be smaller or if for some other reason less data buffer space is needed, some of this space can be used for program storage.

Table 3-4. iSBC 215™ Controller RAM Available for Program and Parameter Storage

Description	Address Range
Data Buffer*	4000 to 440F
Program Storage	4410 to 45FF 46C0 to 473A
Scratch PAD*	4600 to 46BF
Variable Storage**	47B0 to 47CF 47E0 to 47FF

*May be modified by 215 command usage
**Not available if iSBX 218 is installed

PROGRAM STRUCTURE

In writing a program in 8089 assembly code, reference to the *8089 Assembler User's Guide,* Manual Order number 9800938 and the *8086 Family User's Manual,* Manual Order No. 9800722 is essential. The 8089 offers a number of techniques for implementing handshaking between the 8089 and the iSBX bus, including the user of wait states and DMA transfers (essentially an interrupt driven mode) of whole blocks of data. These and other interfacing techniques are discussed in this user's guide.

HARDWARE CONSIDERATIONS

There are two groups of interface control lines between the 8089 and the iSBX bus. The first group includes handshake and control lines; the second group includes program lines.

Table 3-5 lists the first group of lines. The 8089 uses these lines directly to control data transfer through the iSBX bus.

Table 3-5. 8089 Handshake and Control Lines on the iSBX™ Bus

J3 or J4 Pin	Description	iSBX Bus Mnemonic
34	Request DMA Transfer	MDRQT
32	Acknowledge DMA Transfer	MDACK/
16	Initiate Wait State	MWAIT/
6	Multibus Clock	MCLK
15	I/O Read	IORD/
13	I/O Write	IOWRT/
26	Terminate DMA Activity	TDMA

The second group of lines are used for control and status. The 8089 accesses these lines through a read to memory mapped I/O address 8000H for connector J3 and 8008H for connector J4. Table 3-6 lists these lines, their pin assignments and bit assignments.

Jumpers can be connected on the iSBC 215 controller to allow the 8089 to also write bits onto the Option lines (as shown in Table 3-7). The option lines on only one of the interface connectors may be driven at a time. To drive the lines, the 8089 writes to memory mapped I/O port 8018H. Bit 1 drives OP00 or OP01, but not both at one time, bit 2 drives OP10 and OP11, but not both at one time. **All other bit positions in the data word must be set to zero when driving the Option lines.**

Table 3-6. Control and Status Lines on the iSBX™ Interface

Connector 1 J3	Address 8000H	Connector 2 J4	Address 8008H	Pin No.	Description	iSBX Bus Mnemonic
OP00	Bit B	OP01	Bit 3	30	Option 0	OPT0
OP10	Bit C	OP11	Bit 4	28	Option 1	OPT1
INTR00	Bit 9	INTR01	Bit 1	14	Interrupt 0	MINTR0
INTR10	Bit A	INTR11	Bit 2	12	Interrupt 1	MINTR1
M0PST/	Bit 8	M1PST/	Bit 0	8	iSBX Board Present	MPST/

Table 3-7. Jumper Connections Allowing Option Lines to be Driven

Line	iSBX Connector	Jumper Connection
OP00	J3, OP0	W11, 1-2
OP11	J4, OP0	W11, 1-3
OP10	J3, OP1	W12, 1-2
OP11	J4, OP1	W12, 1-3

NOTE

If an iSBX controller is not installed on the iSBC 215 board, or if an iSBX controller that has been installed on a particular iSBX connector does not drive its respective Terminate DMA Activity line, the connector's corresponding jumper (W3 1-2 or W4 1-2) must be installed.

PROGRAM EXECUTION

When loading and executing a user written I/O transfer program or routine, the following procedure is used:

1. Load the program or routine into RAM using the BUFFER I/O command from the iSBC 215 controller firmware.
2. Execute the iSBX EXECUTE command to start the program. Note that the General Address Pointer in the I/O parameter block for this command must point to the address of the start of the program in on-board RAM (see Figure 3-22). Also, upon entering the program, the following 8089 registers are defined as:

   GA: 7E00H Scratch Pad Stack

   IX: 0 to 3 Unit Number

   Exit from the program must always be to ROM location 00C5H and the 8089 BC register must be set to FFH and the 8089 GC register must be set to 7F3BH.

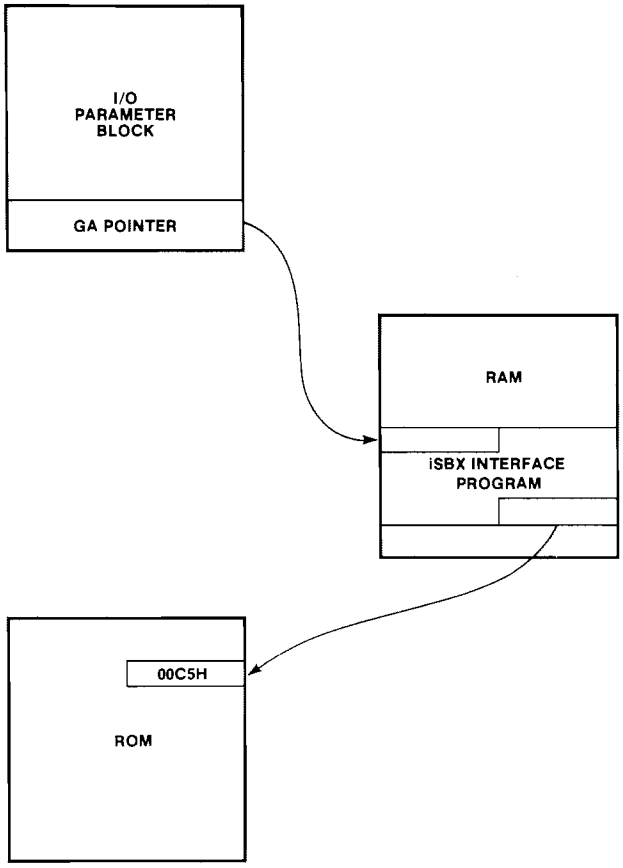


Figure 3-22. Execution of iSBX™ Bus I/O Program From RAM

3-33. EXAMPLE CONTROLLER I/O PROGRAM

Appendix A provides an example of a host processor program to initiate data transfers between the host system memory and disk drives through the iSBC 215 controller.

CHAPTER 4
PRINCIPLES OF OPERATION

4-1. INTRODUCTION

This chapter provides a functional description of the iSBC 215 Winchester Disk Controller circuit operation. The discussion assumes that the reader has a working knowledge of digital electronics and has access to the individual component description of each integrated circuit used on the board. As a prerequisite, the reader should be familiar with the programming conventions discussed in Chapter 3 of this manual, and the functional operation of the Intel 8089 I/O processor and the Multibus interface. Familiarity with the disk drive's operation and interface specifications will also prove beneficial in understanding the controller operation.

4-2. SCHEMATIC INTERPRETATION

A set of schematic diagrams for the controller board (Figure 5-3) and a component location diagram (Figure 5-2) are included in Chapter 5 of this manual.

The schematics are drawn to standard drafting conventions with input signals entering from the left and output signals exiting to the right. Input and output signals between individual sheets of a schematic include a location coordinate code immediately preceeding (input signals) or following (output signals) the signal name. This code defines the location of the origin or destination of the signal within the schematic diagrams. The first digit of the code is the schematic sheet number, and the last two characters specify the zone defined by the horizontal and vertical grid coordinates, which are printed around the perimeter of each schematic sheet. For example, the code "7B8" indicates that the origin or destination of the associated signal appears on sheet 7 of the schematic set within the zone defined by grid coordinates "B" and "8".

An "X" for one of the grid coordinates indicates an entire vertical column or horizontal row on the schematic sheet. For example, the code "7BX" indicates the entire "B" zone on sheet 7.

The logic symbols used in this manual are drawn as specified in ANSI Standards 14.15 and Y32.14. Standard definitions are used for symbols and active line levels on inputs and outputs (see Figure 4-1). A small circle on the input of a logic element indicates that a relative low level is needed to activate the element. The absence of a circle indicates that a relative high level is needed to activate the element. Output levels are indicated in the same manner. Logic gating symbols are drawn according to their circuit function rather than the manufacturer's definition. For example, the gate, which the truth table in Figure 4-1 defines, can be drawn in one of the two configurations shown, depending on its circuit application.

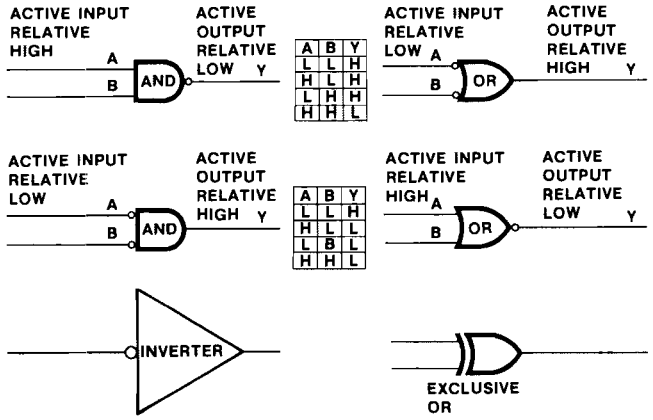


Figure 4-1. Logic Conventions

In addition to the inversion symbol convention, signal nomenclature also follows an active state convention. When a signal (or level) is active in its low state, the signal name is followed by a virgule or "slash" (e.g., XACK/); when a signal is active in its high state, the slash is omitted from the signal name, (e.g., XACK). This convention corresponds to putting a bar over a signal name to indicate it is active in its low state (e.g., XACK).

4-3. FUNCTIONAL OVERVIEW

General. The function of the iSBC 215 Winchester Disk Controller board is to allow the host system to access any location on a specific disk of a selected disk drive and either:

1. Transfer data to that disk location from system (host) memory (write operation), or
2. Transfer data from that disk location to system memory (read operation).

To accomplish this task, the controller circuitry is divided into two sections (see Figure 4-2):

1. Logic that controls communications and data transfer between the host processor and the controller through the Multibus interface, and
2. Logic that controls data transfer between the controller and the disk drive(s) through the disk interface, and between the controller and the iSBX bus through the iSBX bus interface.

The Intel 8089 I/O processor (IOP) controls the data transfer process, using a program stored in on-board ROM. It receives instructions from the host processor through four I/O communications blocks in system memory. Once the host instructs the controller to begin a data transfer, the 8089's internal processor makes a DMA transfer to or from system memory, independent of the host processor.

2K bytes of RAM are included on the board for intermediate storage of data and to allow on-board error checking. This data buffer allows DMA transfer to be made between the controller and host system memory, which minimizes Multibus™ overhead and eliminates disk drive overruns.

Communicating with the host. Figure 4-3 provides a detailed block diagram of the controller. The Bus Arbiter and the Bus Controller manage the transfer of data between system memory and controller through the Multibus interface. The Bus Arbiter negotiates with the current bus master for control of the Multibus interface. The Bus Controller generates control signals that gate data transfers

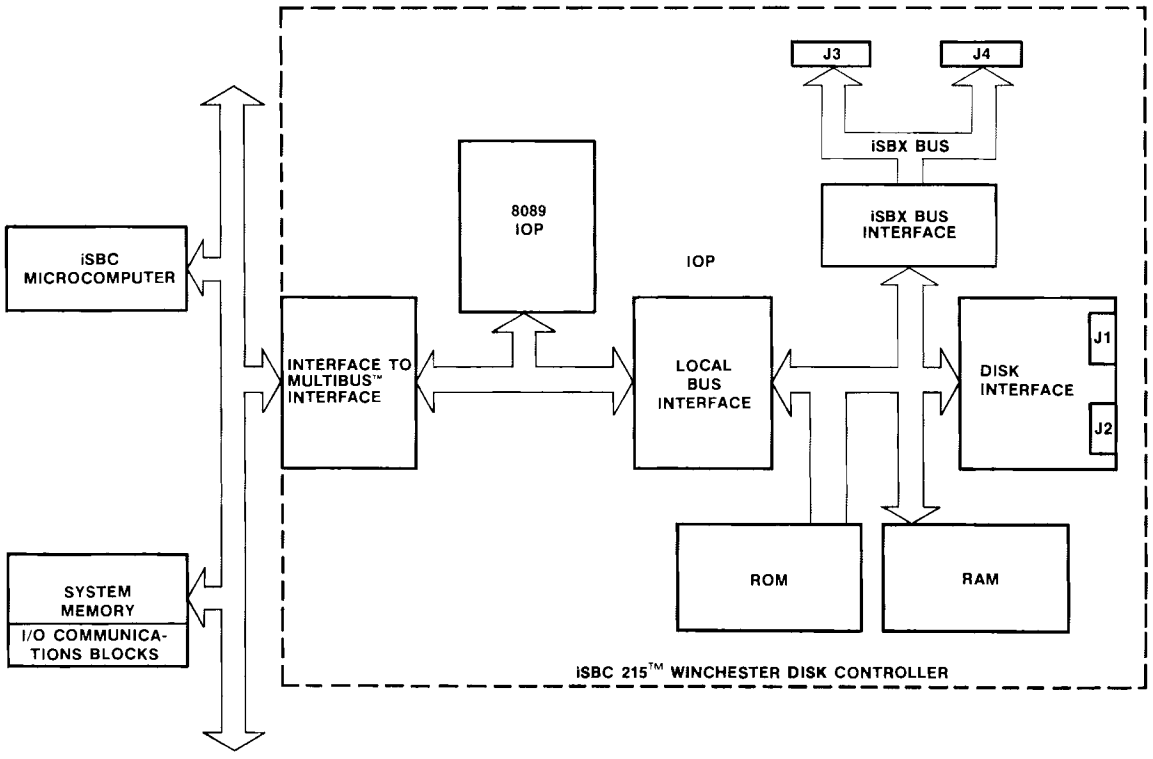


Figure 4-2. Simplified Block Diagram of iSBX 215™ Controller

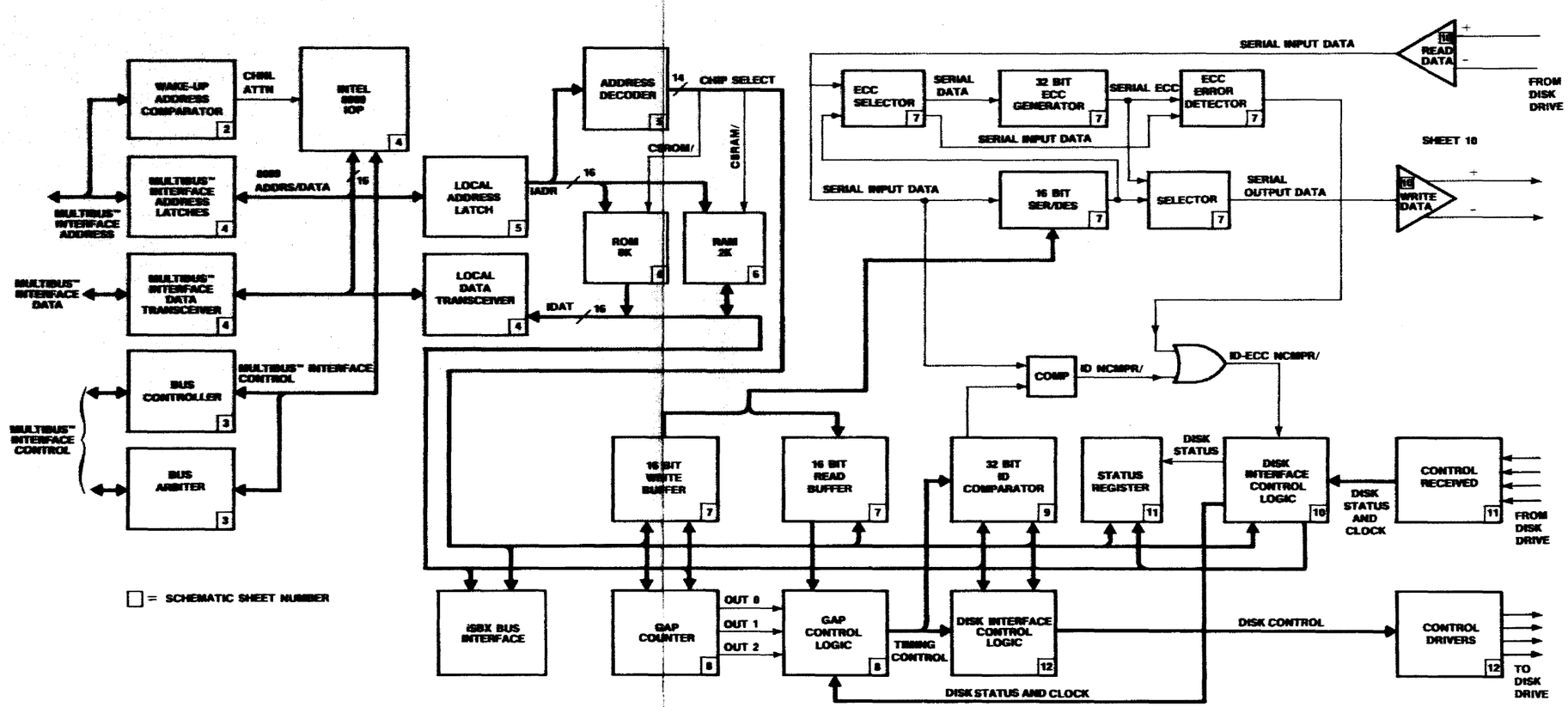


Figure 4-3. iSBC 215™ Controller Functional Block Diagram

between system memory and the on-board RAM. It also controls the transfer of data from RAM to the disk communication circuitry.

The Multibus interface Address Latches transmit 20-bit addresses to system memory via the Multibus interface. The Multibus interface Data Transceiver transmits data either to or from system memory via the Multibus Interface. The controller data bus is 16-bits. The Data Transceiver uses a byte-swap technique to allow data transfer with either an 8-bit or 16-bit system memory.

The Wake-Up Address Comparator is used to assign the controller a host system I/O port address and to set up a communications link between the 8089 IOP and the I/O communications blocks in system memory. (A detailed discussion of the controller initialization procedure is given in Chapter 3 and in Paragraphs 4-12 through 4-15 in this section.)

Communicating with the disk. The 8089 IOP treats the ROM, RAM, iSBX I/O ports and disk communications side of the controller circuitry as local memory. The Local Address Latches transmit 16-bit addresses to local memory. The Local Data Transceiver transmits data either to or from local memory. Some of the addresses in local memory provide access to local I/O ports (see Paragraph 4-20 for a detailed discussion of local I/O ports). The Address Decoder decodes these addresses and generates chip select or enable signals that control the transfer of data to and from the disk. For example, the address 8028H enables the 16-Bit Write Buffer to receive a data word from the local memory. The ROM and RAM are also assigned specific ranges of addresses in local memory.

The 16-Bit SER/DES (Serializer/Deserializer) performs the serial-to-parallel and parallel-to-serial conversion required to transfer data between the disk and system memory. The 16-Bit Write Buffer and the 16-Bit Read Buffer provide intermediate storage for a single 16-bit parallel word between the RAM and the SER/DES. On a write operation, a 16-bit word is transferred from RAM to the write buffer. The SER/DES then converts the word from parallel to serial and transmits it to the disk through the write data driver. On a read operation, a 16-bit serial word is transmitted from the disk through the Read Data Receivers to the SER/DES. The SER/DES then performs a serial-to-parallel conversion and stores the resulting parallel word in the read buffer. The Write Data Driver and the Read Data Receivers are designed to generate and read the differential NRZ drive signals.

The 32-Bit ID Comparator determines when the selected sector on the disk is found during the search for sector ID operation that precedes a write or read function. When a write or read is initiated, the 32-bit sector identification (cylinder, head and sector number) is loaded in the 32-Bit ID Comparator. Sector IDs from the disk are then read and compared with the selected sector ID. When the selected sector is found, data transfer is initiated.

The 32-Bit ECC Generator creates an error checking code (ECC) that is appended to the end of each sector ID field and to each data field (see Figure 3-2). This ECC is used for error checking and correction of data errors. It allows all the errors in a burst of up to 11 bits to be corrected, and allows errors in a burst of 32 bits to be detected.

The Gap Control Logic controls the spacing of data within a sector. Three programmable Counters, which count disk clock pulses, provide timing for the Gap Control Logic. The ability to program the Counters allows the disk(s) to be formatted for a number of different record sizes and gap lengths.

The Disk Control Logic transmits disk control information to the disk drive units through the Control Line Drivers. The Input Control Logic receives status information from the disk drive units and controls the sequencing of the controller read and write operations.

The iSBX Interface provides the ability to connect Intel iSBX Multimodule devices to the controller board in order to control other I/O devices such as flexible disk drives or magnetic tape cartridge drives. The iSBX interface is discussed in more detail in Paragraph 4-25.

A more detailed overview of the read and write operations is given in Paragraph 4-29 through 4-33.

4-4. DETAILED FUNCTIONAL DESCRIPTION

The detailed functional description of the iSBC 215 Winchester Disk Controller circuitry is divided into two major sections: Controller to Host Communications and Controller to Disk Communications. Within each of these sections, the following subjects are discussed:

Controller to Host Communications:

- Multibus™ Interface
- 8089 IOP
- Bus Arbiter
- Bus Controller
- Multibus™ Data Transfer Logic

- Controller Initialization
- Wake-Up Address Comparator
- Controller Reset and Clear
- Establishing a Link with I/O Communications Blocks
- Interrupt Priority
- Memory Map
- ROM
- RAM
- I/O Port Decode Logic

Controller to Disk Communications

- Controller to Disk Drive Interface
- DMA Mode
- Disk Formatting
- Write Data Transfer
- Read Data Transfer
- SER/DES Logic
- Sync Byte Comparator Logic
- 32-Bit ID Comparator Logic
- ECC Generator Logic
- Status Register Logic
- Line Drivers and Receivers

4-5. CONTROLLER TO HOST COMMUNICATIONS

The following discussion provides a detailed functional description of the section of the iSBC 215 Winchester Disk Controller that communicates with the host through the Multibus interface.

4-6. MULTIBUS™ INTERFACE

The 8089 IOP communicates with the host processor and the system memory through the Multibus interface. The Multibus interface signal description and pin configurations are explained in Chapter 2. A detailed description of the Multibus interface operation can be found in the *Intel Multibus™ Specification* Manual Order Number 9800683.

4-7. 8089 I/O PROCESSOR (IOP)

The 8089 IOP, U84 (4X4), is a microprocessor device that has been designed specifically to perform high speed I/O transfers of data between system memory and mass storage devices such as disk drives. Its ability to perform DMA data transfers independent of the host processor allows it to carry out most system memory-to-disk transfers of data simultaneously with other host processor operations. Refer to *The 8086 Family User's Manual*, Manual Order Number 9800722 for a detailed explanation of the 8089 and supporting IC devices.

A number of 8089 control lines have important functions in the controller design. The PWR-RST line (4D1), when pulled high, resets the 8089 to the beginning of its internal firmware control program. Channel Attention line CA (4B4) allows the host to gain the attention of the 8089. On the first channel attention following a reset, the 8089 fetches the contents of address FFFF6H and begins an internal initialization procedure. On subsequent channel attentions, the 8089 looks to the I/O communications blocks in system memory for further instructions. Refer to Paragraphs 4-12 through 4-15 for a detailed discussion of the controller initialization procedure and the use of the CA line.

The Bus Interface Unit (BIU) in the 8089 controls the controller local data bus cycles, transferring instructions and data between the 8089 IOP and external memory or the disk. Every bus access is associated with a register tag bit that indicates to the BIU whether the host system memory or local memory is to be addressed. The BIU outputs the type of bus cycle on status lines S0/, S1/ and S2/. The 8288 Bus Controller decodes these lines and provides signals that selectively enable one bus or the other.

The 8089 is a 16-bit processor, but it is capable of making both single-byte fetches (8-bit system memory) or two-byte fetches (16-bit system memory). The address zero line, IADR-0 (5B7), controls the byte swapping facility of the controller when communicating with an 8-bit system memory.

4-8. CLOCK CIRCUIT

The clock circuit consists of U55, an 8284A Clock/Driver (4C6), and a 15 MHz crystal. The 8284A divides the crystal output by three to produce the 5 MHz CLK necessary to drive the 8089 IOP. The 8284A produces a reset signal (RST), which is used on power-up to reset the 8089, Interrupt Latch U56 (3B5) and the Read/Write Control logic. In addition to the reset signal, the 8284A also produces a synchronized ready (RDY) input to the 8089. A high on the RDY line received from the addressed device (XACK/ from external memory or the iSBX interface, or RDY from the on-board read/write port), indicates that the memory or read/write port has accepted data during a write operation or data is ready to be read during a read operation.

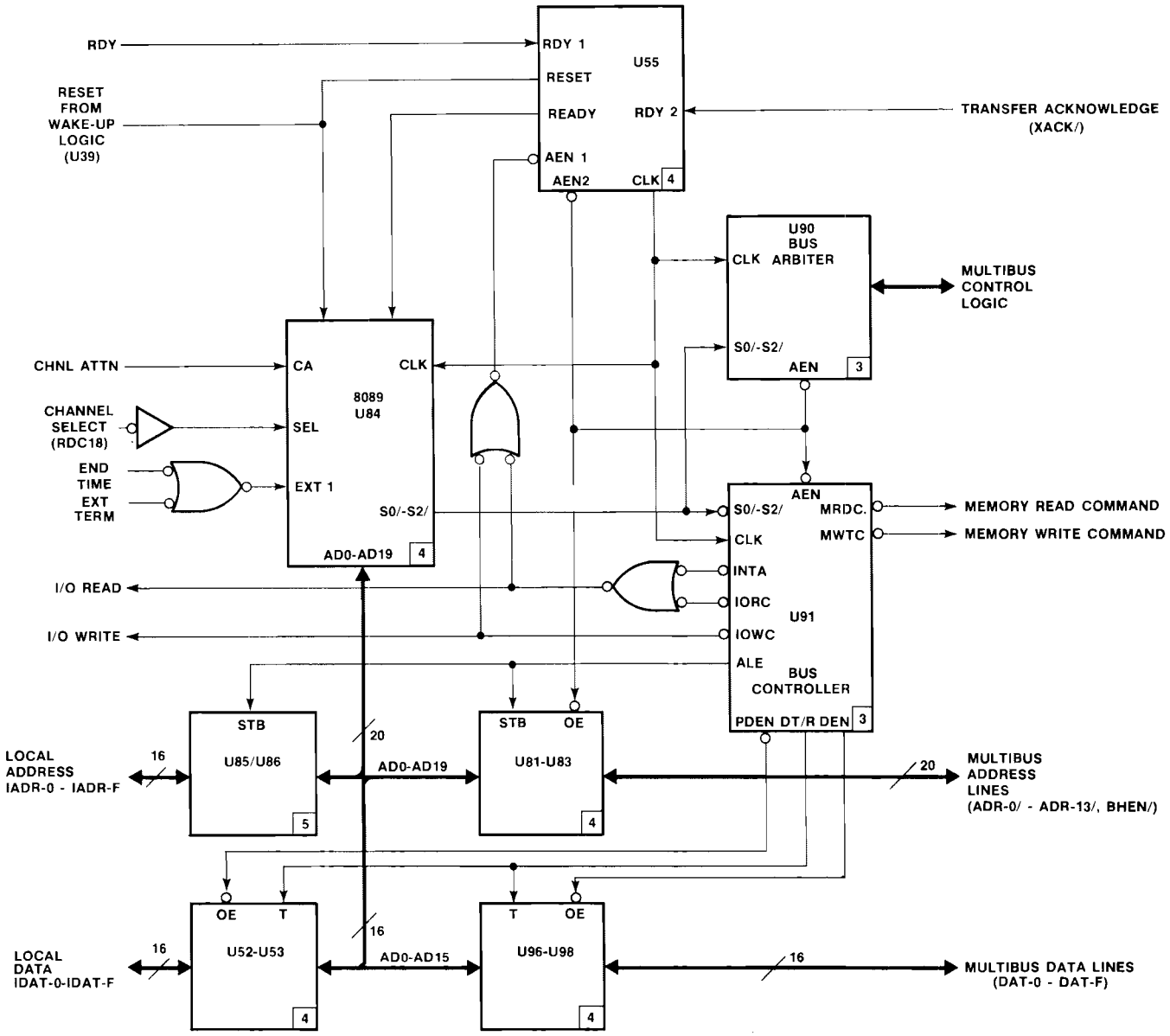


Figure 4-4. Bus Arbiter and Bus Controller Logic

4-9. BUS ARBITER

The 8289 Bus Arbiter, U90 (3D6), controls the 8089 IOP's access to the Multibus interface (see Figure 4-4). The 8289 monitors the 8089's status lines (S0/, S1/ and S2/). When the lines indicate that the 8089 needs a Multibus interface cycle, and the 8089 does not presently control the bus, the 8289 activates a bus request (BREQ/). The low on BREQ/ is transmitted to the bus priority resolving circuitry in the host processor, which returns a low on Bus Priority In line BPRN/, giving the 8089 access to the Multibus interface. Having received access to the Multibus interface, the 8289 activates its busy signal (BUSY/), indicating to the other masters on the system that the Multibus interface is in use. The 8289 then activates the address enable signal (AEN/), which is transmitted to the 8288 Bus Controller, U91 (3C4), to enable its command outputs, to the 8284A Clock Generator, U55 (4C6), to enable its bus ready logic, and to the System Address Latches, U81, U82 and U83 (4X2), to allow an address to be gated on to the Multibus interface.

Jumper pins W18-1, 2 and 3 allow the user to select the Any Request option. A jumper installed between pins W18-1 and 2 causes the controller to relinquish control of the Multibus interface following a request from a higher priority device only. A jumper installed between pins W18-1 and 3 causes the controller to relinquish control of the Multibus interface following a request from any device, higher or lower priority.

4-10. BUS CONTROLLER LOGIC

The 8288 Bus Controller, U91 (3C4), decodes the status line outputs (S0/, S1/ and S2/) from the 8089 IOP and generates the appropriate bus cycle signal. Table 4-1 shows the different signals generated for each configuration of the IOP's status lines.

Table 4-1. 8089 Status Line Decodes

Status Input S2/	S1/	S0/	CPU Cycle	8288 Command
0	0	0	Instruction Fetch, Local	INTA/
0	0	1	Read Memory, Local	IORC/
0	1	0	Write Memory, Local	IOWC/, AIOWC/
0	1	1	Halt	None
1	0	0	Instruction Fetch, System	MRDC/
1	0	1	Read Memory, System	MRDC/
1	1	0	Write Memory, System	MWTC/, AMWC/
1	1	1	Passive	None

These bus cycle signals can be divided into two groups: those which allow the 8089 to access system memory (MWTC/ and MRDC/) and those which allow the 8089 to access local memory (I-AIOWC/ and I-IORC/). The 8089 uses the I/O Read (I-IORC/) and I/O Write (I-AIOWC/) signals to read information from the local ROM, U87 and U88, (6X7), or to read from or write to the local RAM, U99 through U102, (6X4). The 8089 also uses I-IORC/ and I-AIOWC/ to gate on the Read and Write Function Decoders, U35 and U36 (5B2 and 5A2). The function decoders are explained further in Paragraph 4-20.

The 8288 Bus Controller also generates a group of signals that control address and data flow throughout the iSBC 215 controller. The Address Latch Enable line (ALE) is used to strobe addresses from the 8089 into both the system Address Latches, U81-U83 (4X2), and the Local Address Latches, U85-U86 (5X7).

Data Transmit/Receive (DT/R), Data Enable (DEN), and Peripheral Data Enable (PDEN/) control the data flow through the controller. DT/R controls the direction of data transmission through the Multibus interface and local transceivers. If DT/R is high, data is transmitted either on to the Multibus interface through transceivers U96, U97 and U98 (4X7) or on to the local bus through transceivers U52 and U53 (4X6). If DT/R is low, the data transfer is in the opposite direction, into the 8089 through one of the two sets of transceivers. DEN and PDEN controls the selection of the transceivers. If DEN is high the Multibus interface transceivers U96, U97 and U98 are enabled, and if PDEN/ is low (indicating a peripheral cycle) local transceivers U52 and U53 are enabled.

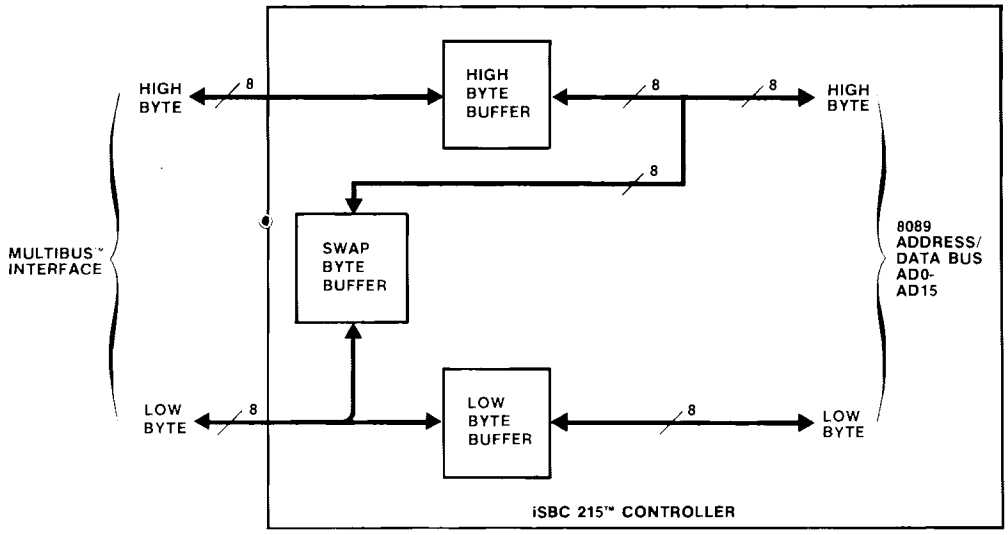
4-11. MULTIBUS™ INTERFACE DATA TRANSFER LOGIC

The controller has three sets of Multibus interface data transceivers: low-byte transceiver U97, which buffers DAT-0/ through DAT-7/, high-byte transceiver U96, which buffers DAT-8/ through DAT-F/, and swap-byte transceiver U98, which takes the data from DAT-0/ through DAT-7/ on the Multibus interface and switches it to high-byte data bus lines AD8 through AD15 on the controller board (see Figure 4-5). This byte-swap is performed only when the controller is interfacing with a 16-bit system memory in byte mode. In this case, every odd address read from system memory is transmitted to the high-byte data lines of the controller. The procedure is reversed when writing to the 8-bit system memory. Three signals control the transceiver: ENBL HI BYTE/ (5C1), which controls the high-byte transceiver; ENBL LO BYTE/ (5C1), which controls the low-byte transceiver (derived from ADRO/); and ENBL SWAP BYTE/ (5C1), which controls the swap byte transceiver. Figure 4-5 shows when each of the control signals is active.

4-12. CONTROLLER INITIALIZATION

Before data can be transferred between system memory and the controller, the controller must be initialized. The initialization procedure, which is described in Paragraph 3-12, involves:

1. Resetting the 8089 IOP.
2. Clearing the reset.
3. Establishing a communication link between the 8089 and the I/O communications blocks in system memory.
4. Reading the disk drive parameters from system memory to the controller on-board RAM.



	8-BIT SYSTEM MEMORY		16-BIT SYSTEM MEMORY	
	I-ADR0/ L	I-ADR0/ H	I-ADR0/ L	I-ADRO/ H
ENBL LO BYTE/	L	H	*	L
ENBL SWAP BYTE/	H	L	*	H
ENBL HI BYTE/	H	H	*	L
*NOT APPLICABLE				

Figure 4-5. Data Transmission Between Multibus™ Interface and Controller Data Transceivers

The following paragraphs describe the hardware operations that take place during this initialization procedure. (See Figure 4-6.)

4-13. WAKE-UP ADDRESS COMPARATOR

For the purpose of resetting the controller, clearing the reset or getting the attention of the 8089 IOP (raising CA), the host addresses the controller as an I/O port in its system I/O space. To perform one of these functions it writes a one byte command to the specified I/O port called the wake-up I/O port. Table 4-2 shows the three possible commands. The user determines the address of the I/O port at which the controller is to reside (called the "Wake-Up Address") and sets the address on the Wake-Up Address switches S1-1 through S1-8 and S2-3 through S2-10 (2X6), on the controller board. When the host issues a write command (IOWC/) to the Wake-Up Address in system I/O space, U77 through U80 (2X5) on the controller compare the address with the switch settings. If they agree, WAKEUP/ is pulled low, enabling the controller to decode the command on the Multibus interface data lines and determine the action to be taken.

The host may use 8-bit or 16-bit I/O port addressing. The user sets switch S2-2 (2A7) to indicate to the controller the type of addressing that is being used. When S2-2 is open (8-bit addressing), pin 9 of U75 is held high, creating a "don't care" situation for the outputs of High-Byte Wake-Up Address Comparators U77 and U78.

Table 4-2. Host Wake-Up Commands

Command	Description
00H	Clear Interrupt and Clear Reset
01H	Channel Attention (Start 8089 IOP)
02H	Reset 8089 IOP

As it is discussed in Chapter 3, the controller also uses the setting of the Wake-Up Address switches to calculate the address of the first byte of the Wake-Up Block, which is the first I/O communications block in system memory.

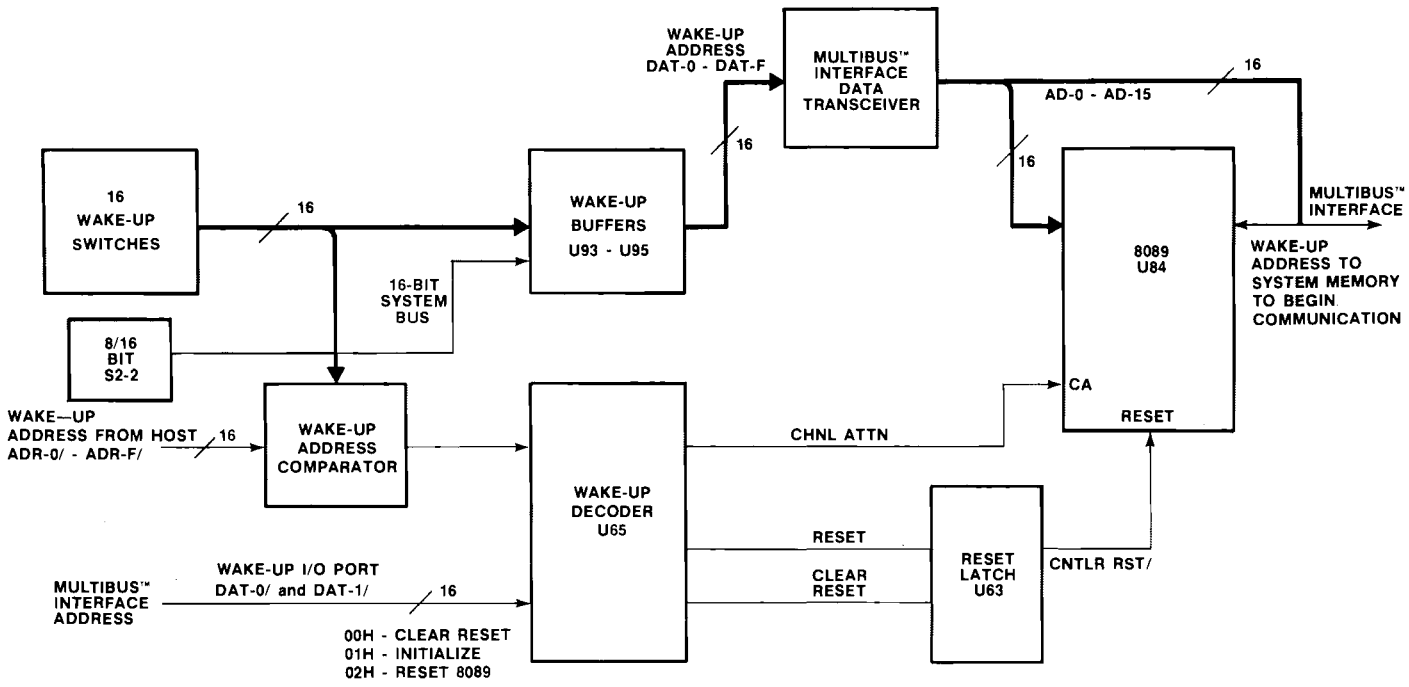


Figure 4-6. Wake-Up Address Logic

4-14. CONTROLLER RESET AND CLEAR

The first operation that must be performed during the initialization of the controller is a reset of the 8089 IOP. To reset the 8089, the host processor writes an 02H to the wake-up address. The WAKE-UP/ line goes low and gates the 02H (DAT-9/ high and DAT-1/ low) into the Wake-Up Decoder, U65 (3B7), producing a low on the controller reset (CNTLR RST/) line. A low on CNTLR RST/ resets the 8089 (4X4), resets Read/Write Control Logic U42 (sheet 8) and clears Control Register U3 (12B5). Once the controller has been reset, the host processor writes a 00H (Clear Interrupt) to the wake-up address, which clears the reset. The Wake-Up Decoder U65 decodes the highs on DAT-0/ and DAT-1/ to raise CNTLR RST/.

4-15. ESTABLISHING A LINK WITH I/O COMMUNICATIONS BLOCKS

Following a power-up event or a software reset (02H written to the wake-up I/O port), the link between the controller and the I/O communications blocks in system memory must be established. To establish this link, a clear reset (00H) is written to the wake-up I/O port followed by a channel attention (01H). The 01H is gated into U65, producing a high on CHNL ATTN, which in turn raises the CA input to the 8089 IOP (4C4).

Being the first Channel Attention following reset, the 8089 begins an internal initialization process. The first step of this process is to do a fetch of address FFFF6H. The address is transmitted on the 8089 Address/Data lines (AD0-AD15) to latches U85 and U86 (5B7). Gates U66 and U72 through U76 (5D4) decode the output of these latches. The output of U76 enables U89 (5D3), gating the status of the 16-bit SYS BUS switch (S2-1) through Data Bit 0 line (DAT-0/) to the 8089. Switch S2-1 on (16 Bit SYS BUS/ low) indicates that the host memory system supports 16-bit data transfers and S2-1 off indicates 8-bit data transfers. Inverter U89 also generates Transfer Acknowledge (XACK/), which is sent to the 8089 (through the 8284A) indicating that the operation has been completed.

After determining the width of the system bus (8-bit or 16-bit) the 8089 fetches the addresses shown in Figure 4-7 as part of the initialization sequence.

Fetching addresses FFFF8/9H gates zeros into the 8089. Fetching addresses FFFFA/BH causes the GATE SWS/ line (5C1) to go low. GATE SWS/ gates the settings of the wake-up address switches, S1-1

through S1-8 and S2-3 through S2-10 through buffers U93, U94 and U95 (2X3) and into the 8089. The 8089 multiplies the settings of the wake-up switch by 2^4, to determine the 20-bit address of the wake-up block, the first I/O communications block in system memory. The 8089 then uses this address to fetch the wake-up block and establish a link with the I/O communications blocks. On subsequent channel attentions (host writes 01H to the wake-up I/O port), the 8089 skips the wake-up block and goes directly to the channel control block, the second I/O communications block. The 8089 uses the channel control block to obtain the starting address of the controller's ROM resident I/O transfer program (also called the channel control program). From this point on, this firmware program directs the controller activities. One of the first operations of the firmware is to again fetch the starting address of the wake-up block. It then links its way through the channel control block and the controller invocation block to the I/O parameter block where it obtains instructions and parameters for a specific I/O operation.

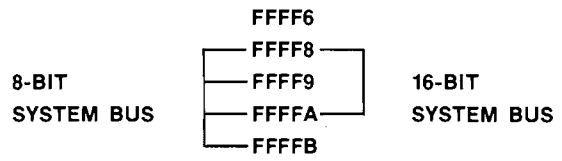


Figure 4-7. Address Fetches In Initialization Sequence.

4-16. INTERRUPT PRIORITY LOGIC

Wire wrap pins W19-C and W19-0 through W19-7 (3B2) allow the user to select the interrupt priority of the controller with respect to other peripherals in the system. To issue an interrupt to the host, the 8089 IOP writes an 0100H to local I/O port 8010H. A high on data line BDAT-8 and a low on write decoder line WDC10/ is then generated, causing interrupt latch U56 (3B5) to pull its output high and pull the selected interrupt line to the Multibus interface low. A 00H written to the system I/O port wake-up address, clears the interrupt (refer to Paragraph 4-14).

4-17. LOCAL MEMORY MAP

As was discussed in the Functional Overview, the 8089 IOP addresses the ROM, RAM, iSBX I/O ports and the disk communications side of the controller circuitry as local memory. Figure 4-8 shows a map of this local memory. The following paragraphs discuss the ROM, RAM and I/O ports.

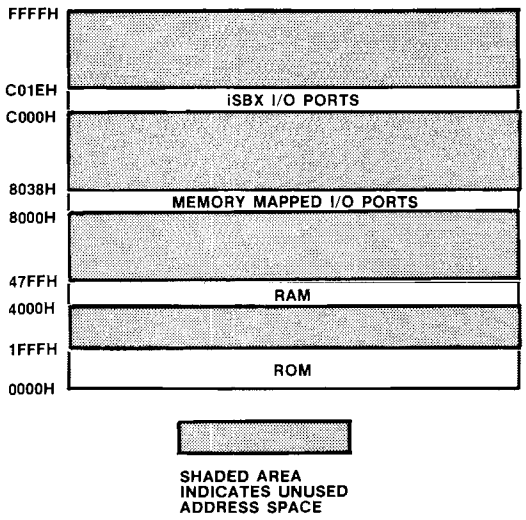


Figure 4-8. Local Memory Map

4-18. ROM

The controller ROM, which contains the 8089 IOP's disk control program, consists of two (4K x 8-bit) ROM devices, U87 and U88 (6X7). On any read from local memory in the range of 0000H to 1FFFH, chip select decoder U65 (5B4) decodes address lines IADR-E and IADR-F and pulls ROM chip-select line CSROM/ low, enabling the ROM devices.

4-19. RAM

The controller RAM consists of four (1K x 4-bit) RAM devices, U99 through U102 (6X4). On any read or write to local memory in the range of 4000H to 47FFH, chip select decoder U65 (5B4) pulls RAM chip-select line CSRAM/ low, enabling the RAM devices.

4-20. LOCAL MEMORY MAPPED I/O PORTS AND iSBX™ I/O PORTS

The 8089 IOP views the controlling devices in the disk control circuitry (such as ID comparators, counters, write buffer, read buffer, etc.) and the iSBX bus ports as local I/O ports, each with an address in local memory space. To enable one of the disk control devices, the 8089 executes a read or a write to the devices respective address. On any read or write to local memory in the range 8000H through 8038H, chip select decoder U65 (5B4) pulls its pin 10 low.

Table 4-3. Local I/O Ports

Address	Read (U33 Enabled)		Write (U32 Enabled)	
	Enable Line	Function	Enable Line	Function
8000H	RDC00/	Read Disk Status	WDC00/	Write control data to disk drive and enable AM SEARCH/, RDGATE and WRT GATE.
8008H			WDC08/	Clear index and ID not compare latches
8010H			WDC10/	Write to disk control register.
8018H	RDC18/	Raise 8089 Ch 2 CA input.	WDC18/	Write to Unit Select and Control register
8020H	RDC20/	Read contents of counter 2	WDC20/	Load counter 0
8022H	RDC20/	Read contents of counter 1	WDC20/	Load counter 1
8024H	RDC20/	Read contents of counter 2	WDC20/	Load counter 2
8026H			WDC20/	Write mode word
8028H	RDC28/	Read contents of read buffer	WDC28/	Write data to write buffer
8030H			WDC30/	Write sector ID to high comparator, start track format operation.
8038H			WDC38/	Write sector ID to low comparator

When this low on pin 10 of U65 is accompanied by a low on I/O read line I-IORC/, read I/O port address decoder U36 (5B2) is enabled; when the low on pin 10 of U65 is accompanied by a low on I/O write line I-AIOWC/, write I/O port address decoder U35 (5A2) is enabled. When enabled, U35 or U36 decode local memory address lines IADR-3 through IADR-5 to select the desired disk control device. Table 4-3 shows the address of each local I/O port and its function.

The two iSBX bus connectors, J3 and J4, on the iSBC 215 board provide access to the controller's iSBX bus. The iSBX bus provides 16 data lines and three address lines, providing a total of sixteen 16-bit I/O ports per connector. Each of these I/O ports has an address in local memory space (see Table 4-4).

Table 4-4. iSBX™ Bus I/O Port Addresses

Port	iSBX Bus Port Address Assignments			
	J3-Channel 0	J3-Channel 1	J4-Channel 0	J4-Channel 1
0	C070	C0B0	C0D0	C0E0
1	C071	C0B1	C0D1	C0E1
3	C073	C0B3	B0D3	C0E3
4	C074	C0B4	C0D4	C0E4
5	C075	C0B5	C0D5	C0E5
6	C076	C0B6	C0D6	C0E6
7	C077	C0B7	C0D7	C0E7

When the 8089 executes a read or a write to one of these ports, chip select decoder U65-9 (5B4) activates the CSMMIO/ line. Gates U30 (13C3) and inverter U31 (13C4) decode the CSMMIO/ and IADR-4 lines to select either J3 or J4. Address lines IADR-1, IADR-2 and IADR-3 are transmitted to connectors J3 and J4, pins 11, 9 and 7, respectively (5C1), to select the I/O port on the selected connector.

4-21. CONTROLLER TO DISK DRIVE COMMUNICATIONS

The following discussion provides a detailed functional description of the section of the iSBC 215 controller that communicates with the disk drive through the Winchester drive interface, and a description of the controllers interface with the iSBX bus through iSBX connectors, J3 and J4. The discussion is broken into four areas: (1) description of the disk interface and iSBX bus signals; (2) explanation of how the controller formats a disk prior to performing the read and write functions; (3) explanation of how writes and reads are performed; and (4) descriptions of the various circuits that perform the data transfer.

4-22. CONTROLLER TO WINCHESTER DISK DRIVE INTERFACE

All the signals that are transmitted between the controller board and the 8" Winchester disk drives are transmitted through either the Control Cable (J1) or the Read/Write Cable (J2). The physical configuration of these cables is described and illustrated in Chapter 2. All the signals transmitted between the drives except for the read, write and clock signals are TTL level. The read, write and clock signals are transmitted as differential signals.

The interface signals that the controller supports are described in the following paragraphs. Each of the drive manufacturers, Shugart/Quantum, Memorex, Priam and Pertec, use the available lines differently. For the specific use of the lines being employed, consult Figure 2-3 through 2-6 and the drive manufacturer's user manual.

4-23. CONTROL CABLE SIGNALS

Control and status information is exchanged between the controller and the drive through the Control Cable. Output signals are defined as those signals that the controller transmits and input signals as those the controller receives. The Control Cable is connected to J1 on the iSBC 215 board and goes to the first drive and up to three subsequent drives in a daisy chain fashion as shown in Figure 2-7. The functions of the 37 Control Cable lines can be divided into five categories:

1. Device Select (Output)
2. Head Select (Output)
3. General Purpose Data Bus (Bidirectional — Priam and Pertec Only)
4. Control (Output)
5. Status (Input)

Table 4-5 describes the function of each of the lines transmitted through the Control Cable.

Table 4-5. Control Cable Line Functions

Line Name	Function	Description
		DEVICE SELECT
US0/-US3/	Unit Select	Four lines; each selects one of four disk drive units.
		HEAD SELECT
HS0/-HS3/	Head Select	Four binary coded lines select one of sixteen heads in selected drive.
		GENERAL PURPOSE DATA BUS (Priam and Pertec Only)
BUS0/-BUS7/	Data Bus	Eight-bit, bi-directional data bus transmits command and status information between controller and drives. Data transmitted includes head and cylinder data.
		COMMAND DATA
WRGATE/	Write Select	Enables the write circuitry in drive, permitting write data that is sent to the drive through the Read/Write cable to be written on the selected disk surface. Used with AD MK EN/ line to write address mark on soft sectored disk.
RDGATE/	Read Select	Enables the read circuitry in drive, permitting data to be read from the selected sector of the disk. Used with AD MK EN/ to read address mark from soft sectored disk.
DIR/	Direction	Controls direction in which head is moved (Low = in, High = out) when stepping head positioner.
STEP/	Step Head	Initiates movement of head in direction that DIR/ has specified.
COMMAND/	Command Data	Indicates command data is present; used in bus cycle handshaking.
PARAMETER/	Parameter Data	Indicates parameter data is present; used in bus cycle handshaking.
DRIVE REQ/	Status Data	Indicates status data is present; used in bus cycle handshaking.
BUS ACK/	Bus Acknowledge	Acknowledges a bus cycle; used in bus cycle handshaking with commands, parameters and status.
AD MK EN/	Address Mark Enable	Enables writing or detecting of address marks (beginning of sectors) when used in conjunction with WRGATE/ and RDGATE/, respectively. Refer to SECTOR/ under status data.
FLT CLR/	Fault Clear	Clears FAULT/ line in selected drive. Signal has no effect if fault condition has not been corrected.
SAFE/	Controller Power Condition	Indicates to drive that power condition of controller is safe.
BA0/ and BA1/	Bus Address	Two binary coded lines specify source or destination register in selected drive for bus data.
		STATUS DATA
INDEX/	Index	Pulse received from selected disk drive once every disk revolution.
SECTOR/	Beginning of Sector	Signal indicates beginning of a sector: address mark for soft sectored disks, sector pulse for hard sectored disks.
FAULT/	Fault Condition	Indicates to controller that an unsafe condition has been detected in the selected drive, which would make the reliability of read/write operation questionable. Normally, logic in drive disables the read, write and positioning circuitry until rezero operation, fault clear or operator intervention occurs.
ILL ADR/	Illegal Address	Indicates drive has received an illegal cylinder address.
SK COM/	Seek Complete	Indicates to controller that selected drive has successfully completed the initial head load, seek operation, or rezero operation within drive specified time limits.
READY/	Drive Ready	Indicates that drive is powered up and is ready to receive or transmit data.

Table 4-5. Control Cable Line Functions (Continued)

Line Name	Function	Description
		STATUS DATA (Continued)
WR PRO/	Write Protected	Indicates that the selected drive is set for write protected operation. Controller is then inhibited from writing to the drive.
TRACK 0/	Track Zero	Indicates that heads of selected drive have been positioned to cylinder (track) zero.

4-24. READ/WRITE CABLE SIGNALS

Read Data, Write Data, Clocks, and two status lines constitute the information exchanged over the Read/Write cables. Output signals are defined as those signals that the controller transmits to the disk drives, and input signals those that the controller receives. For the Memorex or 14″ Shugart drives, the Read/Write cables are connected from the controller to the disk drive in radial fashion, that is one cable from the controller to each of the drives. J2 provides read, write and clock signals for two drives, for example, RD0 (+ and -) and RD1 (+ and -). One of these signals goes to physical address 0 and the other to physical address 1. When using 8″ Shugart, Quantum, Priam or Pertec drives, only the signals associated with physical address 0 are used. These signals are then daisy chained between drive units allowing the controller to communicate with up to four drives. Chapter 1 describes the cabling requirements for the various drive manufacturers. The physical configuration of these cables is explained and illustrated in Chapter 2. Table 4-6 describes the function of each of the lines transmitted through the Read/Write Cables. Note that the read, write and clock signals are differential signals, requiring two lines in the cable; the status lines are TTL level signals.

4-25. CONTROLLER TO iSBX™ CONNECTOR INTERFACE

All the signal and control lines transmitted between the controller and the iSBX bus are transmitted both through connectors J3 and J4. These lines are discussed only in general in this manual as they pertain to the remainder of the discussion of the controller interface with the Winchester drives. For a more detailed discussion of these lines refer to the *Intel iSBX™ Bus Specification,* Manual Order No. 142686.

It should be noted that the controller does not support any parallel-to-serial or serial-to-parallel conversion of data for transmission through the iSBX connectors. It interfaces with any device connected to these connectors through an 8 or 16-bit parallel bus and a number of control and handshake lines. The interface thus resembles the read/write port, made up of the write buffer and the read buffer, that is used in the controller interface to the Winchester drives.

The names in the schematic diagrams for the signal and control lines from the iSBC 215 Controller that are connected to iSBX connectors J3 and J4 often differ from the respective line name from the iSBX bus specifications. Table 4-7 lists both the iSBX bus mnemonic and the controller line name for each line in the iSBX bus that the controller supports.

Table 4-6. Read/Write Cable Line Functions

Line Name	Function	Description
WR0 and WR1 (+ and -)	Write Data	Write Data line pairs transmit serial NRZ data from the controller to the drive for recording on the disk surface. Write Clock synchronizes data transfer.
WRCL0 and WRCL1 (+ and -)	Write Clock	Write Clock line pairs transmit clock signal to drive that is used to synchronize write data transmission. Write Clock is derived from Read Clock, which the controller receives from the selected drive. Since the Read Clock is obtained from the rotating disk, it reflects any speed variations and thus ensures the proper bit rate transmission when writing as well as when reading.
RD0 and RD1 (+ and -)	Read Data	Read Data line pairs transmit serial NRZ data from the disk drive to the controller. The controller converts the differential signal into TTL levels for transmission to the host memory. The Read Clock synchronizes Read Data transfer.
RDCL0 and RDCL1 (+ and -)	Read Clock	Read Clock line pairs transmit clock signal to controller that is used to synchronize read data transmission and as a timing signal for the controller disk interface circuitry. Read Clock is derived from rotating disk.
SECT0/ and SECT1/	Beginning of Sector	Same as SECTOR/ signal transmitted to controller through Control Cable, one signal from each physical address.
SKCOM0/ and SKCOM1/	Seek Complete	Same as SKCOM/ signal transmitted to controller through Control Cable, one signal for each physical address.
RD WR CUR/	Reduced Write Current	Output signal used to control the write electronics for the inner tracks with higher bit densities.

Table 4-7. iSBX™ Bus Mnemonic-to-Controller Line Name

Pin	iSBX Bus Mnemonic	J3	J4	Pin	iSBX Bus Mnemonic	J3	J4
43	MD8	IDAT-8	IDAT-8	44	MD9	IDAT-9	IDAT-9
41	MDA	IDAT-A	IDAT-A	42	MDB	IDAT-B	IDAT-B
39	MDC	IDAT-C	IDAT-C	40	MDD	IDAT-D	IDAT-D
37	MDE	IDAT-E	IDAT-E	38	MDF	IDAT-F	IDAT-F
35	GND	GND	GND	36	+5V	+5V	+5V
33	MD0	IDAT-0	IDAT-0	34	MDRQT	DREQ0	DREQ1
31	MD1	IDAT-1	IDAT-1	32	MDACK/	N/C	N/C
29	MD2	IDAT-2	IDAT-2	30	OPT0	OP00	OP01
27	MD3	IDAT-3	IDAT-3	28	OPT1	OP10	OP11
25	MD4	IDAT-4	IDAT-4	26	TDMA	EXTR0	EXTR1
23	MD5	IDAT-5	IDAT-5	24			
21	MD6	IDAT-6	IDAT-6	22	MCS0/	CSMMIO0/	CSMMIO2/
19	MD7	IDAT-7	IDAT-7	20	MCS1/	CSMMIO1/	CSMMIO3/
17	GND	GND	GND	18	+5V	+5V	+5V
15	IORD/	I-IORC/	I-IORC/	16	MWAIT/	MWAIT0/	MWAIT1/
13	IOWRT/	I-AIOWC/	I-AIOWC/	14	MINTR0	INTR00	INTR01
11	MA0	IADR-0	IADR-0	12	MINTR1	INTR10	INTR11
9	MA1	IADR-1	IADR-1	10			
7	MA2	IADR-2	IADR-2	8	MPST/	M0PST/	M1PST/
5	RESET	PWR RST	PWR RST	6	MCLK	CCLK	CCLK
3	GND	GND	GND	4	+5V	+5V	+5V
1	+12V	+12V	+12V	2	-12V	-12V	-12V

All undefined pins are reserved for future use.

4-26. CONTROLLER TO DISK DRIVE INTERFACE TIMING

The following paragraphs provide a detailed discussion of the inter-circuit timing that occurs when formatting a disk, writing to a disk or reading from a disk. The discussion is provided to describe the interaction of the timing logic shown on Sheet 8 of the Schematic Diagram, with the disk drive interface receivers and drives shown on sheets 9 through 12 and the other data transfer circuitry described in Paragraphs 4-31 through 4-36.

4-27. DMA MODE

In general, when the controller is performing a read or a write function it locates the area of the disk where the read or write is to be performed, then enters its DMA mode to perform the actual transfer. (The process of locating the area to be read or written to is discussed in the following paragraphs.) In the DMA mode, the 8089 IOP (see Figure 4-2) controls the transfer of data between the local RAM block and the write and read buffers (called the read/write port). The data transfer circuitry on the controller board controls the transfer of data between the read/write port and the disk.

The RDY (Ready) line (8D1) is used for hand shaking between the 8089 and the data transfer circuitry. When RDY is low, the 8089 is quiescent; when RDY is high, the 8089 performs a DMA transfer of data either from local RAM to the write buffer (block-to-port) or from the read buffer to local RAM (port-to-block). Gates U40, U41 and U12 (8D3) control the RDY line.

To perform a write or a read, the 8089 executes firmware to set up data (write only) and condition the hardware for the selected operation. It then enters the DMA mode and attempts to transfer data. At this time: the TIME OUT line (8D8) is low; the MWAIT/ line is high; the R/W GATE line (8D1) is high (see Figure 4-9); U21-8 (8D3) is high, held so by the low on the ENBL XFER line (8D1); and the R/WDC 28 line, the output of U11-11 (8D7), is low. The low on R/WDC 28 is thus keeping RDY activated. On this first attempt to transfer data in the DMA mode, the 8089 activates either RDC 28/ or WDC 28/ (8D8), depending on whether a read or a write is being performed, respectively (refer to Paragraph 4-31). When RDC 28/ or WDC28/ is activated, the R/WDC 28 lines is activated, lowering RDY and putting the 8089 into its quiescent (wait) state. When the controller's data transfer circuitry has found the area on the disk where the read or write is to begin, it activates ENBL XFER (8D1). On

the next occurance of a Bit Ring-0 pulse, BR-0 (8D1), following the activation of ENBL XFER, U21-8 (8D3) is activated, activating RDY. The 8089 then immediately performs the data transfer (writes a word into the write buffer or reads a word from the read buffer) and lowers R/WDC 28. On the next clock into U21-11, U21-8 is raised. On the 8089's next attempt to perform a data transfer, R/WDC 28 is also raised, lowering RDY. The data transfer does not occur and the 8089 goes into its wait state. During this time, the SER/DES either transfers the word from the write buffer to the disk or reads another word from the disk into the read buffer. Then on the next BR-0 pulse, RDY is again activated and the next DMA data transfer occurs. The 8089 continues in this DMA mode until the R/W GATE line is lowered.

Note that two other lines have potential control over the RDY line. The TIME OUT line (8D8) is provided to allow the 8089 to be activated if a sector cannot be found on a cylinder. While the drive is searching for a sector, the RDY line is held low. If after two revolutions, the drive does not locate a sync byte, the time out line is raised. U41 (8D3) gates the TIME OUT signal through to U12 (8D1) and activates RDY.

The MWAIT/ line (8D8) is an iSBX Interface control line, derived from MWAIT0/ and MWAIT1/ (13D8). MWAIT/ exercises the same control over the RDY line as U40 (8D3) and can thus be used to set up a handshaking arrangement between an I/O controller connected to one of the iSBX interface connectors (J3 or J4) and the 8089. Refer to the discussion of the 8089 in the *8086 Family User's Manual* for a more detailed explanation of the various uses of the 8089 wait states.

4-28. DISK FORMATTING

Before the surfaces of a disk can be used for the writing and reading of data, the disk must be formatted. Formatting is the operation of writing all the address fields, gaps, ID headers, etc. for the complete disk. The controller performs this operation under software control. The software routine that controls this disk formatting operation allows only a single track to be formatted for each Format command. The host thus issues a new Format command to the controller board for each track to be formatted until the formatting of the entire disk is complete.

The implementation of the Format command is divided into two operations. During the first operation, address marks (soft sectored disks only), gaps and ID fields are written during a single disk revolution. During the second operation, data fields

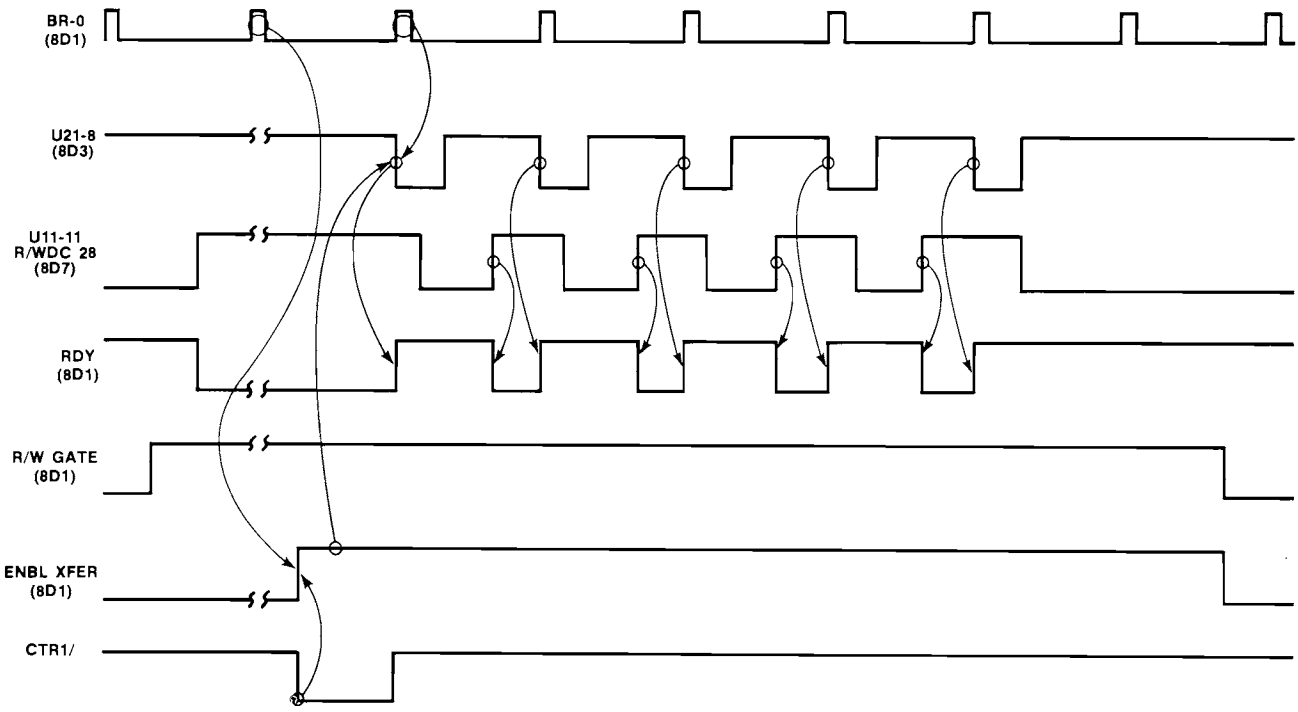


Figure 4-9. Timing Diagram for RDY Signal

are written (using the write data sequence described in Paragraph 4-31) with user supplied data. The second operation requires two disk revolutions, one to write the odd physical data fields (1, 3, 5, ...) and one to write the even physical data fields (0, 2, 4, ...). Three disk revolutions are thus required to format a single track. The hardware execution portion of the format operation is discussed in the following paragraphs. This discussion pertains to the formatting of a soft sectored disk. The iSBC 215 controller supports both soft and hard sectored disks.[1] The formatting procedure, however, is essentially the same. The differences are described at the end of this section, along with the slight differences in the sector format used with the Shugart/Quantum drives. When the Format command is issued to the controller, the 8089 IOP performs a seek to the desired track (cylinder) to begin the format operation.

[1] *A soft sectored disk (as used in Shugart/Quantum and Pertec drives) requires an address mark to be written at the beginning of each sector during the formatting operation. Hard sectored disks (as used in Memorex and Priam drives) provide a sector pulse at the beginning of each sector, thus address marks do not need to be written.*

When the heads are positioned over the selected track, the 8089 writes a C0H (for unit 0), a C8H (for unit 1), a D0H (for unit 2) and a D8H (for unit 3) to I/O port 8018 (decoded as WDC 18/). The activation of WDC18/ enables U3 (12A5) and activates the WRT GAT-F and FORMAT lines (12B1) and WRT GATE (12C1) (see Figure 4-10). WRT GAT-F and FORMAT enable the controller format control circuitry. The controller then writes all zeros to the drive while the 8089 waits for the receipt of the first INDEX/ pulse (11D8).

The receipt of INDEX/ sets latch U34 (11D6), which in turn sets bit F of the Status Register, U44 (11D5). To monitor the Status Register, the 8089 polls (reads) I/O port 8000H bit F (decoded as RDC 00/). Upon detecting Index, the 8089 writes a XXXXH to I/O port 3030H (decoded as WDC 30), which triggers U63 (8B7), activating the WRT AM/ line (8B1) and causing the first address mark to be written on the disk through the ADMKEN/ line (12D1).

The time that the 8089 allows between the detecting of Index and the activating of U63 (8B7) is approximately 11 byte times, which is the time the controller requires to perform a number of firmware steps in preparation for writing the first address mark and ID field, (see Figure 3-2 for a pictorial representation of the track format). During this time, the 8089

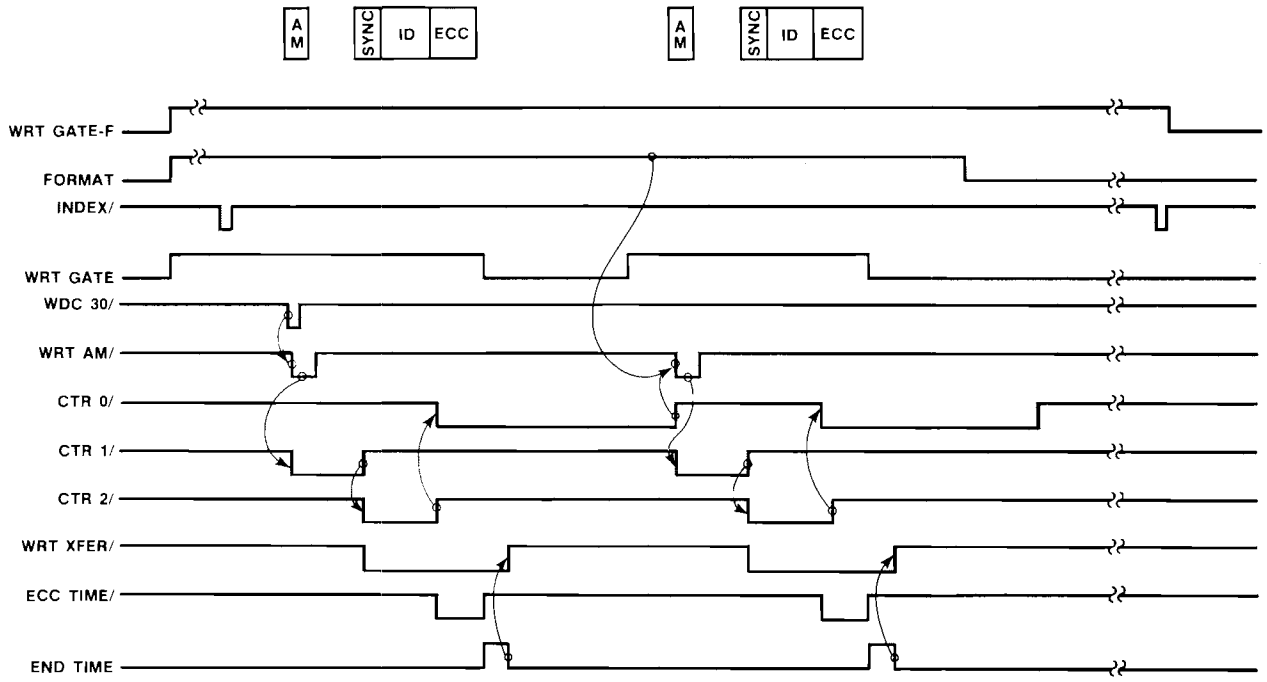


Figure 4-10. Timing Diagram for Disk Formatting Sequence

writes the sync byte (0019H) to the write buffer, U46 and U49 (7C7 and 7D7), by writing to I/O port 8028H (decoded as WDC 28/). It performs this operation in preparation for writing the ID field on the track.

The activation of WRT AM/ also starts counter 1, CTR 1 of U69 (8A7). (The 8089 preset the counters in U69 at the beginning of the format operation.) When CTR 1 times out at the end of 11 byte times, it activates the WRT XFER/ line through U63-7 (8C3), and starts CTR 2. The activation of WRT XFER/ initiates the 8089's DMA mode (as discussed in Paragraph 4-27), during which time the sync byte and the sector ID are written onto the disk. CTR 2 times out at the end of the ID field, starting CTR 0 and activating the ECC TIME line (8B1). During the ECC TIME, the ECC code from the ECC generator is written following the ID field (refer to Paragraph 4-34 for a description of the operation of the ECC generator). At the end of ECC TIME, the END TIME line is enabled, which lowers the WRT XFER/ line and takes the 8089 out of the DMA mode. After the last ID field is written, the FORMAT line is deactivated, which inhibits the writing of any additional address marks.

CTR 0 is set for a time equal to the ECC+G3+DATA+ G4, which the 8089 sets according to the sector size selected for the drive. When CTR 0 times out, it activates WRT AM/ and CTR 1, which begins the formatting of the second sector. This procedure is repeated until the 8089 determines that the last ID field has been formatted. The 8089 then begins searching for the Index pulse. Upon receipt of Index, the RST FRMT/ line is activated, resetting WRITE GATE-F and FORMAT, and inhibiting the writing of the next address mark. The 8089 then continues through the Format routine to the second operation, which is the writing of the data fields with user supplied data. The write data function, discussed in Paragraphs 4-29, describes the write data operation.

For hard sectored disks, a jumper is connected between terminals W16 1-3 (8B8). The formatting of the first sector thus begins when the first SECTOR/ pulse from the disk following the INDEX/ is received, rather than when WDC 30/ is activated. When the SECTOR/ line (11B8) is activated, it activates the INDEX-SECTOR/ line (11C1), which starts CTR 1. Formatting then continues in the same manner as with soft sectored disks, except that the beginning of the next sector occurs at the receipt of the next SECTOR/ pulse rather than at the timing out of CTR 0.

The 8" Shugart/Quantum drive sector format differs in two ways from that of the other three drive types. In the 8" Shugart/Quantum drives, an address mark is placed before both the ID field and the data field, with no gap between the address mark and the sync byte. In addition, a D9H is used for the sync byte in the data field rather than a 19H. When the controller sync byte detector circuit, U54, U68 and U73 (7B5), detects a sync byte (19 or D9) following an address mark and, the SR-6 (7B1) line is activated, (D9 only detected), the DATA SYNC and IDNCMPRL lines are activated through latch U37 (9A6). DATA SYNC and IDNCMPRL then set bits 3 and 6, respectively, of status register U10 (11C5) indicating to the controller the presence of the data field instead of an ID field. In the Memorex, 14" Shugart, Pertec and Priam drives, a data field is assumed to follow an ID field without an intervening address mark.

A second difference between the 8" Shugart/ Quantum drive and the other three drives is that with the 8" Shugart/Quantum drives, a 4EH pattern is written in the gaps rather than zeros. Inverters U58 and U17 (8D6) and gates U19 (8D5) creates the 4EH pattern. U40 and U60 (8A3) gate the pattern through to the SER/DES when the SHUGART and WRT GAT-F lines are activated during a format.

4-29. WRITE DATA TRANSFER

The write operation is divided into two steps: (1) read sector ID and (2) write data. When a write is initiated, the 8089 IOP writes 0006H to I/O port 8000H (decoded as WDC00). Latch U24 (12C5) then: activates the AM SEARCH/, ADMKEN/ and RD GATE/ lines, which enables the drive to search for the address mark and enables the controllers read circuitry (see Figure 4-11).

The 8089 has previously written to I/O port 8020H (decoded as WDC20/) to load counters 0, 1 and 2 of U69 (8A7). It also writes to I/O ports 8030H and 8038H (decoded as WDC30/ and WDC38/), loading the ID of the sector to be written to, into the 32-bit ID comparator logic.

When the address mark (or sector pulse) is detected, SECTOR/ is activated, which activates the AMFND-SECTOR/ line (11B1). The low on AMFND-SECTOR/ resets U34 (8C7) and deactivates the ID FIELD line. The low on the ID FIELD line, deactivates the AMMKEN/ line and activates the ALW SYNC SRCH, initiating the search for the sync byte. (Note that with the Shugart drives, the sync byte follows the address mark directly. The activating of AM FND-SECTOR/ thus activates ALW SYNC SRCH directly through jumper W14 1-2 (12C3).)

In searching for the sync byte, serial data from the disk is read into the SER-DES. Sync byte comparator U73 and U54 (7B5) monitors the outputs of the SER-DES and pulls the SYNC BYTE/ line (7B1) low

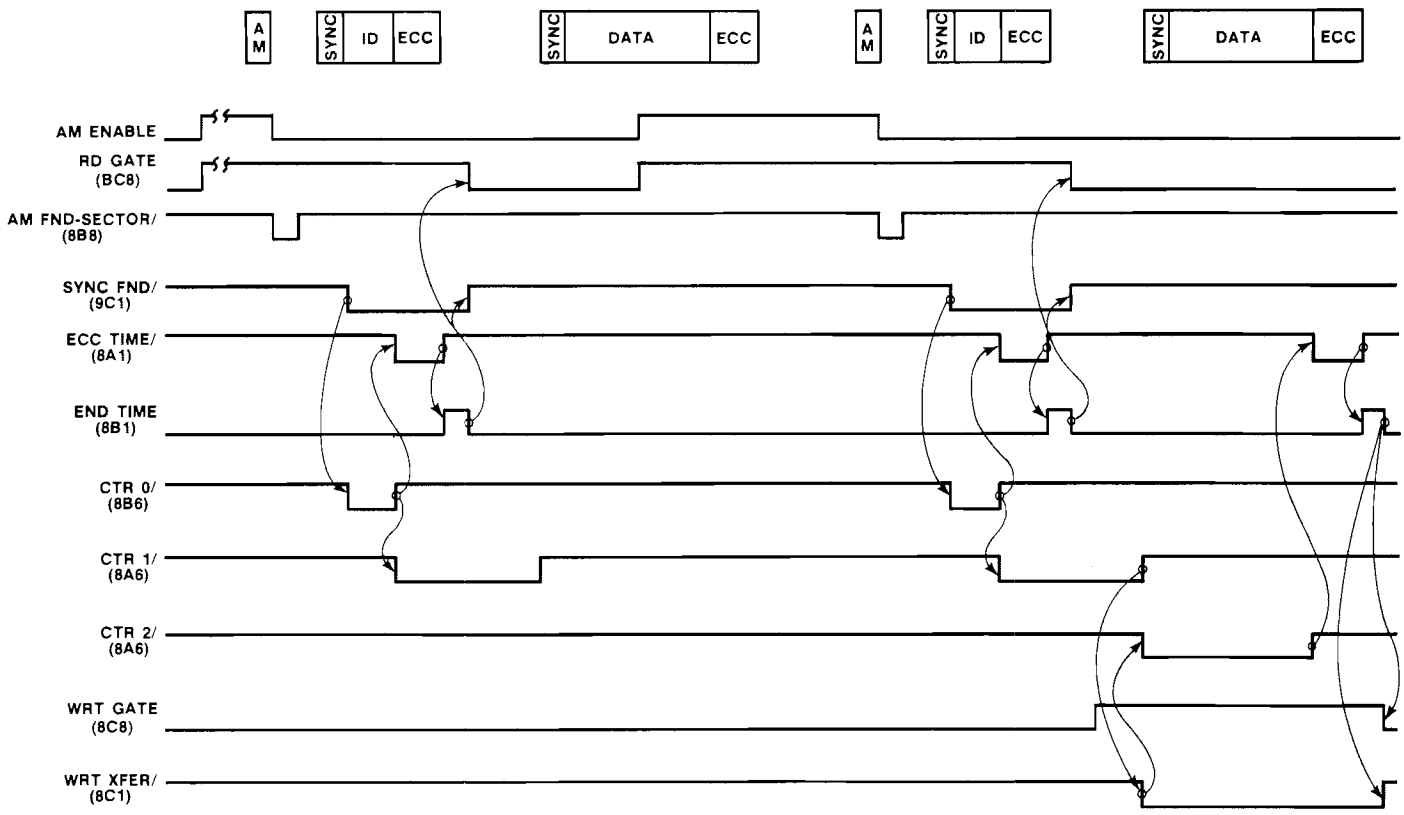


Figure 4-11. Timing Diagram for Write Data

when 19H — the sync byte — is detected. The enabling of SYNC BYTE/, enables the SYNC FND/ lines (9C1), which in turn activates the ID comparator U1, U2, U22 and U23 (9DX) and word clock U20 (8D7). (See the discussion of the Sync Byte Comparator Logic in Paragraph 4-32.)

SYNC FND/ also raises the ENBL XFER line, (8C1), which enables the ECC Generator logic (7AX) and Ready Latch U21 (8D4), and gates on counter 0 of U69 (8A7).

The 32-bit comparator (see Paragraph 4-33) compares the ID read from the disk with the ID of the selected sector. At the end of the ID time, counter 0 times out, pulling the ECC TIME/ line (7A8) low and initiating the ECC compare (see Paragraph 4-34). If the ID and the ECC are valid, bit 6 of the controller status register U10 (11C5) is reset. At the end of ECC time, U42-10 (8B2) activates the END TIME line which resets RD GATE. The 8089 then checks bit 6 of control status register U10 (11C5). If the bit is inactive, the 8089 continues with the write operation. If the ID or ECC are not valid (bit 6 active), the AM ENABLE and RD GATE lines are then reasserted and the controller searches for the next address mark.

To begin the second step of the write operation, the 8089 writes a 01H to I/O port 8000H (decoded WDC00/) and enables the write gate (WRT GATE), through U24 (12B5), enabling the drive's write circuitry. When counter 0 times out, counter 1 is started. Counter 1 is set for a time interval equivalent to the ECC time plus GAP 2. When counter 1 times out, counter 2 is started and the U63-7 (8C3) is set, activating WRT XFER/. WRT XFER/ enables write buffers U46 and U49 (7C7) and the ECC comparator logic (7AX), and raises the RDY line high indicating to the 8089 that the write buffer is ready to receive data.

The 8089 then enters its DMA mode to write data from local RAM to the disk (see the discussion of the DMA mode in Paragraph 4-27). The controller continues transferring data to the disk in this manner until Counter 2 times out, indicating the end of the data field, and raises the ECC TIME line. With the ECC TIME line activated, the ECC generated during the data transfer is written to the disk. END TIME then terminates the write operation.

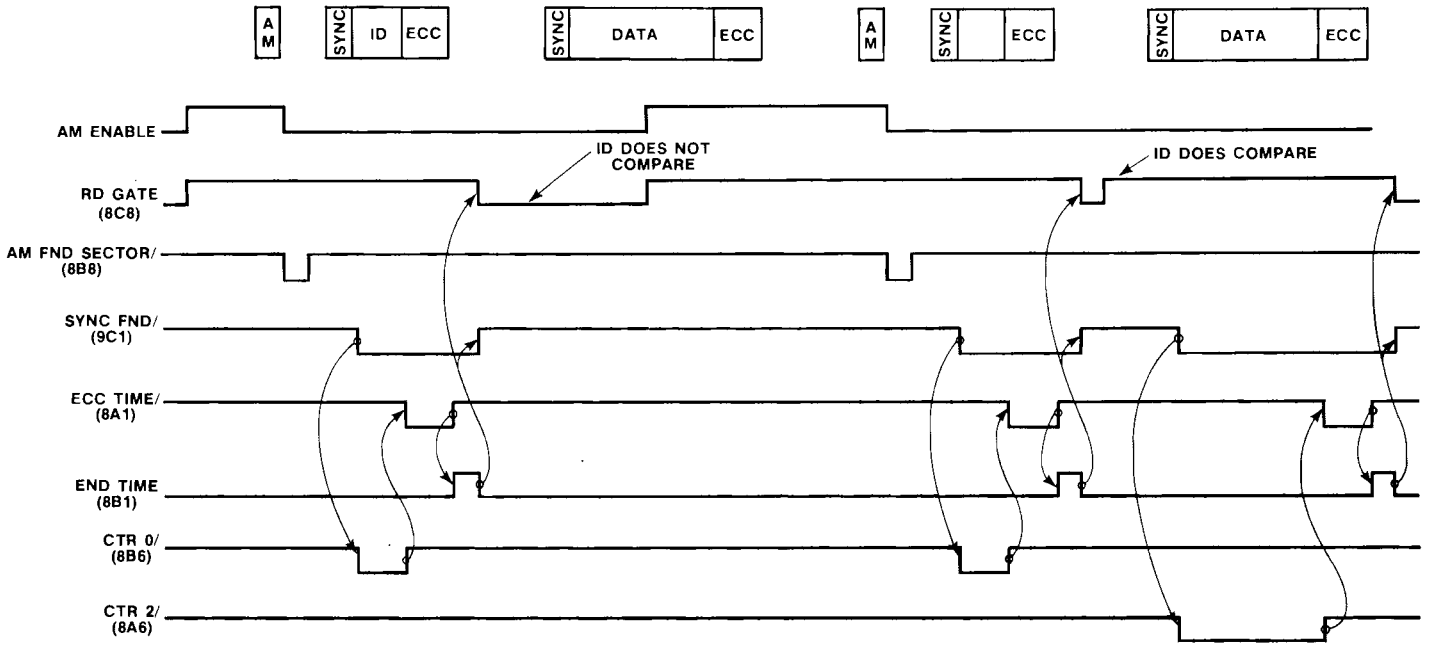


Figure 4-12. Timing Diagram for Read Data Transfer

4-30. READ DATA TRANSFERS

The read operation is divided into two steps: (1) read sector ID and (2) read data. The reading of the sector ID is performed in the same manner as for the write operation (see Figure 4-12).

When the desired sector is located, the RD GATE is again raised to search for the sync byte of the data field. When SYNC FND/ is activated, counter 2 is started through U61-8 (8C4) and U59 (8B6), the ECC generator is enabled and the RDY line is activated, initiating the DMA read data transfer mode. Data is then transferred from the disk to local RAM for the duration of counter 2.

When counter 2 times out, ECC TIME is activated. Following ECC TIME, END TIME is raised, terminating the read operation.

4-31. SER/DES LOGIC

The serial/deserialize logic performs two functions: (1) converts parallel data words into a serial string of bits to be sent to the disk drive during a write operation, and (2) converts a serial string of bits into 16-bit words during a read operation. The SER/DES logic is made up of Write Buffer U46 and U49 (7C7), SERializer/DESerializer U47 and U50 (7C5), Read Buffer U48 and U51 (7C4), and Selector U70 (7A7).

During a write operation (WRT XFER/ low), the 8089 IOP writes to I/O port address 8028H. Write I/O port address decoder U35 (5A2) decodes this address and pulls WDC28/ low, clocking the data to be written to the disk (BDAT-0 through BDAT-F) into write buffer U46 and U49 (7C7). A high on load serial register line LDSR (7C6), derived from word clock U20 (8C7) loads the contents of the write buffer (SR-0 through SR-F) into the SER/DES (7C5). Read/write clock R/W CLK-B (7B8) then clocks the data bit by bit through the QH' output of U50 (7D5), and through selector U70 (7A7) to the WRT DATA line. R/W CLK-A clocks the serial data string on WRT DATA through U18 (10C3) to the selected drive.

During a read operation, the R/W CLK-B (10B1) gates the serial data string (RD DATA) from the disk drive through U18 (10B4) and selector U70 (7A7) and into the SI input of U47 (7C5), creating a 16-bit parallel word. Bit ring-0 line BR-0 (7B8) then clocks this word into read buffer U48 and U51 (7C4). BR-0 is derived from word clock U20 (8C7). With the read buffer loaded, the 8089 initiates a read to I/O port address 8028H. Read I/O port address decoder U36 (5B2) decodes this address and pulls RDC28/ low, which clocks the data word from the read buffer onto internal data bus IDAT-0 through IDAT-F.

4-32. SYNC BYTE COMPARATOR LOGIC

The sync byte comparator detects the presence of a sync byte during a read operation and synchronizes word clock U20 (8C7) with the data. A sync byte is written preceding each sector ID and each data field to indicate to the controller that data to be read is forthcoming (see Figure 3-2). The sync byte value is always 19H except for the Shugart/Quantum drives, which use a D9H for data fields.

During a read operation, sync byte decoder U54 and U73 (7B5) monitors the output of the SER/DES, U47 and U50 (7C5). When a 19H is detected, SYNC BYTE/ goes low indicating the presence of the sync byte. SYNC BYTE/ and the next output of R/W CLK-B set the SYNC FND flip-flop, U57 (9C6). SYNC FND activates word clock U20 (8C6), and activates the read/write logic (sheet 8). A further explanation of the sync byte logic can be found in Paragraphs 4-29 through 4-31.

4-33. 32-BIT ID COMPARATOR LOGIC

The 32-bit ID comparator logic compares the sector ID of the record being searched for with the sector ID being read from the disk drive. The sector ID is made up of the flags, cylinder number, sector number and head address.

To load the sector ID of the record being searched for into 32-bit ID comparator U1, U2, U22 and U23 (9DX), the 8089 IOP writes to I/O ports 8030H, enabling the WDC30/ and WDC38/ lines, respectively. WDC30/ and WDC38/ initiate the loading of the sector ID into the ID comparator. This loading occurs prior to performing either a read or write data operation. The ID compare operation begins after the sync byte of an ID field has been detected (SYNC FND). R/W CLK-B clocks the ID information, which is stored in the ID comparator, out of U22 (pins 7 and 9) bit by bit. U26 (9D2) compares the serial string of bits with the sector ID from the disk drive (RD-DATA). If the two sector IDs differ, ID no-compare line ID NCMPR/ is activated; if they are the same, ID NCMPR/ is raised. Selector U70 (7A7) ORs the ID NCMPR/ and the ECC NCMPR/ lines (see Paragraph 4-37). The resulting ID-ECC NCMPR/ lines is latched into U37 (9B6). The Q/ output of U37, ID NCMPR-L, is transmitted to bit 6 of status register U10 (11C5). The 8089 IOP then reads the contents of the status register and checks the condition of bit 6. Bit 6 being set high indicates that the record read from the disk was either not the record being searched for or had an ECC error; conversely, bit 6 being set low indicates that the ID field compared and that there was not an ECC error. The 8089 IOP can then read or write the data portion of the record.

4-34. ECC GENERATOR LOGIC

The error checking code (ECC) logic performs two functions: (1) during a write operation, it generates a four byte ECC polynomial that is appended to the ID field (format operation only) and the data field (normal write) of a record (see Figure 3-2), (2) during a read operation, it regenerates the ECC polynomial and compares it to the ECC field read from the disk record to ensure that the correct data was read from the drive.

During a write operation, serial data (either an ID field or a data field) is transmitted from the SER/DES (7C5) through selector U70 (7A7) and into the ECC generator through pins 1 and 2 of U103 (7A6), where the ECC polynomial is generated. At the same time a high on the WRT XFER DLYD line (7B8), transmitted through gate U68 (7B4), enables the serial data to be transmitted through U71 (7A2) and selector U70 (7A7) to the WRT DATA line, where it is transmitted to the disk. At ECC time (end of data field), WRT XFER DLYD goes low, inhibiting write data from being transferred through gate U68 (7B4). The ECC TIME/ line goes low, causing the ECC polynomial to be written onto the disk through U71 (7A3), U70 (7A7) and the WRT DATA line.

During a read operation, serial data (again either a sector ID or a data field) is read into the ECC generator through selector U70 (7A7) and into the SER/DES through U71 (7A3) and U70. At ECC time, U71 compares the ECC polynomial from the ECC generator bit by bit with the ECC polynomial from the disk and transmits the difference through U70 to the SER/DES for storage in RAM. If the difference is zero, the ID-ECC NCMPR/ line is pulled high indicating correct data or sector ID (Paragraph 4-33). If the result of the comparison is non-zero, the difference is called the error syndrome. The 8089 uses syndrome to correct errors in a sector ID or data field (if correctable).

4-35. STATUS REGISTER LOGIC

Status register U10 and U44 (11X5) and U9 (11B3) transmit status information from the selected disk drive, the iSBX interface and various lines within the controller disk interface circuitry to the controller. When the 8089 IOP issues a Read Status command, or checks status as an internal operation, read decode enable lines RDC 00/ and RDC 08/ are acticated, causing the contents of status registers U10 and U4, and U9, respectively to be transferred onto the internal bus (IDAT-8 through IDAT-F). The 8089 then analyzes the status information and either uses it for an internal operation or communicates the

Table 4-8. Status Register Bits

Bits	8000H (Upper Byte) U44 (11D5)	Function 8000H (Lower Byte) U10 (11C5)	8008H (Lower Byte) U9 (11B3)
F	Index		
E	Drive Request		
D	Illegal Address		
C	Option Bit 10*		
B	Option Bit 00*		
A	Interrupt 10*		
9	Interrupt 00*		
8	iSBX Board Present at J3*		
7		Time Out	Write Protected
6		ID No Compare	Track Zero
5		Bus Acknowledge	Vendor
4		Fault	Option Bit 11*
3		Data Sync	Option Bit 01*
2		Seek Complete	Interrupt 11*
1		Ready	Interrupt 01*
0		0	iSBX Board Present at J4*

*iSBX Bus lines.

status of the data transfer operation to the host processor through system memory (Controller Invocation Block). Table 4-8 lists the status register bits. Refer to Chapter 3 for information on the status information transmitted to the host.

4-36. LINE DRIVERS AND RECEIVERS

All the serial data and high speed clock signals transmitted between the controller and the disk drive use differential pair line drivers and receivers. The polarity on these lines is positive true logic i.e., when the + side of the line is more positive than the - side of line, a positive logic "1" is being transmitted.

The controller's differential drivers, U16 (10X3) are referenced to 0 volts and +5 volts. The controller's receivers that receive differential signals from the Memorex, 14" Shugart, Pertec and Priam drives, U13 (10X6), are also referenced to 0 volts and +5 volts. The receivers for the 8" Shugart and Quantum drives receive differential signals, U15 (10X5), are referenced to -5 volts and +5 volts.

CHAPTER 5
SERVICE INFORMATION

5-1. INTRODUCTION

This chapter provides service and repair assistance instructions, service diagrams, a complete electronic parts list for the printed circuit board assembly and a reference to the controller's self diagnostic.

5-2. SERVICE DIAGRAMS

The controller board jumper and component locations, and schematic diagrams (Figure 5-1 through 5-3) are included at the end of this chapter. Note that these diagrams are intended only for reference; they reflect the iSBC 215 controller design at the time this manual was printed. The schematics and component location diagrams packaged with the controller reflect the design version shipped and thus supercede the diagrams in this manual.

5-3. SERVICE AND REPAIR ASSISTANCE

United States customers can obtain service and repair assistance from Intel by contacting the Intel Product Service Hotline in Phoenix, Arizona. Customers outside the United States should contact their sales source (Intel Sales Office or Authorized Distributor) for service information and repair assistance.

Before calling the Product Service Hotline, you should have the following information available:

a. Date you received the product.

b. Complete part number of the product (including dash number). On boards, this number is usually silk-screened onto the board. On other MCSD products, it is usually stamped on a label.

c. Serial number of product. On boards, this number is usually stamped on the board. On other MCSD products, the serial number is usually stamped on a label.

d. Shipping and billing addresses.

e. If your Intel Product warranty has expired, you must provide a purchase order number for billing purposes.

f. If you have an extended warranty agreement, be sure to advise the Hotline personnel of this agreement.

Use the following numbers for contacting the Intel Product Service Hotline:

Telephone:

All U.S. locations,
Except Alaska, Arizona, & Hawaii

(800) 528-0595

All other locations: (602) 869-4600

TWX Number:

910 - 951 - 1330

Always contact the Product Service Hotline before returning a product to Intel for repair. You will be given a repair authorization number, shipping instructions, and other important information which will help Intel provide you with fast, efficient service. If you are returning the product because of damage sustained during shipment or if the product is out of warranty, a purchase order is required before Intel can initiate the repair.

5-4. SELF DIAGNOSTIC

A self diagnostic is provided with the iSBC 215 controller, stored in the on-board PROM. It performs a go/no-go test of the controller hardware and firmware. If the controller passes the test, it indicates with a high degree of certainty that the controller is operating properly. See the discussion of the diagnostic in Chapter 3 for a description of the program and instructions for initiating the operation.

5-5. REPLACEABLE COMPONENTS

This section contains the information necessary to procure replacement components directly from commercial sources. Component manufacturers have been abbreviated in the parts list with a two to five character code. Table 5-1 cross-references the manufacturer's code with the name and location of the prime commercial source. Table 5-2 lists all the replaceable components on the controller board. Note that the components that are available commercially are listed in the "MFR CODE" column as "COML" and that they are ordered by description (OBD). Procure commercially-available components from a local distributor whenever possible.

Table 5-1. Code for Manufacturers

Mfr. Code	Manufacturer	Location
BECK	Beckman Instruments Inc.	Fullerton, CA
BOUR	Bourns, Inc.	Riverside, CA
CRYST	Crystek	Ft. Meyers, FL
CTSK	CTS Keene, Inc.	Paso Robles, CA
DALE	Dale Electronics	Columbus, NE
FAI	Fairchild Semiconductor	Mt. View, CA
INTEL	Intel	Santa Clara, CA
MOT	Motorola	Phoenix, AZ
SNGMO	Sangamo-Weston, Inc.	Pickens, SC
SPEC	Spectrol Electronics Corp.	City of Industry, CA
SPRG	Sprague Electronic Co.	Adams, MA
3M	3M Co.	St. Paul, MN
TI	Texas Instruments	Dallas, TX
VIK	Viking Industries, Inc.	Chatsworth, CA
COML	Any Commercial Source; Order By Description (OBD)	

Table 5-2. Controller Board Electrical Parts List

Reference Designation	Description	Mfr. Part No.	Mfr. Code	Qty.
C1, C2	Capacitor, 22μF, Tant, ±10%, 15V	150D226X9015B2	SPRG	2
C3	Capacitor, 2.2μF, Tant, ±10%, 20V	150D225X9020A2	SPRG	1
C4	Capacitor, 0.33μF, Cer. Z5U	OBD	COML	1
C5	Capacitor, 10μF, Tant, ±10%, 20V	150D106X9020B	SPRG	1
C6	Capacitor, 10pF, Mica, ±5%,	D15-5C100J03	SNGMO	1
C7 through C12 C14 through C44	Capacitor, 0.10μF, Cer. Z5U	OBD	COML	37
J1	Connector, Header 50 Pin	3433-1302	3M	1
J2	Connector, Header 40 Pin	3432-1302	3M	1
J3, J4	Connector, 44 Pin	68-369	VIK	2
RP1	Resistor Pack, 220/330 Ω, 10 Pin	765-5-R220/330	BECK	1
RP3	Resistor Pack, 100 Ω, 8 Pin	764-3-R100	BECK	1
RP4	Resistor Pack, 56 Ω, 6 Pin	763-1-R56	BECK	1
RP5, RP7 through RP13	Resistor Pack, 10 kΩ, 8 Pin	764-1-R10K	BECK	8
RP6	Resistor Pack, 220/330 Ω, 8 Pin	764-5-R220/330	BECK	1
R1, R4, R7 through R9, R13, R14	Resistor, Carb., 10 KΩ, ¼W, ±5%	OBD	COML	7
R2, R3, R6, R12, R15, R16	Resistor, Carb., 270 Ω, ¼W, ±5%	OBD	COML	6
R5	Resistor, Carb, 100 kΩ, ¼W, ±5%	OBD	COML	1
R10, R11	Resistor, Carb, 680 Ω, ¼W, ±5%	OBD	COML	2
S1	Switch, 8 Position, DIP	206-08LPST	CTSK	1
S2	Switch, 10 Position, DIP	206-10LPST	CTSK	1
U1, U2, U22, U23	IC, 8 Bit Shift Reg.	SN74LS165N	TI	4
U3	IC, Octal, D Type, Flip-Flop	SN74LS273N	TI	1
U4 through U6, U81 through U83	IC, Octal Latch, Inverting	8283	INTEL	6
U7, U27	IC, Quad Driver, Inverting, OC	7438		2
U8	IC, Dual 4 to 1 Selector/MUX	SN74LS153N	TI	1
U9, U85, U86	IC, Octal D Type Latch	SN74LS373N	TI	3

Table 5-2. Controller Board Electrical Parts List (Continued)

Reference Designation	Description	Mfr. Part No.	Mfr. Code	Qty.
U10, U44, U46, U48, U49, U51	IC, Octal D Type Flip-Flop	SN74LS74N	TI	6
U11, U61, U68,	IC Quad 2 Input NAND	SN74LS00N	TI	3
U12, U29, U59, U72	IC, Quad 2 Input AND	SN74LS08N	TI	4
U13	IC, Quad Line Receiver	3486		1
U14	IC, Dual Line Receiver	75107A	TI	1
U15, U89	IC, Quad 3 State Buffer	SN74LS125N	TI	3
U16	IC, Quad Line Driver	3487	TI	1
U17	IC, Hex Inverter	SN74S04N	TI	1
U18	IC, Dual Pos. Edge Trig. Flip-Flop	SN74S4		1
U19, U26	IC, 2 Wide, 3 in, 2 in, AND-OR-INV	SN74LS51N	TI	2
U20	IC, 4 Bit Binary Counter	SN74LS161N	TI	1
U21, U37, U56, U57, U62	IC, Dual Pos Edge Trip. Flip-Flop	SN74LS74N	TI	6
U24	IC, Quad D Type Flip-Flop	SN74LS175N	TI	2
U25, U28	IC, Hex Inverter	SN74LS04N	TI	2
U30, U32, U38, U41, U67, U76, U92	IC, Quad Input OR	SN74LS32N	TI	7
U31	IC, Hex Schmidt Trigger	SN74LS14N	TI	1
U33, U73	IC, Quad 2 Input NOR	SN74LS02N	TI	2
U34, U63	IC, Quad R-S Type Latch	SN74LS279N	TI	2
U35, U36	IC, 3 to 8 Decoder	SN74LS138N	TI	2
U40, U75	IC, Tri 3 Input NAND	SN74LS10N	TI	2
U42	IC, Hex Type Flip-Flop	SN74LS743N	TI	1
U43, U45, U93 through U95	IC, Octal Three State Buffer	SN74LS244N	TI	5
U47, U50	IC, 8 Bit Shift/Storage Register	SN74LS299N	TI	2
U52, U53	IC, Octal Bus Transceiver	8286	INTEL	2
U54	IC, Dual 4 Input NAND	SN74LS20N	TI	1
U55	IC, Clock Generator	8284A	INTEL	1
U58, U74	IC, Hex Inverting Buf/Drvr	SN74LS06N	TI	2
U60	IC, Quad 2 Input NOR	SN74S02N	TI	1
U65	IC, Dual 2 to 4 Line Decoder	SN74LS139N	TI	1
U66	IC, 13 Input NAND	SN74LS133N	TI	1
U69	IC, Programmable Counter/Timer	8253-5	INTEL	1
U70	IC, Quad 2:1 MUX	SN74LS257N	TI	1
U71	IC, 9 Bit Parity Generator	SN74LS280N	TI	1
U77 through U80	IC, Quad 2 Input XNOR OC	SN74LS266N	TI	4
U84	IC, Input/Output Processor	8089	INTEL	1
U87	IC, PROM	Open Loop (Low Byte)	INTEL[1]	1
		Closed Loop (Low Byte)	INTEL[1]	1
U88	IC, PROM	Open Loop (High Byte)	INTEL[1]	1
		Closed Loop (High Byte)	INTEL[1]	1
U90	IC, Bus Arbiter	8289	INTEL	1
U91	IC, Bus Controller	8288	INTEL	1
U96 through U98	IC, Octal Bus Transceiver, Invert.	8287	INTEL	3
U99 through U102	IC, Static RAM	2114-5	INTEL	4
U103 through U106	IC, 8 Bit Shift Register	SN74LS164N	TI	4
VR1	Voltage Regulator, -5V	MC7905CT	MOT	1
Y1	Crystal, 15.000 MHz	Type 44 Miniature HC454	CRYST	1

[1] Call Intel Product Service Hotline for current part number

NOTES CONTINUED FROM SHEET 1 OF 14

[3] WHEN THE SHUGART DRIVE IS USED, A -5 VOLTAGE NEEDS TO BE SELECTED AS WELL. THE FOLLOWING JUMPERS ALLOW THE CONTROLLER TO DEVELOP A -5 VOLT SUPPLY. REFER TO ZONE 1C6 AND 1C7.

W 21 1-2 -5 VOLTS FROM MULTIBUS
1-3 -5 VOLTS FROM REGULATOR
(REQUIRES W20 TO BE SELECTED)

W 20 1-2 -10 VOLTS INPUT TO REGULATOR
1-3 -12 VOLTS INPUT TO REGULATOR

[4] SBX BOARD CONTROL:
FOUR JUMPERS ARE PROVIDED FOR SBX BOARD CONTROL SEE THEIR RESPECTIVE SPECIFICATIONS FOR USE OF W3 AND W4, W11 AND W12. IF A SBX CONNECTOR IS NOT USED, THEN ITS CORRESPONDING W3 PR W4 MUST BE INSTALLED.

J3 - W3 - 1-2
J4 - W4 - 1-2

[5] PRIORITY ARBITRATION SELECT:
W18 IS USED TO SELECT WHICH TYPE OF MULTIBUS ARBITRATION TO USE. W18 -1-2 IS THE NORMAL MODE WHERE THE CONTROLLER ARBITRATES ON A PRIORITY BASIS. W18-1-3 SELECTS A MODE IN WHICH THE CONTROLLER SURRENDERS THE MULTIBUS FOR ANY REQUEST IRRESPECTIVE OF ANY PRIORITY. REFER TO ZONE 3C7.

[6] COMMON BUS REQUEST SELECT W23 IS USED TO ALLOW THE CONTROLLER TO MAINTAIN CONTROL OF THE MULTIBUS OR TO SURRENDER THE BUS UPON COMPLETION OF EACH BUS ACCESS. W23-1-2 IS THE NORMAL MODE. W23-1-3 DIRECTS THE CONTROLLER TO SURRENDER THE BUS ON EACH ACCESS. FOR OTHER BUS MASTERS WHICH DO NOT DRIVE CBRQ/, W23 MUST BE WIRED W23-1-3. REFER TO ZONE 3D7.

TABLE I

DRIVE VENDOR SELECTION					
W-	SHUGART [3]	MEMOREX	PERTEC	PRIAM	LOCATED BY 1C
1	1 - 3	1 - 3	1 - 2	1 - 2	U28
2	-	1 - 2	-	1 - 2	U11
3					U32 [4]
4					U38
5	1 - 3	1 - 2	1 - 2	1 - 2	U13
6	1 - 3	1 - 2	1 - 2	1 - 2	U13
7	1 - 3	1 - 2	1 - 2	1 - 2	U13
8	1 - 3	1 - 2	1 - 2	1 - 2	U13
9	1 - 2	-	-	-	U20
10	1 - 2	-	1 - 2	1 - 2	U20
11					U24 [4]
12					U24
13	1 - 2	1 - 3	1 - 2	1 - 3	U25
14	1 - 2	1 - 3	1 - 3	1 - 3	U29
15	-	1 - 2	1 - 2	1 - 2	U42
16	1 - 2	1 - 3	1 - 2	1 - 3	U62
17	1 - 2	1 - 2	1 - 2	-	U59
18					U50 [5]
19					U94 — See Table II
20					U98 [3]
21					U84
22	1 - 2	1 - 2	1 - 3	1 - 2	U18
23	1 - 2	1 - 2	1 - 2	1 - 2	U92 [6]

TABLE II

INTERRUPT LEVEL SELECTION (W19)		
LEVEL SELECTED	JUMPERS FROM	REQUIRED TO
0	C	0
1	C	1
2	C	2
3	C	3
4	C	4
5	C	5
6	C	6
7	C	7

SEE SHEET 1

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 14 of 14)

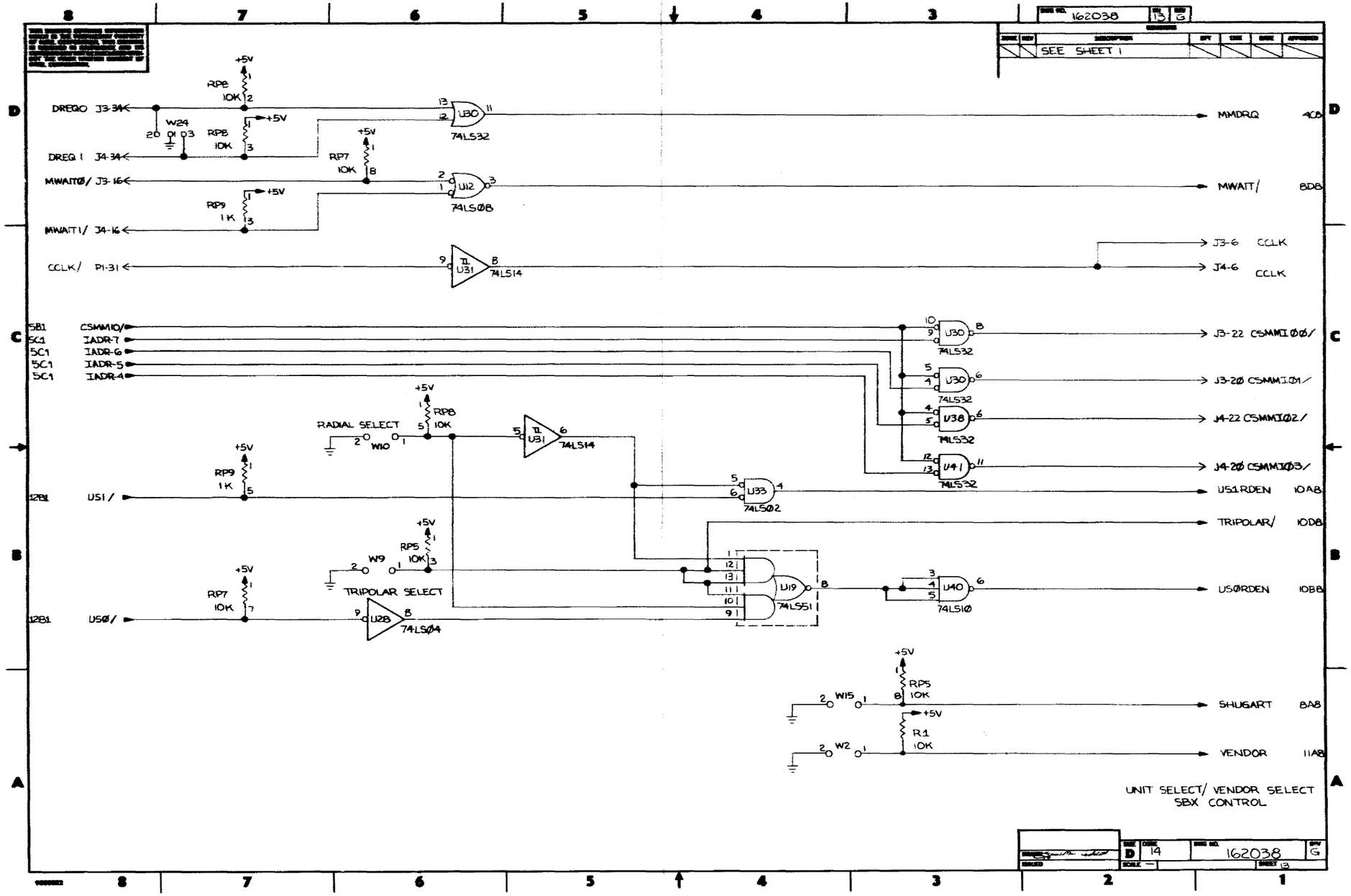


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 13 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

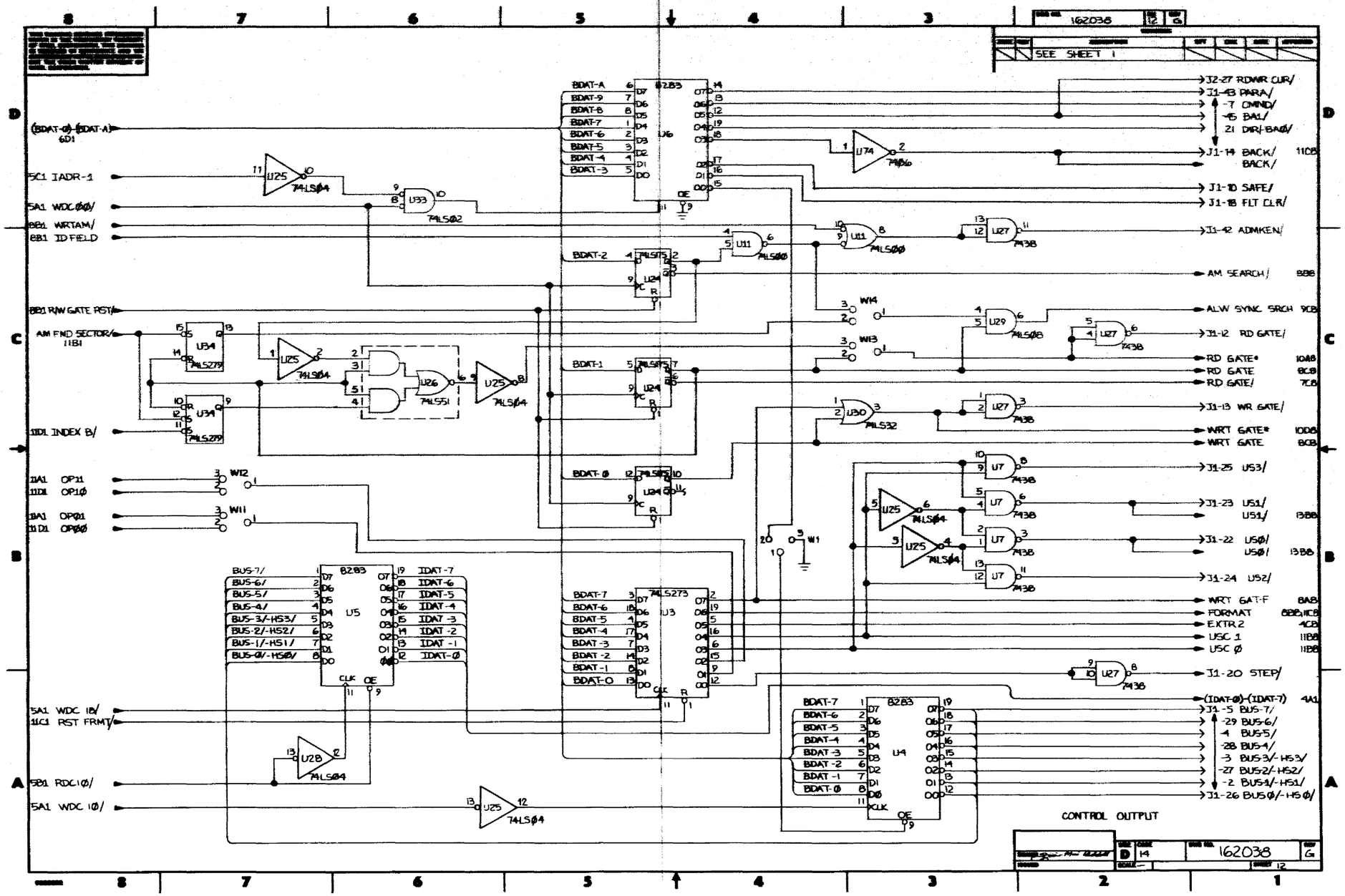


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 12 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

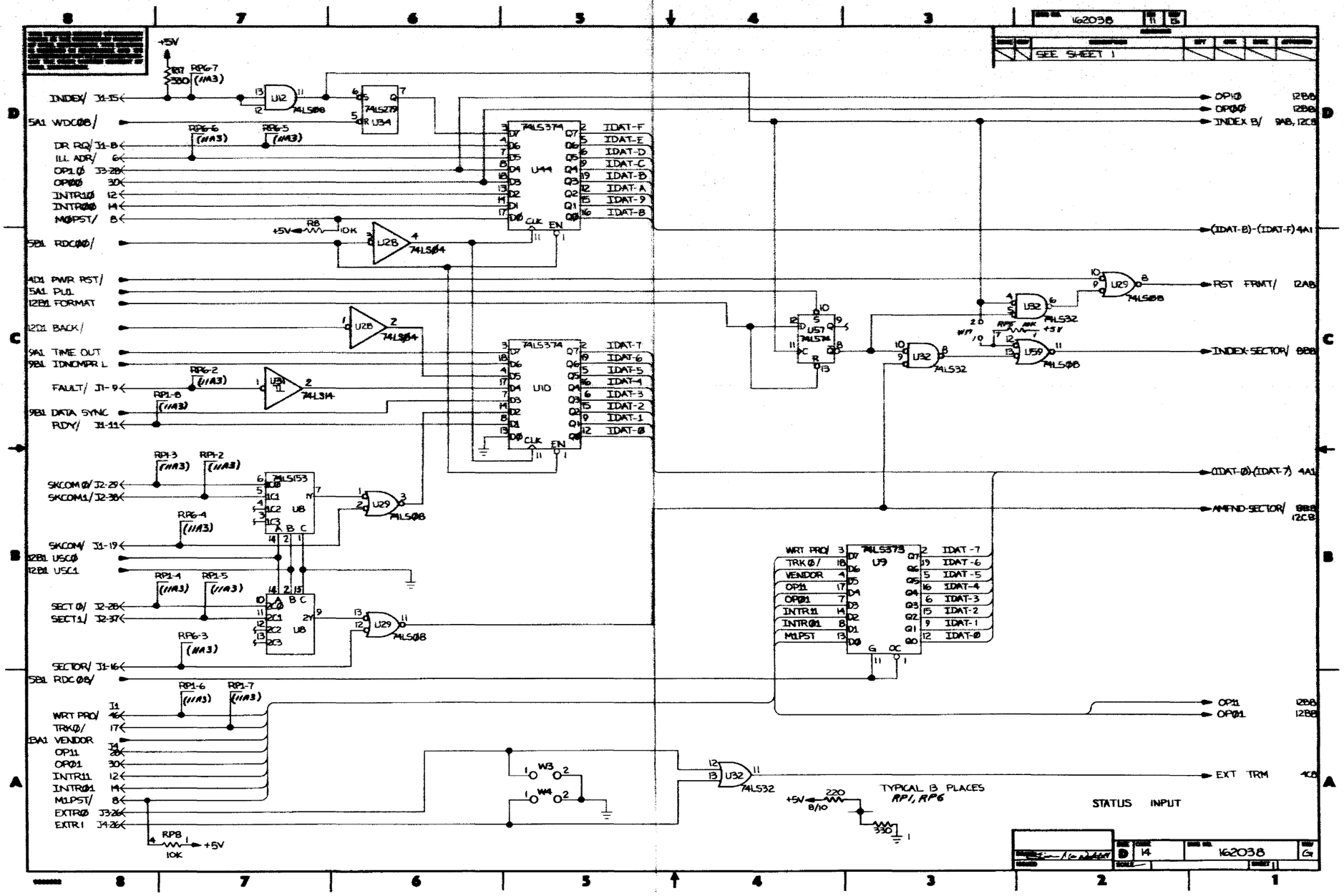


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 11 of 14)

CAUTION: Theee schematic diagrams may have been revised. See "Service Information" chapter for details.

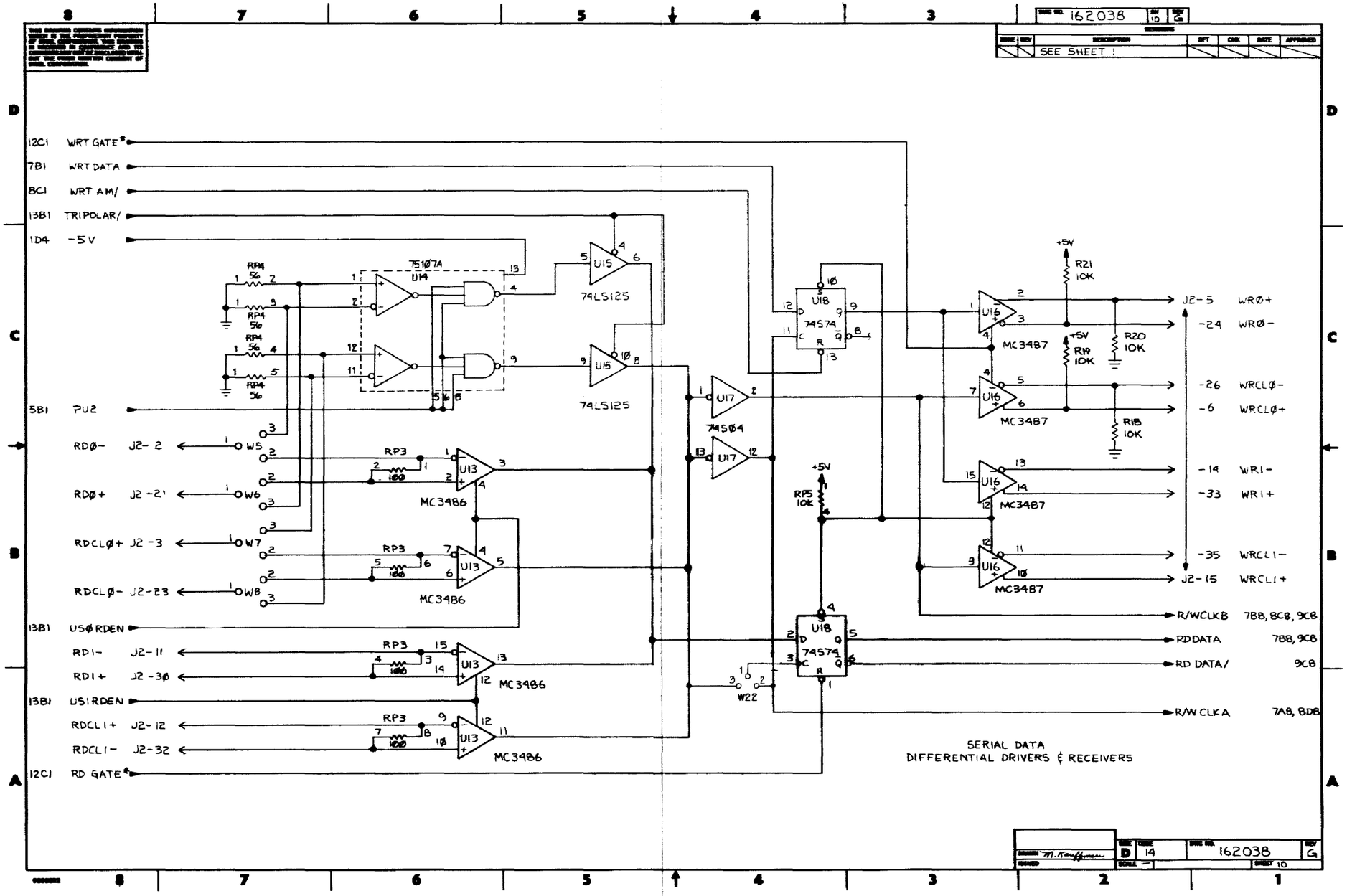


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 10 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

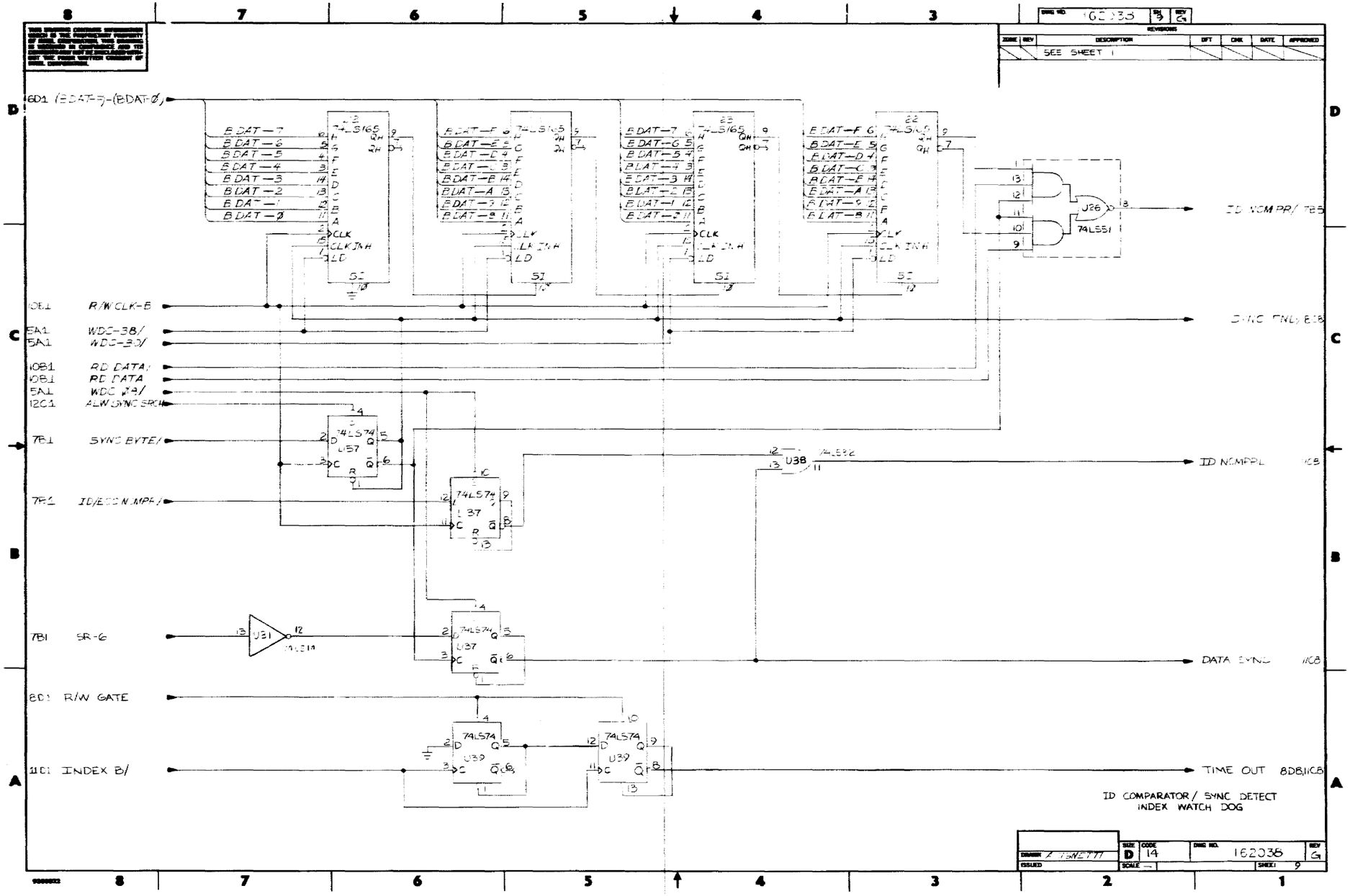


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 9 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

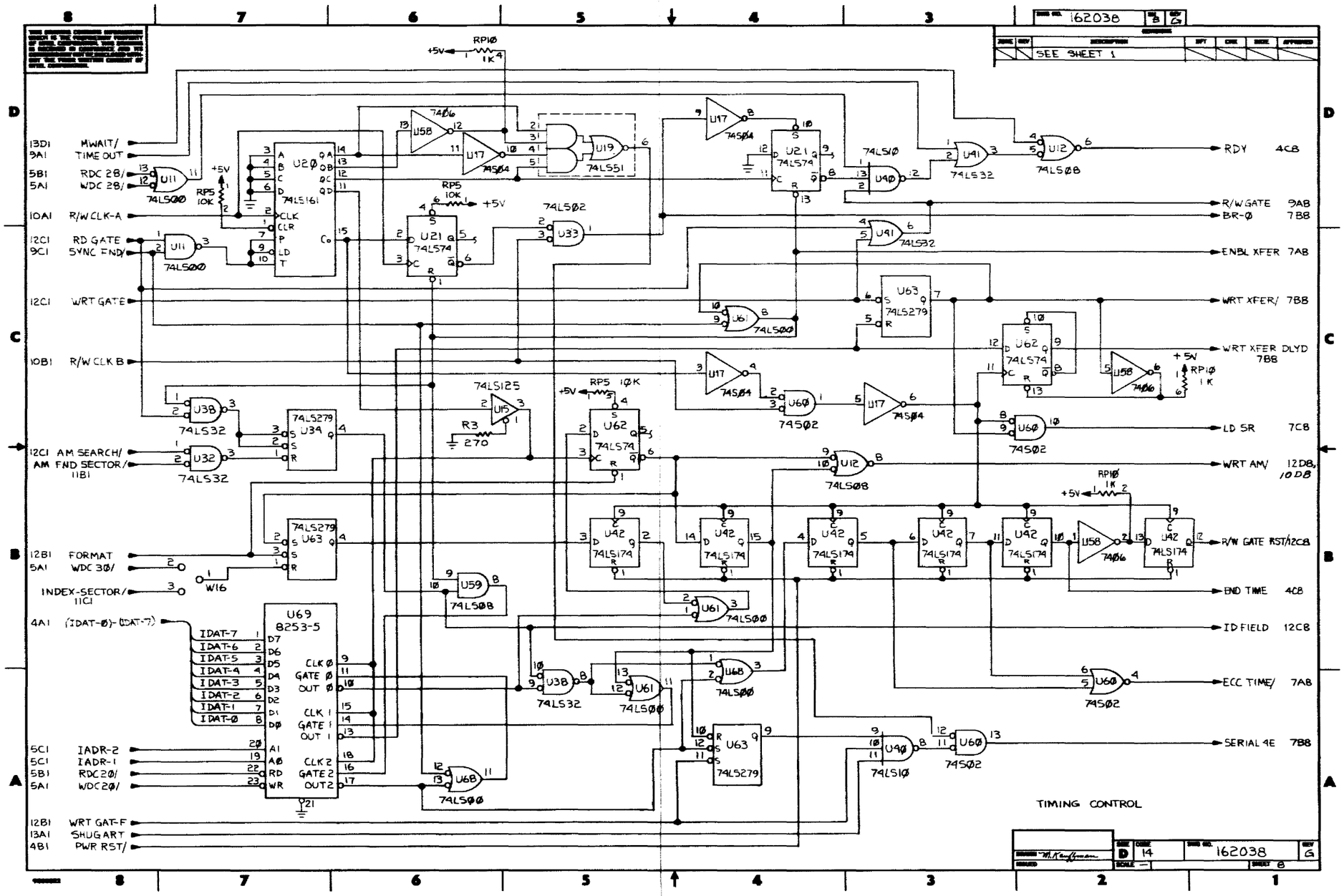


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 8 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

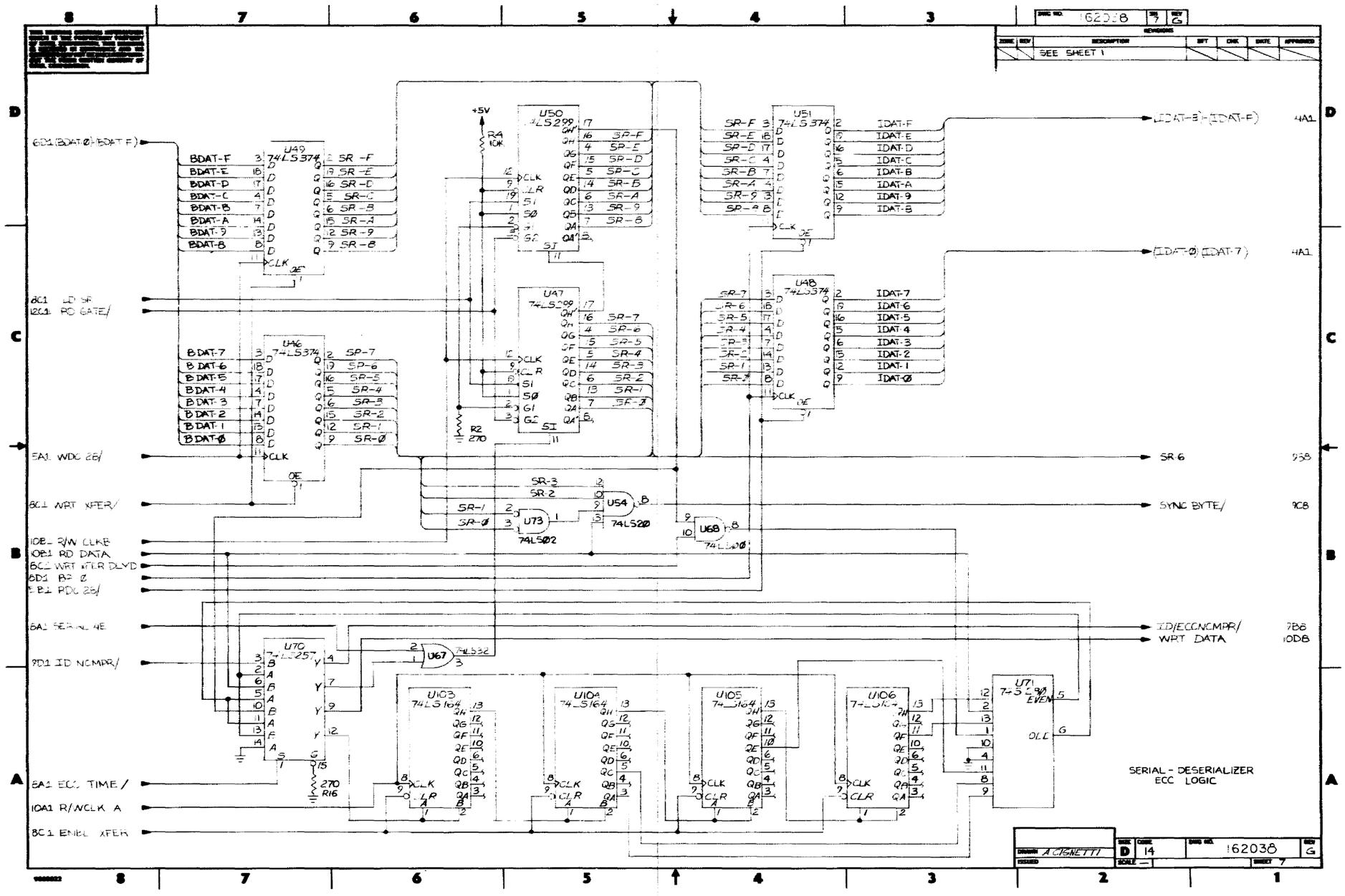


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 7 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

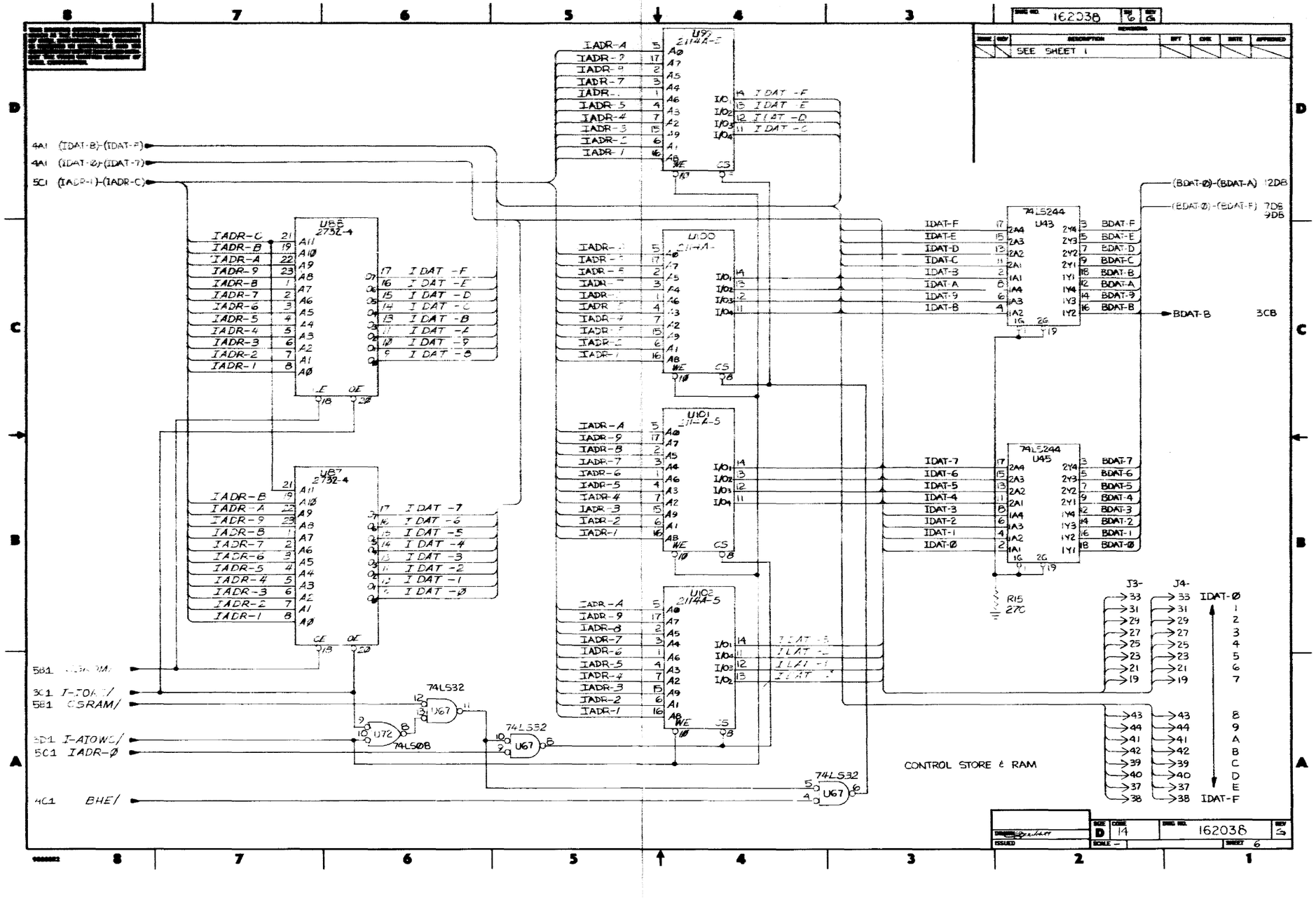


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 6 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

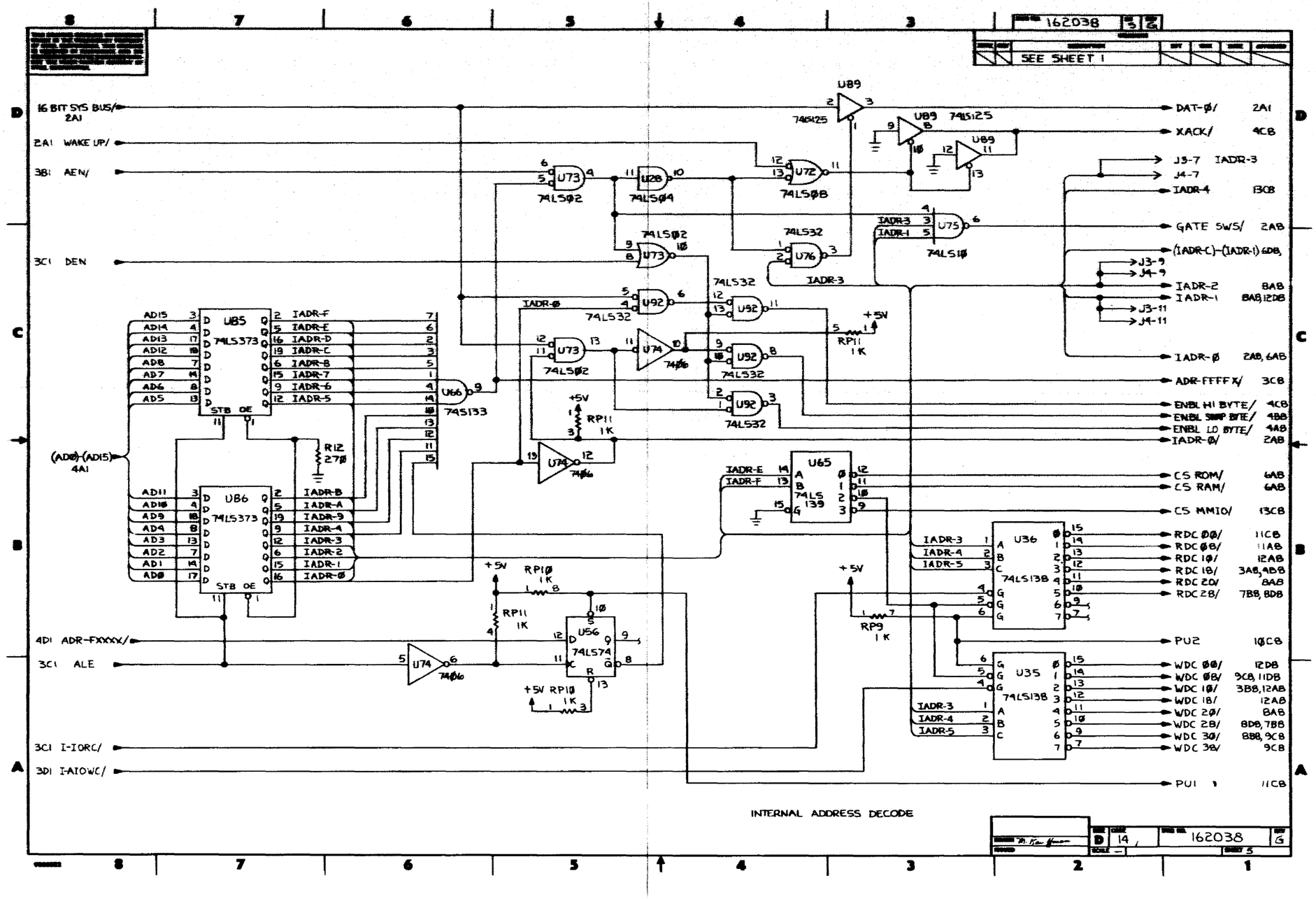


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 5 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

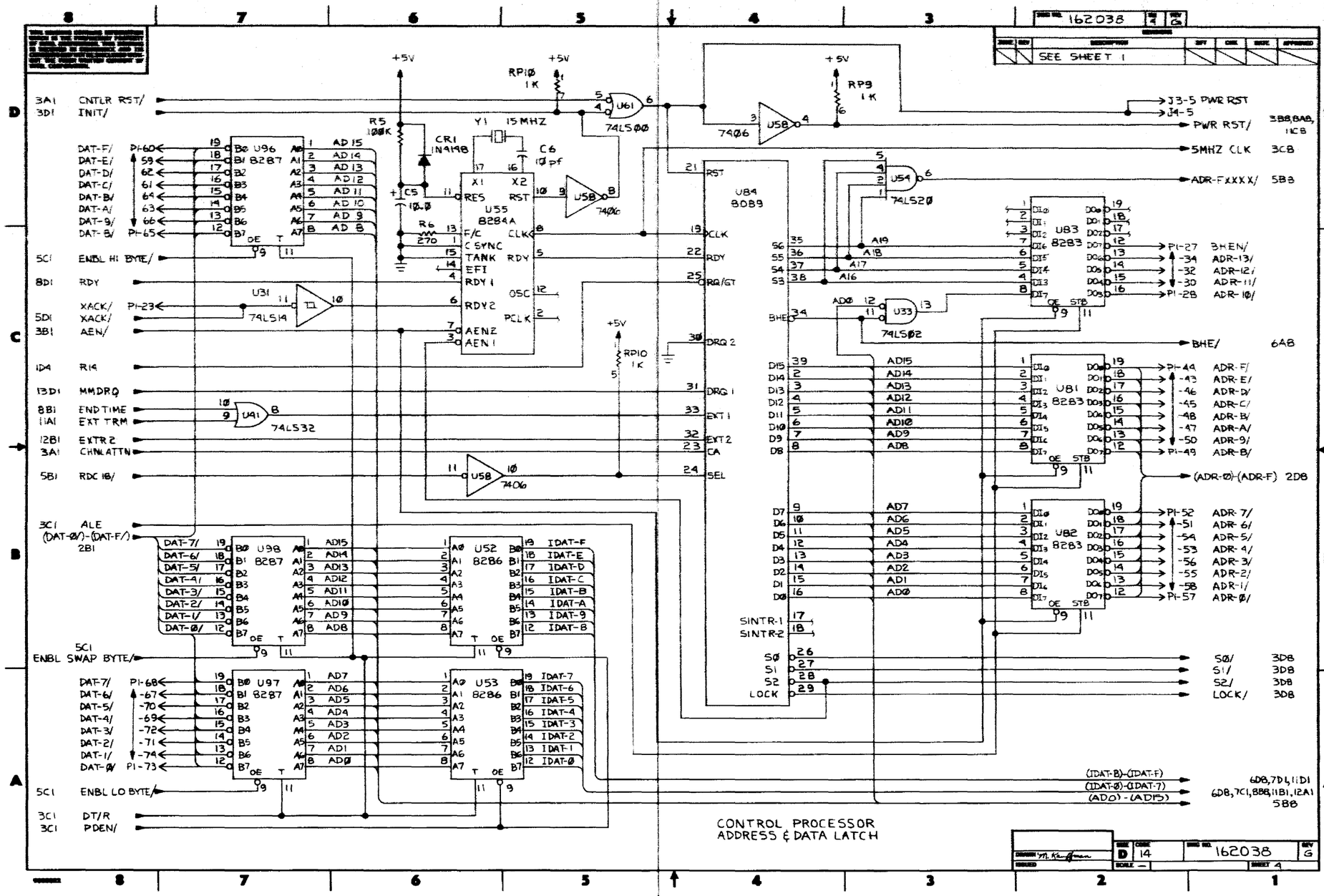


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 4 of 14)

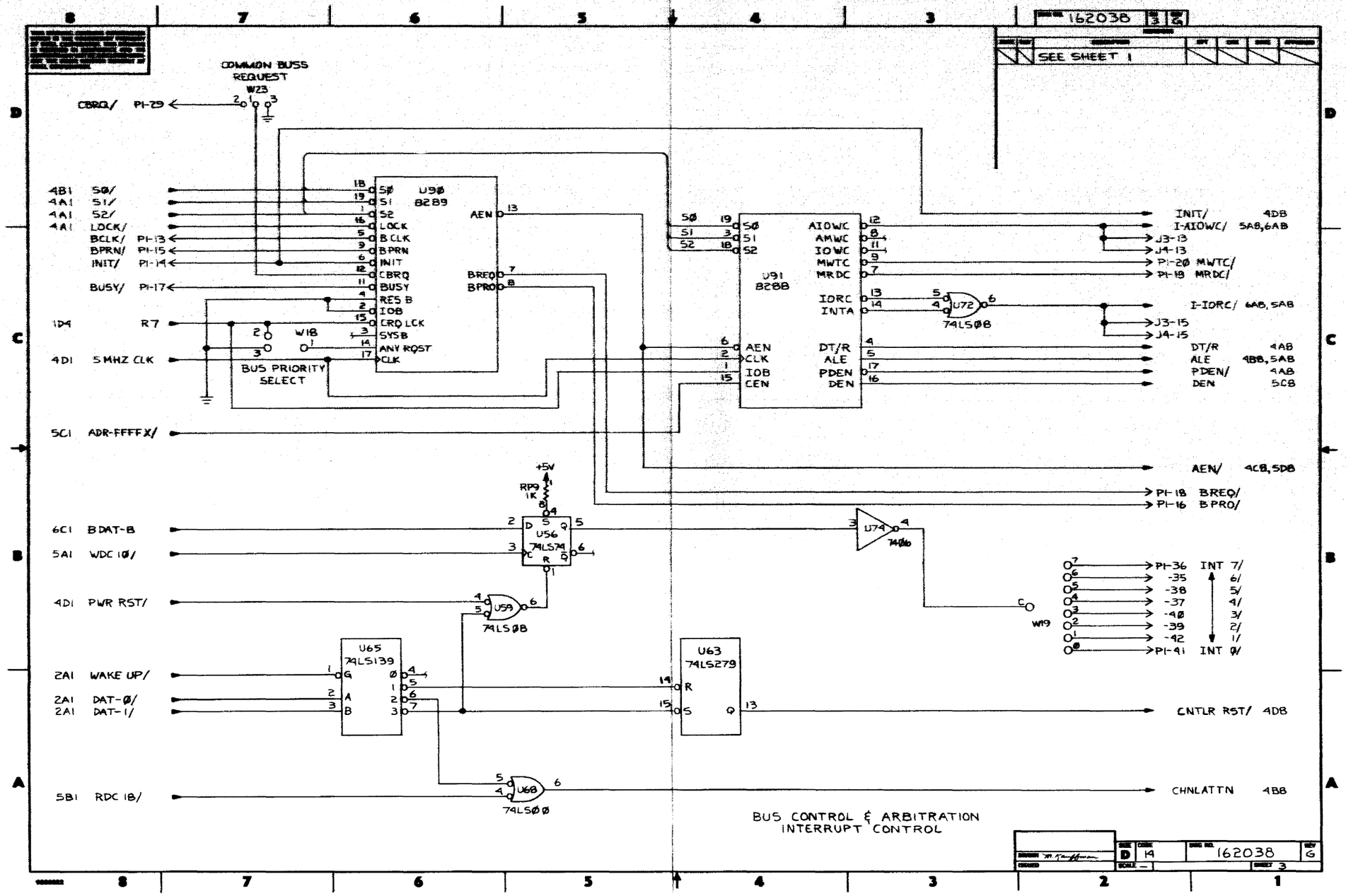


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 3 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

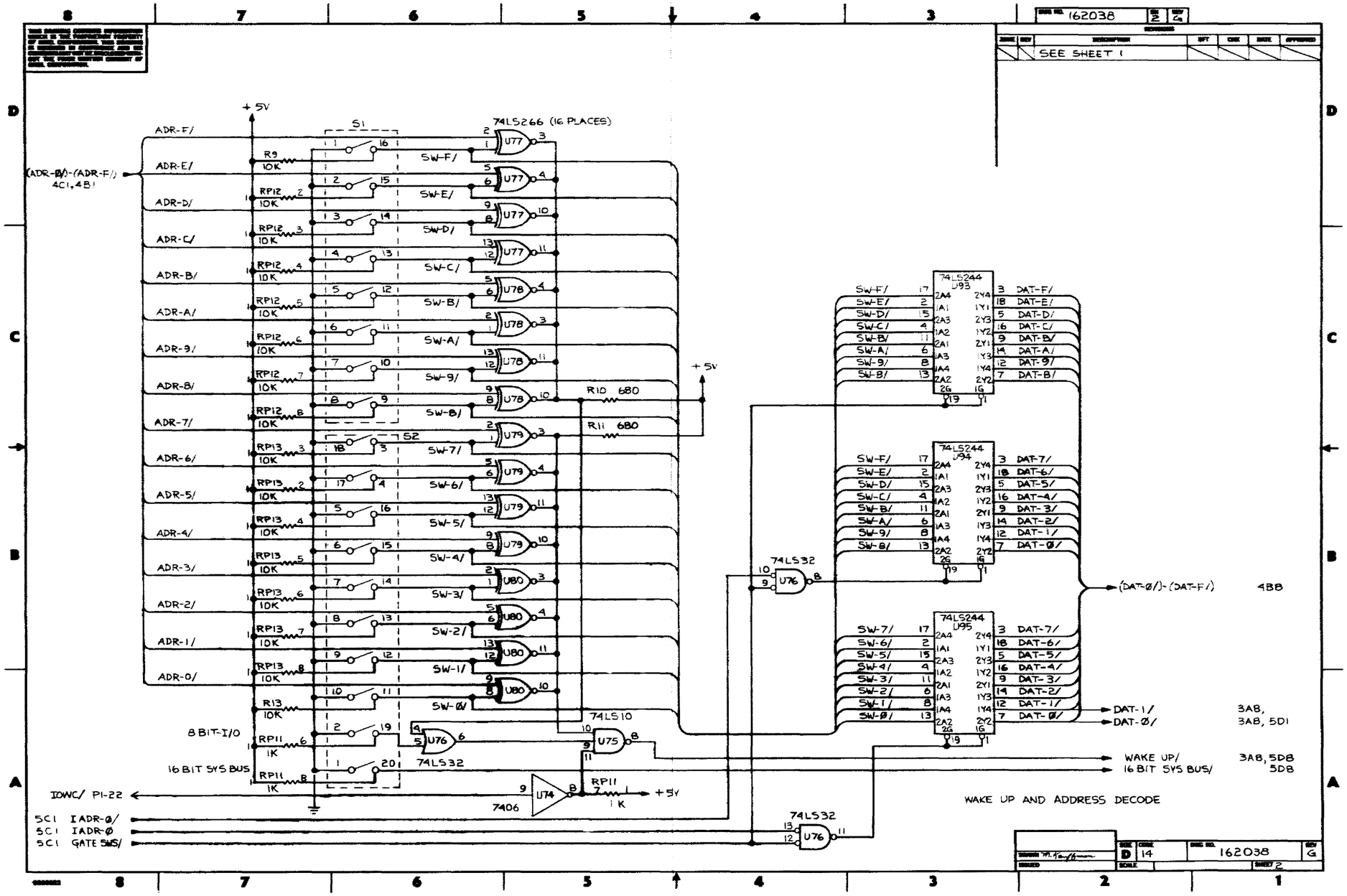
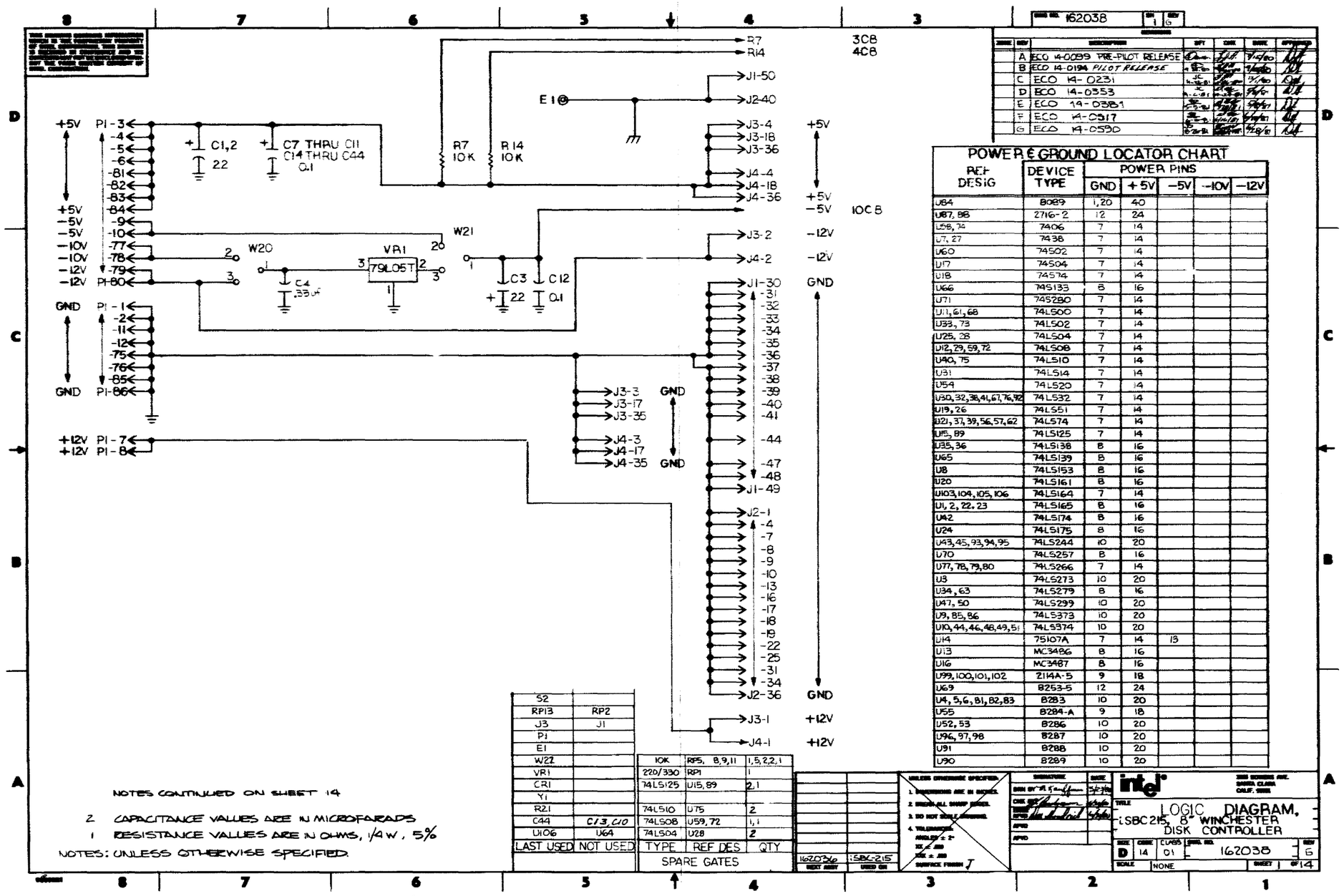


Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 2 of 14)

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.



REV	DESCRIPTION	BY	CHK	DATE	APPD
A	ECO 14-0089 PRE-PILOT RELEASE				
B	ECO 14-0194 PILOT RELEASE				
C	ECO 14-0231				
D	ECO 14-0353				
E	ECO 14-0381				
F	ECO 14-0517				
G	ECO 14-0590				

POWER & GROUND LOCATOR CHART

REF DESIG	DEVICE TYPE	POWER PINS GND	+5V	-5V	-10V	-12V
U84	8089	1,20	40			
U87, 88	2716-2	12	24			
U56, 74	7406	7	14			
U7, 27	7438	7	14			
U60	74S02	7	14			
U17	74S04	7	14			
U18	74S74	7	14			
U66	74S133	8	16			
U71	74S280	7	14			
U1, 61, 68	74LS00	7	14			
U33, 73	74LS02	7	14			
U25, 28	74LS04	7	14			
U12, 29, 59, 72	74LS08	7	14			
U90, 75	74LS10	7	14			
U31	74LS14	7	14			
U54	74LS20	7	14			
U30, 32, 38, 41, 67, 76, 92	74LS32	7	14			
U19, 26	74LS51	7	14			
U21, 37, 39, 56, 57, 62	74LS74	7	14			
U15, 89	74LS125	7	14			
U35, 36	74LS138	8	16			
U65	74LS139	8	16			
U8	74LS153	8	16			
U20	74LS161	8	16			
U103, 104, 105, 106	74LS164	7	14			
U1, 2, 22, 23	74LS165	8	16			
U42	74LS174	8	16			
U24	74LS175	8	16			
U43, 45, 93, 94, 95	74LS244	10	20			
U70	74LS257	8	16			
U77, 78, 79, 80	74LS266	7	14			
U8	74LS273	10	20			
U34, 63	74LS279	8	16			
U47, 50	74LS299	10	20			
U9, 85, 86	74LS373	10	20			
U10, 44, 46, 48, 49, 51	74LS374	10	20			
U14	75107A	7	14	13		
U13	MC3486	8	16			
U16	MC3487	8	16			
U99, 100, 101, 102	2114A-5	9	18			
U69	B253-5	12	24			
U4, 5, 6, 81, 82, 83	B283	10	20			
U55	B284-A	9	18			
U52, 53	B286	10	20			
U96, 97, 98	B287	10	20			
U91	B288	10	20			
U90	B289	10	20			

CAUTION: These schematic diagrams may have been revised. See "Service Information" chapter for details.

Figure 5-3. iSBC 215™ Winchester Disk Controller Schematic Diagram (Sheet 1 of 14)

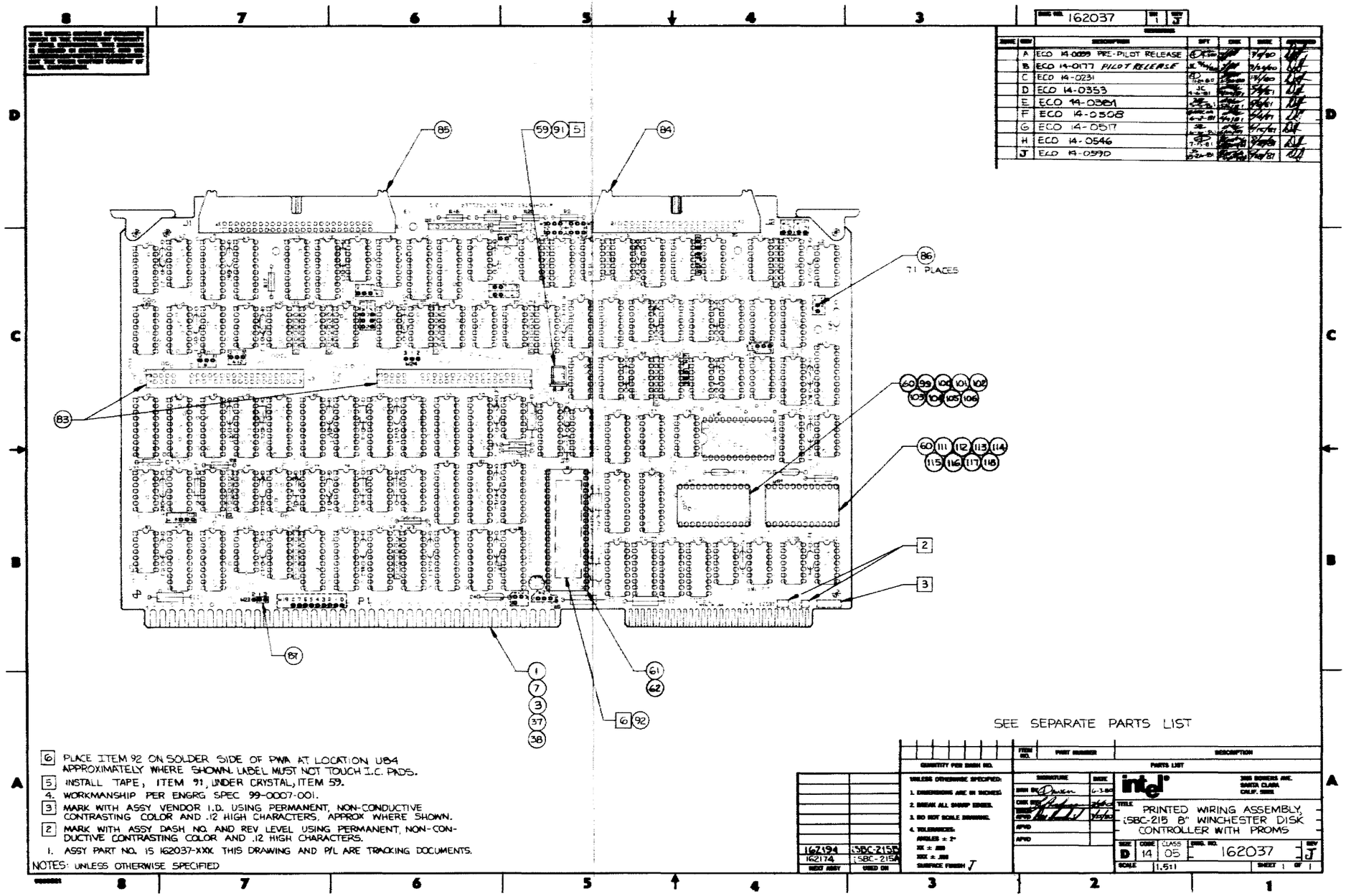
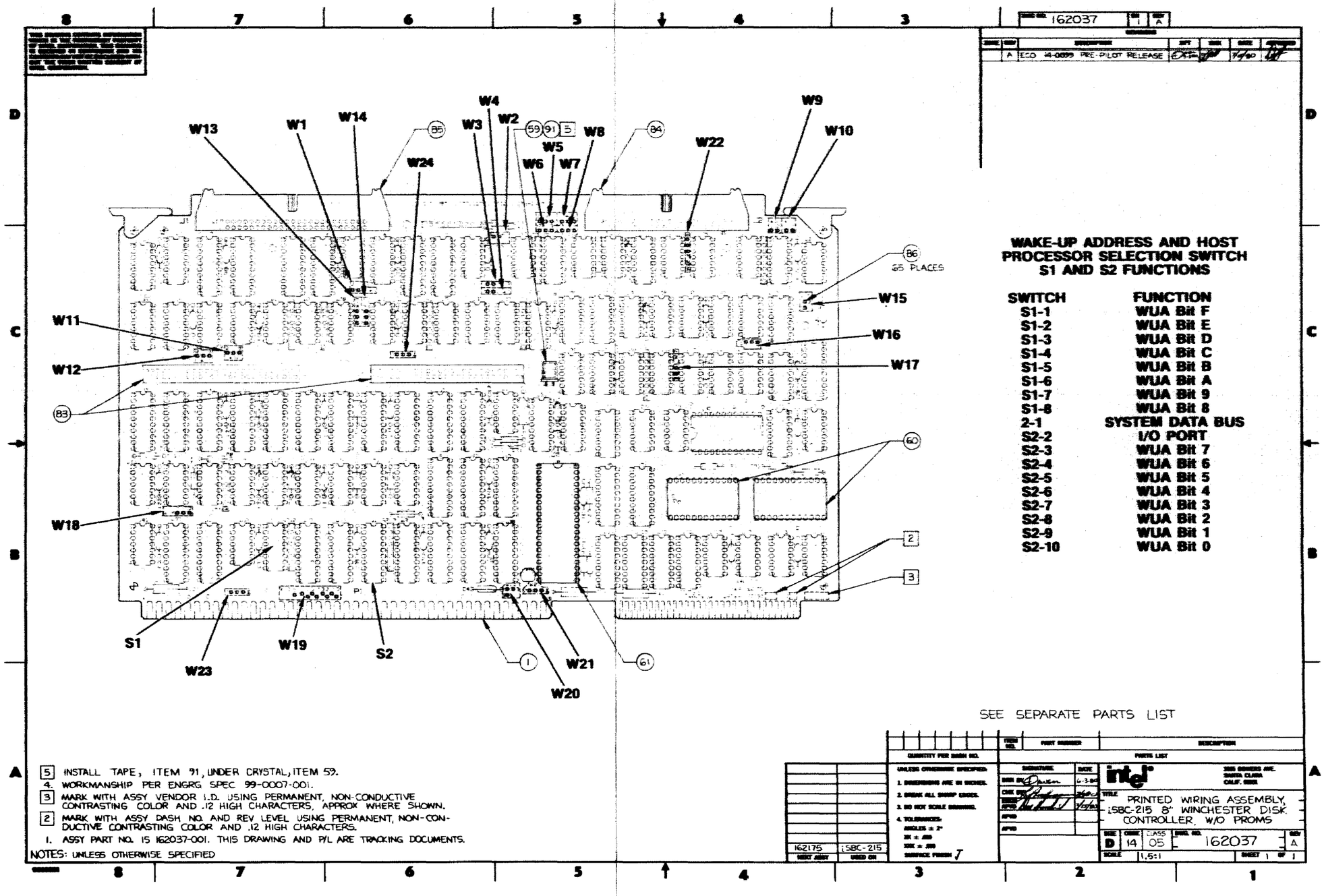


Figure 5-2. iSBC 215™ Winchester Disk Controller Parts Location Diagram



WAKE-UP ADDRESS AND HOST PROCESSOR SELECTION SWITCH S1 AND S2 FUNCTIONS

SWITCH	FUNCTION
S1-1	WUA Bit F
S1-2	WUA Bit E
S1-3	WUA Bit D
S1-4	WUA Bit C
S1-5	WUA Bit B
S1-6	WUA Bit A
S1-7	WUA Bit 9
S1-8	WUA Bit 8
2-1	SYSTEM DATA BUS
S2-2	I/O PORT
S2-3	WUA Bit 7
S2-4	WUA Bit 6
S2-5	WUA Bit 5
S2-6	WUA Bit 4
S2-7	WUA Bit 3
S2-8	WUA Bit 2
S2-9	WUA Bit 1
S2-10	WUA Bit 0

Figure 5-1. iSBC 215™ Controller Jumpers and Switch Locations

APPENDIX A HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR DISK CONTROL PROGRAM

INTRODUCTION

The information contained in this appendix is provided to illustrate various methods of implementing data transfers between one or more host processors and the iSBC 215 controller. The flow charts illustrate the handshake procedures required between a host processor and the controller. User sequences are shown both for single and multi-user processing environments. A sequence for initiating overlapped seeks is also given.

The program listing provides an example program that a host processor would run to direct data transfer between the host and the iSBC 215 controller. The program is written in MCS-86™ Macro Assembler language. It illustrates the data structures that the iSBC 215 controller requires and shows a few simple disk operations drivers.

SINGLE USER SEQUENCE

The flow chart in Figure A-1 shows the handshake sequence between a single host processor and the controller for basic data transfer operations (with no overlapping seeks). Note that communication between the host and the controller is through the Status Semaphore and Operation Status bytes of the Controller Invocation Block.

SINGLE USER SEQUENCE WITH OVERLAPPING SEEKS

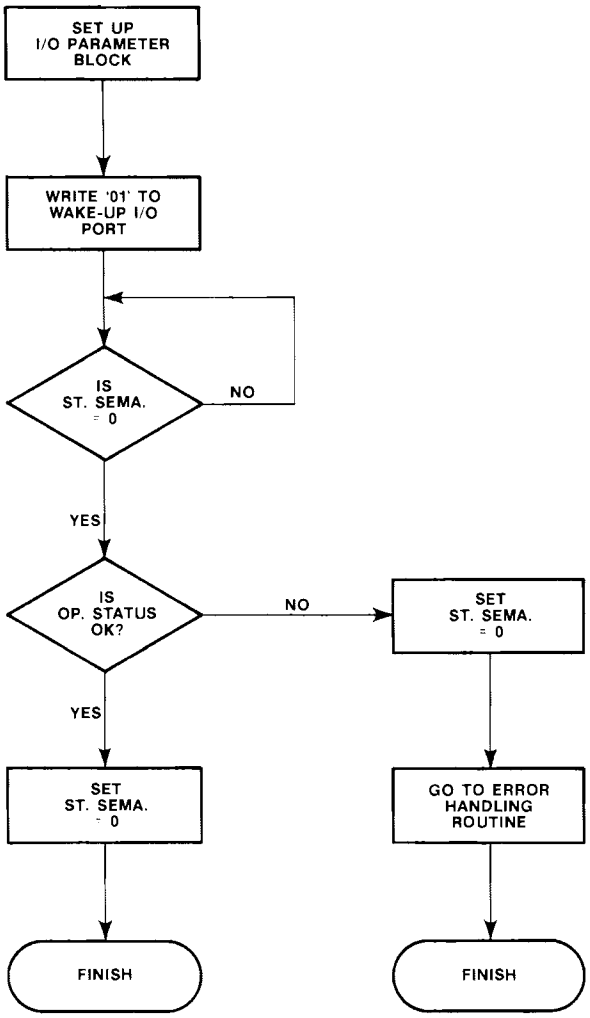
The flow chart in Figure A-2 shows the handshake sequence between a single host processor and the controller for data transfer operations that user overlapping seeks.

MULTI-USER SEQUENCE

The flow chart in Figure A-3 shows the handshake sequence between a host processor and the controller when more than one processor is transferring data between the disk drives through the same controller (multi-processor environment). Note that in this case the Command Semaphore byte in the Controller Invocation Block is also used. Overlapping seeks in a multi-processor environment are implemented the same as in single processor environments.

EXAMPLE HOST PROCESSOR DISK CONTROL PROGRAM

The following program example is for a single user environment. Some of the techniques illustrated in the flow charts in this appendix are implemented in this program, but not all.



Set up command and parameters for desired data transfer operation.

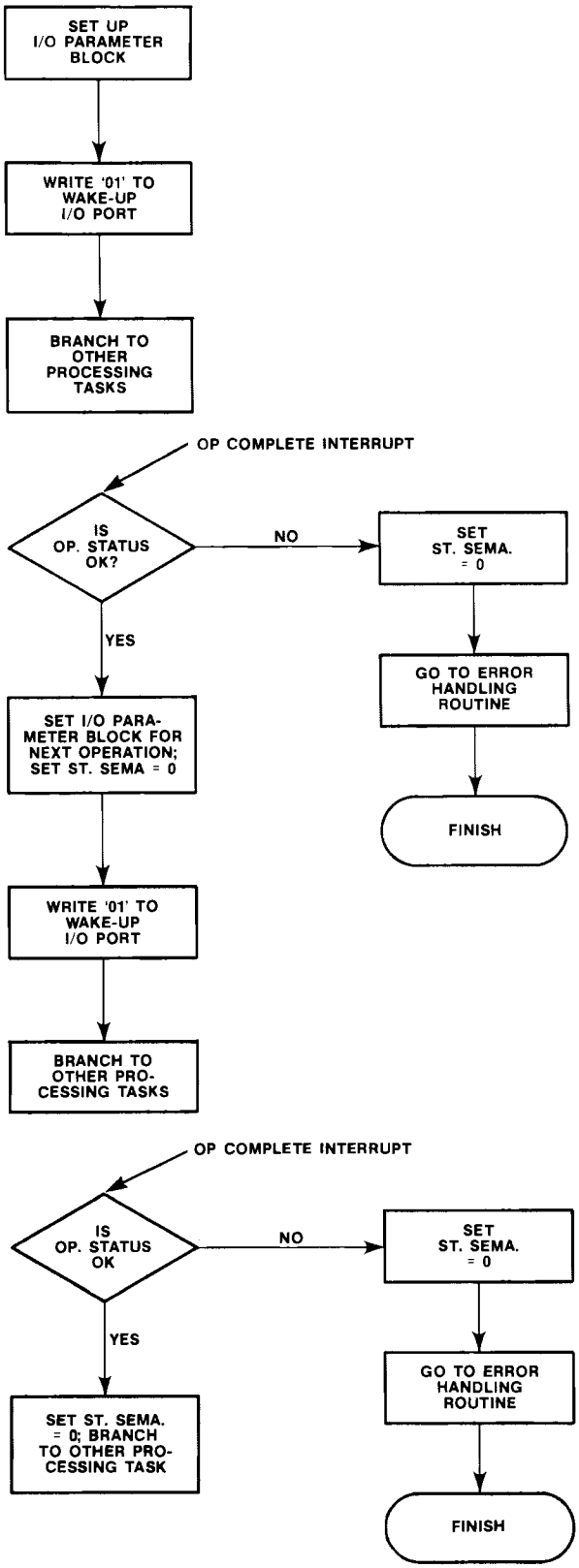
Initiate data transfer operation.

Check to see if controller has completed operation (STATUS SEMAPHORE byte is non-zero).

Check OPERATION STATUS byte to see if operation was completed without error.

Check Error Status buffer and process results.

Figure A-1. Flow Chart for Single User Handshake Sequence Without Overlapping Seeks



Set up command and parameters for seek operation.

Initiate data transfer operation.

Perform other processing tasks.

Respond to operation complete interrupt; Check OPERATION STATUS byte to see if operation was completed without error.

Set up command and parameters for next seek operation.

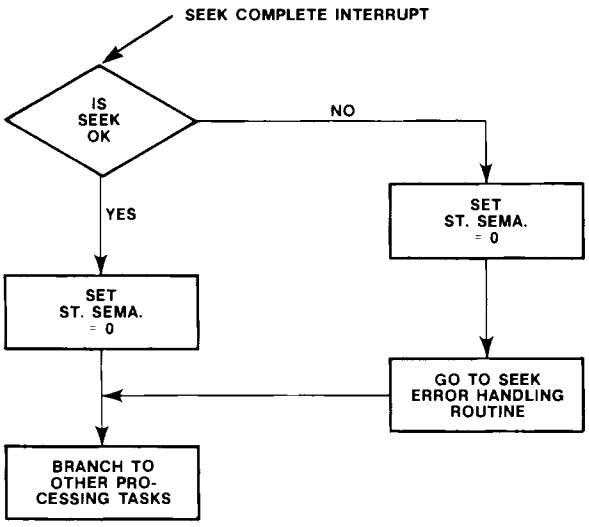
Initiate next data transfer operation.

Perform other processing tasks.

Respond to operation complete interrupt; check OPERATION STATUS byte to see if operation was completed without error.

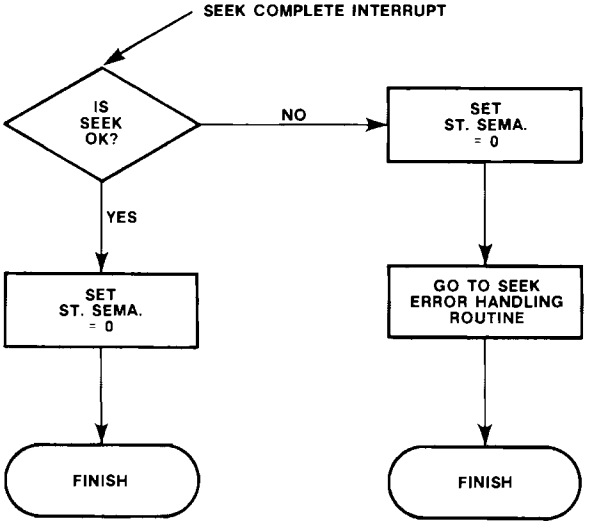
Continue with other processing tasks.

Figure A-2. Flow Chart for Single User Handshake Sequence With Overlapping Seeks



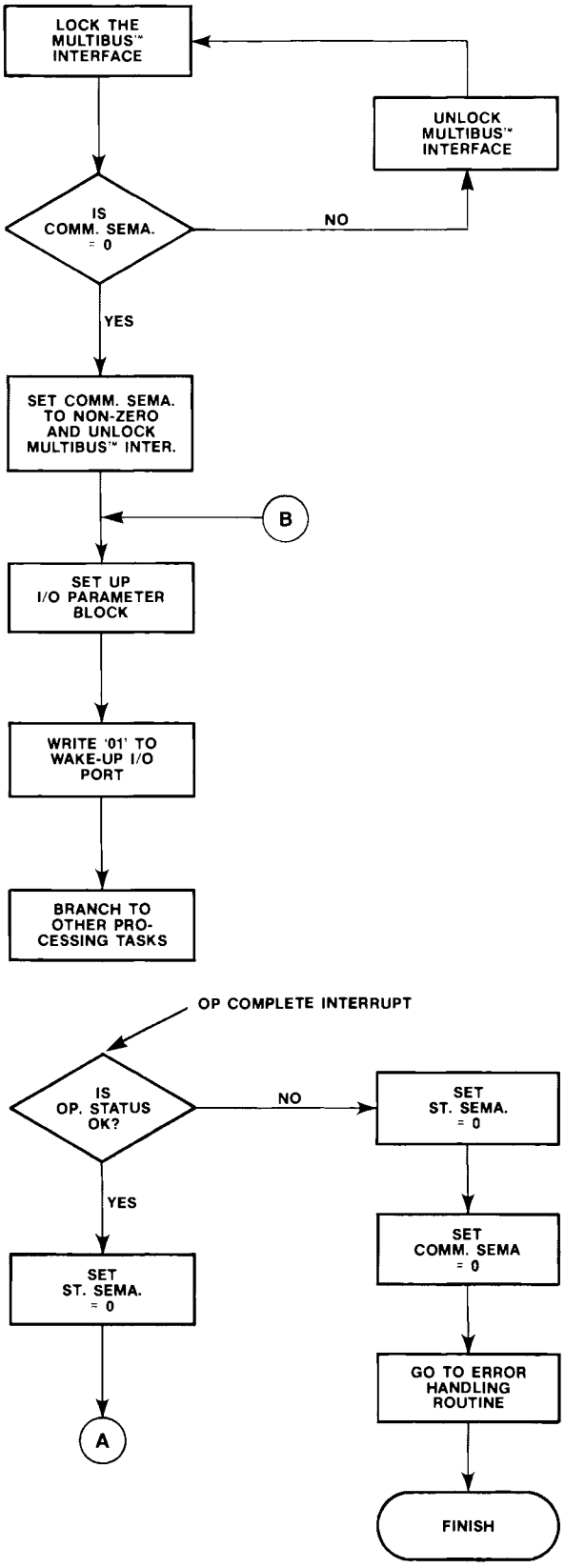
Respond to seek complete interrupt for first operation; check status buffer to determine if seek operation was completed without error.

Continue with other processing tasks.



Respond to seek complete interrupt for second operation; check status buffer to determine if seek operation was completed without error.

Figure A-2. Flow Chart for Single User Handshake Sequence With Overlapping Seeks (Continued)



Obtain control of disk controller. Test and set COMMAND SEMAPHORE byte.

Set up command and parameters for desired data transfer operation.

Initiate data transfer operation.

Continue with other processing tasks.

Respond to operation complete interrupt; check OPERATION STATUS byte to see if operation was completed without error.

Relinquish controller of disk controller.

Figure A-3. Flow Chart for Multi-User Handshake Sequence

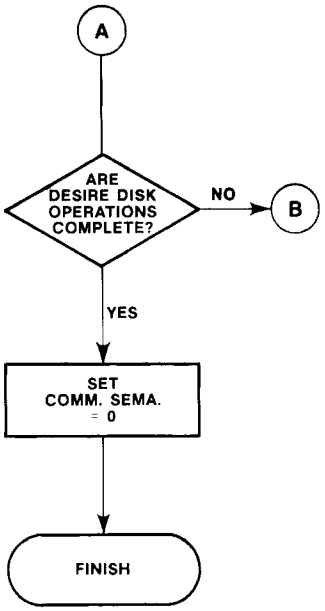


Figure A-3. Flow Chart for Multi-User Handshake Sequence (Continued)

```
MCS-86 MACRO ASSEMBLER    iSBC 215 8" WINCHESTER DISK CONTROLLER PROGRAMMING EXAMPLE          10/27/80   PAGE     1


ISIS-II MCS-86 MACRO ASSEMBLER V2.1 ASSEMBLY OF MODULE EXMPRG
OBJECT MODULE PLACED IN :F1:EXMPRG.OBJ
ASSEMBLER INVOKED BY:  ASM86 :F1:EXMPRG.MMD DATE(10/27/80) XREF DEBUG


LOC  OBJ                  LINE     SOURCE

                             1     $PAGELENGTH(85) PAGEWIDTH(115) TITLE(iSBC 215 8" WINCHESTER DISK CONTROLLER PROG
                                   RAMMING EXAMPLE) XREF
                             2     ;
                             3     ; ##########################################################################
                             4     ; ##                                                                      ##
                             5     ; ##     iSBC 215  DISK CONTROLLER PROGRAMMING EXAMPLE    ##
                             6     ; ##                                                                      ##
                             7     ; ##########################################################################
                             8     ;
                             9     ; THIS PROGRAM ILLUSTRATES THE DATA STRUCTURES REQUIRED BY THE iSBC 215
                            10     ; DISK CONTROLLER.  A FEW SIMPLE DISK OPERATION DRIVERS ARE ALSO SHOWN.
                            11     ;
                            12     ; THE HARDWARE CONFIGURATION SUPPORTED IS:
                            13     ;
                            14     ;       1.  iSBC 86/12A HOST CPU
                            15     ;       2.  20 BIT SYSTEM MEMORY ADDRESS WIDTH
                            16     ;       3.  16 BIT SYSTEM DATA BUS WIDTH
                            17     ;       4.  16 BIT SYSTEM I/O ADDRESS WIDTH
                            18     ;       5.  iSBC 215
                            19     ;               a.  WAKE UP ADDRESS ( WUA ) AT I/O PORT 0635H
                            20     ;               b.  INTERRUPT 5
                            21     ;               c.  -12 VOLTS INPUT
                            22     ;               d.  RELINQUISH BUS CONTROL ON ANY REQUEST
                            23     ;
                            24     ; FOR (2), PROGRAMMING OF DATA TRANSFERS MUST TAKE THIS INTO ACCOUNT,e.g. THERE
                            25     ;          IS NO WRAPAROUND IN SEGMENTS IF MORE THAN 64K BYTES ARE TRANSFERRED.
                            26     ;
                            27     ; iSBC 215 SWITCH AND JUMPER SETTINGS:
                            28     ;
                            29     ; FOR (3), SWITCH S2-1 IS CLOSED.
                            30     ; FOR (4), SWITCH S2-2 IS CLOSED.
                            31     ; FOR (5a), SWITCHES S1-6,S1-7,S2-5,S2-6,S2-8, AND S2-10 ARE CLOSED, THE
                            32     ;          REMAINING ADDRESS SELECT SWITCHES ARE OPEN.
                            33     ; FOR (5b), W19-C CONNECTS TO W19-5; INTERRUPT VECTORS MUST BE SET UP PROPERLY.
                            34     ; FOR (5c), W21-1 CONNECTS TO W21-3
                            35     ; FOR (5d), W2-1 CONNECTS TO W2-2.
                            36     ;
                            37 +1  $INCLUDE(:F1:COMBLK.MMD)
                        =1  38 +1  $EJECT TITLE(iSBC 215 COMMUNICATION BLOCKS)
```

```
MCS-86 MACRO ASSEMBLER    iSBC 215 COMMUNICATION BLOCKS                10/27/80  PAGE    2

LOC  OBJ                  LINE     SOURCE

                     =1     39     ; ---------------------------------
                     =1     40     ; |                               |
                     =1     41     ; |     COMMUNICATION BLOCKS      |
                     =1     42     ; |                               |
                     =1     43     ; ---------------------------------
                     =1     44     ;
                     =1     45     ; ==============
                     =1     46     ;   I.  SCB
                     =1     47     ; ==============
                     =1     48     ;
                     =1     49     ;       THE SCB TELLS THE 8089 ON THE iSBC 215 THE WIDTH OF THE 8089's LOCAL
                     =1     50     ;       BUS AND POINTS TO THE CCB.
                     =1     51     ;
                     =1     52     ;       ************************************************************************
                     =1     53     ;       *  THE MEMORY ADDRESS OF THE SCB IS EQUAL TO THE I/O WAKE-UP ADDRESS  *
                     =1     54     ;       *                ( WUA ) OF THE iSBC 215 MULTIPLIED BY 16.           *
                     =1     55     ;       ************************************************************************
                     =1     56     ;
  0635               =1     57             WUA     EQU     0635H           ; WAKE-UP ADDRESS I/O PORT NUMBER
                     =1     58     ;
----                 =1     59     SCBSEG  SEGMENT AT WUA                  ; PUTS SCB AT ADDRESS 06350H
                     =1     60     ;
0000                 =1     61     SCB     LABEL   FAR
0000 01              =1     62     SOC     DB      01H                     ; TELL 8089 IT IS ON A 16 BIT LOCAL BUS
0001 00              =1     63             DB      00H                     ; RESERVED
0002 0000----      R =1     64     CCBPTR  DD      CCB                     ; POINTER (SEGMENT + OFFSET) TO CCB
                     =1     65     ;
----                 =1     66     SCBSEG  ENDS
                     =1     67     ;
                     =1     68     ; ===============
                     =1     69     ;   II.  CCB
                     =1     70     ; ===============
                     =1     71     ;
                     =1     72     ;       THIS BLOCK CONTAINS THE CONTROL BYTES, BUSY FLAGS, AND POINTERS TO THE
                     =1     73     ;       STARTING ADDRESSES OF THE CHANNEL PROGRAMS FOR THE 8089.
                     =1     74     ;
----                 =1     75     CCBSEG  SEGMENT                         ; CCB MUST BE CONTIGUOUS
                     =1     76     ;
0000                 =1     77     CCB     LABEL   FAR
0000 01              =1     78     CCW1    DB      01H                     ; START CH. 1 PROGRAM IN LOCAL MEMORY
0001 00              =1     79     BSYFLG1 DB      00H                     ; CH. 1 BUSY FLAG
0002 0400----      R =1     80     CH1PTR  DD      CH1PC                   ; POINTER TO FIFTH BYTE OF CIB, WHICH
                     =1     81                                             ;  CONTAINS STARTING ADDRESS OF CH. 1
                     =1     82                                             ;  FIRMWARE PROGRAM
0006 0000            =1     83             DW      0000H                   ; RESERVED
0008 01              =1     84     CCW2    DB      01H                     ; START CH. 2 PROGRAM IN LOCAL MEMORY
0009 00              =1     85     BSYFLG2 DB      00H                     ; CH. 2 BUSY FLAG
000A 0E00----      R =1     86     CH2PTR  DD      CH2PC                   ; POINTER TO LAST WORD OF CCB, WHICH
                     =1     87                                             ;  CONTAINS STARTING ADDRESS OF CH. 2
                     =1     88                                             ;  FIRMWARE PROGRAM
000E                 =1     89     CH2PC   LABEL   FAR
000E 0400            =1     90             DW      0004H                   ; STARTING ADDRESS OF CH. 2 PROGRAM
                     =1     91     ;
----                 =1     92     CCBSEG  ENDS
                     =1     93     ;
                     =1     94 +1  $EJECT
```

```
MCS-86 MACRO ASSEMBLER    iSBC 215 COMMUNICATION BLOCKS                    10/27/80   PAGE    3

LOC  OBJ                  LINE     SOURCE

                     =1     95     ; ===============
                     =1     96     ;   III.  CIB
                     =1     97     ; ===============
                     =1     98     ;
                     =1     99     ;       THIS BLOCK CONTAINS GENERAL PURPOSE COMMAND AND STATUS BYTES, SEMA-
                     =1    100     ;       PHORES, AND POINTERS TO ALLOW THE USE OF THE iSBC 215 IN A MULTI-
                     =1    101     ;       PROCESSOR/MULTI-PROCESSING SYSTEM.
                     =1    102     ;
----                 =1    103     CIBSEG  SEGMENT                         ; CIB MUST BE CONTIGUOUS
                     =1    104     ;
0000                 =1    105     CIB     LABEL   FAR
0000 00              =1    106     CIBCMD  DB      00H                     ; CIB COMMAND BYTE NOT USED BY iSBC 215
0001 00              =1    107     OPSTS   DB      00H                     ; CIB STATUS BYTE IS USED BY iSBC 215
0002 00              =1    108     CMDSEM  DB      00H                     ; COMMAND BYTE SEMAPHORE
0003 00              =1    109     STSSEM  DB      00H                     ; STATUS BYTE SEMAPHORE
0004                 =1    110     CH1PC   LABEL   FAR
0004 00000000        =1    111             DD      0000H                   ; STARTING ADDRESS OF CH. 1 PROGRAM
0008 0000            =1    112     IOPBOFF DW      OFFSET IOPB             ; POINTER TO IOPB
000A ----          R =1    113     IOPBSG  DW      IOPBSEG
000C 00000000        =1    114             DD      0000H                   ; RESERVED
                     =1    115     ;
----                 =1    116     CIBSEG  ENDS
                     =1    117     ;
                     =1    118     ; ===============
                     =1    119     ;   IV.  IOPB
                     =1    120     ; ===============
                     =1    121     ;
                     =1    122     ;       THIS BLOCK CONTAINS THE DEVICE DEPENDENT CONTROL INFORMATION FOR THE
                     =1    123     ;       iSBC 215 CONTROLLER.
                     =1    124     ;
----                 =1    125     IOPBSEG SEGMENT                         ; IOPB MUST BE CONTIGUOUS
                     =1    126     ;
0000                 =1    127     IOPB    LABEL   FAR
0000 00000000        =1    128             DD      0000H                   ; RESERVED
0004 00000000        =1    129     ACTCNT  DD      0000H                   ; ACTUAL TRANSFER COUNT (32 BIT INTEGER)
0008 0000            =1    130     DEVCOD  DW      0000H                   ; DEVICE CODE (0H-WINCHESTER 01H-FLOPPY)
000A 00              =1    131     UNIT    DB      00H                     ; UNIT NUMBER (0 <= UNIT <= 3)
000B 00              =1    132     FUNC    DB      00H                     ; FUNCTION CODE (0 <= FUNCTION <= 0FH)
000C 0000            =1    133     MODIFY  DW      0000H                   ; MODIFIER WORD
000E 0000            =1    134     CYLNDR  DW      0000H                   ; CYLINDER NUMBER
0010 00              =1    135     HEAD    DB      00H                     ; HEAD NUMBER
0011 00              =1    136     SECTOR  DB      00H                     ; SECTOR NUMBER
0012 0000            =1    137     BUFOFF  DW      0000H                   ; POINTER TO DATA BUFFER
0014 0000            =1    138     BUFSEG  DW      0000H
0016 00000000        =1    139     REQCNT  DD      0000H                   ; REQUESTED TRANSFER COUNT (INTEGER)
001A 00000000        =1    140             DD      0000H                   ; RESERVED
                     =1    141     ;
----                 =1    142     IOPBSEG ENDS
                           143     ;
                           144 +1  $INCLUDE(:F1:INITBL.MMD)
                     =1    145 +1  $EJECT TITLE(DISK DRIVE INITIALIZATION TABLES)
```

```
MCS-86 MACRO ASSEMBLER    DISK DRIVE INITIALIZATION TABLES                    10/27/80   PAGE    4

LOC  OBJ                  LINE     SOURCE

                         =1   146      ; ------------------------------------------------------
                         =1   147      ; |                                                    |
                         =1   148      ; |     DISK DRIVE INITIALIZATION PARAMETER TABLES     |
                         =1   149      ; |                                                    |
                         =1   150      ; ------------------------------------------------------
                         =1   151      ;
                         =1   152      ;       THIS SEGMENT CONTAINS THE DRIVE CONFIGURATION DATA TABLES THAT ARE USED
                         =1   153      ;       BY THE INITIALIZATION ROUTINE.  THEY MUST BE MODIFIED TO REFLECT THE
                         =1   154      ;       PARTICULAR DRIVES BEING USED WITH THE iSBC 215 DISK CONTROLLER.
                         =1   155      ;
                         =1   156      ; - IF A DRIVE IS NOT PRESENT, ITS INITIALIZATION TABLE MUST BE ALL ZEROES.
                         =1   157      ;
                         =1   158      ;    ..........................................................
                         =1   159      ;    |          8 " WINCHESTER HARD DISK DRIVES               |
                         =1   160      ;    ..........................................................
                         =1   161      ;    |  BYTES PER SECTOR   |    MAXIMUM SECTORS PER TRACK    |
                         =1   162      ;    |---------------------+---------------------------------|
                         =1   163      ;    |        128          |            54                   |
                         =1   164      ;    |        256          |            31                   |
                         =1   165      ;    |        512          |            17                   |
                         =1   166      ;    |       1024          |            9                    |
                         =1   167      ;    ..........................................................
                         =1   168      ;-------------------------------------------------------------------------------
                         =1   169      ;
----                     =1   170      INITBLSEG       SEGMENT
                         =1   171      ;
                         =1   172      ; DRIVE #0 ---SHUGART MODEL SA1004 (10.6 MEGABYTES STORAGE)
                         =1   173      ;
0000 0001                =1   174              DW      256                     ; NUMBER OF CYLINDERS
0002 04                  =1   175              DB      4                       ; NUMBER OF FIXED READ/WRITE SURFACES
0003 00                  =1   176              DB      0                       ; NUMBER OF REMOVABLE R/W SURFACES
0004 1F                  =1   177              DB      31                      ; NUMBER OF SECTORS PER TRACK
0005 0001                =1   178              DW      256                     ; NUMBER OF BYTES PER SECTOR
0007 05                  =1   179              DB      5                       ; NUMBER OF ALTERNATE CYLINDERS
                         =1   180      ;
                         =1   181      ; DRIVE #1 ---SHUGART MODEL SA1002 (5.3 MEGABYTES STORAGE)
                         =1   182      ;
0008 0001                =1   183              DW      256                     ; NUMBER OF CYLINDERS
000A 02                  =1   184              DB      2                       ; NUMBER OF FIXED READ/WRITE SURFACES
000B 00                  =1   185              DB      0                       ; NUMBER OF REMOVABLE R/W SURFACES
000C 11                  =1   186              DB      17                      ; NUMBER OF SECTORS PER TRACK
000D 0002                =1   187              DW      512                     ; NUMBER OF BYTES PER SECTOR
000F 05                  =1   188              DB      5                       ; NUMBER OF ALTERNATE CYLINDERS
                         =1   189      ;
                         =1   190      ; DRIVE #2 --- NONEXISTENT
                         =1   191      ;
0010 0000                =1   192              DW      0000H                   ; NUMBER OF CYLINDERS
0012 00                  =1   193              DB      00H                     ; NUMBER OF FIXED READ/WRITE SURFACES
0013 00                  =1   194              DB      00H                     ; NUMBER OF REMOVABLE R/W SURFACES
0014 00                  =1   195              DB      00H                     ; NUMBER OF SECTORS PER TRACK
0015 0000                =1   196              DW      0000H                   ; NUMBER OF BYTES PER SECTOR
0017 00                  =1   197              DB      00H                     ; NUMBER OF ALTERNATE CYLINDERS
                         =1   198      ;
                         =1   199      ; DRIVE #3 --- NONEXISTENT
                         =1   200      ;
0018 0000                =1   201              DW      0000H                   ; NUMBER OF CYLINDERS
001A 00                  =1   202              DB      00H                     ; NUMBER OF FIXED READ/WRITE SURFACES
001B 00                  =1   203              DB      00H                     ; NUMBER OF REMOVABLE R/W SURFACES
001C 00                  =1   204              DB      00H                     ; NUMBER OF SECTORS PER TRACK
001D 0000                =1   205              DW      0000H                   ; NUMBER OF BYTES PER SECTOR
001F 00                  =1   206              DB      00H                     ; NUMBER OF ALTERNATE CYLINDERS
                         =1   207      ;
                         =1   208 +1   $EJECT
```

```
MCS-86 MACRO ASSEMBLER    DISK DRIVE INITIALIZATION TABLES                10/27/80   PAGE    5

LOC  OBJ                  LINE     SOURCE

                     =1    209      ;
                     =1    210      ;    .....................................................
                     =1    211      ;    |          8"  FLEXIBLE  DISK DRIVES                |
                     =1    212      ;    .....................................................
                     =1    213      ;    |  BYTES PER SECTOR   |    MAXIMUM SECTORS PER TRACK   |
                     =1    214      ;    |---------------------+--------------------------------|
                     =1    215      ;    |        128          |            26 (FM)             |
                     =1    216      ;    |        256          |            26 (MFM)            |
                     =1    217      ;    |        512          |            15 (MFM)            |
                     =1    218      ;    |       1024          |            8  (MFM)            |
                     =1    219      ;    .....................................................
                     =1    220      ;-------------------------------------------------------------------------------
                     =1    221      ;
                     =1    222      ; DRIVE #0 ---SHUGART MODEL 850     (1.0 MEGABYTES STORAGE)
                     =1    223      ;
0020 4D00            =1    224              DW      77                      ; NUMBER OF CYLINDERS
0022 00              =1    225              DB      0                       ; NUMBER OF FIXED READ/WRITE SURFACES
0023 02              =1    226              DB      2                       ; NUMBER OF REMOVABLE R/W SURFACES
0024 1A              =1    227              DB      26                      ; NUMBER OF SECTORS PER TRACK
0025 0001            =1    228              DW      256                     ; NUMBER OF BYTES PER SECTOR
0027 01              =1    229              DB      01                      ; MFM(1) OR FM(0) RECORDING MODE
                     =1    230      ;
                     =1    231      ; DRIVE #1 ---SHUGART MODEL 850    (1.0) MEGABYTES STORAGE)
                     =1    232      ;
0028 4D00            =1    233              DW      77                      ; NUMBER OF CYLINDERS
002A 00              =1    234              DB      0                       ; NUMBER OF FIXED READ/WRITE SURFACES
002B 02              =1    235              DB      2                       ; NUMBER OF REMOVABLE R/W SURFACES
002C 1A              =1    236              DB      26                      ; NUMBER OF SECTORS PER TRACK
002D 8000            =1    237              DW      128                     ; NUMBER OF BYTES PER SECTOR
002F 00              =1    238              DB      00                      ; MFM(1) OR FM(0) RECORDING MODE
                     =1    239      ;
                     =1    240      ; DRIVE #2 --- NONEXISTENT
                     =1    241      ;
0030 0000            =1    242              DW      0000H                   ; NUMBER OF CYLINDERS
0032 00              =1    243              DB      00H                     ; NUMBER OF FIXED READ/WRITE SURFACES
0033 00              =1    244              DB      00H                     ; NUMBER OF REMOVABLE R/W SURFACES
0034 00              =1    245              DB      00H                     ; NUMBER OF SECTORS PER TRACK
0035 0000            =1    246              DW      0000H                   ; NUMBER OF BYTES PER SECTOR
0037 00              =1    247              DB      00H                     ; MFM(1) OR FM(0) RECORDING MODE
                     =1    248      ;
                     =1    249      ; DRIVE #3 --- NONEXISTENT
                     =1    250      ;
0038 0000            =1    251              DW      0000H                   ; NUMBER OF CYLINDERS
003A 00              =1    252              DB      00H                     ; NUMBER OF FIXED READ/WRITE SURFACES
003B 00              =1    253              DB      00H                     ; NUMBER OF REMOVABLE R/W SURFACES
003C 00              =1    254              DB      00H                     ; NUMBER OF SECTORS PER TRACK
003D 0000            =1    255              DW      0000H                   ; NUMBER OF BYTES PER SECTOR
                     =1    256      ;       DB      00H                     ; MFM(1) OR FM(0) RECORDING MODE
                     =1    257      ;
----                 =1    258      INITBLSEG       ENDS
                           259      ;
                           260 +1   $INCLUDE(:F1:DATSEG.MMD)
                     =1    261 +1   $EJECT TITLE(DATA SEGMENT)
```

```
MCS-86 MACRO ASSEMBLER    DATA SEGMENT                                10/27/80  PAGE    6

LOC  OBJ                  LINE     SOURCE

                     =1    262      ; -------------------------
                     =1    263      ; |                       |
                     =1    264      ; |     DATA SEGMENT      |
                     =1    265      ; |                       |
                     =1    266      ; -------------------------
                     =1    267      ;
----                 =1    268      DATASEG SEGMENT
                     =1    269      ;
                     =1    270      ;       THIS SEGMENT CONTAINS VARIOUS DATA THAT ARE USED BY THE iSBC 215 DRIVER
                     =1    271      ;       SOFTWARE.
                     =1    272      ;
                     =1    273      ; - THE FLAGS ARE SET BY THE INTERRUPT SERVICE ROUTINE, AND ARE COPIES OF THE
                     =1    274      ;       CIB STATUS POSTED BY THE iSBC 215.  THE ROUTINES THAT USE THE FLAGS ARE
                     =1    275      ;       RESPONSIBLE FOR CLEARING THEM AFTER USE.
                     =1    276      ;-------------------------------------------------------------------------------
                     =1    277      ;
                     =1    278              PUBLIC  OPCMP,SKCMP,PKCHG,ERRSTS
                     =1    279      ;
                     =1    280      ;       OPERATION COMPLETE FLAGS
                     =1    281      ;
0000                 =1    282      OPCMP   LABEL   BYTE
0000 00              =1    283      OPCMP0  DB      00H                     ; OPERATION COMPLETE ON UNIT 0
0001 00              =1    284      OPCMP1  DB      00H                     ; OPERATION COMPLETE ON UNIT 1
0002 00              =1    285      OPCMP2  DB      00H                     ; OPERATION COMPLETE ON UNIT 2
0003 00              =1    286      OPCMP3  DB      00H                     ; OPERATION COMPLETE ON UNIT 3
                     =1    287      ;
                     =1    288      ;       SEEK COMPLETE FLAGS
                     =1    289      ;
0004                 =1    290      SKCMP   LABEL   BYTE
0004 00              =1    291      SKCMP0  DB      00H                     ; SEEK COMPLETE ON UNIT 0
0005 00              =1    292      SKCMP1  DB      00H                     ; SEEK COMPLETE ON UNIT 1
0006 00              =1    293      SKCMP2  DB      00H                     ; SEEK COMPLETE ON UNIT 2
0007 00              =1    294      SKCMP3  DB      00H                     ; SEEK COMPLETE ON UNIT 3
                     =1    295      ;
                     =1    296      ;       PACK CHANGE FLAGS
                     =1    297      ;
0008                 =1    298      PKCHG   LABEL   BYTE
0008 00              =1    299      PKCHG0  DB      00H                     ; PACK CHANGE ON UNIT 0
0009 00              =1    300      PKCHG1  DB      00H                     ; PACK CHANGE ON UNIT 1
000A 00              =1    301      PKCHG2  DB      00H                     ; PACK CHANGE ON UNIT 2
000B 00              =1    302      PKCHG3  DB      00H                     ; PACK CHANGE ON UNIT 3
                     =1    303      ;
                     =1    304      ;       ERROR STATUS BLOCK
                     =1    305      ;       (LOADED FROM CONTROLLER BY ERROR HANDLER)
                     =1    306      ;
000C 0000            =1    307      ERRSTS  DW      0000H                   ; ERROR STATUS WORD
000E 00              =1    308      SFERST  DB      00H                     ; SOFT ERROR STATUS BYTE
000F 0000            =1    309      DESCYL  DW      0000H                   ; DESIRED CYLINDER
0011 00              =1    310      DESHD   DB      00H                     ; DESIRED HEAD
0012 00              =1    311      DESSEC  DB      00H                     ; DESIRED SECTOR
0013 0000            =1    312      ACTCYL  DW      0000H                   ; ACTUAL CYLINDER + FLAG BITS
0015 00              =1    313      ACTHD   DB      00H                     ; ACTUAL HEAD
0016 00              =1    314      ACTSEC  DB      00H                     ; ACTUAL SECTOR
0017 00              =1    315      NMRTRY  DB      00H                     ; NUMBER OF RETRIES MADE
                     =1    316      ;
                     =1    317      ;       LAST OPERATION COMPLETE BYTE
                     =1    318      ;       (COPIED FROM CIB BY WAIT215)
                     =1    319      ;
0018 00              =1    320      LSTSTS  DB      00H
                     =1    321      ;
0019 90              =1    322              EVEN
                     =1    323      ;
001A                 =1    324      ENDDAT  LABEL   FAR                     ; END OF DATA SEGMENT
                     =1    325      ;
----                 =1    326      DATASEG ENDS
                           327      ;
                           328 +1   $INCLUDE(:F1:USER.MMD)
                     =1    329 +1   $EJECT TITLE(SYSTEM DEPENDENT INITIALIZATION)
```

```
MCS-86 MACRO ASSEMBLER    SYSTEM DEPENDENT INITIALIZATION                    10/27/80   PAGE    7


LOC  OBJ                  LINE     SOURCE

                     =1    330     ; ------------------------------------------
                     =1    331     ; |                                        |
                     =1    332     ; |     SYSTEM DEPENDENT INITIALIZATION    |
                     =1    333     ; |                                        |
                     =1    334     ; ------------------------------------------
                     =1    335     ;
                     =1    336     ;       THIS ROUTINE SETS UP THE INTERRUPT VECTOR FOR AN iSBC 86/12A CPU
                     =1    337     ;       RUNNING UNDER THE iSBC 957A INTERFACE/EXECUTION PACKAGE.
                     =1    338     ;
                     =1    339     ; - THE 8259 INTERRUPT CONTROLLER AND OTHER INITIALZATIONS ARE PERFORMED
                     =1    340     ;       BY THE iSBC 957A FIRMWARE.
                     =1    341     ; ------------------------------------------------------------------------------
                     =1    342     ;
                     =1    343     ;       ===========================
                     =1    344     ;       INTERRUPT VECTOR DEFINITION
                     =1    345     ;       ===========================
                     =1    346     ;
  0005               =1    347             INTRPT  EQU     5               ; iSBC 220 INTERRUPT NUMBER
                     =1    348     ;
----                 =1    349     SEG0000 SEGMENT AT 0000H                ; INTERRUPT VECTORS ARE FROM ABSOLUTE
                     =1    350                                             ;    ADDRESSES 00000H TO 00FF0H
                     =1    351     ;
0094                 =1    352             ORG     80H + 4*INTRPT          ; LOCATION OF INTERRUPT VECTOR WITH
                     =1    353                                             ;    iSBC 957A FIRMWARE
0094 0000            =1    354     INTRIP  DW      0000H                   ; - INSTRUCTION POINTER
0096 0000            =1    355     INTRCS  DW      0000H                   ; - CODE SEGMENT
                     =1    356     ;
----                 =1    357     SEG0000 ENDS
                     =1    358     ;
                     =1    359     ;       ================
                     =1    360     ;       STACK ALLOCATION
                     =1    361     ;       ================
                     =1    362     ;
----                 =1    363     STACK   SEGMENT                         ; STACK SEGMENT
                     =1    364     ;
0000 (64             =1    365             DB      64 DUP(00H)             ; ALLOW 64 BYTES FOR STACK
     00
     )
                     =1    366     ;
0040                 =1    367     ENDSTK  LABEL   FAR
                     =1    368     ;
----                 =1    369     STACK   ENDS
                     =1    370     ;
                     =1    371     ;       ========================================
                     =1    372     ;       STACK AND INTERRUPT CONFIGURATION ROUTINE
                     =1    373     ;       ========================================
                     =1    374     ;
----                 =1    375     USERSEG SEGMENT
                     =1    376     ;
                     =1    377             PUBLIC  CONFIG
                     =1    378             ASSUME  DS:SEG0000
                     =1    379     ;
0000                 =1    380     CONFIG  PROC FAR
                     =1    381     ;
0000 FA              =1    382             CLI                             ; DISABLE INTERRUPTS WHILE SETTING UP
0001 B8----        R =1    383             MOV     AX,STACK                ;;; SET UP STACK
0004 8ED0            =1    384             MOV     SS,AX                   ;;;
0006 BC4000          =1    385             MOV     SP,OFFSET ENDSTK        ;;;
0009 B80000          =1    386             MOV     AX,0000H                ;;; GET POINTER TO SEGMENT 0000H
000C 8ED8            =1    387             MOV     DS,AX                   ;;;
000E C70694003D02    =1    388             MOV     INTRIP,OFFSET INT215    ;;; SET UP INTERRUPT VECTOR
0014 C7069600----  R =1    389             MOV     INTRCS,SEG INT215       ;;;
001A E4C2            =1    390             IN      AL,0C2H                 ;;; INPUT INTERRUPT MASK FROM 8259
001C 24DF            =1    391             AND     AL,11011111B            ;;; ENABLE INTERRUPT 5
001E E6C2            =1    392             OUT     0C2H,AL                 ;;; WRITE NEW MASK OUT TO 8259
0020 FB              =1    393             STI                             ;;; ENABLE INTERRUPTS
0021 CC              =1    394             INT     3                       ;;; GO TO MONITOR
                     =1    395     ;
                     =1    396     CONFIG  ENDP
                     =1    397     ;
----                 =1    398     USERSEG ENDS
                           399     ;
----                       400     SBC215DRIVER    SEGMENT
                           401     ;
                           402             ASSUME CS:SBC215DRIVER
                           403     ;
                           404 +1  $INCLUDE(:F1:RESET.MMD)
                     =1    405 +1  $EJECT TITLE(CONTROLLER RESET ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    CONTROLLER RESET ROUTINE                    10/27/80   PAGE    8

LOC  OBJ                  LINE     SOURCE

                          =1   406      ; ------------------------------------
                          =1   407      ; |                                  |
                          =1   408      ; |     CONTROLLER RESET ROUTINE     |
                          =1   409      ; |                                  |
                          =1   410      ; ------------------------------------
                          =1   411      ;
                          =1   412      ;       RES215 SETS UP THE COMMUNICATION BLOCKS FOR THE iSBC 215, LINKS THEM
                          =1   413      ;       TOGETHER AND GIVES A RESET, CLEAR RESET, CHANNEL ATTENTION SEQUENCE TO
                          =1   414      ;       THE CONTROLLER.  THIS CAUSES THE 8089 ON THE CONTROLLER TO SET UP ITS
                          =1   415      ;       INTERNAL POINTER TO THE CCB BY THREADING DOWN THE LINKS STARTING WITH
                          =1   416      ;       THE SWITCHES ON THE CONTROLLER.  SUBSEQUENT CA's WILL CAUSE THE 8089 TO
                          =1   417      ;       FETCH ITS POINTERS STARTING AT THE CCB.
                          =1   418      ;
                          =1   419      ; - IF THE CH. 1 BUSY FLAG IS NOT CLEARED WITHIN A "REASONABLE" AMOUNT OF TIME,
                          =1   420      ;       THEN THE iSBC 215 IS PROBABLY NOT RESPONDING TO THE CHANNEL ATTENTION.
                          =1   421      ;       ON THE CONTROLLER: CHECK SWITCH SETTINGS; VOLTAGES; RESET, CLEAR RESET,
                          =1   422      ;       CHANNEL ATTENTION SIGNALS; READY INPUT TO 8089; 8089 STATUS LINES; R/W
                          =1   423      ;       STROBES.
                          =1   424      ;
                          =1   425      ; - THE SYSTEM INTERRUPT LOGIC AND VECTORS FOR THE CONTROLLER ARE ASSUMED TO BE
                          =1   426      ;       CONFIGURED BY AN EXTERNAL PROGRAM.
                          =1   427      ;
                          =1   428      ; INPUT DATA:
                          =1   429      ;       NONE
                          =1   430      ;
                          =1   431      ; OUTPUT DATA:
                          =1   432      ;       CARRY FLAG:     = 0 IF RESET OKAY
                          =1   433      ;                       = 1 IF CH. 1 BUSY FLAG NOT RESET (NOT RESPONDING)
                          =1   434      ; ------------------------------------------------------------------------------
                          =1   435      ;
                          =1   436              PUBLIC  RES215
                          =1   437      ;
0000                      =1   438      RES215  PROC    FAR
                          =1   439      ;
0000 50                   =1   440              PUSH    AX                      ; SAVE REGISTERS
0001 53                   =1   441              PUSH    BX
0002 51                   =1   442              PUSH    CX
0003 52                   =1   443              PUSH    DX
0004 1E                   =1   444              PUSH    DS
                          =1   445      ;
                          =1   446      ;       SET UP LINKS BETWEEN COMMUNICATION BLOCKS
                          =1   447      ;
                          =1   448      ; SCB
                          =1   449      ;
                          =1   450              ASSUME  DS:SCBSEG
0005 B83506               =1   451              MOV     AX,SCBSEG               ; GET POINTER TO SCB
0008 8ED8                 =1   452              MOV     DS,AX
000A C706000000100        =1   453              MOV WORD PTR SOC,0001H          ; SET SOC BYTE AND CLEAR RESERVED BYTE
0010 C70602000000         =1   454              MOV WORD PTR CCBPTR,OFFSET CCB  ; SET POINTER TO CCB
0016 C7060400----      R  =1   455              MOV WORD PTR CCBPTR+2,SEG CCB
                          =1   456      ;
                          =1   457      ; CCB
                          =1   458      ;
001C C5060200             =1   459              LDS     AX,CCBPTR               ; GET POINTER TO CCB
                          =1   460              ASSUME  DS:CCBSEG
0020 C706000001FF         =1   461              MOV WORD PTR CCW1,0FF01H        ; SET CCW1 AND CH. 1 BUSY FLAG
0026 C70602000400         =1   462              MOV WORD PTR CH1PTR,OFFSET CH1PC; SET POINTER TO FIFTH BYTE OF CIB
002C C7060400----      R  =1   463              MOV WORD PTR CH1PTR+2,SEG CH1PC ;    (HAS STARTING ADDRESS FOR CH. 1)
0032 C70608000100         =1   464              MOV WORD PTR CCW2,0001H         ; SET CCW2 AND CLEAR CH. 2 BUSY FLAG
0038 C7060A000E00         =1   465              MOV WORD PTR CH2PTR,OFFSET CH2PC; SET POINTER TO CH. 2 STARTING ADDRESS
003E C7060C00----      R  =1   466              MOV WORD PTR CH2PTR+2,SEG CH2PC
0044 C7060E000400         =1   467              MOV WORD PTR CH2PC,0004H        ; SET CH. 2 STARTING ADDRESS
                          =1   468      ;
                          =1   469      ; CIB
                          =1   470      ;
                          =1   471              ASSUME  DS:CIBSEG
004A B8----            R  =1   472              MOV     AX,CIBSEG               ; GET POINTER TO CIB
004D 8ED8                 =1   473              MOV     DS,AX
004F C70600000000         =1   474              MOV WORD PTR CIBCMD,0000H       ; CLEAR CIB COMMAND AND CIB STATUS BYTES
0055 C70602000000         =1   475              MOV WORD PTR CMDSEM,0000H       ;    ...AND SEMAPHORES
005B C70604000000         =1   476              MOV WORD PTR CH1PC,0000H        ; SET CH. 1 STARTING ADDRESS
0061 C70608000000         =1   477              MOV     IOPBOFF,OFFSET IOPB     ; SET IOPB POINTER
0067 C7060A00----      R  =1   478              MOV     IOPBSG,SEG IOPB
                          =1   479      ;
                          =1   480 +1   $EJECT
```

```
MCS-86 MACRO ASSEMBLER    CONTROLLER RESET ROUTINE                    10/27/80   PAGE    9


LOC  OBJ                  LINE     SOURCE

                     =1    481     ;       CLEAR OUT DATA SEGMENT
                     =1    482     ;
                     =1    483             ASSUME  DS:DATASEG
006D B8----        R =1    484             MOV     AX,DATASEG              ; GET POINTER TO DATA SEGMENT
0070 8ED8            =1    485             MOV     DS,AX
0072 B90D00          =1    486             MOV     CX,(OFFSET ENDDAT)/2    ; GET COUNT (# WORDS IN DATA SEGMENT)
0075 BB0000          =1    487             MOV     BX,0000H                ; CLEAR INDEX REGISTER
0078 C7070000        =1    488     CLRLP:  MOV WORD PTR [BX],0000H         ; CLEAR NEXT WORD IN DATA SEGMENT
007C 43              =1    489             INC     BX                      ; POINT TO NEXT WORD
007D 43              =1    490             INC     BX
007E E0F8            =1    491             LOOPNE  CLRLP                   ; DONE?
                     =1    492                                             ; NO--CLEAR ANOTHER WORD
                     =1    493                                             ; YES--INITIALIZE COMMUNICATION LINKS
                     =1    494     ;
                     =1    495     ;       OUTPUT RESET/CLEAR RESET/CHANNEL ATTENTION TO CONTROLLER
                     =1    496     ;
0080 BA3506          =1    497             MOV     DX,WUA                  ; GET WAKE-UP I/O PORT ADDRESS
0083 B002            =1    498             MOV     AL,02H                  ; GET RESET COMMAND BYTE
0085 EE              =1    499             OUT     DX,AL                   ; OUTPUT TO WAKE-UP I/O PORT
0086 B000            =1    500             MOV     AL,00H                  ; GET CLEAR RESET COMMAND BYTE
0088 EE              =1    501             OUT     DX,AL                   ; OUTPUT TO WAKE-UP I/O PORT
0089 B001            =1    502             MOV     AL,01H                  ; GET CHANNEL ATTENTION COMMAND BYTE
008B EE              =1    503             OUT     DX,AL                   ; OUTPUT  TO WAKE-UP I/O PORT
                     =1    504             ASSUME  DS:CCBSEG
008C B8----        R =1    505             MOV     AX,CCBSEG               ; GET POINTER TO CCB
008F 8ED8            =1    506             MOV     DS,AX
                     =1    507                                             ; (OTHER IMPLEMENTATIONS OF RES215 COULD
                     =1    508                                             ; INITIALIZE OTHER DEVICES WHILE THE
                     =1    509                                             ; iSBC 215 DOES ITS RESET SEQUENCE HERE)
0091 B90010          =1    510             MOV     CX,1000H                ; SET TIME-OUT COUNTER
0094 F8              =1    511             CLC                             ; CLEAR CARRY FLAG
0095 F6060100FF      =1    512     RESLP:  TEST    BSYFLG1,00FFH           ; CHECK CH. 1 BUSY FLAG:
                     =1    513                                             ;    ZERO FLAG = BSYFLG1 & FFH
009A 7403            =1    514             JZ      RESDN                   ; BUSY FLAG CLEARED?
                     =1    515                                             ; YES--RETURN CARRY CLEAR
009C E0F7            =1    516             LOOPNE  RESLP                   ; NO--DECREMENT COUNTER
                     =1    517                                             ; IF CX = 0, THEN BSYFLG1 NEVER GOT
009E F9              =1    518             STC                             ;    CLEARED, SO SET CARRY FLAG
009F 1F              =1    519     RESDN:  POP     DS                      ; RESTORE REGISTERS
00A0 5A              =1    520             POP     DX
00A1 59              =1    521             POP     CX
00A2 5B              =1    522             POP     BX
00A3 58              =1    523             POP     AX
00A4 CB              =1    524             RET                             ; RETURN
                     =1    525     ;
                     =1    526     RES215  ENDP
                           527     ;
                           528 +1  $INCLUDE(:F1:INITEX.MMD)
                     =1    529 +1  $EJECT TITLE(INITIALIZATION ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    INITIALIZATION ROUTINE                    10/27/80   PAGE   10

LOC  OBJ                  LINE     SOURCE

                     =1    530      ; ------------------------------------
                     =1    531      ; |                                  |
                     =1    532      ; |      INITIALIZATION ROUTINE      |
                     =1    533      ; |                                  |
                     =1    534      ; ------------------------------------
                     =1    535      ;
                     =1    536      ;       INIT215 INITIALIZES THE iSBC 215 CONTROLLER BY LOADING PERTINENT INFOR-
                     =1    537      ;       MATION ABOUT THE DISK DRIVE(S) ATTACHED.
                     =1    538      ;
                     =1    539      ; - IF A DRIVE THAT IS SPECIFIED AS PRESENT WILL NOT RESPOND, INIT215 RETURNS
                     =1    540      ;       IMMEDIATELY WITH THE CARRY FLAG SET.
                     =1    541      ;
                     =1    542      ; INPUT DATA:
                     =1    543      ;       DISK DRIVE INITIALIZATION TABLES, IN SEGMENT "INITBLSEG".
                     =1    544      ;
                     =1    545      ; OUTPUT DATA:
                     =1    546      ;       CARRY FLAG              = 0 IF CONTROLLER INITIALIZED SUCCESSFULLY
                     =1    547      ;                               = 1 IF INITIALIZATION ERROR
                     =1    548      ; ------------------------------------------------------------------------------
                     =1    549      ;
                     =1    550              PUBLIC  INIT215
                     =1    551              ASSUME  DS:IOPBSEG
                     =1    552      ;
00A5                 =1    553      INIT215 PROC    FAR
                     =1    554      ;
00A5 50              =1    555              PUSH    AX                      ; SAVE REGISTERS
00A6 1E              =1    556              PUSH    DS
00A7 B8----        R =1    557              MOV     AX,IOPBSEG              ; GET POINTER TO IOPB
00AA 8ED8            =1    558              MOV     DS,AX                   ; PUT IN DS REGISTER
00AC C706080000000   =1    559              MOV     DEVCOD,00H              ; WINCHESTER DRIVES INITIALIZED FIRST
00B2 C6060B0000      =1    560              MOV     FUNC,00H                ; SET IOPB FUNCTION BYTE = INITIALIZE
00B7 C7060C000000    =1    561              MOV     MODIFY,0000H            ; CLEAR MODIFIER (ENABLE RETRIES AND
                     =1    562                                              ;    INTERRUPT ON COMPLETION)
00BD C7061400----  R =1    563              MOV     BUFSEG,INITBLSEG        ; PUT INITIALIZATION TABLES' SEGMENT IN
                     =1    564                                              ;   IOPB DATA BUFFER POINTER
00C3 C7061200F8FF    =1    565              MOV     BUFOFF,-8               ; START INITIALIZE WITH UNIT 0
00C9 B000            =1    566              MOV     AL,00H                  ; CLEAR UNIT COUNTER
00CB 8306120008      =1    567      INITLP: ADD     BUFOFF,8                ; POINT TO NEXT DRIVE's INITIALIZE TABLE
00D0 A20A00          =1    568              MOV     UNIT,AL                 ; PUT UNIT INTO IOPB
00D3 E8EC00          =1    569              CALL    GO215                   ; DO INITIALIZE
                     =1    570                                              ;    (RETURNS CARRY FLAG SET OR CLEAR)
00D6 7214            =1    571              JC      INITDN                  ; UNIT INITIALIZED?
                     =1    572                                              ; NO--TERMINATE WITH CARRY BIT SET
00D8 40              =1    573              INC     AX                      ; YES--INCREMENT UNIT COUNTER
00D9 3C04            =1    574              CMP     AL,4                    ; CHECK UNIT COUNTER (CLEARS CARRY)
00DB 75EE            =1    575              JNZ     INITLP                  ; LAST DRIVE INITIALIZED?
                     =1    576                                              ; NO--INITIALIZE NEXT DRIVE
00DD A10800          =1    577              MOV     AX,DEVCOD               ; YES--FLOPPIES INITIALIZED YET?
00E0 3C00            =1    578              CMP     AL,0
00E2 7508            =1    579              JNZ     INITDN                  ; YES--INITIALIZE FUNCTION FINISHED
00E4 C706080000100   =1    580              MOV     DEVCOD,01               ; NO---INITIALIZE FLOPPY DRIVES
00EA EBDF            =1    581              JMP     INITLP
00EC 1F              =1    582      INITDN: POP     DS                      ; RESTORE REGISTERS
00ED 58              =1    583              POP     AX
00EE CB              =1    584              RET                             ; RETURN
                     =1    585      ;
                     =1    586      INIT215 ENDP
                           587      ;
                           588 +1   $INCLUDE(:F1:FORMAT.MMD)
                     =1    589 +1   $EJECT TITLE(FORMAT TRACK ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    FORMAT TRACK ROUTINE                    10/27/80   PAGE   11

LOC  OBJ                  LINE     SOURCE

                     =1    590     ; --------------------------------
                     =1    591     ; |                              |
                     =1    592     ; |     FORMAT TRACK ROUTINE     |
                     =1    593     ; |                              |
                     =1    594     ; --------------------------------
                     =1    595     ;
                     =1    596     ;       FMTTRK SETS UP THE IOPB FOR A FORMAT TRACK FUNCTION, AND
                     =1    597     ;       INVOKES THE iSBC 215 CONTROLLER TO PERFORM THE OPERATION.
                     =1    598     ;
                     =1    599     ; INPUT DATA:
                     =1    600     ;       BP + 10 =>  DEVICE CODE
                     =1    601     ;       BP + 9  =>  INTERLEAVE FACTOR
                     =1    602     ;       BP + 8  =>  USER DATA BYTE 3
                     =1    603     ;       BP + 7  =>  USER DATA BYTE 2
                     =1    604     ;       BP + 6  =>  USER DATA BYTE 1
                     =1    605     ;       BP + 5  =>  USER DATA BYTE 0
                     =1    606     ;       BP + 4  =>  TYPE OF FORMAT
                     =1    607     ;       BP + 3  =>  HEAD
                     =1    608     ;       BP + 1  =>  CYLINDER
                     =1    609     ;       BP      =>  UNIT
                     =1    610     ;
                     =1    611     ; OUTPUT DATA:
                     =1    612     ;       CARRY FLAG      = 0 IF TRACK FORMATTED SUCCESSFULLY
                     =1    613     ;                       = 1 IF NON-RECOVERABLE ERROR OCCURRED
                     =1    614     ;
                     =1    615     ; - INTERLEAVE FACTOR OF 1 IMPLIES SEQUENTIAL SECTOR NUMBERING.
                     =1    616     ; - USER DATA BYTES 0 - 3 ARE REPLICATED THROUGHOUT THE DATA FIELD.
                     =1    617     ; - INTERLEAVE TYPES:
                     =1    618     ;       00 = NORMAL TRACK (ONLY FORMAT FOR FLOPPY)
                     =1    619     ;       40 = ALTERNATE TRACK (POINTED TO BY EXACTLY ONE DEFECTIVE TRACK,
                     =1    620     ;               CANNOT SUBSEQUENTLY BE FORMATTED DEFECTIVE)
                     =1    621     ;       80 = DEFECTIVE TRACK (DATA FIELD POINTS TO ALTERNATE TRACK)
                     =1    622     ; - TO SET UP A POINTER TO AN ALTERNATE TRACK, SET:
                     =1    623     ;       USER DATA BYTE 0 = ALTERNATE CYLINDER LOW BYTE
                     =1    624     ;       USER DATA BYTE 1 = ALTERNATE CYLINDER HIGH BYTE
                     =1    625     ;       USER DATA BYTE 2 = ALTERNATE HEAD
                     =1    626     ;       USER DATA BYTE 3 = 00H
                     =1    627     ; ------------------------------------------------------------------------------
                     =1    628     ;
                     =1    629             PUBLIC  FMT215
                     =1    630             ASSUME  DS:IOPBSEG
                     =1    631     ;
00EF                 =1    632     FMT215  PROC    FAR
                     =1    633     ;
00EF 50              =1    634             PUSH    AX                      ; SAVE REGISTERS
00F0 1E              =1    635             PUSH    DS
00F1 B8----        R =1    636             MOV     AX,IOPBSEG              ; GET POINTER TO IOPB
00F4 8ED8            =1    637             MOV     DS,AX
00F6 8B460A          =1    638             MOV     AX,[BP+10]              ; GET DEVICE CODE INTO IOPB
00F9 A30800          =1    639             MOV     DEVCOD,AX
00FC 8A4600          =1    640             MOV     AL,[BP]                 ; GET UNIT NUMBER INTO IOPB
00FF A20A00          =1    641             MOV     UNIT,AL
0102 8B4601          =1    642             MOV     AX,[BP+1]               ; GET CYLINDER NUMBER INTO IOPB
0105 A30E00          =1    643             MOV     CYLNDR,AX
0108 8A4603          =1    644             MOV     AL,[BP+3]               ; GET HEAD INTO IOPB
010B A21000          =1    645             MOV     HEAD,AL
010E 892E1200        =1    646             MOV     BUFOFF,BP               ; GET POINTER TO FORMAT ARGUMENT LIST
0112 8306120004      =1    647             ADD     BUFOFF,4                ;    INTO DATA BUFFER POINTER
0117 8C161400        =1    648             MOV     BUFSEG,SS
011B C6060B0002      =1    649             MOV     FUNC,02H                ; SET FUNCTION = FORMAT
0120 C7060C000000    =1    650             MOV     MODIFY,0000H            ; CLEAR MODIFIER (ALLOW ERROR RECOVERY
                     =1    651                                             ;    AND INTERRUPT ON COMPLETION)
0126 E89900          =1    652             CALL    GO215                   ; START iSBC 215 AND WAIT FOR DONE
                     =1    653                                             ;    (RETURNS CARRY FLAG SET OR CLEAR)
0129 1F              =1    654     FMTDN:  POP     DS                      ; RESTORE REGISTERS
012A 58              =1    655             POP     AX
012B CA0A00          =1    656             RET     10                      ; RETURN (AND POP INPUT DATA OFF STACK)
                     =1    657     ;
                     =1    658     FMT215  ENDP
                           659     ;
                           660 +1  $INCLUDE(:F1:RDWRT.MMD)
                     =1    661 +1  $EJECT TITLE(READ DATA ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    READ DATA ROUTINE                          10/27/80   PAGE   12

LOC  OBJ                  LINE     SOURCE

                          =1   662      ; ---------------------
                          =1   663      ; |                   |
                          =1   664      ; |     READ DATA     |
                          =1   665      ; |                   |
                          =1   666      ; ---------------------
                          =1   667      ;
                          =1   668      ;       RD215 SETS UP THE IOPB FOR A READ OPERATION, AND
                          =1   669      ;       INVOKES THE iSBC 215 TO PERFORM THE OPERATION.
                          =1   670      ;
                          =1   671      ; INPUT DATA:
                          =1   672      ;       BP + 13 =>  DEVICE CODE
                          =1   673      ;       BP + 11 =>  BYTE COUNT HIGH WORD
                          =1   674      ;       BP +  9 =>  BYTE COUNT LOW WORD
                          =1   675      ;       BP +  7 =>  DATA BUFFER SEGMENT
                          =1   676      ;       BP +  5 =>  DATA BUFFER OFFSET
                          =1   677      ;       BP +  4 =>  SECTOR
                          =1   678      ;       BP +  3 =>  HEAD
                          =1   679      ;       BP +  1 =>  CYLINDER
                          =1   680      ;       BP      =>  UNIT
                          =1   681      ;
                          =1   682      ; OUTPUT DATA:
                          =1   683      ;       CARRY FLAG      = 0 IF TRANSFER OCCURRED WITH NO OR RECOVERABLE ERROR
                          =1   684      ;                       = 1 IF UNRECOVERABLE ERROR OCCURRED
                          =1   685      ;       DATA BUFFER     FILLED WITH DATA FROM DISK IF NO UNRECOVERABLE ERROR
                          =1   686      ; ------------------------------------------------------------------------------
                          =1   687      ;
                          =1   688              PUBLIC  RD215
                          =1   689              ASSUME  DS:IOPBSEG
                          =1   690      ;
012E                      =1   691      RD215   PROC    FAR
                          =1   692      ;
012E 50                   =1   693              PUSH    AX                      ; SAVE REGISTERS
012F 1E                   =1   694              PUSH    DS
0130 B8----         R     =1   695              MOV     AX,IOPBSEG              ; GET POINTER TO IOPB
0133 8ED8                 =1   696              MOV     DS,AX
0135 8B460D               =1   697              MOV     AX,[BP+13]              ; GET DEVICE CODE INTO IOPB
0138 A30800               =1   698              MOV     DEVCOD,AX
013B 8A4600               =1   699              MOV     AL,[BP]                 ; GET UNIT INTO IOPB
013E A20A00               =1   700              MOV     UNIT,AL
0141 8B4601               =1   701              MOV     AX,[BP+1]               ; GET CYLINDER INTO IOPB
0144 A30E00               =1   702              MOV     CYLNDR,AX
0147 8B4603               =1   703              MOV     AX,[BP+3]               ; GET HEAD AND SECTOR INTO IOPB
014A A31000               =1   704              MOV WORD PTR HEAD,AX
014D 8B4605               =1   705              MOV     AX,[BP+5]               ; GET DATA BUFFER POINTER INTO IOPB
0150 A31200               =1   706              MOV     BUFOFF,AX
0153 8B4607               =1   707              MOV     AX,[BP+7]
0156 A31400               =1   708              MOV     BUFSEG,AX
0159 8B4609               =1   709              MOV     AX,[BP+9]               ; GET BYTE COUNT INTO IOPB
015C A31600               =1   710              MOV WORD PTR REQCNT,AX
015F 8B460B               =1   711              MOV     AX,[BP+11]
0162 A31800               =1   712              MOV WORD PTR REQCNT+2,AX
0165 C7060C000000         =1   713              MOV     MODIFY,0000H            ; CLEAR MODIFIER (ENABLE INTERRUPT ON
                          =1   714                                              ;     COMPLETION AND RETRIES)
016B C6060B0004           =1   715              MOV     FUNC,04H                ; SET FUNCTION = READ DATA
0170 E84F00               =1   716              CALL    GO215                   ; START FUNCTION AND WAIT FOR COMPLETION
                          =1   717                                              ;    (RETURNS CARRY FLAG SET OR CLEAR)
0173 1F                   =1   718              POP     DS                      ; RESTORE REGISTERS
0174 58                   =1   719              POP     AX
0175 CA0D00               =1   720              RET     13                      ; POP PARAMETERS OFF STACK AND RETURN
                          =1   721      ;
                          =1   722      RD215   ENDP
                          =1   723      ;
                          =1   724 +1   $EJECT TITLE(WRITE DATA ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    WRITE DATA ROUTINE                    10/27/80   PAGE   13

LOC  OBJ                  LINE     SOURCE

                     =1    725     ; ---------------------
                     =1    726     ; |                   |
                     =1    727     ; |     WRITE DATA    |
                     =1    728     ; |                   |
                     =1    729     ; ---------------------
                     =1    730     ;
                     =1    731     ;       WRT215 SETS UP THE IOPB FOR A WRITE OPERATION, AND
                     =1    732     ;       INVOKES THE iSBC 215 TO PERFORM THE OPERATION.
                     =1    733     ;
                     =1    734     ; INPUT DATA:
                     =1    735     ;       BP + 13 =>  DEVICE CODE
                     =1    736     ;       BP + 11 =>  BYTE COUNT HIGH WORD
                     =1    737     ;       BP +  9 =>  BYTE COUNT LOW WORD
                     =1    738     ;       BP +  7 =>  DATA BUFFER SEGMENT
                     =1    739     ;       BP +  5 =>  DATA BUFFER OFFSET
                     =1    740     ;       BP +  4 =>  SECTOR
                     =1    741     ;       BP +  3 =>  HEAD
                     =1    742     ;       BP +  1 =>  CYLINDER
                     =1    743     ;       BP      =>  UNIT
                     =1    744     ;
                     =1    745     ;       DATA BUFFER CONTAINS INFORMATION TO BE WRITTEN TO DISK
                     =1    746     ;
                     =1    747     ; OUTPUT DATA:
                     =1    748     ;       CARRY FLAG      = 0 IF TRANSFER OCCURRED WITH NO OR RECOVERABLE ERROR
                     =1    749     ;                       = 1 IF UNRECOVERABLE ERROR OCCURRED
                     =1    750     ; ------------------------------------------------------------------------------
                     =1    751     ;
                     =1    752             PUBLIC  WRT215
                     =1    753             ASSUME  DS:IOPBSEG
                     =1    754     ;
0178                 =1    755     WRT215  PROC    FAR
                     =1    756     ;
0178 50              =1    757             PUSH    AX                      ; SAVE REGISTERS
0179 1E              =1    758             PUSH    DS
017A 88----        R =1    759             MOV     AX,IOPBSEG              ; GET POINTER TO IOPB
017D 8ED8            =1    760             MOV     DS,AX
017F 8B460D          =1    761             MOV     AX,[BP+13]              ; PUT DEVICE CODE IN IOPB
0182 A30800          =1    762             MOV     DEVCOD,AX
0185 8A4600          =1    763             MOV     AL,[BP]                 ; GET UNIT INTO IOPB
0188 A20A00          =1    764             MOV     UNIT,AL
018B 8B4601          =1    765             MOV     AX,[BP+1]               ; GET CYLINDER INTO IOPB
018E A30E00          =1    766             MOV     CYLNDR,AX
0191 8B4603          =1    767             MOV     AX,[BP+3]               ; GET HEAD AND SECTOR INTO IOPB
0194 A31000          =1    768             MOV WORD PTR HEAD,AX
0197 8B4605          =1    769             MOV     AX,[BP+5]               ; GET DATA BUFFER POINTER INTO IOPB
019A A31200          =1    770             MOV     BUFOFF,AX
019D 8B4607          =1    771             MOV     AX,[BP+7]
01A0 A31400          =1    772             MOV     BUFSEG,AX
01A3 8B4609          =1    773             MOV     AX,[BP+9]               ; GET BYTE COUNT INTO IOPB
01A6 A31600          =1    774             MOV WORD PTR REQCNT,AX
01A9 8B460B          =1    775             MOV     AX,[BP+11]
01AC A31800          =1    776             MOV WORD PTR REQCNT+2,AX
01AF C7060C000000    =1    777             MOV     MODIFY,0000H            ; CLEAR MODIFIER (ENABLE INTERRUPT ON
                     =1    778                                             ;     COMPLETION AND RETRIES)
01B5 C6060B0006      =1    779             MOV     FUNC,06H                ; SET FUNCTION = WRITE DATA
01BA E80500          =1    780             CALL    GO215                   ; START iSBC 215 AND WAIT FOR DONE
                     =1    781                                             ;    (RETURNS WITH CARRY SET OR CLEAR)
01BD 1F              =1    782             POP     DS                      ; RESTORE REGISTERS
01BE 58              =1    783             POP     AX
01BF CA0D00          =1    784             RET     13                      ; POP PARAMETERS OFF STACK AND RETURN
                     =1    785     ;
                     =1    786     WRT215  ENDP
                           787     ;
                           788 +1  $INCLUDE(:F1:CORE.MMD)
                     =1    789 +1  $EJECT TITLE(START FUNCTION AND WAIT FOR COMPLETION)
```

```
MCS-86 MACRO ASSEMBLER    START FUNCTION AND WAIT FOR COMPLETION                10/27/80   PAGE   14

LOC  OBJ                  LINE     SOURCE

                     =1    790      ; -------------------------------------------------
                     =1    791      ; |                                               |
                     =1    792      ; |     START FUNCTION AND WAIT FOR COMPLETION     |
                     =1    793      ; |                                               |
                     =1    794      ; -------------------------------------------------
                     =1    795      ;
                     =1    796      ;       THIS ROUTINE GIVES A CHANNEL ATTENTION (WAKE-UP) TO THE iSBC 215 AND
                     =1    797      ;       WAITS FOR THE FUNCTION SPECIFIED (BY THE CALLING PROCEDURE) TO FINISH.
                     =1    798      ;       IF AN ERROR OCCURRED, THE ERROR HANDLER IS INVOKED.
                     =1    799      ;
                     =1    800      ; INPUTS:
                     =1    801      ;       NONE
                     =1    802      ;
                     =1    803      ; OUTPUTS:
                     =1    804      ;       CARRY FLAG:     = 0 IF NO ERROR OR A RECOVERABLE ERROR OCCURRED
                     =1    805      ;                       = 1 IF UNRECOVERABLE ERROR OCCURRED.
                     =1    806      ; ------------------------------------------------------------------------------
                     =1    807      ;
01C2                 =1    808      GO215   PROC    NEAR
                     =1    809      ;
01C2 50              =1    810              PUSH    AX                      ; SAVE REGISTERS
01C3 52              =1    811              PUSH    DX
01C4 BA3506          =1    812              MOV     DX,WUA                  ; GET ADDRESS OF WAKE-UP I/O PORT
01C7 B001            =1    813              MOV     AL,01H                  ; GET WAKE-UP COMMAND BYTE
01C9 EE              =1    814              OUT     DX,AL                   ; GIVE WAKE-UP TO iSBC 215
01CA E80800          =1    815              CALL    WAIT215                 ; WAIT FOR FUNCTION COMPLETE
01CD 7303            =1    816              JNC     DONE                    ; ERROR?
                     =1    817                                              ; NO--RETURN
01CF E82900          =1    818              CALL    ERROR                   ; YES--CALL ERROR HANDLER (RETURNS WITH
                     =1    819                                              ;    CARRY FLAG SET OR CLEAR)
01D2 5A              =1    820      DONE:   POP     DX                      ; RESTORE REGISTERS
01D3 58              =1    821              POP     AX
01D4 C3              =1    822              RET                             ; RETURN
                     =1    823      ;
                     =1    824      GO215   ENDP
                     =1    825      ;
                     =1    826 +1   $EJECT TITLE(WAIT FOR FUNCTION COMPLETE ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    WAIT FOR FUNCTION COMPLETE ROUTINE                10/27/80   PAGE   15

LOC  OBJ                  LINE     SOURCE

                     =1    827     ; ------------------------------------
                     =1    828     ; |                                  |
                     =1    829     ; |     WAIT FOR FUNCTION COMPLETE   |
                     =1    830     ; |                                  |
                     =1    831     ; ------------------------------------
                     =1    832     ;
                     =1    833     ;       NORMALLY, THIS WAIT ROUTINE WOULD TRAP TO THE SYSTEM DISPATCHER/
                     =1    834     ;       SCHEDULER TO ALLOW ANOTHER TASK TO EXECUTE WHILE THE iSBC 215 COMPLETED
                     =1    835     ;       ITS FUNCTION.  HOWEVER, FOR THIS EXAMPLE, THE ROUTINE SIMPLY WAITS FOR
                     =1    836     ;       THE INTERRUPT SERVICE ROUTINE TO LOAD THE OPERATION COMPLETE STATUS
                     =1    837     ;       FROM THE CIB OPERATION STATUS INTO THE DATA SEGMENT.  IF AN ERROR
                     =1    838     ;       OCCURRED, THE STATUS IS AVAILABLE THERE FOR SUBSEQUENT PROCESSING BY
                     =1    839     ;       AN ERROR HANDLER.
                     =1    840     ;
                     =1    841     ; INPUT DATA:
                     =1    842     ;       OPERATION COMPLETE STATUS FROM THE CIB, COPIED INTO THE DATA SEGMENT
                     =1    843     ;               BY THE INTERRUPT ROUTINE
                     =1    844     ;
                     =1    845     ; OUTPUT DATA:
                     =1    846     ;       OPERATION COMPLETE BYTE         CLEARED
                     =1    847     ;       CARRY FLAG                      = 0 IF NO ERROR
                     =1    848     ;                                       = 1 IF ERROR OCCURRED
                     =1    849     ;       COPY OF CIB OPERATION STATUS    IN "LSTSTS" IF ERROR OCCURRED
                     =1    850     ;
                     =1    851     ;       ( OPERATION COMPLETE BYTE AND "LSTSTS" ARE IN SEGMENT "DATASEG" )
                     =1    852     ;-------------------------------------------------------------------------------
                     =1    853     ;
                     =1    854             ASSUME  DS:DATASEG
                     =1    855     ;
01D5                 =1    856     WAIT215 PROC    NEAR
                     =1    857     ;
01D5 50              =1    858             PUSH    AX                      ; SAVE REGISTERS
01D6 53              =1    859             PUSH    BX
01D7 1E              =1    860             PUSH    DS
01D8 BB----        R =1    861             MOV     BX,DATASEG              ; GET POINTER TO DATA SEGMENT
01DB 8EDB            =1    862             MOV     DS,BX
01DD BBFFFF          =1    863             MOV     BX,-1                   ; INITIALIZE INDEX REGISTER
01E0 FB              =1    864             STI                             ; MAKE SURE INTERRUPT CAN GET THROUGH
01E1 F4              =1    865             HLT                             ; *****  WAIT FOR INTERRUPT  *****
01E2 43              =1    866     WAITLP: INC     BX                      ; GET INDEX FOR NEXT UNIT
01E3 81E30300        =1    867             AND     BX,0003H                ; MASK UPPER BITS
01E7 F607FF          =1    868             TEST    BYTE PTR [BX],0FFH      ; OPERATION COMPLETE STATUS = 00H?
                     =1    869                                             ;    (SIGN FLAG = BIT 7 OF OP. STATUS,
                     =1    870                                             ;     TEST INSTR. CLEARS CARRY FLAG)
01EA 74F6            =1    871             JZ      WAITLP                  ; STATUS <> 00H (OPERATION COMPLETE)?
                     =1    872                                             ; NO--CHECK NEXT UNIT
01EC 7906            =1    873             JNS     WAITDN                  ; YES--ERROR OCCURRED DURING FUNCTION?
                     =1    874                                             ; NO--RETURN WITH CARRY FLAG CLEAR
01EE 8A07            =1    875             MOV     AL,[BX]                 ; YES--SAVE CIB OP. STATUS IN "LSTSTS"
01F0 A21800          =1    876             MOV     LSTSTS,AL
01F3 F9              =1    877             STC                             ; SET CARRY FLAG TO INDICATE ERROR
01F4 C60700          =1    878     WAITDN: MOV BYTE PTR [BX],00H           ; CLEAR OPERATION COMPLETE BYTE
01F7 1F              =1    879             POP     DS                      ; RESTORE REGISTERS
01F8 5B              =1    880             POP     BX
01F9 58              =1    881             POP     AX
01FA C3              =1    882             RET                             ; RETURN
                     =1    883     ;
                     =1    884     WAIT215 ENDP
                           885     ;
                           886 +1  $INCLUDE(:F1:ERROR.MMD)
                     =1    887 +1  $EJECT TITLE(ERROR HANDLER)
```

```
MCS-86 MACRO ASSEMBLER    ERROR HANDLER                                10/27/80   PAGE   16

LOC  OBJ                  LINE     SOURCE

                     =1   888      ; -------------------------
                     =1   889      ; |                       |
                     =1   890      ; |     ERROR HANDLER     |
                     =1   891      ; |                       |
                     =1   892      ; -------------------------
                     =1   893      ;
                     =1   894      ;       THIS ROUTINE IS SYSTEM DEPENDENT.  IN THIS EXAMPLE, THE ERROR INFOR-
                     =1   895      ;       MATION FROM THE CONTROLLER IS READ INTO SOFTWARE REGISTERS IN DATASEG,
                     =1   896      ;       WHERE IT CAN BE EXAMINED.  MORE SOPHISTICATED SYSTEMS MIGHT LOG THE
                     =1   897      ;       ERRORS TO DETERMINE WHEN A TRACK IS GOING BAD, FOR EXAMPLE.
                     =1   898      ;
                     =1   899      ; - THE TRANSFER STATUS FUNCTION WILL NOT RETURN AN ERROR.
                     =1   900      ; - THE UNIT NUMBER IN THE IOPB IS NOT CHANGED, SO THAT THE OPERATION COMPLETE
                     =1   901      ;       STATUS FOR THE TRANSFER STATUS FUNCTION WILL BE POSTED AGAINST THE SAME
                     =1   902      ;       UNIT AS CAUSED THE ERROR.
                     =1   903      ;
                     =1   904      ; INPUT DATA:
                     =1   905      ;       CIB OPERATION STATUS    IN "LSTSTS" IN DATA SEGMENT
                     =1   906      ;
                     =1   907      ; OUTPUT DATA:
                     =1   908      ;       ERROR STATUS FROM CONTROLLER    IN DATA SEGMENT
                     =1   909      ;       CIB OPERATION STATUS            IN "LSTSTS" IN DATA SEGMENT
                     =1   910      ;       CARRY FLAG                      = 0 IF SOFT (RECOVERABLE) ERROR
                     =1   911      ;                                       = 1 IF HARD (UNRECOVERABLE) ERROR
                     =1   912      ; ---------------------------------------------------------------------------
                     =1   913      ;
                     =1   914              ASSUME  DS:IOPBSEG
                     =1   915      ;
01FB                 =1   916      ERROR   PROC    NEAR
                     =1   917      ;
01FB 50              =1   918              PUSH    AX                      ; SAVE REGISTERS
01FC 1E              =1   919              PUSH    DS
01FD B8----        R =1   920              MOV     AX,IOPBSEG              ; GET POINTER TO IOPB
0200 8ED8            =1   921              MOV     DS,AX
0202 A11200          =1   922              MOV     AX,BUFOFF               ; SAVE IOPB DATA BUFFER POINTER
0205 50              =1   923              PUSH    AX
0206 A11400          =1   924              MOV     AX,BUFSEG
0209 50              =1   925              PUSH    AX
020A C70612000C00    =1   926              MOV     BUFOFF,OFFSET ERRSTS    ; GET POINTER TO DATA SEGMENT ERROR
0210 C7061400----  R =1   927              MOV     BUFSEG,DATASEG          ;    STATUS REGISTERS
0216 C6060B0001      =1   928              MOV     FUNC,01H                ; SET FUNCTION = TRANSFER STATUS
021B C7060C000000    =1   929              MOV     MODIFY,0000H            ; CLEAR MODIFIER (ENABLE INTERRUPT ON
                     =1   930                                              ;    COMPLETION AND RETRIES)
0221 E89EFF          =1   931              CALL    GO215                   ; START FUNCTION AND WAIT FOR COMPLETE
0224 58              =1   932              POP     AX                      ; RESTORE IOPB DATA BUFFER POINTER
0225 A31400          =1   933              MOV     BUFSEG,AX
0228 58              =1   934              POP     AX
0229 A31200          =1   935              MOV     BUFOFF,AX
022C B8----        R =1   936              MOV     AX,DATASEG              ; GET POINTER TO DATA SEGMENT
022F 8ED8            =1   937              MOV     DS,AX
0231 F8              =1   938              CLC                             ; CLEAR CARRY FLAG
0232 A01800          =1   939              MOV     AL,DS:LSTSTS            ; GET OLD (ERROR) CIB OPERATION STATUS
0235 2440            =1   940              AND     AL,40H                  ; CHECK HARD ERROR BIT
0237 7401            =1   941              JZ      SFTERR                  ; HARD ERROR BIT SET?
                     =1   942                                              ; NO--LEAVE CARRY FLAG CLEAR
0239 F9              =1   943              STC                             ; YES--SET CARRY FLAG
023A 1F              =1   944      SFTERR: POP     DS                      ; RESTORE REGISTERS
023B 58              =1   945              POP     AX
023C C3              =1   946              RET
                     =1   947      ;
                     =1   948      ERROR   ENDP
                          949      ;
                          950 +1   $INCLUDE(:F1:INTRPT.MMD)
                     =1   951 +1   $EJECT TITLE(INTERRUPT SERVICE ROUTINE)
```

```
MCS-86 MACRO ASSEMBLER    INTERRUPT SERVICE ROUTINE                    10/27/80  PAGE  17

LOC  OBJ                  LINE     SOURCE

                     =1    952     ; ------------------------------------
                     =1    953     ; |                                  |
                     =1    954     ; |     INTERRUPT SERVICE ROUTINE    |
                     =1    955     ; |                                  |
                     =1    956     ; ------------------------------------
                     =1    957     ;
                     =1    958     ;       THIS ROUTINE SERVICES THE INTERRUPT GENERATED BY THE iSBC 215 UPON
                     =1    959     ;       OPERATION COMPLETE, SEEK COMPLETE, OR DISK PACK CHANGE.  IT COPIES THE
                     =1    960     ;       CIB OPERATION STATUS INTO ONE OF FOUR BYTES ASSOCIATED WITH EACH OF
                     =1    961     ;       THESE EVENTS.  IT IS ASSUMED THAT SYSTEM PROGRAMS MAKE USE OF THE
                     =1    962     ;       INFORMATION TO RESUME TASKS, HANDLE ERROR LOGGING/RECOVERY, AND KEEP
                     =1    963     ;       TRACK OF DIRECTORY INFORMATION.  FOR THIS PROGRAMMING EXAMPLE, ONLY
                     =1    964     ;       THE OPERATION COMPLETE BYTES ARE USED.
                     =1    965     ;
                     =1    966     ; - THE SYSTEM INTERRUPTS ARE CONFIGURED BY EXTERNAL PROGRAMS.
                     =1    967     ; ------------------------------------------------------------------------------
                     =1    968     ;
                     =1    969             PUBLIC  INT215
                     =1    970     ;
023D                 =1    971     INT215  PROC    FAR
                     =1    972     ;
023D FB              =1    973             STI                             ;;; ENABLE HIGHER PRIORITY INTERRUPTS
023E 50              =1    974             PUSH    AX                      ;;; SAVE REGISTERS
023F 53              =1    975             PUSH    BX
0240 52              =1    976             PUSH    DX
0241 1E              =1    977             PUSH    DS
                     =1    978             ASSUME  DS:CIBSEG
0242 B8----        R =1    979             MOV     AX,CIBSEG               ; GET POINTER TO CIB
0245 8ED8            =1    980             MOV     DS,AX
0247 A00100          =1    981             MOV     AL,OPSTS                ; GET CIB OPERATION STATUS
024A 8AD0            =1    982             MOV     DL,AL                   ; SAVE IT
024C C606030000      =1    983             MOV     STSSEM,00H              ; CLEAR CIB STATUS SEMAPHORE
0251 8AD8            =1    984             MOV     BL,AL                   ; MOVE IT TO INDEX REGISTER
0253 81E33000        =1    985             AND     BX,0030H                ; MASK ALL BITS EXCEPT UNIT NUMBER
0257 D1EB            =1    986             SHR     BX,1                    ; SHIFT UNIT NUMBER TO BITS 0 AND 1
0259 D1EB            =1    987             SHR     BX,1
025B D1EB            =1    988             SHR     BX,1
025D D1EB            =1    989             SHR     BX,1
025F 250600          =1    990             AND     AX,0006H                ; MASK ALL BITS EXCEPT SEEK COMPLETE
                     =1    991                                             ;    AND PACK CHANGE
0262 D1E0            =1    992             SHL     AX,1                    ; SHIFT LEFT TO GET OFFSET INTO PROPER
                     =1    993                                             ;    BYTE IN DATA SEGMENT
0264 03D8            =1    994             ADD     BX,AX                   ; COMBINE WITH UNIT IN INDEX REGISTER
                     =1    995             ASSUME  DS:DATASEG
0266 B8----        R =1    996             MOV     AX,DATASEG              ; GET POINTER TO DATA SEGMENT
0269 8ED8            =1    997             MOV     DS,AX
026B 8817            =1    998             MOV     [BX],DL                 ; MOVE OPERATION STATUS TO DATA SEGMENT
026D BA5063          =1    999             MOV     DX,WUA*16               ; GET POINTER TO I/O WAKE-UP ADDRESS
0270 B002            =1   1000             MOV     AL,02H                  ; GET CLEAR INTERRUPT COMMAND BYTE
0272 EE              =1   1001             OUT     DX,AL                   ; OUTPUT TO iSBC 215
                     =1   1002     ;
0273 1F              =1   1003             POP     DS                      ; RESTORE REGISTERS
0274 5A              =1   1004             POP     DX
0275 5B              =1   1005             POP     BX
0276 FA              =1   1006             CLI                             ; DISABLE INTERRUPTS FOR RESTORE
                     =1   1007                                             ; (RESTORATION OF INTERRUPT LOGIC STATE
                     =1   1008                                             ; IS SYSTEM DEPENDENT. THIS EXAMPLE USES
                     =1   1009                                             ; THE iSBC 86/12A CPU.)
0277 B020            =1   1010             MOV     AL,20H                  ;;; GET END-OF-INTERRUPT COMMAND
0279 E6C0            =1   1011             OUT     0C0H,AL                 ;;; OUTPUT EOI COMMAND TO 8259
027B 58              =1   1012             POP     AX                      ;;;
027C CF              =1   1013             IRET                            ;;; INTERRUPT RETURN ENABLES INTERRUPTS
                     =1   1014     ;
                     =1   1015     INT215  ENDP
                          1016     ;
----                      1017     SBC215DRIVER    ENDS                    ; END OF iSBC 215 DRIVER CODE
                          1018     ;
                          1019 +1  $TITLE(SYMBOL TABLE AND CROSS REFERENCE)
                          1020     ;
                          1021             END                             ; END OF PROGRAMMING EXAMPLE
```

MCS-86 MACRO ASSEMBLER SYMBOL TABLE AND CROSS REFERENCE 10/27/80 PAGE 18

XREF SYMBOL TABLE LISTING

```
NAME            TYPE      VALUE  ATTRIBUTES, XREFS

??SEG . . . . SEGMENT          SIZE=0000H PARA PUBLIC
ACTCNT. . . . V DWORD   0004H  IOPBSEG  129#
ACTCYL. . . . V WORD    0013H  DATASEG  312#
ACTHD . . . . V BYTE    0015H  DATASEG  313#
ACTSEC. . . . V BYTE    0016H  DATASEG  314#
BSYFLG1 . . . V BYTE    0001H  CCBSEG  79# 512
BSYFLG2 . . . V BYTE    0009H  CCBSEG  85#
BUFOFF. . . . V WORD    0012H  IOPBSEG  137# 565 567 646 647 706 770 922 926 935
BUFSEG. . . . V WORD    0014H  IOPBSEG  138# 563 648 708 772 924 927 933
CCB . . . . . L FAR     0000H  CCBSEG  64 77# 454 455
CCBPTR. . . . V DWORD   0002H  SCBSEG  64# 454 455 459
CCBSEG. . . . SEGMENT          SIZE=0010H PARA     75# 92 460 504 505
CCW1. . . . . V BYTE    0000H  CCBSEG  78# 461
CCW2. . . . . V BYTE    0008H  CCBSEG  84# 464
CH1PC . . . . L FAR     0004H  CIBSEG  80 110# 462 463 476
CH1PTR. . . . V DWORD   0002H  CCBSEG  80# 462 463
CH2PC . . . . L FAR     000EH  CCBSEG  86 89# 465 466 467
CH2PTR. . . . V DWORD   000AH  CCBSEG  86# 465 466
CIB . . . . . L FAR     0000H  CIBSEG  105#
CIBCMD. . . . V BYTE    0000H  CIBSEG  106# 474
CIBSEG. . . . SEGMENT          SIZE=0010H PARA     103# 116 471 472 978 979
CLRLP . . . . L NEAR    0078H  SBC215DRIVER  488# 491
CMDSEM. . . . V BYTE    0002H  CIBSEG  108# 475
CONFIG. . . . L FAR     0000H  USERSEG PUBLIC  377 380# 396
CYLNDR. . . . V WORD    000EH  IOPBSEG  134# 643 702 766
DATASEG . . . SEGMENT          SIZE=001AH PARA     268# 326 483 484 854 861 927 936 995 996
DESCYL. . . . V WORD    000FH  DATASEG  309#
DESHD . . . . V BYTE    0011H  DATASEG  310#
DESSEC. . . . V BYTE    0012H  DATASEG  311#
DEVCOD. . . . V WORD    0008H  IOPBSEG  130# 559 577 580 639 698 762
DONE. . . . . L NEAR    01D2H  SBC215DRIVER  816 820#
ENDDAT. . . . L FAR     001AH  DATASEG  324# 486
ENDSTK. . . . L FAR     0040H  STACK  367# 385
ERROR . . . . L NEAR    01FBH  SBC215DRIVER  818 916# 948
ERRSTS. . . . V WORD    000CH  DATASEG PUBLIC  278 307# 926
FMT215. . . . L FAR     00EFH  SBC215DRIVER PUBLIC  629 632# 658
FMTDN . . . . L NEAR    0129H  SBC215DRIVER  654#
FUNC. . . . . V BYTE    000BH  IOPBSEG  132# 560 649 715 779 928
G0215 . . . . L NEAR    01C2H  SBC215DRIVER  569 652 716 780 808# 824 931
HEAD. . . . . V BYTE    0010H  IOPBSEG  135# 645 704 768
INIT215 . . . L FAR     00A5H  SBC215DRIVER PUBLIC  550 553# 586
INITBLSEG . . SEGMENT          SIZE=003FH PARA     170# 258 563
INITDN. . . . L NEAR    00ECH  SBC215DRIVER  571 579 582#
INITLP. . . . L NEAR    00CBH  SBC215DRIVER  567# 575 581
INT215. . . . L FAR     023DH  SBC215DRIVER PUBLIC  388 389 969 971# 1015
INTRCS. . . . V WORD    0096H  SEG0000  355# 389
INTRIP. . . . V WORD    0094H  SEG0000  354# 388
INTRPT. . . . NUMBER    0005H    347# 352
IOPB. . . . . L FAR     0000H  IOPBSEG  112 127# 477 478
IOPBOFF . . . V WORD    0008H  CIBSEG  112# 477
IOPBSEG . . . SEGMENT          SIZE=001EH PARA     113 125# 142 551 557 630 636 689 695 753 759 914 920
IOPBSG. . . . V WORD    000AH  CIBSEG  113# 478
LSTSTS. . . . V BYTE    0018H  DATASEG  320# 876 939
MODIFY. . . . V WORD    000CH  IOPBSEG  133# 561 650 713 777 929
NMRTRY. . . . V BYTE    0017H  DATASEG  315#
OPCMP . . . . V BYTE    0000H  DATASEG PUBLIC  278 282#
OPCMP0. . . . V BYTE    0000H  DATASEG  283#
OPCMP1. . . . V BYTE    0001H  DATASEG  284#
OPCMP2. . . . V BYTE    0002H  DATASEG  285#
OPCMP3. . . . V BYTE    0003H  DATASEG  286#
OPSTS . . . . V BYTE    0001H  CIBSEG  107# 981
PKCHG . . . . V BYTE    0008H  DATASEG PUBLIC  278 298#
PKCHG0. . . . V BYTE    0008H  DATASEG  299#
PKCHG1. . . . V BYTE    0009H  DATASEG  300#
PKCHG2. . . . V BYTE    000AH  DATASEG  301#
PKCHG3. . . . V BYTE    000BH  DATASEG  302#
RD215 . . . . L FAR     012EH  SBC215DRIVER PUBLIC  688 691# 722
REQCNT. . . . V DWORD   0016H  IOPBSEG  139# 710 712 774 776
RES215. . . . L FAR     0000H  SBC215DRIVER PUBLIC  436 438# 526
RESDN . . . . L NEAR    009FH  SBC215DRIVER  514 519#
RESLP . . . . L NEAR    0095H  SBC215DRIVER  512# 516
SBC215DRIVER. SEGMENT          SIZE=027DH PARA     400# 402 1017
SCB . . . . . L FAR     0000H  SCBSEG  61#
SCBSEG. . . . SEGMENT          SIZE=0006H PARA ABS    59# 66 450 451
SECTOR. . . . V BYTE    0011H  IOPBSEG  136#
SEG0000 . . . SEGMENT          SIZE=0098H PARA ABS    349# 357 378
```

```
MCS-86 MACRO ASSEMBLER    SYMBOL TABLE AND CROSS REFERENCE                10/27/80   PAGE   19


NAME           TYPE     VALUE  ATTRIBUTES, XREFS

SFERST. . . . V BYTE   000EH  DATASEG  308#
SFTERR. . . . L NEAR   023AH  SBC215DRIVER  941 944#
SKCMP . . . . V BYTE   0004H  DATASEG PUBLIC  278 290#
SKCMP0. . . . V BYTE   0004H  DATASEG  291#
SKCMP1. . . . V BYTE   0005H  DATASEG  292#
SKCMP2. . . . V BYTE   0006H  DATASEG  293#
SKCMP3. . . . V BYTE   0007H  DATASEG  294#
SOC . . . . . V BYTE   0000H  SCBSEG  62# 453
STACK . . . . SEGMENT         SIZE=0040H PARA
STSSEM. . . . V BYTE   0003H  CIBSEG  109# 983
UNIT. . . . . V BYTE   000AH  IOPBSEG  131# 568 641 700 764
USERSEG . . . SEGMENT         SIZE=0022H PARA    375# 398
WAIT215 . . . L NEAR   01D5H  SBC215DRIVER  815 856# 884
WAITDN. . . . L NEAR   01F4H  SBC215DRIVER  873 878#
WAITLP. . . . L NEAR   01E2H  SBC215DRIVER  866# 871
WRT215. . . . L FAR    0178H  SBC215DRIVER PUBLIC  752 755# 786
WUA . . . . . NUMBER   0635H    57# 59 497 812 999

ASSEMBLY COMPLETE, NO ERRORS FOUND
```