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 prior written consent of Intel Corporation.

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local sales office to obtain the latest specifications before placing your order.

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

BITBUS
COMMputer
CREDIT
Data Pipeline
Genius
INTEL
ICE
I2ICE
ICF
ICF
IDRP
IDIS
iLBX
IMDDX
IMMX
Inside
Intell
Intel
Intelligent Identifier
Intelligent Programming
Intelli
Intellink
iSP
iPDS
iRMX
iSB
iSBX
iSDM
iXM
Library Manager
MCS
Megachasis
MICROMAINFRAME
MULTIBUS
MULTICHANNEL
MULTIMODULE
OpenNET
Plug-A-Buble
PROMPT
Promware
QueX
QUEST
Ripplemode
RMX/80
RUPI
Seamless
SFD
SYSTEM 2000
UPI

MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation.

*MULTIBUS is a patented Intel bus.

Copyright 1984, Intel Corporation
<table>
<thead>
<tr>
<th>REV.</th>
<th>REVISION HISTORY</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>-001</td>
<td>Original Issue</td>
<td>7/82</td>
</tr>
<tr>
<td>-002</td>
<td>Manual updated to include new information for iSBX 217B/C board and iSBX 218A board support. Interconnect tables in chapter 2 are corrected and expanded. Additional programming information is provided in Chapter 3. New schematics diagram in Chapter 5.</td>
<td>12/84</td>
</tr>
</tbody>
</table>
This manual includes several kinds of information. For introductory materials on the iSBC 215 Generic Winchester Disk Controller Board, refer to Chapter 1. To configure the board and install it in your system, refer to Chapter 2. For programming information, refer to Chapter 3. For the functional description, refer to Chapter 4. For service assistance information or the schematic diagram, refer to Chapter 5. This manual is not intended as a tutorial document. The manual assumes that you are familiar with the standards of Intel single-board computers and the associated peripheral control boards and are familiar with programming in general and Intel device programming in particular.

In addition to this manual, you will need the manuals for the system of which the iSBC 215G board is a part. The following listed manuals provide information pertaining to the iSBC 215G board, its use, and its component parts. Intel documents are available from the Intel Literature Department (refer to page ii for the address).

Microsystem Components Handbook, Volumes I and II, Order No. 230843

Intel MULTIBUS® Handbook, Order No. 2108833
(This handbook includes information on the Intel iSBX™ Bus.)

8089 Assembler User's Guide, Order No. 9800938

8086 Family User's Manual, Order No. 9800722

Also, if you intend to use the iSBC 215G board with flexible-diskette and/or cartridge-tape drives, you will need one, two, or all of the following manuals.

iSBX™ 218A Flexible Diskette Controller Hardware Reference Manual, Order No. 145911

iSBX™ 217B Magnetic Cartridge Tape Interface MULTIMODULE™ Board Hardware Reference Manual, Order No. 145497

iSBX™ 217C Magnetic Cartridge Tape Interface MULTIMODULE™ Board Hardware Reference Manual, Order No. 146704

Tape interface support is limited to Archive Corporation and other QIC-02 tape drives. Programming information in this manual is current with and applicable to the following iSBC 215G, iSBX 218A, and iSBX 217B/C boards:
Differences between this version of the board firmware and earlier versions are noted as appropriate. Significant differences between operation of the iSBC 215G board and the iSBC 215A/B board are also noted as appropriate.
# CONTENTS

## CHAPTER 1
**GENERAL INFORMATION**

1.1 Introduction .................................................. 1-1
1.2 Description .................................................. 1-1
1.3 Specifications .............................................. 1-4

## CHAPTER 2
**PREPARATION FOR USE**

2.1 Introduction .................................................. 2-1
2.2 Unpacking and Inspection .................................... 2-1
2.3 Installation Considerations .................................. 2-1
2.3.1 Power Requirements ...................................... 2-1
2.3.2 Cooling Requirements .................................... 2-2
2.3.3 Physical Characteristics .................................. 2-2
2.4 Jumper Configurations and Jumper Installation .......... 2-2
2.4.1 Wake-Up Address Selection ................................ 2-3
2.4.2 Wake-Up I/O Port Address Selection ..................... 2-4
2.4.3 System Data Bus Width Selection ......................... 2-4
2.4.4 Interrupt Priority Number ................................ 2-4
2.4.5 Bus Priority Arbitration .................................. 2-5
2.4.5.1 Common Bus Request (CBRQ*)/Any Request (ANYRQST) Signal Selection .................................................. 2-5
2.4.6 MULTIBUS® Lock (LOCK*) Signal ........................ 2-6
2.4.7 iSBX™ Bus Selection ...................................... 2-6
2.4.8 Hard-Disk Drive Interface ................................ 2-7
2.4.9 -5 V Selection ............................................. 2-8
2.4.10 I/O Communication Blocks Page Selection .............. 2-9
2.4.11 RAM Jumper ............................................... 2-9
2.5 Interface Connections ....................................... 2-9
2.5.1 MULTIBUS® Interface ...................................... 2-9
2.5.2 iSBX™ MULTIMODULE™ Interface ......................... 2-14
2.5.3 Cabling Requirements ..................................... 2-16
2.6 Board Installation ........................................... 2-35
2.6.1 Drive Installation ......................................... 2-35
2.6.1.1 14/8-Inch Drives With Integral Data Separator .... 2-36
2.6.1.2 5 1/4-Inch Drives ..................................... 2-37
2.6.1.3 Other Drives Without Data Separator ................. 2-37
2.6.2 iSBX™ 218A Board Installation .......................... 2-38
2.6.3 iSBX™ 217B/C Board Installation ......................... 2-39

## CHAPTER 3
**PROGRAMMING**

3.1 Introduction .................................................. 3-1
3.2 Programming Options ......................................... 3-1
3.3 Mass Storage Principles ..................................... 3-2
3.3.1 Hard-Disk Organization .................................. 3-2
3.3.2 Flexible-Disk Organization ............................... 3-3
CHAPTER 3
PROGRAMMING (continued)

3.3.3 Cartridge-Tape Organization ............................................. 3-3
3.4 Host/Board Communications .......................................................... 3-3
3.4.1 Wake-Up I/O Port .................................................................. 3-4
3.4.2 I/O Communications Blocks .......................................................... 3-5
3.4.2.1 Wake-Up Block .................................................................. 3-7
3.4.2.2 Channel Control Block ......................................................... 3-8
3.4.2.3 Controller Invocation Block ................................................... 3-9
3.4.2.4 I/O Parameter Block ............................................................. 3-11
3.4.3 Cold-Start Board Initialization ..................................................... 3-13
3.5 Function Commands ..................................................................... 3-14
3.5.1 Initialize (00H) ..................................................................... 3-15
3.5.1.1 Number of Cylinders .......................................................... 3-17
3.5.1.2 Number of Heads ............................................................... 3-17
3.5.1.3 Sectors Per Track ............................................................... 3-17
3.5.1.4 Bytes Per Sector ................................................................. 3-17
3.5.1.5 Number of Alternate Cylinders .............................................. 3-17
3.5.1.6 Drive Variables .................................................................. 3-18
3.5.1.7 Tape Parameters ................................................................. 3-19
3.5.2 Transfer Status Buffer (01H) ...................................................... 3-19
3.5.3 Format (02H) ....................................................................... 3-19
3.5.4 Read Sector ID (03H) ............................................................... 3-21
3.5.5 Read Data (04H) .................................................................... 3-22
3.5.5.1 Disk Read Details ............................................................... 3-23
3.5.5.2 Tape Read Details ............................................................... 3-23
3.5.6 Read to Buffer and Verify (05H) ................................................ 3-24
3.5.7 Write Data (06H) .................................................................... 3-25
3.5.7.1 Disk Write Details ............................................................... 3-25
3.5.7.2 Tape Write Details ............................................................... 3-26
3.5.8 Write Buffer Data (07H) ............................................................ 3-26
3.5.9 Initiate Track Seek (08H) ............................................................. 3-27
3.5.10 Spin Down (0BH) .................................................................... 3-28
3.5.11 iSBX™ Execute (0CH) .............................................................. 3-28
3.5.12 iSBX™ Transfer (0DH) ............................................................. 3-28
3.5.13 Buffer I/O (0EH) ................................................................... 3-29
3.5.14 Diagnostic (0FH) .................................................................. 3-31
3.5.15 Tape Initialization (10H) ........................................................... 3-31
3.5.16 Rewind (11H) ...................................................................... 3-32
3.5.17 Space Forward One File Mark (12H) ........................................ 3-32
3.5.18 Write File Mark (14H) ............................................................. 3-33
3.5.19 Erase Tape (17H) ................................................................... 3-33
3.5.20 Load Tape (18H) ................................................................... 3-34
3.5.21 Tape Reset (19H) ................................................................... 3-34
3.5.22 Retension Tape (1DH) .............................................................. 3-34
3.5.23 Read Tape Status (1EH) ........................................................... 3-35
3.5.24 Read/Write Terminate (1FH) .................................................... 3-35
3.6 Function Modifiers ........................................................................ 3-36
3.7 Extended Status ............................................................................ 3-37
CHAPTER 3
PROGRAMMING (continued)

3.8 Status Operations............................................. 3-38
3.8.1 Status Buffer Format........................................ 3-39
3.8.2 Detailed Error Status....................................... 3-40
3.8.2.1 Status Byte Ø............................................. 3-41
3.8.2.2 Status Byte 1............................................. 3-42
3.8.2.3 Status Byte 2............................................. 3-44
3.9 Interrupts.................................................... 3-46
3.10 iSBX™ Bus Expansion.......................................... 3-46
3.10.1 Firmware Drivers.......................................... 3-47
3.10.2 User-Provided Drivers...................................... 3-47
3.10.2.1 Random-Access Memory Map............................. 3-49
3.10.2.2 Execution From On-Board RAM........................... 3-50
3.10.2.3 Execution From System Memory.......................... 3-50
3.10.2.4 Program Execution........................................ 3-51
3.10.2.5 Example I/O Program..................................... 3-51

CHAPTER 4
FUNCTIONAL DESCRIPTION

4.1 Introduction.................................................. 4-1
4.2 Schematic Interpretation..................................... 4-1
4.3 General Description........................................... 4-2
4.3.1 Communications With Host................................. 4-3
4.3.2 Communications With Hard Disk............................ 4-4
4.3.3 Communications Via iSBX™ Bus.............................. 4-5
4.4 Detailed Description.......................................... 4-5
4.4.1 Board/Host Communications.................................. 4-5
4.4.1.1 MULTIBUS® Interface Signals........................... 4-5
4.4.1.2 I/O Processor............................................. 4-5
4.4.1.3 Bus Controller............................................ 4-7
4.4.2 Board/Drive Communications............................... 4-13
4.4.2.1 Board/Hard-Disk Drive Interface........................ 4-14
4.4.2.2 Board/Flexible-Disk and Cartridge-Tape Drive Interface 4-15
4.4.2.3 Interface Timing.......................................... 4-15
4.4.2.4 Serializer/De-serializer.................................. 4-22
4.4.2.5 Sync Byte Comparator..................................... 4-23
4.4.2.6 Sector Identity Comparator................................ 4-23
4.4.2.7 Error Checking Code Generator........................... 4-23
4.4.2.8 Status Register............................................ 4-24
4.4.2.9 Line Drivers and Receivers................................ 4-25
4.4.3 Local Memory Organization................................. 4-25
4.4.3.1 Read-Only Memory........................................ 4-26
4.4.3.2 Random-Access Memory................................... 4-26
4.4.3.3 I/O Port Decoding.......................................... 4-26
CONTENTS (continued)

CHAPTER 5
SERVICE INFORMATION
5.1 Introduction................................................. 5-1
5.2 Service Diagrams............................................. 5-1
5-3 Service and Repair Assistance................................ 5-1

APPENDIX A
ANSI INTERFACE PROGRAMMING.......................................................... A-1

APPENDIX B
EXAMPLE I/O PROGRAM
B-1. Introduction................................................. B-1
B-2. Single User Sequence......................................... B-1
B-3. Single User Sequence with Overlapping Seeks............... B-1
B-4. Multi-User Sequence.......................................... B-1
B-5. Example Host Processor Disk Control Program............... B-2

TABLES

<table>
<thead>
<tr>
<th>Table Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Board Specifications</td>
<td>1-5</td>
</tr>
<tr>
<td>1-2</td>
<td>Allowable Sectors Per Track for Non-ANSI Hard-Disk Drives</td>
<td>1-6</td>
</tr>
<tr>
<td>1-3</td>
<td>Allowable Sectors Per Track for ANSI Hard-Disk Drives</td>
<td>1-7</td>
</tr>
<tr>
<td>1-4</td>
<td>Formatted Capacity Per Hard-Disk Drive (in Mbytes)</td>
<td>1-7</td>
</tr>
<tr>
<td>1-5</td>
<td>Wake-Up Address Jumpers</td>
<td>2-3</td>
</tr>
<tr>
<td>1-6</td>
<td>Interrupt Priority Number Selection</td>
<td>2-4</td>
</tr>
<tr>
<td>1-7</td>
<td>Bus Arbitration Options</td>
<td>2-5</td>
</tr>
<tr>
<td>1-8</td>
<td>iSBX™ Bus Control Jumpers</td>
<td>2-6</td>
</tr>
<tr>
<td>1-9</td>
<td>Jumper Configuration for Various Hard-Disk Drives</td>
<td>2-8</td>
</tr>
<tr>
<td>1-10</td>
<td>-5 V Selection Jumper Configuration</td>
<td>2-9</td>
</tr>
<tr>
<td>1-11</td>
<td>Connector P1 Pin Assignments</td>
<td>2-10</td>
</tr>
<tr>
<td>1-12</td>
<td>Connector P2 Pin Assignments</td>
<td>2-11</td>
</tr>
<tr>
<td>1-13</td>
<td>Connector P1 Input/Output Signals</td>
<td>2-12</td>
</tr>
<tr>
<td>1-14</td>
<td>Connector P2 Input/Output Signals</td>
<td>2-14</td>
</tr>
<tr>
<td>1-15</td>
<td>iSBX™ Bus Mnemonics/iSBC© 215G Board Mnemonics</td>
<td>2-14</td>
</tr>
<tr>
<td>1-16</td>
<td>Drive Interface Pin-Out Data</td>
<td>2-15</td>
</tr>
<tr>
<td>1-17</td>
<td>Control Cable Functions (J1 Complete/J5 Partial)</td>
<td>2-17</td>
</tr>
<tr>
<td>1-18</td>
<td>Read/Write Cable Functions (J2 Complete/J5 Partial)</td>
<td>2-22</td>
</tr>
<tr>
<td>1-19</td>
<td>I/O Control Commands</td>
<td>3-5</td>
</tr>
<tr>
<td>1-20</td>
<td>Wake-Up Block Byte Contents</td>
<td>3-8</td>
</tr>
<tr>
<td>1-21</td>
<td>Channel Control Block Byte Contents</td>
<td>3-9</td>
</tr>
<tr>
<td>1-22</td>
<td>Controller Invocation Block Byte Contents</td>
<td>3-10</td>
</tr>
<tr>
<td>1-23</td>
<td>I/O Parameter Block Byte Contents</td>
<td>3-12</td>
</tr>
<tr>
<td>1-24</td>
<td>Function Command Summary</td>
<td>3-15</td>
</tr>
<tr>
<td>1-25</td>
<td>iSBX™ Bus I/O Port Addresses (Hexadecimal)</td>
<td>3-29</td>
</tr>
<tr>
<td>1-26</td>
<td>Error Status Buffer</td>
<td>3-39</td>
</tr>
</tbody>
</table>
TABLES (continued)

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-9.</td>
<td>IOP Handshake and Control Lines on iSBBX™ Bus</td>
<td>3-48</td>
</tr>
<tr>
<td>3-10.</td>
<td>Control and Status Lines on iSBBX™ Interface</td>
<td>3-48</td>
</tr>
<tr>
<td>3-11.</td>
<td>Jumper Connections Allowing Option Lines to be Driven</td>
<td>3-49</td>
</tr>
<tr>
<td>3-12.</td>
<td>On-Board Program and Parameter Storage RAM Space</td>
<td>3-49</td>
</tr>
<tr>
<td>4-1.</td>
<td>IOP Status Line Decodes</td>
<td>4-8</td>
</tr>
<tr>
<td>4-2.</td>
<td>Data Transfer Signal Relationships</td>
<td>4-10</td>
</tr>
<tr>
<td>4-3.</td>
<td>Host Wake-Up Commands</td>
<td>4-11</td>
</tr>
<tr>
<td>4-4.</td>
<td>Status Register Bits</td>
<td>4-25</td>
</tr>
<tr>
<td>4-5.</td>
<td>Local I/O Ports</td>
<td>4-27</td>
</tr>
</tbody>
</table>

FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1.</td>
<td>Example of Multiple Drive System Using Non-ANSI Interface</td>
<td>1-2</td>
</tr>
<tr>
<td>1-2.</td>
<td>Example of Multiple Drive System Using ANSI Interface</td>
<td>1-2</td>
</tr>
<tr>
<td>1-3.</td>
<td>Example of Multiple Drive System Using iSBBX™ 218A Flexible Diskette Controller</td>
<td>1-3</td>
</tr>
<tr>
<td>1-4.</td>
<td>Automatic Error Checking and Correction</td>
<td>1-4</td>
</tr>
<tr>
<td>2-1.</td>
<td>8-Inch Shugart/Quantum Drive Interconnection Listing</td>
<td>2-23</td>
</tr>
<tr>
<td>2-2.</td>
<td>Fujitsu 23ØØ/Memorex/14-Inch Shugart Drive Interconnection Listing</td>
<td>2-24</td>
</tr>
<tr>
<td>2-3.</td>
<td>Priam Drive Interconnection Listing</td>
<td>2-25</td>
</tr>
<tr>
<td>2-4.</td>
<td>5 1/4-Inch Drive Interconnection Listing</td>
<td>2-26</td>
</tr>
<tr>
<td>2-5.</td>
<td>CDC Drive Interconnection Listing</td>
<td>2-27</td>
</tr>
<tr>
<td>2-6.</td>
<td>ANSI Drive Interconnection Listing</td>
<td>2-28</td>
</tr>
<tr>
<td>2-7.</td>
<td>8-Inch Shugart/Quantum Drive Interconnecting Cable</td>
<td>2-29</td>
</tr>
<tr>
<td>2-8.</td>
<td>Fujitsu 23ØØ/Memorex/14-Inch Shugart Drive Interconnecting Cable</td>
<td>2-30</td>
</tr>
<tr>
<td>2-9.</td>
<td>Priam Drive Interconnecting Cable</td>
<td>2-31</td>
</tr>
<tr>
<td>2-10.</td>
<td>5 1/4-Inch Drive Interconnecting Cable</td>
<td>2-32</td>
</tr>
<tr>
<td>2-11.</td>
<td>CDC Drive Interconnecting Cable</td>
<td>2-33</td>
</tr>
<tr>
<td>2-12.</td>
<td>ANSI Drive Interconnecting Cable</td>
<td>2-34</td>
</tr>
<tr>
<td>2-13.</td>
<td>ANSI Drive Interface</td>
<td>2-35</td>
</tr>
<tr>
<td>2-14.</td>
<td>Pin Numbering Conventions</td>
<td>2-36</td>
</tr>
<tr>
<td>2-15.</td>
<td>Interface for Drive with Integral Data Separator</td>
<td>2-36</td>
</tr>
<tr>
<td>2-16.</td>
<td>Interface for Drives and iSBC® 215G Kit</td>
<td>2-37</td>
</tr>
<tr>
<td>2-17.</td>
<td>Interface for Drives without Data Separator</td>
<td>2-38</td>
</tr>
<tr>
<td>2-18.</td>
<td>iSBBX™ 218A Board Installation</td>
<td>2-40</td>
</tr>
<tr>
<td>2-19.</td>
<td>iSBBX™ 217B/C Board Installation</td>
<td>2-41</td>
</tr>
<tr>
<td>3-1.</td>
<td>Disk Drive Organization and Terminology</td>
<td>3-1</td>
</tr>
<tr>
<td>3-2.</td>
<td>Sector Fields</td>
<td>3-3</td>
</tr>
<tr>
<td>3-3.</td>
<td>Host CPU/Board Interaction</td>
<td>3-6</td>
</tr>
<tr>
<td>3-4.</td>
<td>Wake-Up Block</td>
<td>3-7</td>
</tr>
<tr>
<td>3-5.</td>
<td>Channel Control Block</td>
<td>3-8</td>
</tr>
<tr>
<td>3-6.</td>
<td>Controller Invocation Block</td>
<td>3-10</td>
</tr>
<tr>
<td>3-7.</td>
<td>I/O Parameter Block</td>
<td>3-11</td>
</tr>
<tr>
<td>3-8.</td>
<td>Initialize Function Data Buffer Format</td>
<td>3-16</td>
</tr>
<tr>
<td>3-9.</td>
<td>Drive Variables Byte Format</td>
<td>3-18</td>
</tr>
</tbody>
</table>
### FIGURES (continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1Ø.</td>
<td>Sector Data Format</td>
<td>3-2Ø</td>
</tr>
<tr>
<td>3-11.</td>
<td>Format Function Data Buffer Format</td>
<td>3-2Ø</td>
</tr>
<tr>
<td>3-12.</td>
<td>Read-Sector-ID Function Data Buffer and Flag Byte</td>
<td>3-22</td>
</tr>
<tr>
<td>3-13.</td>
<td>Transfer Parameter Byte Format (iSBX™ MULTIMODULE™ Board)</td>
<td>3-3Ø</td>
</tr>
<tr>
<td>3-14.</td>
<td>Modifier Word Format</td>
<td>3-36</td>
</tr>
<tr>
<td>3-15.</td>
<td>Operation Status Byte Format</td>
<td>3-38</td>
</tr>
<tr>
<td>3-16.</td>
<td>Status Buffer Byte 8 Format for Disk</td>
<td>3-4Ø</td>
</tr>
<tr>
<td>3-17.</td>
<td>Status Buffer Byte Ø Format</td>
<td>3-41</td>
</tr>
<tr>
<td>3-18.</td>
<td>Status Buffer Byte 1 Format</td>
<td>3-43</td>
</tr>
<tr>
<td>3-19.</td>
<td>Status Buffer Byte 2 Format</td>
<td>3-45</td>
</tr>
<tr>
<td>3-2Ø.</td>
<td>Execution of iSBX™ Bus I/O Program from RAM</td>
<td>3-51</td>
</tr>
<tr>
<td>4-1.</td>
<td>Logic Conventions</td>
<td>4-2</td>
</tr>
<tr>
<td>4-2.</td>
<td>Board Simplified Block Diagram</td>
<td>4-3</td>
</tr>
<tr>
<td>4-3.</td>
<td>Bus Arbiter and Controller Logic</td>
<td>4-7</td>
</tr>
<tr>
<td>4-4.</td>
<td>Data Transfer from MULTIBUS† to iSBC† 215G Board</td>
<td>4-9</td>
</tr>
<tr>
<td>4-5.</td>
<td>Wake-Up Address Logic</td>
<td>4-11</td>
</tr>
<tr>
<td>4-6.</td>
<td>Initialization Sequence Address Fetches</td>
<td>4-13</td>
</tr>
<tr>
<td>4-7.</td>
<td>Ready Signal Timing Diagram</td>
<td>4-16</td>
</tr>
<tr>
<td>4-8.</td>
<td>Disk Formatting Sequence Timing Diagram</td>
<td>4-18</td>
</tr>
<tr>
<td>4-9.</td>
<td>Write Data Transfer Timing Diagram</td>
<td>4-2Ø</td>
</tr>
<tr>
<td>4-1Ø.</td>
<td>Read Data Transfer Timing Diagram</td>
<td>4-22</td>
</tr>
<tr>
<td>4-11.</td>
<td>Local Memory Map</td>
<td>4-26</td>
</tr>
<tr>
<td>5-1.</td>
<td>Parts Location Diagram</td>
<td>5-3</td>
</tr>
<tr>
<td>5-2.</td>
<td>Jumper Location Diagram</td>
<td>5-7</td>
</tr>
<tr>
<td>5-3.</td>
<td>Schematic Diagram</td>
<td>5-9</td>
</tr>
<tr>
<td>A-1.</td>
<td>Recommended Sector Format</td>
<td>A-1</td>
</tr>
</tbody>
</table>

***

xii
1.1 INTRODUCTION

The Intel iSBC 215 Generic Winchester Disk Controller Board (referred to as the iSBC 215G board in this manual) allows as many as four hard-disk drives (typically, Winchester technology), as many as four flexible-diskette drives, and as many as four magnetic cartridge-tape drives to be interfaced to any Intel MULTIBUS interface compatible computer system. It supports disk drives that use open-loop head positioning, closed-loop head positioning, or ANSI X3T9/1226 interfaces. Figures 1-1 and 1-2 show examples of multiple hard-disk drive applications in non-ANSI and ANSI configurations, respectively.

The iSBC 215G board design is based on the Intel 8089 8/16-Bit HMOS I/O Processor, which features direct-memory-access (DMA) transfers, multiple-sector transfers, transparent error detection and correction (with automatic recovery and retry), and data management. The board operates in a multiprocessor environment and is fully compatible with all Intel 8- and 16-bit computers. The number of tracks per surface is software selectable for each drive unit. Seek operations on more than one drive can be overlapped with read/write operations on other drives. The iSBC 215G board is fully compatible with Intel 8086 16-Bit HMOS Microprocessor 20-bit addressing, and can be used in Intel MULTIBUS 24-bit address systems.

The board includes two Intel iSBX bus connectors, J3 and J4, that allow other storage devices such as flexible-diskette drives or magnetic cartridge-tape drives to be operated with MULTIBUS interface compatible systems. For example, the Intel iSBX 218A Flexible Diskette Controller Board attaches to iSBX bus connector J4, allowing the iSBC 215G board to control as many as four flexible-diskette drives. Figure 1-3 shows an example multiple-drive system using four 5 1/4- or 8-inch flexible-diskette drives, the iSBC 215G board, and the iSBX 218A Flexible Diskette Controller. As another example, the Intel iSBX 217B/C Magnetic Cartridge Tape Interface Board attaches to iSBX bus connector J3, allowing the iSBC 215G board to control as many as four tape drives.

1.2 DESCRIPTION

The iSBC 215G board is a single, multi-layer printed-circuit board assembly. It may be installed in any Intel backplane or custom-designed configuration that is physically and electrically compatible with the Intel MULTIBUS interface.

The host central processing unit (CPU) communicates with the iSBC 215G board via four blocks of information in host memory. Once the iSBC 215G board is initialized, a CPU I/O write to the board wake-up address initiates activities. The board accesses the four blocks in the host memory to determine the specific operation being performed, fetches the required parameters, and completes the specified operation without CPU intervention.
Figure 1-1. Example of Multiple Drive System Using Non-ANSI Interface

Figure 1-2. Example of Multiple Drive System Using ANSI Interface
The iSBC 215G 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 board 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 the CPU memory. During a disk write operation, the board 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 functions, the board appends an error checking code (ECC) at the end of each sector ID and data field. This ECC is used for checking and correcting data errors. It corrects all errors in bursts of as many as 11 bits, and detects all errors in bursts of as many as 32 bits (see Figure 1-4).

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 2764 EPROM's provide on-board storage of the board I/O control program and a resident diagnostic exerciser, and Intel 2114 Static RAM's provide local memory data buffering and temporary storage for read/write parameters.
GENERAL INFORMATION

1.3 SPECIFICATIONS

Table 1-1 lists the specifications of the iSBC 215G board; Tables 1-2, 1-3, and 1-4 list typical characteristics of compatible disk drives. Note that the drives listed in Tables 1-2, 1-3, and 1-4 are representative only and are not qualified or endorsed by Intel, and that Intel assumes no responsibility to update or keep the list current.
### Table 1-1. Board Specifications

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor</strong></td>
<td>Any Intel mainframe or any MULTIBUS interface compatible CPU. (The iSBC 215G board can operate with 16-, 20-, or 24-bit addresses and with 8- or 16-bit data bus widths.)</td>
</tr>
<tr>
<td><strong>Drive Type</strong></td>
<td>Disk Drives -- Either hard-disk (Winchester) or flexible-disk (through optional on-board iSBX Flexible Diskette Controller Board).</td>
</tr>
<tr>
<td><strong>Drives per Controller</strong></td>
<td>Hard-Disk Drives -- As many as two 5 1/4-inch or four 8-inch non-ANSI drives. As many as four ANSI X3T9/1226 drives. Flexible-Disk Drives -- As many as four 5 1/4- or 8-inch drives through iSBX 218A Flexible Disk Controller connected to iSBC 215G board iSBX bus connector J4. Tape Drives -- As many as four 1/4-inch magnetic cartridge-tape drives through iSBX 217B or C Magnetic Cartridge Tape Interface Board connected to iSBC 215G board iSBX bus connector J3.</td>
</tr>
<tr>
<td><strong>Error Detection and Correction</strong></td>
<td>Error detection to 32 bits in length; error correction to 11 consecutive bits in length.</td>
</tr>
<tr>
<td><strong>Power Requirements</strong></td>
<td>+5 V ±5 % @ 4.52 A maximum</td>
</tr>
<tr>
<td></td>
<td>-5 V ±5 % @ 0.015 A maximum</td>
</tr>
<tr>
<td><strong>Mounting</strong></td>
<td>Occupies one card slot in MULTIBUS compatible card cage or backplane connector. Occupies two slots in most card cages when optional iSBX MULTIMODULE is installed.</td>
</tr>
</tbody>
</table>

Note: Jumper selection and on-board voltage regulator allow use of -10 V or -12 V from MULTIBUS connector as alternate to direct -5 V source.
## GENERAL INFORMATION

### Table 1-1. Board Specifications (continued)

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Physical</strong></td>
<td></td>
</tr>
<tr>
<td>Width</td>
<td>17.2 cm (6.8 inches)</td>
</tr>
<tr>
<td>Length</td>
<td>30.5 cm (12.0 inches)</td>
</tr>
<tr>
<td>Thickness</td>
<td>1.3 cm (0.5 inch)</td>
</tr>
<tr>
<td>Weight</td>
<td>0.54 kg (19 ounces)</td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td></td>
</tr>
<tr>
<td>Temperature</td>
<td>Operating: 0°C to +55°C,</td>
</tr>
<tr>
<td></td>
<td>Non-Operating: -40°C to +70°C</td>
</tr>
<tr>
<td></td>
<td>(+32°F to +131°F)</td>
</tr>
<tr>
<td></td>
<td>(-40°F to +158°F)</td>
</tr>
<tr>
<td>Humidity</td>
<td>5 to 95 %,</td>
</tr>
<tr>
<td></td>
<td>non-condensing</td>
</tr>
</tbody>
</table>

### Table 1-2. Allowable Sectors Per Track for Non-ANSI Hard-Disk Drives

<table>
<thead>
<tr>
<th>Disk Drive</th>
<th>Data Bytes per Sector</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>256</td>
</tr>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td></td>
<td>1024</td>
</tr>
<tr>
<td>Priam 8-in</td>
<td>72</td>
</tr>
<tr>
<td>Priam 14-in</td>
<td>107</td>
</tr>
<tr>
<td>Ampex, RMS, CMI, Shugart, Quantum</td>
<td>54</td>
</tr>
<tr>
<td>Fujitsu, Memorex</td>
<td>64</td>
</tr>
<tr>
<td>Shugart 14-in</td>
<td>96</td>
</tr>
<tr>
<td>CDC</td>
<td>64</td>
</tr>
</tbody>
</table>
### Table 1-3. Allowable Sectors Per Track for ANSI Hard-Disk Drives

<table>
<thead>
<tr>
<th>Disk Drive</th>
<th>Data Bytes per Sector</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>128</td>
</tr>
<tr>
<td>3M</td>
<td>82</td>
</tr>
<tr>
<td>Kennedy, BASF</td>
<td>74</td>
</tr>
<tr>
<td>Micropolis</td>
<td>71</td>
</tr>
<tr>
<td>Pertec</td>
<td>85</td>
</tr>
</tbody>
</table>

### Table 1-4. Formatted Capacity Per Hard-Disk Drive (in Mbytes)

<table>
<thead>
<tr>
<th>Disk Drive</th>
<th>Data Bytes per Sector</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>128</td>
</tr>
<tr>
<td>Shugart SA 1ØØ4</td>
<td>7.Ø8</td>
</tr>
<tr>
<td>Quantum Q2Ø1Ø</td>
<td>7.Ø8</td>
</tr>
<tr>
<td>Shugart SA4ØØ8</td>
<td>19.86</td>
</tr>
<tr>
<td>Priam 345Ø, 941Ø-32</td>
<td>23.96</td>
</tr>
<tr>
<td>Fujitsu 23ØØ, Memorex 1Ø1</td>
<td>7.99</td>
</tr>
<tr>
<td>RMS 512, CMI</td>
<td>8.4Ø</td>
</tr>
<tr>
<td>CDC</td>
<td>19.5Ø</td>
</tr>
<tr>
<td>3M 8432</td>
<td>11.76</td>
</tr>
<tr>
<td>Kennedy, BASF 6172</td>
<td>17.45</td>
</tr>
<tr>
<td>Micropolis 12Ø3</td>
<td>26.65</td>
</tr>
<tr>
<td>Pertec D8Ø35 (ANSI)</td>
<td>21.85</td>
</tr>
</tbody>
</table>

***

1-7
2.1 INTRODUCTION

This chapter provides information for use in preparing and installing the iSBC 215G board. Included are instructions for unpacking and inspection, installation, installing jumpers, connecting the board to the MULTIBUS interface, and preparing and connecting cabling to the disk drives.

2.2 UNPACKING AND INSPECTION

Inspect the shipping carton immediately upon receipt for evidence of mishandling during transit. If the shipping carton is severely 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 when the carton is opened and the contents are damaged, keep the carton and packing materials for subsequent agent inspection.

For repair of a product damaged during shipment, contact the Intel Product Service Center to obtain a Return Authorization Number and further instructions. (Chapter 5 lists the telephone numbers for the various centers.) A purchase order is required to complete the repair. Submit a copy of the purchase order to the carrier with your claim.

2.3 INSTALLATION CONSIDERATIONS

The iSBC 215G board can be installed in any Intel cardcage/backplane or any user-designed backplane that meets the MULTIBUS interface specification. The board occupies one backplane slot. An additional slot may be required if an iSBX MULTIMODULE is installed.

Because the iSBC 215G board operates as a system master, the slot into which it is installed must include bus priority arbitration capability. Priority resolution can be done in either serial or parallel fashion.

2.3.1 POWER REQUIREMENTS

The board requires a source of +5 V ±5 % power, at a maximum current of 4.52 A. This is supplied through the MULTIBUS connector. When interfacing with 8-inch Shugart and Quantum drives, an additional source of -5 V ± 5 % power, at a maximum current of 15 mA, is required. This supply can be obtained directly from the MULTIBUS connector or from an on-board regulator that uses the MULTIBUS -10 V or -12 V source.

When interfacing with an iSBX bus through J3 or J4, additional voltage sources of +12 V, -12 V, or both, may be required. These can also be supplied through the MULTIBUS connector (see the individual iSBX Board
specifications for tolerances and current requirements). Before installing the iSBC 215G board in a system chassis, make certain that the associated power supplies can supply the required additional current.

NOTE

If power is applied to or removed from the iSBC 215G board while a drive is ready, a spurious disk write operation could occur. To prevent this, always make certain that the drives are not turning when the iSBC 215G board power is switched on or off.

2.3.2 COOLING REQUIREMENTS

The iSBC 215G board (with no iSBX boards installed) dissipates 338.2 gram calories of heat (1.34 Btu's) per minute. Sufficient cooling air circulation (approximately 2Ø linear feet per minute under ordinary conditions) must be provided to keep the board within the required operating temperature range (Ø to 55° C).

2.3.3 PHYSICAL CHARACTERISTICS

The dimensions, outline, and connector and jumper locations of the iSBC 215G board are shown in Figure 5-2.

2.4 JUMPER CONFIGURATIONS AND JUMPER INSTALLATION

Various configurations of the iSBC 215G board can be accommodated through the jumper stake pins provided on the board. A variable number of jumpers may be installed by the user on pairs of these stake pins to conveniently set up the board for the system environment in which it is to operate (8- or 16-bit system data bus; 16-, 2Ø-, or 24-bit addressing, etc.) and for the type of device to which it is to be interfaced (Shugart, Quantum, Memorex, etc. drive, or iSBX board). The default configuration includes approximately 4Ø jumpers, 2 of which are soldered and 1 wire-wrapped in place.

Each jumper is identified by its "W" number and the numbers of the two stake pins used (for example: W21-1 -- 2, or W3Ø-1 -- 2Ø). For jumper stake pin physical locations and details on jumper layouts, refer to Figure 5-2. In Figure 5-2, a § symbol following a jumper number denotes the default configuration. The board should be configured, as described in the following paragraphs, before its installation in a system.
NOTE

An asterisk or slash following a signal mnemonic denotes that the signal is active when in the low state.

2.4.1 WAKE-UP ADDRESS SELECTION

The iSBC 215G board communicates with the host CPU through four I/O communication blocks located in the host memory. The board receives instructions by reading the contents of the beginning address of the first I/O communication block. These contents are called the wake-up address, and may be at any address for a 20- or 16-bit host system. Omitting jumper W36-1 -- 2 allows a 24-bit address host system to place the I/O communication blocks in the first 1-Mbyte page (that is, address 0XXXXXXXX); installing jumper W36-1 -- 2 allows placement in the last page (that is, address FXXXXXXXX). If the host CPU does not provide 24-bit addressing, or if the backplane does not provide for the upper four address lines, this jumper must not be installed.

Sixteen stake pin pairs are provided on the iSBC 215G board to allow the user to set the wake-up address. Eight of the pairs are identified as jumper W29. The other eight are identified as jumper W3Ø, which also includes two more pairs. (Of the additional W3Ø pairs, one specifies an 8-bit or 16-bit wake-up I/O port address and one specifies the system data bus width – see Sections 2.4.2 and 2.4.3.) The function, number, and location of each jumper are shown in Table 2-1 and Figure 5-2. An installed jumper represents a logical 1.

<table>
<thead>
<tr>
<th>Jumper</th>
<th>From</th>
<th>To</th>
<th>Wake-Up Address Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>W29-1</td>
<td>W29-16</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>W29-2</td>
<td>W29-15</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>W29-3</td>
<td>W29-14</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>W29-4</td>
<td>W29-13</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>W29-5</td>
<td>W29-12</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>W29-6</td>
<td>W29-11</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>W29-7</td>
<td>W29-10</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>W29-8</td>
<td>W29-9</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>W3Ø-3</td>
<td>W3Ø-18</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>W3Ø-4</td>
<td>W3Ø-17</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>W3Ø-5</td>
<td>W3Ø-16</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>W3Ø-6</td>
<td>W3Ø-15</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>W3Ø-7</td>
<td>W3Ø-14</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>W3Ø-8</td>
<td>W3Ø-13</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>W3Ø-9</td>
<td>W3Ø-12</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>W3Ø-1Ø</td>
<td>W3Ø-11</td>
<td>Ø</td>
<td></td>
</tr>
</tbody>
</table>
The board 8089 I/O processor (IOP) treats the wake-up address as the segment portion of the standard segment and offset 20-bit addressing. For the wake-up address, the IOP uses an offset of 0. This multiplies the settings of the wake-up address jumpers by $2^4$ (that is, it shifts the number four places to the left) to create a 20-bit wake-up address from 16-bits.

2.4.2 WAKE-UP I/O PORT ADDRESS SELECTION

The host CPU communicates with the iSBC 215G board through an I/O port, the number of which is also set by the wake-up address jumpers. For a host CPU with 8-bit I/O port addressing, bits 0 through 7 of the wake-up address determine the wake-up I/O port number; for a host CPU with 16-bit I/O port addressing, bits 0 through F determine the port number. Jumper W30-2 -- 19 (see Figure 5-2) determines the type of I/O port addressing used by the host CPU. It is installed for use with a 16-bit host CPU such as the Intel 8086; not installed for use with an 8-bit host CPU such as the Intel 8085.

2.4.3 SYSTEM DATA BUS WIDTH SELECTION

System data bus width selection jumper W30-1 -- 20 (see Figure 5-2) sets the board for the type of system data bus with which the iSBC 215G board is to interface. It is installed for a 16-bit data path, not installed for an 8-bit data path. Installing the jumper allows use of 16-bit data transfer mode to access the system bus (if the system memory supports 16-bit accesses), even though the host CPU supports only 8-bit accesses.

2.4.4 INTERRUPT PRIORITY NUMBER

The iSBC 215G board internal interrupt request signal can be assigned to any of eight MULTIBUS interrupt priority numbers (0* through INT7*). The number is selected by wire wrapping two jumper stake pins (see Figure 5-2) together as indicated in Table 2-2.

Table 2-2. Interrupt Priority Number Selection

<table>
<thead>
<tr>
<th>Interrupt Number</th>
<th>Install Wire-Wrap Jumper</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>From Stake Pin:</td>
</tr>
<tr>
<td>Ø</td>
<td>W19-C</td>
</tr>
<tr>
<td>1</td>
<td>W19-C</td>
</tr>
<tr>
<td>2</td>
<td>W19-C</td>
</tr>
<tr>
<td>3</td>
<td>W19-C</td>
</tr>
<tr>
<td>4</td>
<td>W19-C</td>
</tr>
<tr>
<td>5</td>
<td>W19-C</td>
</tr>
<tr>
<td>6</td>
<td>W19-C</td>
</tr>
<tr>
<td>7</td>
<td>W19-C</td>
</tr>
</tbody>
</table>
2.4.5 BUS PRIORITY ARBITRATION

Bus priority arbitration controls the sequence in which access is allowed to the MULTIBUS interface. Access priority is determined by three signals in combination: ANYRQST, CBRQ*, and BPRO*. These are described in the following paragraphs.

2.4.5.1 Common Bus Request (CBRQ*)/Any Request (ANYRQST) Signal Selection

The CBRQ* and ANYRQST signals provide the required mode select inputs to the 8289 Bus Arbiter. The arbitration options are shown in Table 2-3.

CBRQ* is a bi-directional interface signal that improves bus access time by allowing a bus master to retain control of the MULTIBUS interface without contending for it on each transfer cycle, as long as no other master is requesting control of the bus. The signal is either supplied from the bus via connector P1 or connected to ground, dependent upon the position of jumper W23. This signal operates the same in parallel and serial priority resolution schemes.

ANYRQST is a bus arbiter input signal that controls whether the iSBC 215G board will allow a lower-priority device to gain access to the MULTIBUS interface by the CBRQ* signal. The signal is either high (connected to +5 V through a resistor), or low (connected to ground), dependent upon the position of jumper W18. When ANYRQST is high, a lower-priority device may gain control of the bus by activating the CBRQ* signal. When ANYRQST is low, a lower-priority device cannot gain control of the bus until it gains priority through the BPRN* signal.

Table 2-3. Bus Arbitration Options

<table>
<thead>
<tr>
<th>Signal</th>
<th>Jumper</th>
<th>Connect To</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBRQ*</td>
<td>W23-1</td>
<td>Bus</td>
<td>Arbitrate to gain access to MULTIBUS interface. If continued access is required, iSBC 215G board retains control until higher-priority device requests bus, at which time board arbitrates again and surrenders bus control to only that device.</td>
</tr>
<tr>
<td></td>
<td>and</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANYRQST</td>
<td>W18-1</td>
<td>Ground</td>
<td></td>
</tr>
<tr>
<td>CBRQ*</td>
<td>W23-1</td>
<td>Bus</td>
<td>Arbitrate to gain access to MULTIBUS interface. If continued access is required, iSBC 215G board retains control until another device requests bus, at which time board arbitrates again and surrenders bus control to requesting device (either higher or lower priority).</td>
</tr>
<tr>
<td></td>
<td>and</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANYRQST</td>
<td>W18-1</td>
<td>+5 V</td>
<td></td>
</tr>
<tr>
<td>CBRQ*</td>
<td>W23-1</td>
<td>Ground</td>
<td>Arbitrate for every bus access.</td>
</tr>
<tr>
<td></td>
<td>and</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANYRQST</td>
<td>W18-1</td>
<td>+5 V</td>
<td></td>
</tr>
</tbody>
</table>
PREPARATION FOR USE

2.4.5.2 Bus Priority Out (BPRO*) Signal Selection

The BPRO* signal is used in serial MULTIBUS priority schemes. BPRO* must be connected to the BPRN* input of the bus master with the next lower priority. The BPRO* signal is enabled for serial resolution by installing jumper W28-1 -- 2 (see Figure 5-2), or disabled for parallel resolution by omitting the jumper.

2.4.6 MULTIBUS® LOCK (LOCK*) SIGNAL

The LOCK* signal is used by the current bus master to exclude a dual-port RAM from use through the alternate port (for instance, the iSBC 86/35 single board computer, the iSBC Ø12CX memory board, etc.) when a multi-transfer operation (for instance, a read-modify-write) is required. The LOCK* signal is enabled by installing jumper W32-1 -- 2 (see Figure 5-2), or disabled by omitting the jumper.

2.4.7 iSBX™ BUS SELECTION

The iSBX bus control jumpers, W3, W4, W11, W12, and W24 (see Figure 5-2) select the external-terminate and DMA-request lines on the iSBX bus as shown in Table 2-4. Instructions are included in Chapter 3 for writing iSBC 215G board-to-drive interface software for I/O modules designed to iSBX Bus Specifications.

Table 2-4. iSBX™ Bus Control Jumpers

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Installed?</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>W3-1 -- 2</td>
<td>Yes</td>
<td>EXT TRM -- External terminate (J3); terminated on iSBC 215G board.</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>EXT TRM -- External terminate (J3); driven by iSBX I/O controller.</td>
</tr>
<tr>
<td>W4-1 -- 2</td>
<td>Yes</td>
<td>EXT TRM -- External terminate (J4); terminated on iSBC 215G board.</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>EXT TRM -- External terminate (J4); driven by iSBX I/O controller.</td>
</tr>
<tr>
<td>W11-1 -- 2</td>
<td>Yes</td>
<td>OPØØ -- Option Ø (J3) driving.</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>OPØØ -- Option Ø (J3) receiving.</td>
</tr>
<tr>
<td>W11-1 -- 3</td>
<td>Yes</td>
<td>OPØ1 -- Option Ø (J4) driving.</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>OPØ1 -- Option Ø (J4) receiving.</td>
</tr>
</tbody>
</table>
### Table 2-4. iSBX™ Bus Control Jumpers (continued)

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Installed?</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>W12-l -- 2</td>
<td>Yes</td>
<td>OP10 -- Option 1 (J3) driving.</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>OP10 -- Option 1 (J3) receiving.</td>
</tr>
<tr>
<td>W12-l -- 3</td>
<td>Yes</td>
<td>OP11 -- Option 1 (J4) driving.</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>OP11 -- Option 1 (J4) receiving.</td>
</tr>
<tr>
<td>W24-l -- 2</td>
<td>Yes</td>
<td>DREQ0 -- iSBX controller on J4 uses DMA request; iSBX controller on J3 does not use DMA request or is not installed.</td>
</tr>
<tr>
<td>W24-l -- 3</td>
<td>Yes</td>
<td>DREQ1 -- iSBX controller on J3 uses DMA request; iSBX controller on J4 does not use DMA request or is not installed.</td>
</tr>
<tr>
<td>W24-l -- 2</td>
<td>No</td>
<td>DREQ0/DREQ1 -- Both iSBX controllers use DMA requests, or neither uses DMA requests, or neither is installed.</td>
</tr>
<tr>
<td>and W24-l -- 3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### 2.4.8 HARD-DISK DRIVE INTERFACE

The iSBC 215G board is designed to communicate with either ANSI compatible (X3T9/1226) or proprietary non-ANSI hard-disk (Winchester technology) drive interfaces. It can control as many as four disk drives, except for certain units (for instance, Memorex, Shugart 14-inch, Priam, or CDC Finch Series). Two drives are supported for the excepted types. In all instances, drives from only one manufacturer at a time may be used, unless the drives are 100-percent compatible.

The jumpers listed in Table 2-5 allow the user to configure the iSBC 215G board for the listed drive types (see Figure 5-2). Other drive types may be used; however, Intel assumes no obligation to determine the appropriate jumper configuration. Interface cables must also be constructed and installed (according to the type of drive being used) as described later in this chapter.
Table 2-5. Jumper Configuration for Various Hard-Disk Drives

<table>
<thead>
<tr>
<th>VENDOR CONFIGURATION TABLE</th>
<th>DRIVE INTERFACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>WIRE NO.</td>
<td>FUNCTION</td>
</tr>
<tr>
<td>W1</td>
<td>CMD BUS ENG*</td>
</tr>
<tr>
<td>W2</td>
<td>VENDOR #</td>
</tr>
<tr>
<td>W3</td>
<td>ROAT</td>
</tr>
<tr>
<td>W4</td>
<td>RD*</td>
</tr>
<tr>
<td>W5</td>
<td>ROCIF</td>
</tr>
<tr>
<td>W6</td>
<td>ROCUP*</td>
</tr>
<tr>
<td>W7</td>
<td>T/R (BA)</td>
</tr>
<tr>
<td>W8</td>
<td>RACIAL SELECT</td>
</tr>
<tr>
<td>W9</td>
<td>RD GATE</td>
</tr>
<tr>
<td>W10</td>
<td>DF CONTROL</td>
</tr>
<tr>
<td>W11</td>
<td>SHUGART (SA 1000)</td>
</tr>
<tr>
<td>W12</td>
<td>SHUGART (SA 4000)</td>
</tr>
<tr>
<td>W13</td>
<td>SECTORING</td>
</tr>
<tr>
<td>W14</td>
<td>RD GATE</td>
</tr>
<tr>
<td>W15</td>
<td>INDEX SELECT</td>
</tr>
<tr>
<td>W16</td>
<td>RD CL</td>
</tr>
<tr>
<td>W17</td>
<td>VENDOR 1</td>
</tr>
<tr>
<td>W18</td>
<td>VENDOR 2</td>
</tr>
<tr>
<td>W19</td>
<td>VENDOR 3</td>
</tr>
<tr>
<td>W20</td>
<td>VENDOR 4</td>
</tr>
<tr>
<td>W21</td>
<td>SECTOR</td>
</tr>
<tr>
<td>W22</td>
<td>DCF PROG</td>
</tr>
<tr>
<td>W23</td>
<td>PROG. SECTOR</td>
</tr>
<tr>
<td>W24</td>
<td>10V/12V</td>
</tr>
<tr>
<td>W25</td>
<td>-5V</td>
</tr>
</tbody>
</table>

2.4.9 -5 Volt SELECTION

For interfacing with drives that require -5 V power (8-inch Shugart or Quantum or CDC drives), the power source and regulator voltage source must be selected. Install the two jumpers (see Figure 5-2) as indicated in Table 2-6 to select: 1) -5 V from either the MULTIBUS connector or the on-board regulator, and 2) the voltage source for the regulator.
Table 2-6. -5 V Selection Jumper Configuration

<table>
<thead>
<tr>
<th>Install Jumper:</th>
<th>For:</th>
</tr>
</thead>
<tbody>
<tr>
<td>W21-1 -- 2</td>
<td>-5 V from MULTIBUS interface</td>
</tr>
<tr>
<td>W21-1 -- 3</td>
<td>-5 V from regulator</td>
</tr>
<tr>
<td>W2Ø-1 -- 2</td>
<td>-12 V to regulator</td>
</tr>
<tr>
<td>W2Ø-1 -- 3</td>
<td>-10 V to regulator</td>
</tr>
</tbody>
</table>

2.4.10 I/O COMMUNICATION BLOCKS PAGE SELECTION

In the default configuration, all I/O communication blocks are located in the lowest 1-Mbyte page of the 16-Mbyte MULTIBUS address space. The user can select the highest page by installing jumper W36-1 -- 2 (see Figure 5-2).

2.4.11 RAM JUMPER

Jumper W31 is factory default connected as W31-1 to W31-2. It is not re-configurable.

2.5 INTERFACE CONNECTIONS

The iSBC 215G board communicates with the CPU and other boards via the MULTIBUS interface (connectors P1 and P2), and with the various storage drives via special cables (connector J1, J2, or J5, as appropriate).

2.5.1 MULTIBUS® INTERFACE

All interconnections between the iSBC 215G board and the MULTIBUS interface are accomplished through the two MULTIBUS edge connectors, P1 and P2. Tables 2-7 and 2-8 list the pins and signals for connectors P1 and P2, respectively. Tables 2-9 and 2-10 describe the signals listed for the connector pins. With reference to Tables 2-7 through 2-10, see the considerations below.

1. All odd-numbered pins (1, 3, 5, etc.) are on the component side of the board; even-numbered pins are opposite. Pin 1 is the left-most pin when viewed from the component side with the extractors at the top.

2. Cable and board connector numbering convention may not agree.

3. An asterisk or slash following a signal mnemonic denotes that the signal is active when in the low state.
Table 2-7. Connector P1 Pin Assignments

<table>
<thead>
<tr>
<th>(Component Side)</th>
<th>(Circuit Side)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pin</strong></td>
<td><strong>Mnemonic</strong></td>
</tr>
<tr>
<td>1</td>
<td>GND</td>
</tr>
<tr>
<td>3</td>
<td>+5 V</td>
</tr>
<tr>
<td>5</td>
<td>+5 V</td>
</tr>
<tr>
<td>7</td>
<td>+12 V</td>
</tr>
<tr>
<td>9</td>
<td>-5 V</td>
</tr>
<tr>
<td>11</td>
<td>GND</td>
</tr>
<tr>
<td>13</td>
<td>BCLK*</td>
</tr>
<tr>
<td>15</td>
<td>BPRN*</td>
</tr>
<tr>
<td>17</td>
<td>BUSY*</td>
</tr>
<tr>
<td>19</td>
<td>MRDC*</td>
</tr>
<tr>
<td>21</td>
<td>IORC*</td>
</tr>
<tr>
<td>23</td>
<td>XACK*</td>
</tr>
<tr>
<td>25</td>
<td>LOCK*</td>
</tr>
<tr>
<td>27</td>
<td>BHEN*</td>
</tr>
<tr>
<td>29</td>
<td>CBREQ*</td>
</tr>
<tr>
<td>31</td>
<td>CCLK*</td>
</tr>
<tr>
<td>33</td>
<td>INTA*</td>
</tr>
<tr>
<td>35</td>
<td>INT6*</td>
</tr>
<tr>
<td>37</td>
<td>INT4*</td>
</tr>
<tr>
<td>39</td>
<td>INT2*</td>
</tr>
<tr>
<td>41</td>
<td>INT0*</td>
</tr>
<tr>
<td>43</td>
<td>ADRE*</td>
</tr>
<tr>
<td>45</td>
<td>ADRC*</td>
</tr>
<tr>
<td>47</td>
<td>ADRA*</td>
</tr>
<tr>
<td>49</td>
<td>ADR8*</td>
</tr>
<tr>
<td>51</td>
<td>ADR6*</td>
</tr>
<tr>
<td>53</td>
<td>ADR4*</td>
</tr>
<tr>
<td>55</td>
<td>ADR2*</td>
</tr>
<tr>
<td>57</td>
<td>ADR0*</td>
</tr>
<tr>
<td>59</td>
<td>DATE*</td>
</tr>
<tr>
<td>61</td>
<td>DATC*</td>
</tr>
<tr>
<td>63</td>
<td>DATA*</td>
</tr>
<tr>
<td>65</td>
<td>DAT8*</td>
</tr>
<tr>
<td>67</td>
<td>DAT6*</td>
</tr>
<tr>
<td>69</td>
<td>DAT4*</td>
</tr>
<tr>
<td>71</td>
<td>DAT2*</td>
</tr>
<tr>
<td>73</td>
<td>DAT0/</td>
</tr>
<tr>
<td>75</td>
<td>GND</td>
</tr>
<tr>
<td>77</td>
<td>-10 V</td>
</tr>
<tr>
<td>79</td>
<td>-12 V</td>
</tr>
<tr>
<td>81</td>
<td>+5 V</td>
</tr>
<tr>
<td>83</td>
<td>+5 V</td>
</tr>
<tr>
<td>85</td>
<td>GND</td>
</tr>
</tbody>
</table>
**Table 2-8. Connector P2 Pin Assignments**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Mnemonic</th>
<th>Description</th>
<th>Pin</th>
<th>Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Not Connected</td>
<td>2</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Not Connected</td>
<td>4</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Not Connected</td>
<td>6</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>Not Connected</td>
<td>8</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>Not Connected</td>
<td>10</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>Not Connected</td>
<td>12</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>Not Connected</td>
<td>14</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>Not Connected</td>
<td>16</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>Not Connected</td>
<td>18</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td>Not Connected</td>
<td>20</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>Not Connected</td>
<td>22</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>Not Connected</td>
<td>24</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>25</td>
<td></td>
<td>Not Connected</td>
<td>26</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>27</td>
<td></td>
<td>Not Connected</td>
<td>28</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>29</td>
<td></td>
<td>Not Connected</td>
<td>30</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>Not Connected</td>
<td>32</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>33</td>
<td></td>
<td>Not Connected</td>
<td>34</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td>Not Connected</td>
<td>36</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>37</td>
<td></td>
<td>Not Connected</td>
<td>38</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>39</td>
<td></td>
<td>Not Connected</td>
<td>40</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>41</td>
<td></td>
<td>Not Connected</td>
<td>42</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>43</td>
<td></td>
<td>Not Connected</td>
<td>44</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>Not Connected</td>
<td>46</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>47</td>
<td></td>
<td>Not Connected</td>
<td>48</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>49</td>
<td></td>
<td>Not Connected</td>
<td>50</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>51</td>
<td></td>
<td>Not Connected</td>
<td>52</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>53</td>
<td></td>
<td>Not Connected</td>
<td>54</td>
<td></td>
<td>Not Connected</td>
</tr>
<tr>
<td>55</td>
<td>ADR16*</td>
<td>Address Bus</td>
<td>56</td>
<td>ADR17*</td>
<td>Address Bus</td>
</tr>
<tr>
<td>57</td>
<td>ADR14*</td>
<td>Address Bus</td>
<td>58</td>
<td>ADR15*</td>
<td>Address Bus</td>
</tr>
<tr>
<td>59</td>
<td></td>
<td>Not Connected</td>
<td>60</td>
<td></td>
<td>Not Connected</td>
</tr>
</tbody>
</table>
## Table 2-9. Connector PI Input/Output Signals

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADR0* -- ADR13*</td>
<td><strong>Address Bits:</strong> Specify part of memory address or I/O port address to be accessed. ADR0* through ADR13* are used for normal 16-bit address selection and are shifted 4 places to derive 20-bit addresses. Address bits ADR14* through ADR17* (which are listed in Table 2-9), are also used for address selection. When bits specify memory address, ADR0*, in conjunction with BHEN*, enables even-byte bank on MULTIBUS interface. When bits specify I/O port, only address bits ADR0* through ADRF* are used.</td>
</tr>
<tr>
<td>BHEN*</td>
<td><strong>Byte High Enable:</strong> Determines, in conjunction with ADR0*, byte bank data to be transferred.</td>
</tr>
<tr>
<td>BCLK*</td>
<td><strong>Bus Clock:</strong> Synchronizes bus contention logic on all bus masters.</td>
</tr>
<tr>
<td>CCLK*</td>
<td><strong>Constant Clock:</strong> Provides for synchronization of all devices using MULTIBUS interface. Master clock signal.</td>
</tr>
<tr>
<td>BPRN*</td>
<td><strong>Bus Priority In:</strong> Indicates to particular bus master that no higher priority master is requesting use of bus. BPRN* is synchronized with BCLK*.</td>
</tr>
<tr>
<td>BPRO*</td>
<td><strong>Bus Priority Out:</strong> In serial priority resolution scheme, indicates to lower priority bus master that neither it (master issuing BPRO* signal) nor higher master is requesting use of bus.</td>
</tr>
<tr>
<td>BREQ*</td>
<td><strong>Bus Request:</strong> In parallel priority resolution scheme, indicates that issuing bus master requires control of bus for one or more data transfers. BREQ* is synchronized with BCLK*.</td>
</tr>
<tr>
<td>BUSY*</td>
<td><strong>Busy:</strong> Indicates that bus is in use and prevents other bus masters from gaining control. BUSY* is synchronized with BCLK*.</td>
</tr>
<tr>
<td>CBRQ*</td>
<td><strong>Common Bus Request:</strong> Indicates that bus master requires control of bus but does not have such control. As soon as control is attained, controlling master raises CBRQ* signal.</td>
</tr>
</tbody>
</table>

2-12
## Table 2-9. Connector P1 Input/Output Signals (continued)

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCK*</td>
<td><strong>MULTIBUS Bus Lock</strong>: Prevents off-board requests for on-board dual-port RAM use.</td>
</tr>
<tr>
<td>DAT0* -- DATF*</td>
<td><strong>Data Lines</strong>: Provide for transmitting or receiving 16 parallel bits of data to or from selected memory address or I/O port. For byte data operations, bits DAT0* through DAT7* constitute even byte and bits DAT8* through DATF* constitute odd byte.</td>
</tr>
<tr>
<td>INH1*</td>
<td><strong>Inhibit RAM</strong>: Inhibits local RAM cycles.</td>
</tr>
<tr>
<td>INH2*</td>
<td><strong>Inhibit ROM</strong>: Inhibits local ROM cycles.</td>
</tr>
<tr>
<td>INIT*</td>
<td><strong>Initialize</strong>: Resets system to known state.</td>
</tr>
<tr>
<td>INT0* -- INT7*</td>
<td><strong>Interrupt Request</strong>: Provide for transmitting 8 interrupt requests to assigned interrupt handlers.</td>
</tr>
<tr>
<td>INTA*</td>
<td><strong>Interrupt Acknowledge</strong>: Not used.</td>
</tr>
<tr>
<td>IORC*</td>
<td><strong>I/O Read Command</strong>: Indicates that address of I/O port is on MULTIBUS address lines and that port output is to be placed on MULTIBUS data lines.</td>
</tr>
<tr>
<td>IOWC*</td>
<td><strong>I/O Write Command</strong>: Indicates that address of I/O port is on MULTIBUS address lines and that information on MULTIBUS data lines is to be accepted by addressed port.</td>
</tr>
<tr>
<td>MRDC*</td>
<td><strong>Memory Read Command</strong>: Indicates that memory address is on MULTIBUS address lines and that contents of address are to be placed on MULTIBUS data lines.</td>
</tr>
<tr>
<td>MWTC*</td>
<td><strong>Memory Write Command</strong>: Indicates that memory address is on MULTIBUS address lines and that information on MULTIBUS data lines are to be written into address.</td>
</tr>
<tr>
<td>XACK*</td>
<td><strong>Transfer Acknowledge</strong>: Indicates that specified read or write operation has been completed at memory address or I/O port (that is, data have been placed on or accepted from MULTIBUS data lines).</td>
</tr>
</tbody>
</table>
Table 2-10. Connector P2 Input/Output Signals

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADR14* -- ADR17*</td>
<td>Address Bits: Specify high-order four bits of memory address to be accessed. (See Table 2-8 for bits ADR0* -- ADR13*.) ADR14* through ADR17* are used in conjunction with shifted address bits ADR0* through ADR13* to derive 24-bit addresses (for 16-Mbyte MULTIBUS memory), and are transferred in separate CPU operation.</td>
</tr>
</tbody>
</table>

2.5.2 ISBX™ MULTIMODULE™ INTERFACE

Connectors J3 and J4 on the ISBC 215G board are designed to interface with Intel ISBX I/O controllers or other I/O modules designed to meet the Intel ISBX Bus Specifications. A detailed description of the ISBX bus is given in the MULTIBUS Handbook.

Note that the ISBC 215G board does not comply fully with the ISBX Specification in regard to signals DREQ (DMA Request, pin 34), MWAIT (Expansion Module Wait, pin 16), and EXTR (External Terminate, pin 26). According to the specification, these signals must be uniquely identifiable by the base board for each channel. The ISBC 215G board logically OR's these signals, which thus may be active for only one channel at any time.

The Intel ISBX 218A Flexible Diskette Controller Board connects to the J4 connector and provides an interface between the ISBC 215G board and as many as four 5 1/4- or 8-inch double-density flexible disk drives. The ISBX 218A board interfaces directly with the ISBC 215G board software as described in Chapter 3.

The Intel ISBX 217B/C Magnetic Cartridge Tape Interface Board connects to the J3 connector and provides an interface between the ISBC 215G board and as many as four industry-standard QIC-02 type 1/4-inch magnetic cartridge-tape drives. The ISBX 217B/C board interfaces directly with the ISBC 215G board software as described in Chapter 3.

I/O modules that interface the ISBC 215G board with other storage devices such as bubble memories can also be designed and connected to J3 and/or J4. The device select function of the ISBC 215G board software allows the board to be interfaced with as many as 256 different devices through both ISBX connectors J3 and J4.

The schematic diagram mnemonics for the signal and control lines (from the ISBC 215G board) that are connected to ISBX connectors J3 and J4 often differ from the respective line mnemonic from the ISBX bus specifications. Table 2-11 lists both ISBX bus and ISBC 215G board mnemonics for each signal in the ISBX bus that the board supports. Note that DMA acknowledge pin 32 is not connected on the ISBC 215G board.
# Table 2-11. iSBX™ Bus Mnemonics/iSBC® 215G Board Mnemonics

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>iSBX™ Bus Mnemonic</th>
<th>iSBC® 215G Board Mnemonic</th>
<th>J3</th>
<th>J4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+12 V</td>
<td>+12 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-12 V</td>
<td>-12 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>GND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>+5 V</td>
<td>+5 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>RESET</td>
<td>PWR RST</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MCLK</td>
<td>CCLK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MA2</td>
<td>IADR-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MPST*</td>
<td>MOPST*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MA1</td>
<td>IADR-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Reserved</td>
<td>Reserved</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>MAØ</td>
<td>IADR-Ø</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MINTRL</td>
<td>INTRIØ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>IOWRT*</td>
<td>I-ALOWC*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>MINTRØ</td>
<td>INTRØØ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>IORD*</td>
<td>I-IORÇ*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>MWAIT*</td>
<td>MWAITÔ*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>GND</td>
<td>GND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>+5 V</td>
<td>+5 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>MD 7</td>
<td>IDAT-7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>MCS1*</td>
<td>CSMMI01*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>MD6</td>
<td>IDAT-6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>MCS0*</td>
<td>CSMMI00*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>MD5</td>
<td>IDAT-5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Reserved</td>
<td>Reserved</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>MD4</td>
<td>IDAT-4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>TDMA</td>
<td>EXTRØ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>MD3</td>
<td>IDAT-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>OPT1</td>
<td>OP1Ø</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>MD2</td>
<td>IDAT-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>OPTØ</td>
<td>OPØØ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>MD1</td>
<td>IDAT-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>MDACK*</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>MDØ</td>
<td>IDAT-Ø</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>MDRQT</td>
<td>DREQØ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>GND</td>
<td>GND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>+5 V</td>
<td>+5 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>MDE</td>
<td>IDAT-E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>MDF</td>
<td>IDAT-F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>MDC</td>
<td>IDAT-C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>MDD</td>
<td>IDAT-D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>MDA</td>
<td>IDAT-A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>MDB</td>
<td>IDAT-B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>MD8</td>
<td>IDAT-8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>MD9</td>
<td>IDAT-9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.5.3 CABLING REQUIREMENTS

Interface cables between the iSBC 215G board and the disk drives must be fabricated according to the type of drive being used and the number of drives. Tables 2-12, 2-13, and 2-14 and Figures 2-1 through 2-6 show the signal mnemonics and connector pin assignments for the board and for each type of drive. A 5Ø-pin mass-terminated socket connector (3M 3425/6Ø5Ø or equivalent) is recommended for mating with J1 or J5 of the iSBC 215G board. A 4Ø-pin connector (3M 3417-6Ø4Ø or equivalent) is recommended for mating with J2.

The mass-terminated sockets are easily attached to flat ribbon cable using the jig supplied by the connector manufacturer. The control cables that connect to J1 and J5 require a 5Ø-conductor ribbon cable; the read/write cable that connects to J2 requires one or two 2Ø-conductor ribbon cables, depending on the drive configuration. Total length for either the control cable or the read/write cable must not exceed 1Ø feet. See the respective service manual for the type of connectors required for the cable end that connects to the drives.

Interconnecting cables are shown in Figures 2-7 through 2-12. Most of these require a number of wire cross-overs (scrambling) between the iSBC 215G board connectors and the drives. It is suggested that the scrambling be done at the drive interface connector. Scrambling is not required for the ANSI configuration (Figure 2-14).

NOTE

The cabling and drive interconnecting information given in this paragraph, and in Figures 2-3 through 2-14, reflect the specifications at the time this manual was printed. Before proceeding with cable construction, check the drive hardware reference manual for current pin assignments and interface requirements.
Table 2-12. Drive Interface Pin-Out Data

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>J5-1</td>
<td>J1-1</td>
</tr>
<tr>
<td>J5-2</td>
<td>J1-2</td>
</tr>
<tr>
<td>J5-3</td>
<td>J1-3</td>
</tr>
<tr>
<td>J5-4</td>
<td>J1-4</td>
</tr>
<tr>
<td>J5-5</td>
<td>J1-5</td>
</tr>
<tr>
<td>J5-6</td>
<td>J1-6</td>
</tr>
<tr>
<td>J5-7</td>
<td>J1-7</td>
</tr>
<tr>
<td>J5-8</td>
<td>J1-8</td>
</tr>
<tr>
<td>J5-9</td>
<td>J1-9</td>
</tr>
<tr>
<td>J5-10</td>
<td>J1-10</td>
</tr>
<tr>
<td>J5-11</td>
<td>J1-11</td>
</tr>
<tr>
<td>J5-12</td>
<td>J1-12</td>
</tr>
<tr>
<td>J5-13</td>
<td>J1-35</td>
</tr>
<tr>
<td>J5-14</td>
<td>J1-14</td>
</tr>
<tr>
<td>J5-15</td>
<td>J1-13</td>
</tr>
<tr>
<td>J5-16</td>
<td>J1-16</td>
</tr>
<tr>
<td>J5-17</td>
<td>J1-36</td>
</tr>
<tr>
<td>J5-18</td>
<td>J1-18</td>
</tr>
<tr>
<td>J5-19</td>
<td>J1-39</td>
</tr>
<tr>
<td>J5-20</td>
<td>J1-20</td>
</tr>
<tr>
<td>J5-21</td>
<td>J1-43</td>
</tr>
<tr>
<td>J5-22</td>
<td>J1-22</td>
</tr>
<tr>
<td>J5-23</td>
<td>J1-34</td>
</tr>
<tr>
<td>J5-24</td>
<td>J1-24</td>
</tr>
<tr>
<td>J5-25</td>
<td>J1-23</td>
</tr>
<tr>
<td>J5-26</td>
<td>J1-26</td>
</tr>
<tr>
<td>J5-27</td>
<td>J1-25</td>
</tr>
<tr>
<td>J5-28</td>
<td>J1-28</td>
</tr>
<tr>
<td>J5-29</td>
<td>J1-42</td>
</tr>
<tr>
<td>J5-30</td>
<td>J1-30</td>
</tr>
<tr>
<td>J5-31</td>
<td>J1-29</td>
</tr>
<tr>
<td>J5-32</td>
<td>J1-32</td>
</tr>
<tr>
<td>J5-33</td>
<td>J1-31</td>
</tr>
<tr>
<td>J5-34</td>
<td></td>
</tr>
<tr>
<td>J5-35</td>
<td>J1-15</td>
</tr>
<tr>
<td>J5-36</td>
<td></td>
</tr>
<tr>
<td>J5-37</td>
<td>J1-33</td>
</tr>
<tr>
<td>J5-38</td>
<td>J1-38</td>
</tr>
<tr>
<td>J5-39</td>
<td></td>
</tr>
<tr>
<td>J5-40</td>
<td></td>
</tr>
<tr>
<td>J5-41</td>
<td>J1-46</td>
</tr>
<tr>
<td>J5-42</td>
<td></td>
</tr>
<tr>
<td>J5-43</td>
<td></td>
</tr>
<tr>
<td>J5-44</td>
<td>J1-44</td>
</tr>
<tr>
<td>J5-45</td>
<td></td>
</tr>
<tr>
<td>J5-46</td>
<td></td>
</tr>
<tr>
<td>J5-47</td>
<td>J1-48</td>
</tr>
<tr>
<td>J5-48</td>
<td></td>
</tr>
</tbody>
</table>
## Table 2-12. Drive Interface Pin-Out Data (continued)

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>J1</th>
<th>J2</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>J5-49</td>
<td></td>
<td>J2-8</td>
<td>WRØ -</td>
</tr>
<tr>
<td>J5-50</td>
<td>J1-50</td>
<td></td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-20</td>
<td>RD1 +</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-21</td>
<td>RD1 -</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-23</td>
<td>RDCL1 +</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-24</td>
<td>RDCL1 -</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-27</td>
<td>WR1 -</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-26</td>
<td>WR1 +</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-30</td>
<td>WRCL1 -</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>J2-29</td>
<td>WRCL1 +</td>
</tr>
<tr>
<td>J1-17</td>
<td></td>
<td></td>
<td>FAULT*</td>
</tr>
<tr>
<td>J1-21</td>
<td></td>
<td></td>
<td>RDY*</td>
</tr>
<tr>
<td>J1-37</td>
<td></td>
<td></td>
<td>SKCOM*</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-18</td>
<td>SKCOMØ*</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-36</td>
<td>SKCOM1*</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-16</td>
<td>SECTØ*</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-34</td>
<td>SECT1*</td>
</tr>
<tr>
<td>J1-40</td>
<td>J2-14</td>
<td></td>
<td>BA1*</td>
</tr>
<tr>
<td>J1-41</td>
<td></td>
<td></td>
<td>BAØ*</td>
</tr>
<tr>
<td>J1-27</td>
<td></td>
<td></td>
<td>BACK*</td>
</tr>
<tr>
<td>J1-19</td>
<td></td>
<td></td>
<td>SAFE*</td>
</tr>
<tr>
<td>J1-49</td>
<td></td>
<td></td>
<td>US3*</td>
</tr>
<tr>
<td>J1-45</td>
<td></td>
<td></td>
<td>US1*</td>
</tr>
<tr>
<td>J1-47</td>
<td></td>
<td></td>
<td>US2*</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-4</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-10</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-22</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-26</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-27</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-28</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-29</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-30</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-38</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-39</td>
<td>No connection</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>J2-40</td>
<td>No connection</td>
</tr>
</tbody>
</table>
**Table 2-13. Control Cable Functions (J1 Complete/J5 Partial)**

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device Select</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USØ*--US3*</td>
<td>Unit Select</td>
<td>Four lines; each selects one of four disk drives.</td>
</tr>
<tr>
<td><strong>Head Select</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HSØ*--HS3*</td>
<td>Head Select</td>
<td>Four lines; in iSBC 215A/B board, select one of sixteen heads in selected drive. In iSBC 215G board, can be put to various uses; for example:</td>
</tr>
<tr>
<td>(Same as BUS Ø*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>through BUS 3*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>below)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HSØ -- Not used</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HS1 -- SAFE*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HS2 -- SELECTOUT*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HS3 -- PARA*</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>General Purpose Data Bus (Priam and ANSI)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUSØ*--BUS7*</td>
<td>Data Bus</td>
<td>Eight-bit, bi-directional data bus; transmits command and status information between iSBC 215G board and drives (includes head and cylinder data).</td>
</tr>
<tr>
<td><strong>Command Data</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRGATE*</td>
<td>Write Select</td>
<td>Enables write circuitry in drive, permitting write data sent to drive through read/write cable to be written on selected disk surface. Used with ADMKEN* line to write address mark on soft-sectored disk.</td>
</tr>
<tr>
<td>RDGATE*</td>
<td>Read Select</td>
<td>Enables read circuitry in drive, permitting data to be read from selected sector of disk. Used with ADMKEN* line to read address mark from soft-sectored disk.</td>
</tr>
<tr>
<td>DIR*</td>
<td>Direction</td>
<td>Level controls direction of head movement (low for &quot;in&quot;, high for &quot;out&quot;) when stepping head positioner.</td>
</tr>
<tr>
<td>STEP*</td>
<td>Step Head</td>
<td>Initiates head movement in selected direction.</td>
</tr>
<tr>
<td>CMND*</td>
<td>Command Data</td>
<td>Indicates that command data are present. Used in bus cycle handshaking.</td>
</tr>
<tr>
<td>Signal Mnemonic</td>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PARA*</td>
<td>Parameter Data</td>
<td>Indicates that parameter data are present. Used in bus cycle handshaking.</td>
</tr>
<tr>
<td>DRIVE REQ*</td>
<td>Status Data</td>
<td>Indicates that status data are present. Used in bus cycle handshaking.</td>
</tr>
<tr>
<td>(ATTN* for ANSI only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUS ACK</td>
<td>Bus Acknowledge</td>
<td>Acknowledges bus cycle. Used in bus cycle handshaking with commands, parameters, and status.</td>
</tr>
<tr>
<td>BACK*</td>
<td>Bus Acknowledge</td>
<td>Not used.</td>
</tr>
<tr>
<td>ADMKEN*</td>
<td>Address Mark Enable</td>
<td>Enables writing or detecting address marks (beginning of sectors) when used in conjunction with WRGATE* and RDGATE*, respectively. (See SECTOR*.)</td>
</tr>
<tr>
<td>SELECTOUT*</td>
<td>Select Unit</td>
<td>Selects and strobes selected drive.</td>
</tr>
<tr>
<td>BA0*/BA1*</td>
<td>Bus Address</td>
<td>Two binary coded lines; specify source or destination register in selected drive for bus data.</td>
</tr>
<tr>
<td>SAFE*</td>
<td>Power Safe</td>
<td>Indicates that board power is at safe level.</td>
</tr>
</tbody>
</table>

**Status Data**

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX*</td>
<td>Index</td>
<td>Indicates start of each disk revolution on selected disk drive.</td>
</tr>
<tr>
<td>SECTOR*</td>
<td>Start of Sector</td>
<td>Indicates start of sector (address mark for soft-sectored disks, sector pulse for hard-sectored disks).</td>
</tr>
<tr>
<td>FAULT*</td>
<td>Fault Condition</td>
<td>Indicates that unsafe condition has been detected in selected drive, making read/write operation reliability questionable. Ordinarily, drive logic disables read, write, and positioning circuitry until rezero operation, fault clear operation, or operator intervention.</td>
</tr>
</tbody>
</table>
### Table 2-13. Control Cable Functions (J1 Complete/J5 Partial) (continued)

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SKCOM*</td>
<td>Seek Complete</td>
<td>Indicates that selected drive has successfully completed initial head load, seek operation, or rezero operation within specified time limit.</td>
</tr>
<tr>
<td>RDY*</td>
<td>Drive Ready</td>
<td>Indicates that drive is powered up and ready to receive or transmit data.</td>
</tr>
<tr>
<td>BUSY*</td>
<td>Track Zero/Busy</td>
<td>Indicates that heads of selected drive have been positioned to cylinder (track) zero or that command is in progress.</td>
</tr>
<tr>
<td>BUS ACK*</td>
<td>Bus Acknowledge</td>
<td>Indicates that drive acknowledges parameter request or command.</td>
</tr>
<tr>
<td>CHNL ATTN*</td>
<td>Channel Attention</td>
<td>Indicates to IOP that select line should be checked to determine either board master/slave status (following reset), or channel selection.</td>
</tr>
</tbody>
</table>
Table 2-14. Read/Write Cable Functions (J2 Complete/J5 Partial)

<table>
<thead>
<tr>
<th>Signal Mnemonic</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRØ/WR1</td>
<td>Write Data</td>
<td>Line pairs; transmitted serial NRZ data (converted from TTL levels to differential signals) for recording on disk surface. Write clock synchronizes data transfer.</td>
</tr>
<tr>
<td>WRCLØ/WRCL1</td>
<td>Write Clock</td>
<td>Line pairs; transmitted clock signal for synchronizing write data transmission. Write clock is derived from read clock received from selected drive. Being obtained from rotating disk, read clock signal reflects speed variations and thus ensures proper bit transmission rate when writing as well as reading.</td>
</tr>
<tr>
<td>RDØ/RD1</td>
<td>Read Data</td>
<td>Line pairs; transmitted serial NRZ data from disk drive to iSBC 215G board (to be converted from differential signals to TTL levels) for transmission to host memory. Read clock synchronizes read data transfers.</td>
</tr>
<tr>
<td>RDCLØ/RDCL1</td>
<td>Read Clock</td>
<td>Line pairs; transmitted clock signal used to synchronize read data transmission and as timing signal for board/disk interface circuitry. Read clock is derived from rotating disk.</td>
</tr>
<tr>
<td>SECTØ*/SECT1*</td>
<td>Start of Sector</td>
<td>See SECTOR* in Table 2-12. Binary-coded signal outputs; one from each unit.</td>
</tr>
<tr>
<td>SKCOMØ*/SKCOM1*</td>
<td>Seek Complete</td>
<td>See SKCOM* in Table 2-12. Binary-coded signal outputs; one from each unit Ø and 1.</td>
</tr>
<tr>
<td>RDWCUR*</td>
<td>Reduced Write Current</td>
<td>Controls write electronics for inner tracks with higher bit densities.</td>
</tr>
</tbody>
</table>
Figure 2-1. 8-Inch Shugart/Quantum Drive Interconnection Listing
**PREPARATION FOR USE**

Fujitsu/Memorex/14" Shugart Drive iSBC² 215G Board Mating Connector 56-Pin 1

Fujitsu/Memorex/14" Shugart Drive iSBC² 215G Board Mating Connector 56-Pin 2

Fujitsu/Memorex/14" Shugart Drive iSBC² 215G Board Drive 6 26-Pin 4

Fujitsu/Memorex/14" Shugart Drive iSBC² 215G Board Connector J2 46-Pin 3

---

1  1
2  2
3  10
4  3
5  12
6  4
7  14
8  9
9  11
10 8
11 7
12 6
13 5
14 4
15 3
16 2
17 1

---

`*iSBC² 215G Board (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 2-2. Fujitsu 2300/Memorex/14-Inch Shugart Drive Interconnection Listing**

2-24
Figure 2-3. Priam Drive Interconnection Listing
### 5 1/4" Drive

**Mating Connector**
- 34-Pin

**ISBC' 215G Board**
- Mating Connector J1

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>- Head Select 2° (- HS2)</td>
</tr>
<tr>
<td>2</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>3</td>
<td>Write Gate (WRGATE)</td>
</tr>
<tr>
<td>4</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>5</td>
<td>- SEEK COMPLETE (SKCOM)</td>
</tr>
<tr>
<td>6</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>7</td>
<td>Track 000 (Busy)</td>
</tr>
<tr>
<td>8</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>9</td>
<td>Write Fault (FAULT)</td>
</tr>
<tr>
<td>10</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>11</td>
<td>- HEAD SELECT 2° (- HS1)</td>
</tr>
<tr>
<td>12</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>13</td>
<td>- INDEX (INDEX)</td>
</tr>
<tr>
<td>14</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>15</td>
<td>- READY (RDY)</td>
</tr>
<tr>
<td>16</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>17</td>
<td>Step (STEP)</td>
</tr>
<tr>
<td>18</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>19</td>
<td>- DRIVE SELECT 1 (US0)</td>
</tr>
<tr>
<td>20</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>21</td>
<td>- DRIVE SELECT 2 (US1)</td>
</tr>
<tr>
<td>22</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>23</td>
<td>- DRIVE SELECT 3 (US2)</td>
</tr>
<tr>
<td>24</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>25</td>
<td>- DRIVE SELECT 4 (US3)</td>
</tr>
<tr>
<td>26</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>27</td>
<td>- DIRECTION IN (DIR)</td>
</tr>
</tbody>
</table>

**Data Separator**
- Mating Connector J5

**Drive 6**
- Mating Connector

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>- DRIVE SELECTED</td>
</tr>
<tr>
<td>2</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>3</td>
<td>SPARE</td>
</tr>
<tr>
<td>4</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>5</td>
<td>SPARE</td>
</tr>
<tr>
<td>6</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>7</td>
<td>SPARE</td>
</tr>
<tr>
<td>8</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>9</td>
<td>+ TIMING CLK</td>
</tr>
<tr>
<td>10</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>11</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>12</td>
<td>+ MFM Write Clock</td>
</tr>
<tr>
<td>13</td>
<td>- MFM Write Clock</td>
</tr>
<tr>
<td>14</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>15</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>16</td>
<td>+ MFM READ DATA</td>
</tr>
<tr>
<td>17</td>
<td>- MFM READ DATA</td>
</tr>
<tr>
<td>18</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>19</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>20</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>21</td>
<td>Data Separator</td>
</tr>
<tr>
<td>22</td>
<td>ISBC' 215G Board</td>
</tr>
</tbody>
</table>

### Data Separator
- ISBC' 215G Board Connectors J1

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>- READ GATE (RDGATE)</td>
</tr>
<tr>
<td>2</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>3</td>
<td>AMF (SECTOR)</td>
</tr>
<tr>
<td>4</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>5</td>
<td>WRAM (ADMKEN)</td>
</tr>
<tr>
<td>6</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>7</td>
<td>RWC (RDWRCUR)</td>
</tr>
<tr>
<td>8</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>9</td>
<td>+ NRZ WRITE DATA (WR+)</td>
</tr>
<tr>
<td>10</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>11</td>
<td>+ WRITE CLOCK (WRCLK+)</td>
</tr>
<tr>
<td>12</td>
<td>- WRITE CLOCK (WRCLK-)</td>
</tr>
<tr>
<td>13</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>14</td>
<td>+ READ CLOCK (RDCLK+)</td>
</tr>
<tr>
<td>15</td>
<td>+ READ CLOCK (RDCLK+)</td>
</tr>
<tr>
<td>16</td>
<td>Ground (GND)</td>
</tr>
<tr>
<td>17</td>
<td>+ NRZ READ DATA (RD+)</td>
</tr>
<tr>
<td>18</td>
<td>- NRZ READ DATA (RD-)</td>
</tr>
<tr>
<td>19</td>
<td>Ground (GND)</td>
</tr>
</tbody>
</table>

**NOTE:** This interconnection listing is for information only. Interface to an ST412 compatible 5 1/4" hard-disk drive requires one of the ISBC' 215G kits, which provides, in addition to the ISBC' 215G board, the ISBC' 213 data separator board and all required cables.

---

**Figure 2-4. 5 1/4-Inch Drive Interconnection Listing**
PREPARATION FOR USE

Drive Mating Connector  
isBC® 215G Board 
Mating Connector J1

5pin  1  5pin  2

1. Ground (GND)  36. Step (STEP)
2. Read Enable (RDGATE)
3. Fault Reset (Select Out)
4. Head Select 2' (HS1)
5. Ground (GND)
6. Head Select 2" (HS2)
7. Ground (GND)
8. INDEX (INDEX)
9. Drive Ready (RDY)
10. Ground (GND)
11. Drive Block
12. Drive Select 1 (US1)
13. Drive Select 2 (US2)
14. Ground (GND)
15. Write Data + (WRD+)
16. Write Data - (WRD-)
17. Ground (GND)
18. Write Clock + (WRCLK+)
19. Write Clock - (WRCLK-)
20. Servo Clock + (RDCLK+)
21. Servo Clock - (RDCLK-)
22. Ground (GND)
23. Read Data + (RD+)
24. Read Data - (RD-)
25. Ground (GND)
26. Write Gate (WRGATE)
27. Ground (GND)
28. TRACK 0 (TRACK 0)
29. Fault Reset (Select Out)
30. Head Select 2' (HS1)
31. Fault Reset (Select Out)
32. Head Select 2" (HS2)
33. INDEX (INDEX)
34. Head Select 2" (HS2)
35. Head Select 2' (HS1)

NOTE: See Figure 2-8 for cabling diagram.

*Refer to drive manual for application details.

Figure 2-5. CDC Drive Interconnection Listing
Figure 2-6. ANSI Drive Interconnection Listing
Figure 2-7. 8-Inch Shugart/Quantum Drive Interconnecting Cable
Figure 2-8. Fujitsu 230/Memorex/14-Inch Shugart Drive Interconnecting Cable
Figure 2-9. Priam Drive Interconnecting Cable
Figure 2-1Ø. 5 1/4-Inch Drive Interconnecting Cable
PREPARATION FOR USE

Figure 2-11. CDC Drive Interconnecting Cable
Figure 2-12. ANSI Drive Interconnecting Cable
2.6 BOARD INSTALLATION

The iSBC 215G board can be installed in any MULTIBUS compatible cardcage or chassis. Installation is accomplished as outlined in the following procedure:

1. Remove power from cardcage or chassis.

2. Connect all jumpers for desired configuration (refer to paragraph 2.4).

3. Install iSBX MULTIMODULE board or boards as outlined in appropriate procedure (paragraph 2.6.2 for iSBX 218A board; paragraph 2.6.3 for iSBX 217B/C board).

4. Slide board into desired slot and press firmly to make certain that both connectors are properly seated.

5. Apply power to cardcage or chassis.

2.6.1 DRIVE INSTALLATION

The requirements for connecting the iSBC 215G board to the disk drive vary among drive types. Drives that conform to the ANSI X3T9/1226 interface can simply be plugged into iSBC 215G board center connector J5 (see Figure 2-13). Drives that do not conform to the ANSI interface require different interconnection schemes. Connectors J1 (left-hand connector) and J2 (right-hand connector) are used for non-ANSI interfacing. The iSBC 215G board is pin-compatible with earlier board versions. Connectors J1 and J2 pin functions are consistent with those versions, but pin numbering has been altered (see Figure 2-14). (Earlier versions of the board do not support connector J5.) The following paragraphs describe several non-ANSI interconnection schemes.

---

**Figure 2-13. ANSI Drive Interface**

---
2.6.1.1 14/8-Inch Drives With Integral Data Separator

Connecting the iSBC 215G board to drives with integral data separators requires only constructing and attaching the interconnecting cables. Refer to Figure 2-15. It may be helpful also to refer to Figures 2-1 through 2-12.
2.6.1.2 5 1/4-Inch Drives

All 5 1/4-inch drives used with the iSBC 215G board must be ST506/412 interface compatible (the iSBC 215G board does not provide step-pulse buffering). These drives require the use of the iSBC 215G Kit, which consists of an iSBC 215G board, an iSBC 213 data separator, and a cable set. Refer to Figure 2-16.

![Diagram of interface for drives and iSBC® 215G Kit](image)

Figure 2-16. Interface for Drives and iSBC® 215G Kit

2.6.1.3 Other Drives Without Data Separator

Drives without data separators, other than ST506/412 compatible types, require the use of a Shugart, RMS, or equivalent data separator, and require constructing and attaching the interconnecting cables. Refer to Figure 2-17. It may be helpful also to refer to Figures 2-1 through 2-12.
2.6.2 iSBX™ 218A BOARD INSTALLATION

The iSBX 218A board connects to J4 on the iSBC 215G board. Before installing the iSBX 218A board, install iSBC 215G board jumper W4-1 -- 2. Also, install jumper W24 as appropriate for DMA conditions (see Table 2-4). A single cable that transmits both control and read/write information is required to connect the iSBX 218A board to the flexible-disk drives as shown in Figure 1-1. Refer to the iSBX 218A Flexible Diskette Controller Board Hardware Reference Manual for further installation details and operating information.

Install the board as follows:

1. Install supplied threaded spacers on solder side of iSBX 218A board (at holes near connectors J1 and J2 and at J2 end of board), securing spacers by inserting and hand-tightening supplied 1/4-inch screws from component side of iSBX 218A board (see Figure 2-18.)

2. Locate pin 1 on iSBC 215G board iSBX connector J4. Similarly, locate pin 1 on iSBX 218A board connector (see Figure 2-18.)
3. Carefully match connectors at pin 1, insert ISBX 218A board connector into ISBC 215G board ISBX connector, and press gently to seat fully. (ISBX board connector should be oriented in same direction as ISBC 215G board I/O connectors.)

4. Insert remaining 1/4-inch screws from solder side of ISBC 215G board into spacers, and tighten all screws.

2.6.3 ISBX™ 217B/C BOARD INSTALLATION

The ISBX 217B/C board connects to J3 on the ISBC 215G board. Before installing the ISBX 217B/C board, install ISBC 215G board jumper W3-1 -- 2. Also, install jumper W24 as appropriate for DMA conditions (see Table 2-4). A single cable that transmits both control and read/write information is required to connect the ISBX 217B/C board to the magnetic cartridge-tape drive or drives. Refer to the ISBX 217B/C Magnetic Cartridge Tape Interface Board Hardware Reference Manual for further installation details and operating instructions.

Install the board as follows:

1. Install supplied threaded spacer on solder side of ISBX 217B/C board (at hole near connector J1), securing spacer by inserting and hand-tightening one of supplied 1/4-inch screws from component side of ISBX 217B/C board (see Figure 2-19.)

2. Locate pin 1 on ISBC 215G board ISBX connector J3. Similarly, locate pin 1 on ISBX 217B/C board connector (see Figure 2-19.)

3. Carefully match connectors at pin 1, insert ISBX 217B/C board connector into ISBC 215G board ISBX connector, and press gently to seat fully. (ISBX board connector should be oriented in same direction as ISBC 215G board I/O connectors.)

4. Insert remaining 1/4-inch screw from solder side of ISBC 215G board into spacer, and tighten both screws.
Figure 2-18. iSBX™ 218A Board Installation
Figure 2-19. iSBX™ 217B/C Board Installation
3.1 INTRODUCTION

This chapter describes the programming conventions that must be followed to initiate and monitor the transfer of data between the system memory and a disk (hard or flexible) or tape drive. The firmware installed on the iSBC 215G board includes direct support for hard-disk (Winchester technology) interface, flexible-disk interface via the iSBX 218A Flexible Diskette Controller Board, and the QIC-02 cartridge-tape interface via the iSBX 217B/C Magnetic Cartridge Tape Interface Board.

NOTE

If power is applied to or removed from the iSBC 215G board while a drive is ready, a spurious disk-write operation could occur. To prevent this, make certain that the disks are stopped whenever iSBC 215G board power is switched on or off.

Included in this section are: 1) descriptions of disk organization, track sectoring format, disk controller communications protocol, interrupt handling, the use of disk control functions; and 2) special procedures for programming I/O transfers to flexible-disk and cartridge-tape drives through the iSBX interface.

3.2 PROGRAMMING OPTIONS

The iSBC 215G board is designed to interface with hard-disk (Winchester technology) drives as specified in Chapters 1 and 2. The board also has two iSBX connectors for communication with other I/O devices through an iSBX I/O controller such as the iSBX 218A Flexible Diskette Controller Board or the iSBX 217B/C Magnetic Cartridge Tape Interface Board.

The iSBC 215G board contains a ROM-resident I/O transfer program designed to control data transfers between the board and hard-disk drives as well as between the board and flexible-disk and cartridge-tape drives connected to the iSBX 218A and iSBX 217B/C controllers. In addition, the iSBC 215G board can also execute programs that the user has written in 8089 I/O Processor (IOP) assembler code to control other I/O devices through the iSBX bus on the board.
3.3 MASS STORAGE PRINCIPLES

Mass storage, for purposes of this manual, is defined as the capacity of a peripheral device or combination of devices for storage of large amounts of data in applications requiring low-cost, high-capacity, random-access storage, but not the very low access times of such integrated-circuit devices as RAM's. The iSBC 215G board can accommodate storage devices of three types: hard-disk (Winchester technology), flexible-disk, and cartridge-tape. While hard-disk devices offer greater capacity per device and somewhat faster access, flexible-disk and cartridge-tape devices offer virtually unlimited capacity through interchangeable storage media. Thus, with the iSBC 215G board, the user can augment the system memory in the most efficient fashion, using any required combination of storage media.

3.3.1 HARD-DISK ORGANIZATION

In this description, a head is assumed to be associated with a single disk surface. Each surface can have as many as 4096 tracks (circular data paths numbered 0 through 4095). The set of tracks on multiple recording surfaces (one track per surface) 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.

![Disk Drive Organization and Terminology](image)

Each track is divided into equal-sized sectors. Each of these sectors includes a sector identification block with error checking information. The iSBC 215G board allows the user to select the size of the data block, which then determines the maximum number of sectors permitted per track (as shown in Table 1-1). The iSBC 215G board 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 board organizes this information for 8-inch hard-disk drives. Refer to paragraph 3.5.3 for further information on track formatting.

![Figure 3-2. Sector Fields](image)

### 3.3.2 FLEXIBLE-DISK ORGANIZATION

The organization of data on the flexible disk is much the same as that of the hard-disk. The primary difference is that flexible-disk drives use only one disk at a time, which provides one or two recording surfaces. Hard-disk drives, however, can contain multiple disks, thus providing multiple recording surfaces. Refer to the iSBX 218A Flexible Diskette Controller Board hardware reference manual for information on flexible-disk track formatting.

### 3.3.3 CARTRIDGE-TAPE ORGANIZATION

Data stored on tape differs from that on disk in that files are not limited in length as on disk. A file is simply a string of data that ends with an end-of-file character. Refer to the iSBX 217B/C Magnetic Cartridge Tape Interface Board hardware reference manual for information on cartridge-tape file formatting.

### 3.4 HOST/BOARD COMMUNICATIONS

The iSBX 215G board is a full DMA device that is capable of operating as a bus master for transferring information to and from system memory. However, it cannot operate as the system master (host), and depends upon
the system master to provide operation programming. The board responds to any host CPU that provides the necessary operation programming. Thus, the iSBC 215G board can be used in multiprocessor systems if the necessary software interlocks are used to assure that multiple, concurrent mass storage accesses are not attempted. All mass storage operations are initiated by the output from the host CPU of a command byte to the wake-up port assigned to the iSBC 215G board. Once the operation is initiated by the host CPU, all subsequent communication between the host CPU and the board, until the operation is complete, take place using the I/O communications blocks established in memory by the host CPU prior to initiating the mass-storage operation.

The I/O communication blocks structure for the board, exclusive of any data buffer, consists of 68 bytes of memory that are arranged into 4 blocks. The format of each of the 4 blocks is specifically defined. However, the blocks can be arranged in any order or in any location within a 1-Mbyte page of memory (dedicated memory locations excluded). Each of the blocks has a defined format and the memory bytes that make up each must be contiguous. Each of the blocks also has a defined function related to the overall operation of the iSBC 215G board.

The most important advantage of such a communications block structure is flexibility. Though some of the blocks should be limited in use to only one such block in memory, the system may contain multiple copies of blocks used directly to specify operations. Thus, by merely changing a few pointers, software can specify a different storage operation without structuring an all-new I/O block.

3.4.1 WAKE-UP I/O PORT

The wake-up I/O port is the I/O address to which the iSBC 215G board responds. This single I/O address is user selectable through jumpers on the board and may be either 8 or 16 bits, as applicable to the host CPU and the application. The command signal that controls the number of bits in the address to which the board will respond is also jumper selectable.

NOTE

The jumpers that select the I/O port address (shifted to the left four places) also select the first address in the wake-up block. Thus wake-up I/O port address 100H also specifies wake-up block address 100H.

To invoke iSBC 215G board activity, the host CPU transmits a wake-up command byte to the board through the wake-up I/O port. Three wake-up commands are allowed, as shown in Table 3-1. Note that only the two least significant bits of the command are used to determine which of the three hardware functions to implement. Note that only MULTIBUS I/O write operations are recognized.
PROGRAMMING INFORMATION

Refer to paragraph 2.4.2 for detailed information on the 8- and 16-bit I/O port addresses. Also, refer to paragraph 2.4.1 for detailed information on wake-up addresses.

Table 3-1. I/O Control Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>Clear Interrupt -- iSBC 215G board to host CPU interrupt is reset; board reset is cleared.</td>
</tr>
<tr>
<td>01H</td>
<td>Start Operation -- Instructs board to start operation defined by I/O parameter block elements.</td>
</tr>
<tr>
<td>02H</td>
<td>Reset Board -- Performs hardware reset of board. Clear interrupt (00H) must be initiated following this command. (Each time board is reset, communications link between board and host CPU must be re-established through initializing.)</td>
</tr>
<tr>
<td>03H through FFH</td>
<td>Reserved.</td>
</tr>
</tbody>
</table>

3.4.2 I/O COMMUNICATIONS BLOCKS

The host CPU and the iSBC 215G board use the four blocks of system memory and one MULTIBUS I/O port to exchange instructions and status. The I/O communications blocks are entitled: wake-up block, channel control block, controller invocation block, and I/O parameter block. The iSBC 215G board uses these blocks to perform three basic functions: initialize the board, check and transmit status, and obtain user-selected drive access functions and parameters. In addition to these I/O blocks, certain board functions (such as track formatting) also require data/parameter buffers in system memory. Dedicated locations, however, are not required.

NOTE

Following the iSBC 215G board initialization, the wake-up block, channel control block, and controller invocation block must remain at the assigned locations. The location of the I/O parameter block can be changed only if the I/O parameter block pointer in the controller invocation block is changed to indicate the new location.
One I/O port in the host CPU addressable (MULTIBUS) I/O space is also required. The host CPU uses this port, called the wake-up I/O port, to initiate iSBC 215G board activity. The sequence (see Figure 3-3) in which the board 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 control and data blocks, as required, in system memory with the command and parameters for the function the iSBC 215G board is to perform (for example, read-data).

2. The host then transmits a wake-up command (01H) to the wake-up I/O port, signalling the board to read the I/O communications blocks for instructions.

3. The board reads the wake-up block and links its way to the channel control block, through the controller invocation block, to the I/O parameter block. (The wake-up block is used once during board initialization and by host iSBC 215G board IOP firmware. All subsequent wake-up commands cause the iSBC board to read the channel control block.)

4. At the I/O parameter block, the iSBC 215G board reads the command and parameter data into local RAM and begins the data transfer function.

5. The board reads data from the selected drive into local RAM, then DMA-transfers the data from RAM into system memory.

6. When the data transfer is complete, the board posts the status in the controller invocation block, sends an interrupt to the host CPU, and awaits further instructions.

---

**Figure 3-3. Host CPU/Board Interaction**
These I/O communications blocks are accessed in a similar manner when performing a write function.

The host CPU initiates board activity through the wake-up I/O port, which it addresses through the MULTIBUS interface. The 8089 I/O Processor (IOP) handles all communications between the host CPU, host memory, and disk drives, once the host has initiated board activity. Board operations software is contained in on-board PROM. Local RAM on the board facilitates intermediate data storage between the host CPU and the disk drive. The iSBX bus provides a second I/O transfer path between the board and an I/O controller such as the iSBX 218A Flexible Diskette Controller Board.

Note that, in the following command block descriptions, all bytes shown as reserved in the illustrations should be set to 0 unless specified otherwise. Also note that some of the unused bytes are intended for future expansion or are required for compatibility with other devices that use a similar command structure.

3.4.2.1 Wake-Up Block

The wake-up block (see Figure 3-4 and Table 3-2) is used to establish a link between the board and the I/O communications blocks in host system memory. It is the first of the I/O communications blocks and requires 6 bytes of memory. It is used once at board initialization by the host CPU and by iSBC 215G board IOP firmware.

Note that the first address in the wake-up block in system memory is the wake-up address and that it is selected by the same set of jumpers that select the I/O port address. The hexadecimal value contained in the configuration of these jumpers is multiplied by 2^4 (that is, shifted four places to the left) to derive the 20-bit MULTIBUS address. (Thus, the jumper selected wake-up address is used as the segment value, with an offset value of 0, to derive the actual wake-up address.) Upon recognition of the first I/O start command to the wake-up I/O port, the board starts reading the wake-up block at this address. It fetches the wake-up block and internally saves the channel control block address (the next link in the communications blocks chain). This operation is necessary only after a board reset.

Figure 3-4. Wake-Up Block
Table 3-2. Wake-Up Block Byte Contents

<table>
<thead>
<tr>
<th>Byte</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>SYSTEM OPERATION COMMAND -- Must be set to Ø1H.</td>
</tr>
<tr>
<td>1</td>
<td>Reserved.</td>
</tr>
<tr>
<td>2 through 5</td>
<td>CHANNEL CONTROL BLOCK ADDRESS -- Address (segment X 2 + offset) of first byte in channel control block.</td>
</tr>
</tbody>
</table>

3.4.2.2 Channel Control Block

The channel control block (see Figure 3-5 and Table 3-3) indicates the IOP status IOP and invokes program operations. It requires 16 bytes of memory (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 board operations that are transparent to the host CPU. The busy-1 flag is posted (by the iSBC 215G board, except during cold-start initialization) when the board is busy processing a command, and cleared (also by the iSBC 215G board) after the processing is completed (when the IOP halts). It is used in handshaking and status commands between the iSBC 215G board and the host CPU.

The channel control and controller invocation block addresses are stored in the iSBC 215G board logic while processing the first start I/O command to the wake-up I/O port after a reset operation. These addresses may not be changed without also commanding a board reset and initialization.

Figure 3-5. Channel Control Block
Table 3-3. Channel Control Block Byte Contents

<table>
<thead>
<tr>
<th>Byte</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>CHANNEL CONTROL WORD 1 -- Indicates location of IOP control store program. Ø1H -- On-board ROM. Ø3H -- Host system memory (used only when executing user-written I/O program from host memory).</td>
</tr>
<tr>
<td>1</td>
<td>BUSY 1 FLAG -- Indicates board state (busy or idle). Ø0H -- Idle FFH -- Busy</td>
</tr>
<tr>
<td>2 through 5</td>
<td>INVOCATION BLOCK ADDRESS -- Address of fifth byte of controller invocation block.</td>
</tr>
<tr>
<td>6 and 7</td>
<td>Reserved.</td>
</tr>
<tr>
<td>8</td>
<td>CHANNEL CONTROL WORD 2 -- Must contain Ø1H.</td>
</tr>
<tr>
<td>9</td>
<td>BUSY 2 WORD -- Not used by host CPU.</td>
</tr>
<tr>
<td>Ø through 13</td>
<td>CONTROL POINTER ADDRESS -- Must point to control pointer address in bytes 14 and 15.</td>
</tr>
<tr>
<td>14 and 15</td>
<td>CONTROL POINTER -- Must be set to Ø0Ø4H.</td>
</tr>
</tbody>
</table>

3.4.2.3 Controller Invocation Block

The controller invocation block (see Figure 3-6 and Table 3-4) posts status to the host CPU and locates the starting address for the on-board drive interface program. The status semaphore byte (byte 3) has a special purpose. The host CPU uses this byte to indicate to the board 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 of memory.
### Table 3-4. Controller Invocation Block Byte Contents

<table>
<thead>
<tr>
<th>Byte</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved.</td>
</tr>
<tr>
<td>1</td>
<td>BOARD OPERATION STATUS -- Indicates board status (see paragraph 3.7).</td>
</tr>
<tr>
<td>2</td>
<td>COMMAND SEMAPHORE -- Board does not use byte. (Byte functions as multiprocessor interlock when required.)</td>
</tr>
<tr>
<td>3</td>
<td>STATUS SEMAPHORE -- Indicates state of status posting. Board posts status only when byte contains FFH. When new status has been posted, board sets byte to FFH. When host CPU has read status, it sets byte to 00H.</td>
</tr>
<tr>
<td>4 through 7</td>
<td>CONTROL STORE PROGRAM ADDRESS -- Starting address of on-board mass storage interface program. Set to 00000H.</td>
</tr>
<tr>
<td>8 through 11</td>
<td>I/O PARAMETER BLOCK ADDRESS -- Address of first byte of parameter block.</td>
</tr>
<tr>
<td>12 through 15</td>
<td>Reserved.</td>
</tr>
</tbody>
</table>

*Set to all zeros*
3.4.2.4 I/O PARAMETER BLOCK

The I/O parameter block (see Figure 3-7 and Table 3-5) functions as the primary communications channel between the host CPU and the iSBC 215G board. It contains the board operating commands, which define the function the board is to perform (read, write, etc.), and the parameters of the function (memory address, disk head, cylinder, etc). The I/O parameter block requires 30 bytes of memory; however, the first four bytes are reserved for expansion and must always be set to all 0's.

\*Set to all zeros.

**Figure 3-7. I/O Parameter Block**
### Table 3-5. I/O Parameter Block Byte Contents

<table>
<thead>
<tr>
<th>Byte</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø through 3</td>
<td>Reserved.</td>
</tr>
<tr>
<td>4 through 7</td>
<td><strong>ACTUAL TRANSFER COUNT</strong> -- Four-byte binary number, least significant bits in first byte. Indicates count of bytes actually transferred between host CPU and iSBC 215G board.</td>
</tr>
<tr>
<td>8 and 9</td>
<td><strong>DEVICE CODE</strong> -- Code for type of device being accessed.</td>
</tr>
<tr>
<td></td>
<td>- 0000H -- hard-disk.</td>
</tr>
<tr>
<td></td>
<td>- 0001H -- 8-in flexible-disk drive.</td>
</tr>
<tr>
<td></td>
<td>- 0003H -- 5 1/4-in flexible-disk drive.</td>
</tr>
<tr>
<td></td>
<td>- 0004H -- QIC-Ø2 cartridge-tape drive.</td>
</tr>
<tr>
<td>10</td>
<td><strong>UNIT</strong> -- Indicates binary number for drive being accessed (bits Ø and 1 provide four numbers, bits 2 through 7 are reserved).</td>
</tr>
<tr>
<td>11</td>
<td><strong>FUNCTION</strong> -- Indicates code for operation to be performed (see paragraph 3.5)</td>
</tr>
<tr>
<td>12 and 13</td>
<td><strong>MODIFIER</strong> -- Two-byte binary word. Indicates code to modify function codes (see paragraph 3.6).</td>
</tr>
<tr>
<td>14 and 15</td>
<td><strong>CYLINDER</strong> -- Two-byte binary number, bit Ø is least significant. Indicates logical cylinder code.</td>
</tr>
<tr>
<td>16</td>
<td><strong>HEAD</strong> -- One-byte binary number, bit Ø is least significant. Indicates logical head code.</td>
</tr>
<tr>
<td>17</td>
<td><strong>SECTOR</strong> -- One-byte binary number, bit Ø is least significant. Indicates logical sector code.</td>
</tr>
<tr>
<td>18 through 21</td>
<td><strong>DATA BUFFER ADDRESS</strong> -- Indicates address of first byte in host memory data (parameter) buffer. (For 24-bit addressing information, refer to paragraph 3.6.)</td>
</tr>
<tr>
<td>22 through 25</td>
<td><strong>REQUESTED TRANSFER COUNT</strong> -- Four-byte binary number, least significant bits in first byte. Indicates count of bytes requested to be transferred between host CPU and iSBC 215G board.</td>
</tr>
<tr>
<td>26 through 29</td>
<td><strong>GENERAL ADDRESS POINTER</strong> -- Indicates general-purpose address pointer. When iSBC 215G board is configured for ANSI X3T9/1226 interface, bytes have following values, where n is calculated as in Appendix A:</td>
</tr>
<tr>
<td></td>
<td>26 -- nØ</td>
</tr>
<tr>
<td></td>
<td>27 -- n₁</td>
</tr>
<tr>
<td></td>
<td>28 -- n₂</td>
</tr>
<tr>
<td></td>
<td>29 -- n₃</td>
</tr>
</tbody>
</table>
As shown in Table 3-5, the board writes the actual count of bytes transferred into bytes 4 through 7 of the I/O parameter block. This is done following either termination or completion of an operation. If the count does not match the requested transfer, this indicates that the operation was terminated prematurely and that a status check is in order. When the board is to perform the track formatting operation, the host CPU writes a count of 6 into the actual-transfer-count word. When the board is to perform a status transfer, a count of 12 is written. When initializing hard-disk drive Ø, this word is used to display the board firmware and revision numbers. Bits 7 and 6 contain the version number minus 1; bits 5, 4, 3, and 2 contain the revision number.

3.4.3 COLD-START BOARD INITIALIZATION

The iSBC 215G board cold-start initialization must be performed to establish the link between the IOP and the I/O communications blocks in host system memory at any time that power has been removed from and restored to the board (before any data transfer activities between the host system memory and the drives can be initiated). After the board is initialized, any of the data transfer functions can be performed in any sequence. (Refer to paragraph 4.4.1.3.2 for a detailed explanation of board initialization.)

The following procedure outlines the sequence in which the board initializing activities are performed. Prior to initializing the board, make certain that the system data bus jumpers, the host system I/O address jumpers, the wake-up address jumpers, and the interrupt level jumper have been set as described in the jumper configuration procedures in Chapter 2.

To initialize the board, the host CPU performs the following steps:

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

<table>
<thead>
<tr>
<th>Block</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wake-Up Block</td>
<td>6</td>
</tr>
<tr>
<td>Channel Control Block</td>
<td>6</td>
</tr>
<tr>
<td>Invocation Block</td>
<td>6</td>
</tr>
<tr>
<td>I/O Parameter Block</td>
<td>3Ø</td>
</tr>
</tbody>
</table>

NOTE

The address of the wake-up block first byte in host memory must be equal to the wake-up address set in the board wake-up address jumpers times $2^4$. For example, if the jumpers are set to $0673H$, the address of byte Ø of the wake-up block is $06730H$ for 2Ø-bit addressing and $6730H$ for 16-bit addressing.
2. Sets up the bytes in the wake-up block (see Figure 3-4 and Table 3-2).

3. Sets the BUSY 1 FLAG (optional, byte 1 of the channel control block) to other than Ø (FFH). Because the iSBC 215G board resets the BUSY 1 FLAG to Ø at the completion of the cold-start operation (Ø1H), the host CPU can monitor the flag to determine when the initialization procedure is completed.

4. Writes Ø2H to the wake-up I/O port to reset the iSBC 215G board.

5. Writes Ø0H to the wake-up I/O port to clear the reset.

6. Writes Ø1H to the wake-up I/O port to establish the host-CPU-to-iSBC-215G-board communications link. The board reads the wake-up block in host memory and records the address of the channel control block in local RAM, then proceeds to the channel control block and clears the BUSY 1 FLAG. On all subsequent Ø1H commands to the wake-up I/O port, the board reads the channel control block.

3.5 FUNCTION COMMANDS

The function commands included in the iSBC 215G board firmware take full advantage of the capabilities of the board and its attached peripheral devices. These commands provide for a full set of operations for the hard-disk drives attached directly to the board, and also include a set that is used specifically for flexible-disk and cartridge-tape drives attached via iSBX 218A and iSBX 217B/C boards. Modified definitions for some of the hard-disk commands, combined with the additional commands for the iSBX boards, allow direct use of these boards while using the same general programming used with the iSBC 215G board. Each of the function commands is invoked by setting up the command blocks as required for the command and then issuing a start operation (Ø1H) command to the wake-up port address of the iSBC 215G board.

Table 3-6 lists all of the function commands and includes device applicability information. Each of the commands is described in detail in the following paragraphs. With the exception of the spin-down command (ØBH), all of the disk commands are similar to the commands for the iSBC 215A/B board and earlier versions. Some of the functions were enhanced at various times; however, all such enhancements default to compatibility with earlier board versions.

The functions available on the iSBC 215G board are divided into two general types: short-term functions and long-term functions. Short-term functions are those that are performed with the specified device directly on line with the iSBC 215G board. These functions terminate with the board sending a single interrupt to the host CPU (if the interrupt was not suppressed). The long-term functions are those that are initiated by the iSBC 215G board and completed off-line by the selected device. The on-line portion of the long-term function terminates with the board sending an interrupt to the host CPU (if the interrupt was not suppressed). When the selected device completes the off-line portion of
the function, the iSBC 215G board sends a second interrupt to the host CPU. (The second interrupt cannot be suppressed.)

The following description of each of the commands includes a diagram of the I/O parameter block with the mandatory fields (other than reserved fields) shown for each function. In these descriptions, all of the long-term functions are so noted (short-term functions are not noted).

Table 3-6. Function Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Hexadecimal Value</th>
<th>Hard Disk</th>
<th>Flexible Disk</th>
<th>Cartridge Tape</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize</td>
<td>00</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Transfer Status Buffer</td>
<td>01</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Format</td>
<td>02</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Read Sector ID</td>
<td>03</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Read Data</td>
<td>04</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Read Data to Buffer and Verify</td>
<td>05</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Write Data</td>
<td>06</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Write Buffer Data</td>
<td>07</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Spin Down</td>
<td>08</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>iSBX Execute</td>
<td>0C</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>iSBX Transfer</td>
<td>0D</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Buffer I/O</td>
<td>0E</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Diagnostic</td>
<td>0F</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Tape Initialize</td>
<td>10</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Rewind</td>
<td>11</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Space Forward One File Mark</td>
<td>12</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Write File Mark</td>
<td>14</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Erase Tape</td>
<td>17</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Load Tape</td>
<td>18</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Tape Reset</td>
<td>1C</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Retension Tape</td>
<td>1D</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Read Tape Status</td>
<td>1E</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Read/Write Terminate</td>
<td>1F</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

3.5.1 INITIALIZE (00H)

The initialize function transfers device-related parameters to the iSBC 215G board for subsequent use during execution of other functions. To perform this function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code
- Unit Number
- Function Code
- Modifier
- Data Buffer Pointer

Bytes 8 and 9
Byte 10
Byte 11
Bytes 12 and 13
Bytes 18 through 21
The device parameters are specified in the data buffer area and fetched automatically by the ISBC 215G board (via the data buffer pointer) during function execution. Figure 3-8 illustrates the data buffer formats. The device parameters for tape are transferred in a single byte and require a single-byte data buffer for this purpose.

**HARD DISK**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NUMBER OF CYLINDERS</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>NOT USED</td>
<td>NUMBER OF HEADS</td>
</tr>
<tr>
<td>5</td>
<td>BPS (LOW)</td>
<td>SECTORS/TRACK</td>
</tr>
<tr>
<td>7</td>
<td>NO. ALT. CYLS.</td>
<td>BPS (HIGH)</td>
</tr>
</tbody>
</table>

**TAPE**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NOT USED</td>
<td>DEVICE AVAIL</td>
</tr>
<tr>
<td>3</td>
<td>NOT USED</td>
<td>NOT USED</td>
</tr>
<tr>
<td>5</td>
<td>NOT USED</td>
<td>NOT USED</td>
</tr>
<tr>
<td>7</td>
<td>NOT USED</td>
<td>NOT USED</td>
</tr>
</tbody>
</table>

**FLEXIBLE DISK**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NUMBER OF CYLINDERS</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>NUMBER OF HEADS</td>
<td>NOT USED</td>
</tr>
<tr>
<td>5</td>
<td>BPS (LOW)</td>
<td>SECTORS/TRACK</td>
</tr>
<tr>
<td>7</td>
<td>DRIVE VARIABLES</td>
<td>BPS (HIGH)</td>
</tr>
</tbody>
</table>

Figure 3-8. Initialize Function Data Buffer Format

Execution of the initialize function sets the ISBC 215G board logic for a mass storage device based on the device code and unit number specified in the I/O parameter block. Thus, to fully initialize the ISBC 215G board, the initialize function must be performed for each device attached directly to the board, to the board via the ISBX 218A board, and to the board via the ISBX 217B/C board. The ISBC 215G board firmware requires that the initialization procedures for all possible storage devices be performed, even when a particular device is not physically present; however, the ISBX 218A and 217B/C MULTIMODULE board initialization procedures need not be done if the boards are not installed.

The full initialization procedure must be performed following any interruption of power, system hardware reset, or invocation of the reset-board wake-up command. When the device specified is a disk drive (either hard or flexible), the heads are set to track 0. When the device specified is hard-disk drive 0, an on-board memory test is performed (unless inhibited by use of the function modifier) prior to execution of the disk drive initialization. When the on-board memory test is performed, the stored parameters for all of the storage devices are destroyed and the new parameters for hard-disk drive 0 are stored.

The following paragraphs describe the device parameters supplied to the ISBC 215G board during the initialization operation. Note that paragraphs 3.5.1.1 through 3.5.1.5 apply to hard-disk and flexible-disk drives, paragraph 3.5.1.6 applies to flexible-disk drives only, and paragraph 3.5.1.7 applies to cartridge-tape drives only.
3.5.1.1 Number of Cylinders

The number of cylinders is a two-byte hexadecimal value that specifies the total number of cylinders available on the disk drive. (Refer to the reference manual for the particular drive to determine the correct number for this parameter.) Setting the number of cylinders parameter to 0 removes the specified drive from use. A drive thus removed from service can be restored to service by performing the initialize function.

3.5.1.2 Number of Heads

The number of heads parameter is two one-byte hexadecimal values that specify the number of available recording surfaces. It is contained in byte 2 for hard-disk drives and in byte 3 for flexible disk drives. Byte 2 specifies the number of surfaces on the specified drive connected to the iSBC 215G board; byte 3 specifies the number of surfaces available on the specified drive connected to the board through the iSBX 218A board. In either instance, the unused byte should be set to all 0's.

3.5.1.3 Sectors Per Track

The sectors per track parameter is a one-byte hexadecimal value that specifies the number of sectors available on each track on the specified drive and is contained in byte 4. For hard-sectored hard-disk drives and all flexible disk drives, this value can be obtained from the reference manual for the particular drive. For soft-sectored hard-disk drives, the number of sectors per track must be calculated from information provided in the disk drive manual. Some typical values are given in Tables 1-2 and 1-3. For ANSI hard-disk drives other than those listed in Table 1-3, refer to Appendix A to calculate the values.

3.5.1.4 Bytes Per Sector

The bytes per sector parameter is a two-byte hexadecimal value that specifies the number of bytes in a disk sector and is contained in bytes 5 and 6, with byte 6 as the most significant. The parameter value must match the formatted sector size for the specified drive. If the drive is not formatted, the sector size specified during formatting must match this value.

3.5.1.5 Number of Alternate Cylinders

The number of alternate cylinders parameter is a one-byte hexadecimal value that specifies the number of cylinders that are reserved as alternates on the drive. The parameter value must match the number of cylinders formatted as alternates for the specified drive. If the drive is not formatted, the number formatted as alternates during formatting must match this value.
3.5.1.6 Drive Variables

The drive variables parameter in byte 7 is a combination of values that specifies the recording format, head step rate, and head load delay time for flexible-disk drives only. For 5 1/4-inch disk drives, the default values are 22 ms for the head step rate and 36 ms for the head load delay time. For 8-inch drives, the default values are 11 ms for the head step rate and 60 ms for the head load delay time. In either instance, the default values are selected when bits 1 through 7 of this parameter are set to all 0's. (These default values are the same as those for the iSBC 215A/B boards.) Figure 3-9 illustrates the format of the drive-variables parameter. Note that, if the default values are not selected by using all 0's, both the step rate and head load delay must be selected using the values listed in Figure 3-9.

Figure 3-9. Drive Variables Byte Format
3.5.1.7 Tape Parameters

The tape available parameter is a one-byte value that specifies whether the tape device is available for use. The least significant bit indicates the device availability (0 for device not available, 1 for device available). All other bits are reserved and should be set to 0.

3.5.2 TRANSFER STATUS BUFFER (01H)

The transfer status buffer function transfers the contents of the iSBC 215G board 12-byte status buffer into system memory starting at the location specified by the data buffer pointer. To perform this function, the host CPU establishes the following fields in the I/O parameter block:

- **Unit Number**
- **Function Code**
- **Modifier**
- **Data Buffer Pointer**

The host CPU can request the contents of either the short-term command status buffer or the long-term command status buffer (which is used only with some tape functions). When bit 6 of the modifier word is set to 0, the short-term status buffer contents are transferred; when bit 6 is set to 1, the long-term status buffer contents are transferred. When the short-term buffer is specified, its contents are not affected by the transfer status buffer function. However, when the long-term buffer is specified, its contents are written into the short-term buffer over the existing contents. (The status buffer format and definitions of the status conditions are included separately later in this chapter.)

3.5.3 FORMAT (02H)

The format function partitions the addressed track for subsequent data recording (see Figure 3-10). The partitioning is accomplished by writing sector headers and reserving recording space based on the initialization information for the specified disk drive. The sector headers contain information used in subsequent write or read operations to locate the correct sector data area. Each execution of the format function formats one track. To perform this function, the host CPU establishes the following fields in the I/O parameter block:

- **Device Code**
- **Unit Number**
- **Function Code**
- **Modifier**
- **Cylinder Number**
- **Head Number**
- **Sector One Offset**
- **Data Buffer Pointer**

Bytes 8 and 9

Byte 10

Byte 11

Bytes 12 and 13

Bytes 14 and 15

Byte 16

Byte 17

Bytes 18 through 21
Additional format parameters are specified in the data buffer area and fetched automatically by the iSBC 215G board during function execution. Figure 3-11 illustrates the data buffer content for the function.

**Figure 3-11. Format Function Data Buffer Format**
There are three options for formatting a track. Byte 0 in the data buffer specifies the type of format function required. Most tracks are formatted as standard data types. When a track is determined to have a medium defect, it is formatted as a defective track and provides a pointer to the alternate track used in its place. At format time, a few tracks on the disk are reserved as alternate data tracks. When an alternate track is used, it is formatted as an assigned alternate track. Note that all iSBX 218A board format functions also must specify the format type. With 5 1/4-inch disk drives using a 512-byte sector size, either 8 or 9 sectors per track may be specified with iSBC 215G boards identified as PBA number 144263-Ø14 or PBA number 146484-ØØ1.

When formatting data tracks and assigned alternate tracks, bytes 1 through 4 provide a 4-byte user-specified pattern. This pattern is repeatedly recorded into each sector data area during track formatting, and can be any 4-byte pattern. Typically, some form of worst-case pattern is used as a test of the medium integrity. When formatting a defective track, bytes 1 and 2 specify the cylinder number and byte 3 specifies the head number for the alternate track to which it is pointed. As with the data and alternate tracks, the content of bytes 1 through 4 are repeatedly recorded into each sector data area during track formatting as a pointer to the assigned alternate track.

Byte 5 in the data buffer specifies the interleave factor for the track. The interleave factor controls the order of the sectors on the track, and is the minimum number of sector intervals between the start of one sector and the start of the next sequential sector. For example, when an interleave factor of one is specified, the sector numbers are sequential. Greater interleave factors allow increased disk rotation time between sequential numbers. This allows the host CPU to prepare for the next data transfer before the next sequential sector arrives at the read/write head. Host CPU processing time is an important consideration in determining the ideal interleave factor for the iSBC 215G board. Performance tests with typical applications programs are suggested to determine the ideal factor.

3.5.4 READ SECTOR ID (03H)

The read-sector ID function searches for the first error-free sector header on the presently selected cylinder and head and, when the header is located, transfers the contents of the sector ID field into system memory, starting at the location specified by the data buffer pointer.

To perform this function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code
- Unit Number
- Function Code
- Modifier
- Data Buffer Pointer
- Bytes 8 and 9
- Byte 10
- Byte 11
- Bytes 12 and 13
- Bytes 18 through 21
Because the read-sector-ID function is typically used to verify disk position, an implicit seek is not performed. The information from the sector ID field is stored in the data buffer automatically by the iSBC 215G board during function execution. Figure 3-12 illustrates the data buffer and flag byte format.

![Data Buffer and Flag Byte Format](image)

**Figure 3-12. Read-Sector-ID Function Data Buffer and Flag Byte**

### 3.5.5 READ DATA (04H)

The read data function transfers a block of data from the specified device into system memory, starting at the location specified by the data buffer pointer. To perform the read data function, the host CPU establishes the following fields in the I/O parameter block:

- **Device Code**: Bytes 8 and 9
- **Unit Number**: Byte 10
- **Function Code**: Byte 11
- **Modifier**: Bytes 12 and 13
- **Cylinder Number**: Bytes 14 and 15
- **Head Number**: Byte 16
- **Starting Sector Number**: Byte 17
- **Data Buffer Pointer**: Bytes 18 through 21
- **Requested Transfer Count**: Bytes 22 through 25
3.5.5.1 Disk Read Details

When a disk-read function is initiated, the iSBC 215G board compares the presently selected cylinder number with the requested number. If the two are not the same, the board initiates a seek function to the requested cylinder (implicit seek). When the requested cylinder is reached and the requested head selected, the board starts scanning sector headers for the requested sector. When the requested sector is located, the contents of the sector data field are written into the board sector buffer and error checking and correction (if required) are performed. The contents of the sector buffer are then transferred into system memory, starting at the data buffer pointer location.

If the first sector transferred satisfies the requested transfer count, the read-data function is terminated and the status is posted. If the requested transfer count is not satisfied, the next logical sector is transferred in the same manner as the first. This process continues until the requested transfer count is satisfied or end-of-medium is detected. (End-of-medium is defined as the highest cylinder, head, and sector numbers possible for a given volume, as specified in the initialization parameters.)

Additional implicit seeks are performed until the requested transfer count is satisfied. If the requested transfer count does not match an even sector boundary, only the amount of data required to satisfy the requested transfer count is transferred from the last sector accessed.

3.5.5.2 Tape Read Details

The tape read-data function uses the same function code as the disk read operation; however, the cylinder, head, and sector parameters and error correction are not used.

The tape read function is also always a complete operation. A complete tape read operation can consist of one or more tape read functions and must be both opened and terminated. The first tape read function in a sequence opens the tape read operation. Once the operation is opened, disk functions (either hard or flexible) can be interleaved with subsequent tape read functions. (Note that the only tape function permitted until the tape read operation is terminated is a tape read function.) The individual tape read functions within a tape read operation are closed when the requested transfer count is satisfied. The requested transfer count can be any value from 1 byte to 16 Mbytes.

The tape read-data operation remains open until one of the following conditions is satisfied:

FILE MARK TERMINATION -- The tape read operation termination results when a file mark is encountered. In such a case, the operation status byte indicates a summary error and operation-completed status (89H). To determine if the summary error resulted from a normal file-mark termination or an error condition, it is necessary to transfer and examine the 12-byte short-term status buffer. For a file-mark-induced termination, the status buffer contents indicate a
length error (if the requested transfer count was not satisfied) and file mark detected.

**COMMAND TERMINATION** -- The host CPU can terminate a read-data operation by initiating the read/write-termination function. In terminating a normal read-data operation, the read/write-termination function initiates tape rewind to the beginning-of-tape marker. The operation status byte indicates a summary error and operation-completed status (89H), and the short-term status buffer indicates a buffer under-run/over-run error and beginning-of-tape marker detected.

**BLANK TAPE TERMINATION** -- If the read-data function is attempted on blank tape, the function is automatically terminated after a few inches of blank tape have passed the read head. The operation status byte indicates a hard error and a summary error. The short-term status buffer indicates a length error, soft data check, tape data check, and no data detected. The tape is not automatically rewound.

If the iSBC 215G board fails to maintain the data transfer rate required by the tape drive (typically 200 K bytes per second), an over-run occurs, the iSBC 215G board closes the read-data function, and the tape drive automatically stops and repositions the tape. The next read-data function resumes without loss of data. The status for the tape read-data function closed by the over-run indicates that repositioning was required with a buffer under-run/over-run error posted in the short-term status buffer.

### 3.5.6 READ TO BUFFER AND VERIFY (05H)

The read to buffer and verify function transfers a block of data from the specified disk drive, one sector at a time, into the iSBC 215G board RAM buffer and checks each sector read for an error correcting code (ECC). To perform the read to buffer and verify function, the host CPU establishes the following fields in the I/O parameter block.

<table>
<thead>
<tr>
<th>Field</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device code</td>
<td>8 and 9</td>
</tr>
<tr>
<td>Unit Number</td>
<td>10</td>
</tr>
<tr>
<td>Function Code</td>
<td>11</td>
</tr>
<tr>
<td>Modifier</td>
<td>12 and 13</td>
</tr>
<tr>
<td>Cylinder Number</td>
<td>14 and 15</td>
</tr>
<tr>
<td>Head Number</td>
<td>16</td>
</tr>
<tr>
<td>Starting Sector Number</td>
<td>17</td>
</tr>
<tr>
<td>Requested Transfer Count</td>
<td>22 through 25</td>
</tr>
</tbody>
</table>

The data transferred from the disk into the buffer memory can also be sent to a device attached to one of the iSBX connectors. To perform this operation, the read to buffer and verify function is followed by either the iSBX execute function or the write buffer data function.

By specifying one disk device for the read buffer data and verify function and a different disk device for the write buffer data function, device-to-device transfers can be accomplished without transferring the data into system memory. However, this must be done at the sector level.
of granularity. Also, the iSBX 217B/C board and an attached tape drive cannot be specified as one of the devices.

3.5.7 WRITE DATA (06H)

The write data function transfers a block of data from system memory, starting at the location specified by the data buffer pointer to the specified device. To perform the write-data function, the host CPU establishes the following fields in the I/O parameter block:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Code</td>
<td>8 and 9</td>
</tr>
<tr>
<td>Unit Number</td>
<td>10</td>
</tr>
<tr>
<td>Function Code</td>
<td>11</td>
</tr>
<tr>
<td>Modifier</td>
<td>12 and 13</td>
</tr>
<tr>
<td>Cylinder Number</td>
<td>14 and 15</td>
</tr>
<tr>
<td>Head Number</td>
<td>16</td>
</tr>
<tr>
<td>Starting Sector Number</td>
<td>17</td>
</tr>
<tr>
<td>Data Buffer Pointer</td>
<td>18 through 21</td>
</tr>
<tr>
<td>Requested Transfer Count</td>
<td>22 through 25</td>
</tr>
</tbody>
</table>

3.5.7.1 Disk Write Details

The disk write function is very similar to the disk-read function except for the direction of data movement. When a disk-write function is initiated, the iSBC 215G board compares the present cylinder number and the requested cylinder number. If the two are not the same, the board initiates a seek function to the requested cylinder (implicit seek). The board then fills the on-board sector buffer with data from system memory starting at the data buffer pointer location.

When the sector buffer contains enough data to fill one disk sector, the board prepares to write the data to the disk. It starts scanning sector headers in search of the requested sector. When the sector is located, the contents of the sector buffer are written into the sector data field. While the data are being written, a polynomial check number is calculated. It is written immediately following the last data byte.

If the first sector of data transferred satisfies the requested transfer count, the write-data function is terminated and status is posted. If the requested transfer count is not satisfied, the next logical sector is written in the same manner as the first. This process continues until the requested transfer count is satisfied or end-of-medium is detected. Additional implicit seeks are performed until the requested transfer count is satisfied. If the requested transfer count does not match an even sector boundary, only the amount of data required to satisfy the requested transfer count is transferred into the last sector accessed, and the remainder of the sector is filled with Ø's.
3.5.7.2 Tape Write Details

The tape write data function uses the same function code as the disk write operation; however, the cylinder, head, and sector parameters and error correction are not used.

The tape write function is also always a complete operation. A complete tape write operation can consist of one or more tape write functions and must be both opened and terminated. The first tape write function in a sequence opens the tape write operation. Once the operation is opened, disk functions (either hard or flexible) can be interleaved with subsequent tape write functions. (Note that the only tape function permitted until the tape write operation is terminated is a tape write function.) The individual tape write functions within a tape write operation are closed when the requested transfer count is satisfied. The requested transfer count can be any value from 1 byte to 16 Mbytes. However, the total transfer count must be divisible by 512.

The tape write-data operation remains open until the read/write-terminate function is executed to write the file mark and rewind the tape to the beginning-of-tape marker.

If the iSBC 215G board fails to maintain the data transfer rate required by the tape drive (typically 200 K bytes per second), an over-run occurs, the iSBC 215G board closes the tape write function, and the tape drive automatically stops and repositions the tape. The next write-data function resumes without loss of data. The status for the tape write-data function closed by the over-run indicates that tape repositioning was required with a buffer under-run/over-run error posted in the short-term status buffer.

3.5.8 WRITE BUFFER DATA (07H)

The write buffer data function writes the data present in the sector buffer to the specified disk drive. When the requested transfer count exceeds the sector size, the write buffer data function writes the same sector buffer contents into the data field of the next logical sector. To perform the write data buffer function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code: Bytes 8 and 9
- Unit Number: Byte 10
- Function Code: Byte 11
- Modifier: Bytes 12 and 13
- Cylinder Number: Bytes 14 and 15
- Head Number: Byte 16
- Starting Sector Number: Byte 17
- Requested Transfer Count: Bytes 22 through 25

The data transferred from the disk into the buffer memory can also be supplied to a device attached to one of the iSBX connectors. To perform this operation, the read buffer and verify function is followed by either the iSBX execute function or the write buffer data function. By specifying one disk device for the read to buffer and verify function and...
a different disk device for the write buffer data function, device-to-device transfers can be made without transferring the data into system memory. However, this must be done at the sector level of granularity. Also, the iSBC 217B/C board and an attached tape drive cannot be specified as one of the devices.

3.5.9 INITIATE TRACK SEEK (08H)

The initiate track seek function positions the read/write heads of the specified drive without initiating a data transfer. To perform the initiate track seek function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code
- Unit Number
- Function Code
- Modifier
- Cylinder Number
- Head Number
- Bytes 8 and 9
- Byte 10
- Byte 11
- Bytes 12 and 13
- Bytes 14 and 15
- Byte 16

Each data transfer function causes an implicit seek if the read/write heads are not located at the desired cylinder. However, if the implicit seek capability is used, any other device connected the iSBC 215G board is un-available to the host CPU until the selected operation is completed.

The iSBC 215G board accomplishes the initiate track seek function by directing the specified disk drive to perform an off-line seek. Once the off-line seek has been started, the board terminates the initiate track seek function, posts the operation-complete status, and sends an interrupt to the host CPU (if interrupts are not suppressed). When the disk drive completes the off-line seek, the iSBC 215G board posts the seek-complete status and sends a second interrupt to the host CPU. (The second interrupt cannot be suppressed.)

Note that, if it is necessary that the host CPU examine the 12-byte status buffer relative to the off-line seek operation, it must request the contents of the long-term status buffer. If the iSBC 215G board is busy with another device when an off-line seek operation is completed, the board completes the other function, posts the appropriate short-term status, and sends an interrupt to the host CPU (if interrupts are not suppressed). After the host CPU services the interrupt and resets the status semaphore, the iSBC 215G board posts the seek-complete status and again sends an interrupt to the host CPU, this time to indicate completion of the off-line seek operation.

It is possible for two or more disk drives to perform concurrent off-line seek operations; however, the iSBC 215G board is limited to one active short-term function while an off-line seek operation (or operations) is (are) in progress. If another function is attempted with a disk drive that is performing an off-line seek, the result is a seek-in-progress error. An initiate track seek function that specifies a destination cylinder with a number greater than the total number of cylinders available results in an automatic seek to track Ø, and an invalid-address error is posted in the short-term status buffer.
3.5.10 SPIN DOWN (0BH)

The spin down function directs the disk drive to prepare for an interruption of power. Typically, a disk drive that recognizes the spin down function moves the heads to a safe parking area on the disk surface. If the disk drive does not support the spin down function, the iSBC 215G board posts the invalid-command status in the short-term status buffer and terminates the function. To perform the spin down function, the host CPU establishes the following fields in the I/O parameter blocks:

| Unit Number | Function Code | Byte 10 | Byte 11 |

3.5.11 iSBX™ EXECUTE (0CH)

The iSBX execute function transfers iSBC 215G firmware control to a program stored in the on-board RAM. To perform the iSBX execute function, the host CPU establishes the following fields in the I/O parameter block:

| Function Code | Byte 11 |
| General Address Pointer | Bytes 26 through 29 |

The iSBX execute function allows execution of programs associated with iSBX MULTIMODULE boards other than the iSBX 218A and 217B/C boards (for which the iSBX 215G board provides programs in on-board firmware). The execution program for another iSBX MULTIMODULE board must be entered in 8089 assembler code and must be down-loaded to the board RAM using the buffer-I/O function prior to initiation of the iSBX execute function.

Execution of the down-loaded program begins at the memory address specified by the general address pointer. At completion of program execution, the program must exit to iSBC 215G board ROM address $00C5H. The remainder of the bytes in the I/O parameter block are not required by the iSBC 215G board for this function and can be used to pass parameters to the down-loaded program. The iSBX execute function is classed as a short-term operation, and includes a single interrupt at the completion of the function.

3.5.12 iSBX™ TRANSFER (0DH)

The iSBX transfer function transfers a block of data between the specified iSBX device and system memory. To perform the iSBX transfer function, the host CPU establishes the following fields in the I/O parameter block:

| Function Code | Byte 11 |
| Modifier | Bytes 12 and 13 |
| iSBX bus I/O Port Address | Bytes 14 and 15 (cylinder number) |
| Transfer Parameters | Byte 16 (head number) |
| Data Buffer Pointer | Bytes 18 through 21 |
| Requested Transfer Count | Bytes 22 through 25 |

3-28
The iSBX transfer function allows use of the less complex iSBX MULTIMODULE boards on the iSBC 215G board without additional programming in 8089 assembler code. This function re-defines the I/O parameter block slightly to accommodate passing the iSBX MULTIMODULE board parameters to the iSBC 215G board. The iSBX port address is specified in I/O parameter block bytes 14 and 15 and a set of transfer parameters are specified in byte 16. Table 3-7 lists the iSBX port addresses for the iSBC 215G board and Figure 3-13 illustrates the format of the transfer parameter byte. In common with the other data transfer functions, the data buffer pointer specifies the first address of the data buffer in system memory and the requested transfer count specifies the number of bytes to be transferred. The iSBX transfer function is classed as a short-term operation, and includes a single interrupt at the completion of the function.

Table 3-7. iSBX™ Bus I/O Port Addresses (Hexadecimal)

<table>
<thead>
<tr>
<th>Port</th>
<th>J3 Channel 0</th>
<th>J3 Channel 1</th>
<th>J4 Channel 0</th>
<th>J4 Channel 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>C070</td>
<td>C0B0</td>
<td>C0D0</td>
<td>C0E0</td>
</tr>
<tr>
<td>1</td>
<td>C072</td>
<td>C0B2</td>
<td>C0D2</td>
<td>C0E2</td>
</tr>
<tr>
<td>2</td>
<td>C074</td>
<td>C0B4</td>
<td>C0D4</td>
<td>C0E4</td>
</tr>
<tr>
<td>3</td>
<td>C076</td>
<td>C0B6</td>
<td>C0D6</td>
<td>C0E6</td>
</tr>
<tr>
<td>4</td>
<td>C078</td>
<td>C0B8</td>
<td>C0D8</td>
<td>C0E8</td>
</tr>
<tr>
<td>5</td>
<td>C07A</td>
<td>C0BA</td>
<td>C0DA</td>
<td>C0EA</td>
</tr>
<tr>
<td>6</td>
<td>C07C</td>
<td>C0BC</td>
<td>C0DC</td>
<td>C0EC</td>
</tr>
<tr>
<td>7</td>
<td>C07E</td>
<td>C0BE</td>
<td>C0DE</td>
<td>C0EE</td>
</tr>
</tbody>
</table>
3.5.13 BUFFER I/O (OEH)

The buffer I/O function transfers a block of data from the system memory into the iSBC 215G board RAM buffer. To perform the buffer I/O function, the host CPU establishes the following fields in the I/O parameter block:

- Unit Number
- Function Code
- Modifier
- Board Memory Pointer
- I/O Specifier (Input = 00H, Output = FFH)
- Data Buffer Pointer
- Requested Transfer Count

The buffer I/O function allows the host CPU to transfer data between the iSBC 215G board RAM and a system memory buffer. This function is used for diagnostic purposes, to fill the iSBC 215G board sector buffer for a subsequent write buffer data function, and to down-load an iSBX program for later execution. The function redefines the I/O parameter block slightly to accommodate passing the starting on-board RAM address and the transfer direction to the board. (The board memory address pointer is specified in I/O parameter block bytes 14 and 15 and must be in the range of 4000H to 45FFH; the transfer direction is specified in byte 16.) In common with other data transfer functions, the data buffer pointer specifies the first address of the data buffer in system memory and the requested transfer count specifies the number of bytes to be transferred. The buffer I/O function is classed as a short-term operation, and includes a single interrupt at the completion of the function.
3.5.14 DIAGNOSTIC (0FH)

The diagnostic function initiates a go/no-go self test, contained in the iSBC 215G firmware, that verifies the internal data and status logic in the disk drives. To perform the diagnostic function, the host CPU establishes the following fields in the I/O parameter block:

- **Device Code**
- **Unit Number**
- **Function Code**
- **Modifier**
  - Bytes 8 and 9
  - Byte 10
  - Byte 11
  - Bytes 12 and 13

The diagnostic function reserves the use of the highest cylinder (track) number on head 0 for execution of the program. This track cannot be used for data storage. When the function is initiated, the cylinder and head are selected automatically; the unit number is specified in the I/O parameter block. The following three additional modifiers (specified in modifier byte 13) specifically apply to the diagnostic function.

- **00H** -- Re-calibrates, then initiates a seek to the highest cylinder number of head 0. At seek completion, the iSBC 215G board performs a read-sector-ID function to verify track location before performing a write and read test on sector 0 using a 55AAH data pattern.

- **01H** -- Initiates a ROM check-sum test to verify the contents of the iSBC 215G board ROM.

- **02H** -- Initiates a seek to cylinder 0. At seek completion, the iSBC 215G board performs a read-sector-ID function to verify that the heads are located at cylinder 0.

3.5.15 TAPE INITIALIZATION (10H)

The tape initialization function is the second function in a four-step initialization process for the specified QIC-02 tape drive. The initialization process is started in the initialize function (00H) and is completed following the tape initialization function by performing the tape reset function and then the load tape function. To perform the tape initialization function, the host CPU establishes the following fields in the I/O parameter block:

- **Device Code**
- **Unit Number**
- **Function Code**
  - Bytes 8 and 9
  - Byte 10
  - Byte 11

Initializing the tape drives attached to the iSBC 215G board via the iSBX 217B/C board is a four-step process. In the first step, the host CPU performs the initialize function (00H) to initialize the iSBC 215G board. In this, the second step, the iSBC 215G board initializes the iSBX 217B/C board. In the third step, tape reset, the tape drive is initialized. Finally, in the fourth step, the load tape function is performed to position the tape for subsequent operations. The tape initialization function is classed as a short-term operation, and includes a single interrupt at the completion of the function.
3.5.16 REWIND (11H)

The rewind function returns the tape on the specified drive to the beginning of tape marker. To perform the rewind function, the host CPU establishes the following fields in the I/O parameter block:

- **Device Code**  Bytes 8 and 9
- **Unit Number**  Byte 10
- **Function Code**  Byte 11

Typically, the tape on a drive is rewound for one of two reasons: to return the tape to its starting point prior to removing the cartridge from the drive, or to position the tape to a known point before attempting a data transfer. The rewind operation is classed as a long-term function.

The iSBC 215G board accomplishes the rewind function by directing the iSBX 217B/C board to perform an off-line rewind on the specified tape drive. Once the operation is started, the iSBC 215G board terminates the rewind function, posts the operation complete status, and sends an interrupt to the host CPU (if interrupts are not suppressed). When the tape drive completes the off-line rewind, the iSBC 215G board sends a second interrupt to the host CPU.

Note that, if it is necessary that the host CPU examine the 12-byte status buffer relative to the off-line rewind operation, it must request the contents of the long-term status buffer. If the iSBC 215G board is busy with another device when an off-line rewind operation is completed, the board completes the other function, posts the appropriate status, and sends an interrupt to the host CPU (if interrupts are not suppressed). After the host CPU services the interrupt and resets the status semaphore, the iSBC 215G board posts the rewind complete status and again sends an interrupt to the host CPU, this time to indicate completion of the off-line rewind operation.

3.5.17 SPACE FORWARD ONE FILE MARK (12H)

The space forward one file function moves the tape forward until a file mark or end of medium is reached. To perform the function, the host CPU establishes the following fields in the I/O parameter block:

- **Device Code**  Bytes 8 and 9
- **Unit Number**  Byte 10
- **Function Code**  Byte 11

The iSBC 215G board accomplishes the space forward one file function by directing the iSBX 217B/C board to perform an off-line tape movement on the specified tape drive. Once the operation is started, the iSBC 215G board terminates the space forward one file function, posts the operation complete status, and sends an interrupt to the host CPU (if interrupts are not suppressed). When the tape drive reaches a file mark or detects end of medium and completes the function, the iSBC 215G board sends a second interrupt to the host CPU.
Note that, if it is necessary that the host CPU examine the 12-byte status buffer relative to the off-line space forward one file operation, it must request the contents of the long-term status buffer. If the iSBC 215G board is busy with another device when an off-line operation is completed, the board completes the other function, posts the appropriate status, and sends an interrupt to the host CPU (if interrupts are not suppressed). After the host CPU services the interrupt and resets the semaphore, the iSBC 215G board posts the operation complete status and again sends an interrupt to the host CPU, this time to indicate completion of the off-line space forward one file operation. This operation is classed as a long-term function.

3.5.18 WRITE FILE MARK (14H)

The write file mark function allows writing additional file marks at the end of a file written to the tape. (During write operations, the read/write terminate function automatically writes one file mark at the end of a file.) To perform the write file mark function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code: Bytes 8 and 9
- Unit Number: Byte 10
- Function Code: Byte 11

The write file mark function writes a file mark on the tape at the position of the write head at the time of the command. This allows writing several file marks on the tape to denote special meaning to the separation between two files or to indicate end of tape. This function is classed as a short-term function.

3.5.19 ERASE TAPE (17H)

The erase tape function prepares the tape for subsequent recording by removing all existing recorded data. To perform the erase tape function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code: Bytes 8 and 9
- Unit Number: Byte 10
- Function Code: Byte 11

Whenever a write data operation is performed, the tape is automatically erased just before the data is written onto the tape. However, there is some risk that this erase operation may leave some background noise in the area written onto. Also, any data from a previous recording in the area beyond the subject area are not erased. The erase tape function allows for the removal of all previous data by first rewinding the tape to the beginning of tape marker, erasing forward to the end of tape marker, and again rewinding to the beginning of tape marker.

The iSBC 215G board accomplishes the erase tape function by directing the iSBX 217B/C board to perform an off-line erase operation on the specified tape drive. Once the operation is started, the iSBC 215G board
terminates the erase tape function, posts the operation complete status, and sends an interrupt to the host CPU (if interrupts are not suppressed). When the tape drive completes the function, the iSBC 215G board sends a second interrupt to the host CPU.

Note that, if it is necessary that the host CPU examine the 12-byte status buffer relative to the off-line erase tape operation, it must request the contents of the long-term status buffer. If the iSBC 215G board is busy with another device when an off-line operation is completed, the board completes the other function, posts the appropriate status, and sends an interrupt to the host CPU (if interrupts are not suppressed). After the host CPU services the interrupt and resets the status flag, the iSBC 215G board posts the operation complete status and again sends an interrupt to the host CPU, this time to indicate completion of the off-line erase-tape operation. This operation is classed as a long-term function.

3.5.20 LOAD TAPE (18H)

The load tape function is the fourth function in the tape initialization sequence and positions the tape to the beginning of tape marker. The load tape function also initiates the on-going iSBC 215G board check for tape medium change. To perform the load tape function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code
- Unit Number
- Function Code
- Bytes 8 and 9
- Byte 10
- Byte 11

3.5.21 TAPE RESET (1CH)

The tape reset function is the third function in the tape initialization sequence and is used to initialize the tape drive. To perform the tape reset function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code
- Unit Number
- Function Code
- Bytes 8 and 9
- Byte 10
- Byte 11

3.5.22 RETENSION TAPE (1DH)

The retension tape function prepares the tape for subsequent operations by moving the tape forward to the end of tape marker and then rewinding to the beginning of tape marker. This re-stacks the tape in the cartridge and assures free and easy tape movement. To perform the retension tape function, the host CPU establishes the following fields in the I/O parameter block:
The iSBC 215G board accomplishes the retension tape function by directing the iSBX 217B/C board to perform off-line retensioning on the specified tape drive. Once the operation is started, the iSBC 215G board terminates the function, posts the operation complete status, and sends an interrupt to the host CPU (if interrupts are not suppressed). When the tape drive completes the operation, the iSBC 215G board sends a second interrupt to the host CPU.

Note that, if it is necessary that the host CPU examine the 12-byte status buffer relative to the off-line retension tape operation, it must request the contents of the long-term status buffer. If the iSBC 215G board is busy with another device when an off-line operation is completed, the board completes the other function, posts the appropriate status, and sends an interrupt to the host CPU (if interrupts are not suppressed). After the host CPU services the interrupt and resets the status flag, the iSBC 215G board posts the operation complete status and again sends an interrupt to the host CPU, this time to indicate completion of the off-line retension tape operation. This operation is classed as a long-term function.

3.5.23 READ TAPE STATUS (1EH)

The read tape status function transfers the existing tape drive status from the iSBX 217B/C board to the iSBC 215G board short-term status buffer. The tape status is automatically transferred from the iSBX 217B/C board to the iSBC 215G board at the end of each tape operation. Typically, this function is not used unless there exists a relatively long period between tape operations. If it is required that the host CPU examine the tape drive status, it must transfer the contents of the short-term buffer to system memory using the transfer status buffer function. To perform the read tape status function, the host CPU establishes the following fields in the I/O parameter block:

<table>
<thead>
<tr>
<th>Field</th>
<th>Bytes 8 and 9</th>
<th>Byte 10</th>
<th>Byte 11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Code</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unit Number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function Code</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.5.24 READ/WRITE TERMINATE (1FH)

The read/write terminate function marks the end of a read or write operation. When the read/write terminate function is used to terminate a write tape operation, a file mark is automatically written and the tape is not rewound. The read/write terminate function is used to terminate a read operation only when the operation is being aborted (a read operation ordinarily terminates when a file mark is encountered). When a read operation is aborted using the read/write terminate function, the tape is rewound to the beginning of tape marker. This operation is classed as a
short-term function. To perform the read/write terminate function, the host CPU establishes the following fields in the I/O parameter block:

- Device Code
- Unit Number
- Function Code
- Bytes 8 and 9
- Byte 10
- Byte 11

### 3.6 FUNCTION MODIFIERS

The function modifiers allow the user to change easily the default functions and tailor the iSBC 215G board to a particular application. These modifiers are specified in bytes 12 and 13 of the I/O parameter block. Each of the modifier actions is assigned to a single bit in the modifier word and each action is enabled by the presence of a 1 in that bit position. Figure 3-14 illustrates the format of the modifier word; the following paragraphs describe each of the relevant parts of that word.

**SUPPRESS INTERRUPT** -- The suppress interrupt modifier bit, when set to 1, directs the iSBC 215G board to suppress assertion of the interrupt at the end of a short-term function. When a long-term function is executed, the suppress interrupt modifier suppresses the first interrupt when the iSBC 215G board posts the operation complete status. The second interrupt (sent when the board signals that the off-line portion of the function has been completed) is not suppressed.

---

![Modifier Word Format](image)

Figure 3-14. Modifier Word Format
PROGRAMMING INFORMATION

INHIBIT RETRIES -- (Hard Disk Only) The inhibit retries modifier bit, when set to 1, directs the iSBC 215G board to attempt only once to complete a data transfer function.

READ/WRITE DELETED DATA -- The read/write deleted data modifier bit, when set to 1, allows access to the corresponding iSBX 218A board flexible disk controller command.

24-BIT ADDRESSING -- The 24-bit addressing modifier bit, when set to 1, converts the data buffer pointer format from the standard segment and offset addressing to 24-bit linear addressing. This allows placement of the data buffer anywhere in the 16-Mbyte space addressable on the MULTIBUS interface. When 24-bit addressing is used, byte 18 provides address bits $0H$ through $7H$, byte 19 provides address bits $8H$ through $FH$, and byte $20H$ provides address bits $10H$ through $17H$. Byte $21H$ is set to all $0$'s.

BYPASS BOARD TEST -- The bypass board test modifier bit, when set to 1 during initialization of hard-disk drive $0$, causes the iSBC 215G board to skip the RAM test ordinarily executed when disk drive $0$ is initialized. When the RAM test is executed, all sets of parameters in the on-board RAM are destroyed. Thus, if drive $0$ is detached and then later re-attached with this modifier bit set to $0$, it is necessary to re-initialize all of the other drives to again pass the device parameters to the iSBC 215G board.

TRANSFER LONG-TERM STATUS BUFFER -- The transfer long-term status buffer modifier bit, when set to 1, converts the transfer status buffer function from a transfer of the short-term status buffer to transfer of the long-term status buffer. Because transfer of the long-term status buffer destroys the contents of the short-term status buffer, the short-term buffer contents should be transferred first.

DISK MOTOR CONTROL -- The disk motor control bit, when set to 1, invokes the motor on control for those 8-inch flexible-disk drives that require such control. Do not set this bit for 5 1/4-inch drives; motor control is assumed for those units.

ROM CHECKSUM TEST -- The ROM checksum modifier bit, when set to 1, converts the diagnostic function from the full diagnostic test to a checksum test of the iSBC 215G board ROM.

RESTORE TO CYLINDER $0$ -- The restore to cylinder $0$ modifier bit, when set to 1, limits the diagnostic function to restoring the read/write heads to cylinder $0$.

3.7 EXTENDED STATUS

At the end of each iSBC 215G board operation, including operations with any iSBX MULTIMODULE boards mounted on the iSBC 215G board, information pertaining to the operation is stored in one of the two status buffers provided in the on-board RAM. If the completed operation was a short-term function, only the short-term status is posted; if the
completed operation was a long-term function, both the short-term and long-term status are posted. The short-term status is posted at the time the first interrupt is asserted, and the long-term status is posted when the specified device completes the off-line portion of the function. Prior to asserting the interrupt, the iSBC 215G board summarizes the status buffer into the operation status byte. Figure 3-15 illustrates the format of the operation status byte.

Bits 0 through 3 contain the code for a device specific status summary. Bits 4 and 5 are used to specify the device unit number. Bit 6 indicates (when set to 1) that a hard error occurred on the specified device and that a function could not be executed. Bit 7 indicates (when set to 1) that some kind of error occurred. If bit 7 is set to 1 and bit 6 is set to 0, this indicates that a soft (recoverable) error occurred.

![Operation Status Byte Format](image)

Figure 3-15. Operation Status Byte Format

### 3.8 STATUS OPERATIONS

Whenever the operation status byte indicates that an error occurred, the appropriate status block, short-term or long-term, can be requested by the host CPU to obtain additional information pertaining to the error condition. This information is stored in the status buffer, the contents of which are described in the following paragraphs.
3.8.1 STATUS BUFFER FORMAT

Although disk and tape operations use the same status buffers, the definitions of the bytes contained in the status buffer are different for those. Table 3-8 lists the format of the status buffer for both disk and tape operations.

Table 3-8. Error Status Buffer

<table>
<thead>
<tr>
<th>Byte</th>
<th>Disk Function</th>
<th>Tape Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>Detailed Status Byte</td>
<td>Detailed Status Byte</td>
</tr>
<tr>
<td>1</td>
<td>Detailed Status Byte</td>
<td>Detailed Status Byte</td>
</tr>
<tr>
<td>2</td>
<td>Detailed Status Byte</td>
<td>Detailed Status Byte</td>
</tr>
<tr>
<td>3</td>
<td>Desired Cylinder (Low Byte)</td>
<td>Beginning of Tape Marker Detected</td>
</tr>
<tr>
<td>4</td>
<td>Desired Cylinder (High Byte)</td>
<td>Logical Load Point Detected</td>
</tr>
<tr>
<td>5</td>
<td>Desired Head and Volume</td>
<td>File Mark Detected</td>
</tr>
<tr>
<td>6</td>
<td>Desired Sector</td>
<td>Logical End of Tape Detected</td>
</tr>
<tr>
<td>7</td>
<td>Actual Cylinder (Low Byte)</td>
<td>Not Used</td>
</tr>
<tr>
<td>8</td>
<td>Flags, Actual Cylinder (High Byte)</td>
<td>No Data Detected (Blank Tape)</td>
</tr>
<tr>
<td>9</td>
<td>Actual Head and Volume</td>
<td>Not Used</td>
</tr>
<tr>
<td>10</td>
<td>Actual Sector</td>
<td>Not Used</td>
</tr>
<tr>
<td>11</td>
<td>Number of Retries</td>
<td>Not Used</td>
</tr>
</tbody>
</table>

The contents for each of the status bytes are as follows:

Bytes Ø through 2 of the status buffer (for both disk and tape functions) contain the detailed error status for the last function completed.

Bytes 3 through 6 (for disk functions) list the cylinder, head, and sector address requested in the I/O parameter block for the function completed.

Byte 3 (for a tape function), when set to ØFFH, indicates that the beginning of tape marker was detected and that the tape is positioned at the start of the recording area on the tape.

Byte 4 (for a tape function), when set to ØFFH, indicates that the logical load point on the tape was detected. The logical load point is typically the fully rewound position on the tape; the beginning of tape marker is usually located a short distance before the fully rewound position.

Byte 5 (for a tape function), when set to ØFFH, indicates that the tape drive encountered a file mark during function execution.
Byte 6 (for a tape function), when set to $\text{FFH}$, indicates that the tape drive encountered the logical end of tape marker during the execution of the last function.

Bytes 7 through 10 (for disk functions) list the cylinder, head, and sector address actually accessed by the disk drive during the execution of the function. The status byte assigned to return the high byte of the actual cylinder number also returns additional information about the track and sector (flags). Figure 3-16 illustrates byte 8 of the status buffer for disk operations.

Bytes 7, 9, and 10 are not used for tape functions and are returned as all 0's.

Byte 8 (for a tape function), when set to $\text{FFH}$, indicates that the tape drive was unable to detect any valid data after attempting to read several inches of blank tape.

Byte 11 (for disk functions) lists the number of retries attempted by the iSBC 215G board or the specified drive.

---

### Figure 3-16. Status Buffer Byte 8 Format for Disk

#### 3.8.2 Detailed Error Status

The first three bytes of the status buffer provide detailed status pertaining to the last completed function. This is ordinarily called the error status. If any bit in the first three status bytes is set to 1, bit 7 in the operation status byte is set to report that an error occurred. If any bit in the first two status bytes is set to 1, bit 6 in the operation status byte reports the type: set to 1 for hard error; set to 0 for soft error. The definitions of the status bits for disk functions and tape functions are similar, but not identical. In the following descriptions of the status bits, both disk and tape definitions are included where required.
3.8.2.1 Status Byte 0

Figure 3-17 illustrates the format of status byte Ø; the paragraphs that follow the figure describe the bits of this status byte.

Figure 3-17. Status Buffer Byte Ø Format

Bit 0 -- INVALID iSBC 215G FUNCTION -- Bit 0 set to 1 indicates that the function code in byte 11 of the I/O parameter block was not one of the defined function codes. (The present version of the iSBC 215G board checks each function code to determine if it is defined. The iSBC 215A/B board and earlier versions of the iSBC 215G board did not make this verification check.)

Bit 1 -- INVALID iSBX 217B/C FUNCTION -- (Tape Function) Bit 1 set to 1 indicates that the function code (for the iSBX 217B/C board) in byte 11 of the I/O parameter block could not be executed by the iSBX 217B/C board. (Bit 1 is always set to Ø when disk functions are executed.)

Bit 2 -- INVALID TAPE DRIVE FUNCTION -- (Tape Function) Bit 2 set to 1 indicates that the function code (for the tape drive) in byte 11 of the I/O parameter block could not be executed by the tape drive. (Bit 2 is always set to Ø when disk functions are executed.)

Bit 3 -- iSBC 215G BOARD RAM ERROR -- (Disk Function) Bit 3 set to 1 indicates that the iSBC 215G board failed the RAM test portion of the internal diagnostic program. (Though not strictly a disk function, the RAM test function is classed as one; bit 3 is always set to Ø when tape functions are executed.)

Bit 4 -- iSBC 215G BOARD ROM ERROR -- (Disk Function) Bit 4 set to 1 indicates that the iSBC 215G board failed the ROM checksum portion of the internal diagnostic program.
Bit 4 -- iSBC 217B/C BOARD ROM ERROR -- (Tape Function) Bit 4 set to 1 indicates that the iSBX 217B/C board failed the ROM checksum test performed during the tape initialization function.

Bit 5 -- SEEK IN PROGRESS -- (Disk Function) Bit 5 set to 1 indicates that there was an off-line seek in progress in the disk drive when initiation of another function was attempted with the same drive.

Bit 5 -- LONG-TERM FUNCTION IN PROGRESS -- (Tape Function) Bit 5 set to 1 indicates that there was an off-line function in progress in the tape drive when initiation of another function was attempted with the same drive.

Bit 6 -- ILLEGAL FORMAT TYPE -- (Disk Function) Bit 6 set to 1 indicates that one of two illegal operations was attempted or that an illegal I/O parameter block format was detected. (The illegal operations are: 1) an attempt to assign an alternate to the assigned alternate track (that is, attempting to assign a second alternate track directly after finding that the assigned alternate track is defective); or 2) an attempt to access directly an iSBC 215G board unassigned alternate track as a primary data track. Also, bit 6 is set to 1 when a check of the device code, function code, or unit number in the I/O parameter block reveals that a illegal value was used.

Bit 6 -- ILLEGAL CONFIGURATION -- (Tape Function) -- Bit 6 set to 1 indicates that an attempt was made to access a tape drive that is not classed as present. As with disk functions, detection of an illegal device code, function code, or unit number in the I/O parameter block results in bit 6 being set to 1.

Bit 7 -- END OF MEDIUM -- Bit 7 set to 1 indicates that the end of medium marker was detected before the requested transfer count in the I/O parameter block was satisfied. For disk operations, the condition reported in bit 7 is classed as an error.

3.8.2.2 Status Byte 1

Figure 3-18 illustrates the format of status byte 1; the paragraphs that follow the figure describe the bits of this status byte.
Bit 0 -- ILLEGAL SECTOR SIZE -- (Disk Function) -- Bit 0 set to 1 indicates that the sector size information read from the sector header on the disk drive conflicts with the sector size specified when the initialization function was executed.

Bit 0 -- LENGTH ERROR -- (Tape Function) -- Bit 0 set to 1 indicates one of the following conditions:

- The data transfer function specified a requested transfer count of 0.
- A file mark was detected with the requested transfer count un-satisfied.
- The data transfer function was terminated by the iSBX 217B/C board.

Bit 1 -- DIAGNOSTIC FAULT -- (Disk Function) -- Bit 1 set to 1 indicates that the iSBC 215G board and the disk drive failed execution of the internal diagnostic program. (Though not strictly a disk function, the diagnostic fault function is classed as one; bit 1 is always set to 0 when tape functions are executed.)

Bit 2 -- NO INDEX -- (Disk Function) -- Bit 2 set to 1 indicates that the iSBC 215G board did not receive an index pulse from the disk drive. This error indicates typically that the specified disk drive is not attached to the iSBC 215G board or that power is not applied to the disk drive.

Bit 2 -- NO RESPONSE TIME OUT -- (Tape Function) Bit 2 set to 1 indicates that the iSBX 217B/C board failed to respond to an attempted access within the prescribed time.
Bit 3 -- INVALID FUNCTION CODE -- Bit 3 set to 1 (a summary error) indicates that one of the three function code error bits in status byte 0 was set to 1.

Bit 4 -- SECTOR NOT FOUND -- (Disk Function) -- Bit 4 set to 1 indicates that the ISBC 215G board failed to locate the sector number, specified in the I/O parameter block, in any of the sector ID fields in the track.

Bit 4 -- TAPE CARTRIDGE MISSING -- (Tape Function) -- Bit 4 set to 1 indicates that there is no tape cartridge installed in the specified tape drive.

Bit 5 -- INVALID ADDRESS -- (Disk Function) -- Bit 5 set to 1 indicates that an invalid cylinder, head, or sector was specified.

Bit 6 -- SELECTED UNIT NOT READY -- Bit 6 set to 1 indicates that the device specified in the I/O parameter block did not respond to an attempted access. This error typically indicates that the specified device is not attached to the ISBC 215G board, that power is not applied to the device, or that the device was manually switched off-line.

Bit 7 -- DISK/TAPE WRITE PROTECTED -- Bit 7 set to 1 indicates that an attempt was made to write to a medium, installed in the selected device, that was mechanically write-protected.

3.8.2.3 Status Byte 2

Figure 3-19 illustrates the format of status byte 2; the following paragraphs describe the bits of this status byte.

Bit Ø -- Not used. Always set to Ø.

Bit 1 -- TAPE SOFT ERROR -- (Tape Function) Bit 1 set to 1 indicates that the data transfer function with the tape drive connected to the ISBX 217B/C board was completed successfully, but that one or more retries was (were) necessary to complete the transfer. (The cause of the retry or retries can be actual data errors that were successfully written or read during retry.) Bit 1 is always set to Ø when disk functions are executed.

Bit 2 -- PARITY ERROR -- (Tape Function) Bit 2 set to 1 indicates that a data byte parity error was detected by the ISBX 217C board during a data transfer. Bit 2 is not used for disk functions or tape functions with the ISBX 217B board and is always set to Ø for such operations.
Figure 3-19. Status Buffer Byte 2 Format

Bit 3 -- DATA FIELD ECC ERROR -- (Disk Function) -- Bit 3 set to 1 indicates that the iSBC 215G board detected an error in the data field during an error correcting code (ECC) check in one of the sectors transferred. If bit 6 of the operation status byte is set to 0, this indicates that the error was soft and is recoverable; if bit 6 is set to 1, this indicates that the error was hard and is not recoverable. Bit 3 is always set to 0 when tape functions are executed.

Bit 3 -- TAPE DATA ERROR -- (Tape Function) -- Bit 3 set to 1 indicates that the iSBC 215G board detected a data error in the file that was transferred and that the error was hard and is not recoverable. Bit 3 is always set to 0 when disk functions are executed.

Bit 4 -- ID FIELD ECC ERROR -- (Disk Function) Bit 4 set to 1 indicates that the iSBC 215G board detected an error in the ID field during an error correcting code (ECC) check in one of the sectors transferred. If bit 6 of the operation status byte is set to 0, this indicates that the error was soft and is recoverable; if bit 6 is set to 1, this indicates that the error was hard and is not recoverable. Bit 4 is always set to 0 when tape functions are executed.

Bit 5 -- DRIVE FAULT -- Bit 5 set to 1 indicates that there is a hardware problem in the selected device.

Bit 6 -- CYLINDER ADDRESS MISCOMPARE -- (Disk Function) Bit 6 set to 1 indicates that the heads were positioned to the incorrect cylinder. The recovery process for this error is to perform a seek to track 0 and then re-attempt the seek to the desired track.
Bit 6 -- BUFFER OVER-RUN/UNDER-RUN -- (Tape Function) -- Bit 6 set to 1 indicates that the data transfers from the iSBC 215G board did not keep up with the tape drive. This is not an error condition.

Bit 7 -- SEEK ERROR -- (Disk Function) -- Bit 7 set to 1 indicates that the read/write heads were not positioned to the correct track during the seek (implicit or explicit) function. When this error is detected, the internal firmware automatically commands a re-calibrate procedure and will initiate a re-seek unless retries are inhibited. Bit 7 is always set to Ø when tape functions are executed.

3.9 INTERRUPTS

The iSBC 215G board generates interrupts to alert the host CPU of significant changes in mass storage system status by asserting any of the eight MULTIBUS interrupt lines (INT0* through INT7*). The iSBC 215G board ordinarily posts interrupts to the host CPU for three conditions:

1. Operation complete.
2. Seek complete.
3. Medium change.

The interrupt on operation-complete can be disabled by entering a 1 in bit Ø of the modifier word in I/O parameter block bytes 12 and 13. The seek-complete and medium-change interrupts cannot be disabled. Once an interrupt is asserted, it can be cancelled by a clear-I/O command from the host CPU to the board (ØH to the wake-up port), a power-on reset, or assertion of the MULTIBUS INIT* signal.

Interrupt priority level selection in the range of Ø through 7 is done through jumper stake pin connections on the board. Two pins must be connected by wire wrapping to select the priority. (Refer to the description of interrupt priority level selection in Chapter 2.)

3.10 iSBX™ BUS EXPANSION

Connectors J3 and J4 on the iSBC 215G board allow access to the iSBX bus (refer to the Intel MULTIBUS Handbook for detailed information). The iSBX bus includes 16 data lines and 3 address lines, providing a total of eight 16-bit I/O ports per connector. Using both J3 and J4, the iSBC 215G board can communicate through the iSBX bus with as many as 16 separate peripheral ports.

The iSBX 218A Flexible Diskette Controller Board connects to iSBX connector J4 and allows communication with as many as four flexible-disk drives; the iSBX 217B/C Magnetic Cartridge Tape Interface Board connects to iSBX connector J3 and allows communication with as many as four QIC-Ø2 1/4-inch tape drives. In addition, users can design I/O controller devices that interface with the iSBX bus and use the 8Ø89 I/O processor (IOP) to control data transfer.
Two methods are available to control the transfer of data between the iSBC 215G board and a device connected to the iSBX interface:

1. Commands from the iSBC 215G board ROM-based I/O program.

2. User written I/O programs.

Both the iSBX 218A Flexible Diskette Controller Board and the iSBX 217B/C Magnetic Cartridge Tape Interface Board use the ROM-based I/O program to control data transfers to and from the flexible-disk drives and QIC-02 cartridge-tape drives, respectively. The following paragraphs describe data transfer between the iSBC 215G board 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 driver program.

### 3.10.1 FIRMWARE DRIVERS

As described at the beginning of this chapter, the iSBC 215G board includes a ROM-based I/O transfer program that is designed to control hard-disk drives through the on-board drive interface, or flexible-disk and/or QIC-02 cartridge-tape drives through an iSBX 218A board and/or iSBX 217B/C board attached to iSBX connector J4 and/or J3. The I/O-transfer-through-iSBX command in this program can also be used for general data transfer between the host system memory and a user-designed I/O controller connected to the iSBX bus.

The I/O-transfer-through-iSBX command allows transfer of data between the host memory and the iSBX bus in the same manner as with the write-data or read-data commands. With this command, however, the user must provide the necessary interface hardware between the iSBX connector(s) and the I/O device with which the iSBC 215G board 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 218A and iSBC 215G boards. The complexity of the interface will depend on the type of I/O device and the desired data transfer rate.

### 3.10.2 USER-PROVIDED DRIVERS

A second method of initiating and controlling data transfer between the host CPU and the iSBX interface is through a user-designed driver program written in 8089 IOP 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 215G board RAM. In writing a program in 8089 IOP assembler code, reference to the 8089 Assembler User's Guide and the 8086 Family User's Manual is essential. The IOP offers a number of techniques for implementing handshaking with the iSBX bus, the use of wait states, and DMA transfers of whole blocks of data. These and other interfacing techniques are described in the user's guide.

There are two groups of interface control lines between the IOP and iSBX bus. The first group includes handshake and control lines; the second
group includes program lines. Table 3-9 lists the first group of lines. The IOP uses these lines directly to control data transfer through the ISBX bus. The second group of lines is used for control and status. The IOP accesses these lines through a read to memory-mapped I/O address 8000H for connector J3 and 8008H for connector J4. Table 3-10 lists these lines, pin assignments, and bit assignments.

Table 3-9. IOP Handshake and Control Lines on ISBX™ Bus

<table>
<thead>
<tr>
<th>J3 or J4 Pin</th>
<th>Description</th>
<th>ISBX™ Bus Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>34</td>
<td>Request DMA Transfer</td>
<td>MDRQT</td>
</tr>
<tr>
<td>32</td>
<td>Acknowledge DMA Transfer</td>
<td>MDACK*</td>
</tr>
<tr>
<td>16</td>
<td>Initiate Wait State</td>
<td>MWAIT*</td>
</tr>
<tr>
<td>6</td>
<td>MULTIBUS Clock</td>
<td>MCLK</td>
</tr>
<tr>
<td>15</td>
<td>I/O Read</td>
<td>IORD*</td>
</tr>
<tr>
<td>13</td>
<td>I/O Write</td>
<td>IOWRT*</td>
</tr>
<tr>
<td>26</td>
<td>Terminate DMA Activity</td>
<td>TDMA</td>
</tr>
</tbody>
</table>

Table 3-10. Control and Status Lines on ISBX™ Interface

<table>
<thead>
<tr>
<th>Connector J3</th>
<th>Address 8000H</th>
<th>Connector J4</th>
<th>Address 8008H</th>
<th>Pin No.</th>
<th>Description</th>
<th>ISBX™ Bus Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>OP0</td>
<td>Bit B</td>
<td>OP1</td>
<td>Bit 3</td>
<td>30</td>
<td>Option 0</td>
<td>OPT0</td>
</tr>
<tr>
<td>OP1</td>
<td>Bit C</td>
<td>OP11</td>
<td>Bit 4</td>
<td>28</td>
<td>Option 1</td>
<td>OPT1</td>
</tr>
<tr>
<td>INTR0</td>
<td>Bit 9</td>
<td>INTR01</td>
<td>Bit 1</td>
<td>14</td>
<td>Interrupt 0</td>
<td>MINTR0</td>
</tr>
<tr>
<td>INTR1</td>
<td>Bit A</td>
<td>INTR11</td>
<td>Bit 2</td>
<td>12</td>
<td>Interrupt 1</td>
<td>MINTR1</td>
</tr>
<tr>
<td>MIPST*</td>
<td>Bit 8</td>
<td>MIPST*</td>
<td>Bit 0</td>
<td>8</td>
<td>iSBX Board</td>
<td>Present</td>
</tr>
</tbody>
</table>

Jumpers can be connected on the ISBC 215G board to allow the IOP to also write onto the option lines (as shown in Table 3-11). The option lines on only one of the interface connectors may be driven at a time. To drive the lines, the IOP writes to memory mapped I/O port 8018H. Bit 1 drives OP0 and OP1, but not both at one time; bit 2 drives OP1 and OP11, but not both at one time. All other bit positions in the data word must be set to 0 when driving the option lines.
Table 3-11. Jumper Connections Allowing Option Lines to be Driven

<table>
<thead>
<tr>
<th>Line</th>
<th>iSBX™ Connector</th>
<th>Mnemonic</th>
<th>Jumper Connection</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPØØ</td>
<td>J3</td>
<td>OPØ</td>
<td>W11-1 -- 2</td>
</tr>
<tr>
<td>OP11</td>
<td>J4</td>
<td>OPØ</td>
<td>W11-1 -- 3</td>
</tr>
<tr>
<td>OP1Ø</td>
<td>J3</td>
<td>OP1</td>
<td>W12-1 -- 2</td>
</tr>
<tr>
<td>OP11</td>
<td>J4</td>
<td>OP1</td>
<td>W12-1 -- 3</td>
</tr>
</tbody>
</table>

**NOTE**

If an iSBX controller is not installed on the iSBC 215G board, or if an iSBX controller that has been installed on a particular iSBX connector does not drive its respective terminate-DMA-activity line (J3-26, J4-26), the corresponding connector jumper (W3-1 -- 2 or W4-1 -- 2) must be installed.

3.10.2.1 Random-Access Memory Map

The iSBC 215G board RAM is used for a variety of purposes. Thus, 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 3-12. 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 required, some of this space can be used for program storage. If the devices connected to the iSBX bus do not require data from other devices, all of data buffer space can be used.

Table 3-12. On-Board Program and Parameter Storage RAM Space

<table>
<thead>
<tr>
<th>Description</th>
<th>Address Range (Hexadecimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Buffer (Note)</td>
<td>4000 through 440F</td>
</tr>
<tr>
<td>Program Storage</td>
<td>4410 through 45FF</td>
</tr>
<tr>
<td>Scratch Pad (Note)</td>
<td>4600 through 46FF</td>
</tr>
<tr>
<td>Pointers, Error Status, and Save Area</td>
<td>4700 through 47FF</td>
</tr>
</tbody>
</table>

Note: These areas may be modified by iSBC 215G board command use. If data in the data buffer need not be saved, the space may be used as a scratch-pad area between transfers.
3.10.2.2 Execution From On-Board RAM

Executing the program from on-board RAM presents space limitations, but allows data transfers to be performed at the IOP highest program execution speed. To overcome some of the RAM space problems, the program can be divided into shorter routines that 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 command allows an I/O transfer routine or program that is stored in iSBC 215G board RAM to be started from a host program.

3.10.2.3 Execution From System Memory

Executing the program from host memory is inherently slower than executing the program from on-board RAM, because it requires constant access to 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 local memory. To execute a program, the user must:

1. Establish I/O communications blocks in host system memory.
2. Set the wake-up address jumpers on the iSBC 215G board for the address of the first byte of the wake-up block.
3. Program the host CPU to initiate program execution by writing $01H 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. These are:

1. Byte $0 of the channel control block must be set to $03H to indicate to the iSBC 215G board 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 for detailed information on setting up an I/O parameter block when the I/O program is to be executed from host system memory.

3.10.2.4 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 215G board firmware.
2. Execute the execute-iSBX-I/O 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 3-20). Also, upon entering the program, the following IOP registers are defined as:

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>7E00H</td>
<td>Scratch Pad Stack</td>
</tr>
<tr>
<td>IX</td>
<td>Ø to 3</td>
<td>Unit Number</td>
</tr>
</tbody>
</table>

Exit from the program must always be to ROM location ØC5H, and the IOP BC register must be set to FFH.

---

**Figure 3-20. Execution of iSBX™ Bus I/O Program from RAM**

---

### 3.10.2.5 EXAMPLE I/O PROGRAM

Appendix B provides an example of a host CPU program to initiate data transfers between the host system memory and disk drives through the iSBC 215G board.
4.1 INTRODUCTION

This chapter provides a functional description of the ISBC 215G board circuit operation. The description 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 8/16-Bit HMOS I/O Processor and the MULTIBUS interface. Familiarity with the disk drive operation and interface specifications will also prove beneficial in understanding the board operation.

4.2 SCHEMATIC INTERPRETATION

An installation component location diagram (Figure 5-2), a block diagram (Figure 5-3), and a schematic diagram (Figure 5-4) for the ISBC 215G board are included in Chapter 5 of this manual.

The schematic is drawn to standard drafting conventions with input signals entering from the left. Signal connections between individual sheets of the schematic include a location coordinate code immediately preceding (input signals) or following (output signals) the signal mnemonic. This code defines the location of the origin or destination of the signal within the schematic diagram. 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 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 column or 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. A circle on the input of a logic element indicates that a relative low level is required to activate the element. The absence of a circle indicates that a relative high level is required to activate the element. Output levels are indicated in the same manner. Logic gating symbols are drawn according to circuit function rather than the manufacturer's definition. For example, the gate defined by the truth table in Figure 4-1 can be drawn in one of the two configurations shown, depending on its circuit application.

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 mnemonic is followed by a star (for example, XACK*); when a signal is active in its high state, the star is omitted.
FUNCTIONAL DESCRIPTION

from the signal mnemonic, (for example, XACK). This convention corresponds to placing a bar over a signal mnemonic to indicate that it is active in its low state (for example, XACK).

---

4.3 GENERAL DESCRIPTION

The function of the iSBC 215G board is to allow the host system to access any location on a specific disk or tape of a selected drive and either:

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

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

1. Logic that performs communications and data transfers between the host central processor unit (CPU) and the iSBC 215G board through the MULTIBUS interface.

2. Logic that performs data transfers between the iSBC 215G board and the disk drive(s) through the hard-disk interface, and between the board and the flexible-disk or cartridge-tape drive(s) through the iSBX bus interface.

3. Logic that controls both of the above functions.

As shown in Figure 4-2, the iSBC 215G board contains an Intel 8089 8/16-Bit HMOS I/O Processor (IOP), which controls the data transfer process, using a program stored in on-board ROM. It receives instructions from the host CPU through four I/O communications blocks in system memory. Once the host CPU instructs the board to begin a data
transfer, the IOP makes a DMA transfer (independent of the host CPU) to or from system memory.

On-board RAM space (2 K bytes) is included for intermediate storage of data and to allow on-board error checking. This RAM operates as a data buffer and allows DMA transfer between the board and host system memory, which minimizes MULTIBUS overhead and eliminates disk drive overruns.

In support of the following general description of the iSBC 215G board functional logic groups, a detailed block diagram and a schematic diagram are included in Chapter 5 (see Figures 5-3 and 5-4).

4.3.1 COMMUNICATIONS WITH HOST

As shown in the block diagram (Figure 5-3), the board includes the logic to operate within a multi-master system and contend with other masters for control of the MULTIBUS interface.

The bus controller generates control signals that gate data transfers between system memory and the on-board RAM. It also controls the transfer of data from RAM to the drive communication circuitry.
FUNCTIONAL DESCRIPTION

The MULTIBUS interface address latches and PAL U64 transmit 24-bit addresses to system memory via the MULTIBUS interface. The MULTIBUS interface data transceiver transmits data to or from system memory via the MULTIBUS interface. The data transceivers use a byte-swap technique to allow data transfer with either an 8- or 16-bit system memory. Intercommunication among board logic groups is accomplished via the board internal data bus, which is 16 bits wide.

The wake-up address comparator assigns the iSBC 215G board a host system I/O port address and sets up a communications link between the IOP and the I/O communication blocks in system memory.

4.3.2 COMMUNICATIONS WITH HARD DISK

The IOP treats the ROM, RAM, iSBX I/O ports, and disk communications portions of the board circuitry as local memory. The internal address latches transmit 16-bit addresses to local memory. The internal data transceivers transmit data either to or from local memory. (Some of the addresses in local memory provide access to 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, 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 serializer/de-serializer (SER/DES) performs serial-to-parallel and parallel-to-serial conversion operations 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 word between the RAM and the SER/DES. In 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. In a read operation, a 16-bit serial word is transferred from the disk through the read data receiver to the SER/DES. The SER/DES then performs a serial-to-parallel conversion and stores the resulting word in the read buffer. The write data drivers 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 located 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 into the 32-bit ID comparator. Sector ID's from the disk are then read and compared with the selected sector ID. When the selected sector is located, 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 the end of each data field (see Figure 3-2). This ECC is used for error checking and correction of data errors. It corrects all errors in a burst of as many as 11 bits, and detects all errors in a burst of as many as 32 bits.
FUNCTIONAL DESCRIPTION

The gap control logic controls the spacing of data within a sector. Three programmable counters count disk clock pulses to provide timing for the gap control logic. Counter programmability allows disk formatting for a number of different record sizes and gap lengths.

The disk control logic transmits disk control information to the drives through control line drivers. The input control logic receives status information from the disk drive units and controls the sequencing of the board read and write operations.

4.3.3 COMMUNICATIONS VIA iSBX™ BUS

The iSBX bus interface provides the capability to connect Intel iSBX MULTIMODULE devices to the iSBC 215G board in order to control other peripheral devices such as flexible-disk or QIC-02 cartridge-tape drives. For detailed information on the iSBX bus, refer to the Intel MULTIBUS Handbook.

4.4 DETAILED DESCRIPTION

The detailed functional description of the iSBC 215G board circuitry is divided into three sections: board-to-host communications, board-to-drive communications, and local memory organization.

4.4.1 BOARD/HOST COMMUNICATIONS

The following paragraphs provide a detailed functional description of the iSBC 215G board logic that communicates with the host CPU through the MULTIBUS interface.

4.4.1.1 MULTIBUS® Interface Signals

The IOP communicates with the host CPU and the system memory through the MULTIBUS interface. The MULTIBUS interface signal description and pin configurations are included in Chapter 2. For a detailed description of the MULTIBUS interface operation, refer to the Intel MULTIBUS Handbook.

4.4.1.2 I/O Processor

The 8089 I/O Processor (IOP), (U84, 4X4), is a microprocessor device that is 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 without host CPU action allows it to carry out most system memory-to-drive or drive-to-memory transfers of data simultaneously with other host CPU operations. For detailed
FUNCTIONAL DESCRIPTION

information on the 8089 8/16-Bit NMOS I/O Processor, refer to the Intel Microsystem Components Handbook, Volume II.

A number of IOP control lines are important to the board design. The RST* line (4D1), when driven low, resets the IOP to the beginning of its internal firmware control program, and resets the interrupt latch and the read/write logic. Channel attention line CA (4B4) allows the host CPU to gain the attention of the IOP. On the first channel attention following a reset, the IOP fetches the contents of address FFFF6H and begins an internal initialization procedure. On subsequent channel attentions, the IOP reads the I/O communications blocks in system memory for further instructions.

The bus interface unit (BIU) in the IOP controls the board internal data bus cycles, transferring instructions and data between the IOP and external memory or the disk. Every bus access is associated with a register 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.

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

The clock circuit consists of an 8284A Clock Generator/Driver (U55, 4C6), and a 15-MHz crystal. The clock generator/driver divides the crystal output by three to produce the 5-MHz clock signal necessary to drive the IOP, and also produces an IOP reset signal (RST), which is used at power-up, after an initialization, or after a board reset. In addition to the reset signal, the clock/driver also produces a synchronized ready input (RDY) to the IOP. A high on the RDY line 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 that data are ready to be read during a read operation.

The 8289 Bus Arbiter, (U9Ø, 3D6), controls the IOP access to the MULTIBUS interface (see Figure 4-3). The bus arbiter monitors the IOP status lines (S0*, S1*, and S2*). When the lines indicate that the IOP does not presently control the bus, the bus arbiter activates a bus request (BREQ*). The low BREQ* is transmitted to the bus priority resolving circuitry in the host CPU, which returns a low on bus priority in line BPRN*, giving the IOP access to the MULTIBUS interface, and the bus arbiter activates its busy signal (BUSY*), indicating to the other system masters that the MULTIBUS interface is in use. The bus arbiter then activates the address enable signal (AEN*), which is transmitted to the 8288 Bus Controller (U91, 3C4), to enable its command outputs; to the clock generator/driver (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 to the MULTIBUS interface. Jumper stake pins W18-1, 2, and 3 allow the user to select the any-request option. Jumper W18-1 -- 2 installation causes the board to relinquish control of the MULTIBUS.
interface following a request from a higher priority device only. Jumper W18-1 -- 3 installation causes the board to relinquish control of the MULTIBUS interface following a request from any device of either higher or lower priority.

Figure 4-3. Bus Arbiter and Controller Logic

4.4.1.3 Bus Controller

The 8288 Bus Controller (U91, 3C4), decodes the status line outputs (S0*, S1*, and S2*) from the IOP and generates the appropriate bus cycle signal. Table 4-1 lists the different signals generated for each configuration of the IOP status lines.
### FUNCTIONAL DESCRIPTION

**Table 4-1. IOP Status Line Decodes**

<table>
<thead>
<tr>
<th>Status Input</th>
<th>CPU Cycle</th>
<th>Bus Controller Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0*</td>
<td>Instruction Fetch, Local</td>
<td>INTA*</td>
</tr>
<tr>
<td>S0*</td>
<td>Read Memory, Local</td>
<td>IORC*</td>
</tr>
<tr>
<td>S0*</td>
<td>Write Memory, Local</td>
<td>AIOWC*</td>
</tr>
<tr>
<td>S0*</td>
<td>Halt</td>
<td>None</td>
</tr>
<tr>
<td>S0*</td>
<td>Instruction Fetch, System</td>
<td>MRDC*</td>
</tr>
<tr>
<td>S0*</td>
<td>Read Memory, System</td>
<td>MRDC*</td>
</tr>
<tr>
<td>S0*</td>
<td>Write Memory, System</td>
<td>MWTC*</td>
</tr>
<tr>
<td>S0*</td>
<td>Passive</td>
<td>None</td>
</tr>
</tbody>
</table>

These bus cycle signals can be divided into two groups: those that allow the IOP to access system memory (MWTC* and MRDC*) and those that allow the IOP to access local (internal) memory (I-AIOWC* and IORC*). The IOP uses the I/O read (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 IOP also uses I-IO RC* and I-AIOWC* to enable the read and write function decoders (U3S and U36, 5B2, and 5A2).

The bus controller also generates a group of signals that control address and data flow throughout the iSBC 215G board. The address latch enable line (ALE) is used to strobe addresses from the IOP into both the system address latches (U81 through U83, 4X2), and the local address latches (U85 through U86, 5X7).

Data transmit/receive (DT/R), data enable (DEN), and peripheral data enable (PDEN*) signals control the data flow through the iSBC 215G board. The DT/R signal controls the direction of data transmission through the MULTIBUS interface and local transceivers. If DT/R is high, data are transmitted either to the MULTIBUS interface through transceivers U96, U97, and U98 (4X7) or 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 IOP through one of the two sets of transceivers. The DEN and PDEN* signals control the selection of the transceivers. If DEN is high, MULTIBUS interface transceivers U96, U97, and U98 are enabled; if PDEN* is low (indicating a peripheral cycle), local transceivers U52 and U53 are enabled.

4-8
4.4.1.3.1 **MULTIBUS® DATA TRANSFER.** The iSBC 215G board has three sets of MULTIBUS interface data transceivers: low-byte transceiver U97, which buffers DAT-Ø* through DAT-7*; high-byte transceiver U96, which buffers DAT-8* through DAT-F*; and swap-byte transceiver U98, which switches the data from DAT-Ø* through DAT-7* on the MULTIBUS interface to high-byte data bus lines AD8 through AD15 on the iSBC 215G board (see Figure 4-4). This byte-swap is performed only when the iSBC 215G board interfaces with a 16-bit system memory in byte mode. In such an instance, every odd address read from system memory is transmitted to the high-byte data lines of the board. The procedure is reversed when writing to an 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 and is derived from ADRO*; and ENBL SWAP BYTE* (5C1), which controls the swap byte transceiver. Table 4-2 shows when each of the control signals is active.

Figure 4-4. Data Transfer from MULTIBUS® to iSBC® 215G Board
4.4.1.3.2 **INITIALIZATION.** Before data can be transferred between system memory and the ISBC 215G board, the board must be initialized. The initialization procedure, which is described in Chapter 3, involves:

1. Resetting the IOP.
2. Clearing the reset.
3. Establishing a communication link between the IOP and the I/O communications blocks in system memory.
4. Reading the drive parameters from system memory to the ISBC 215G on-board RAM.

The following paragraphs describe the hardware operations that take place during this initialization procedure. (See Figure 4-5.)

4.4.1.3.2.1 **Wake-Up Address Comparator.** For the purpose of resetting the ISBC 215G board, clearing the reset, or getting the attention of the IOP (driving the CA signal true), the host CPU addresses the board 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 (the wake-up I/O port). Table 4-3 shows the three possible commands. The user determines the address of the I/O port through which board/CPU communications are to take place (the wake-up address) and installs the appropriate jumpers on the ISBC 215G board. When the host CPU issues a write command (IOWC*) to the wake-up address in system I/O space, the wake-up address comparator (U77 through U8Ø, 2X5) compares the address with the jumper configuration. If the address and configuration agree, the WAKEUP* signal is driven low, enabling the board to decode the command on the MULTIBUS interface data lines and determine the action to be taken.
FUNCTIONAL DESCRIPTION

Figure 4-5. Wake-Up Address Logic

Table 4-3. Host Wake-Up Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ØØH</td>
<td>Clear interrupt and clear reset.</td>
</tr>
<tr>
<td>Ø1H</td>
<td>Channel attention (start IOP operations).</td>
</tr>
<tr>
<td>Ø2H</td>
<td>Reset IOP.</td>
</tr>
</tbody>
</table>

The host CPU may use 8- or 16-bit I/O port addressing. A user-installed jumper indicates to the board the type of addressing that is being used. When jumper W3Ø-2 -- 19 is not installed, (8-bit I/O address), pin 9 of U75 is held high, creating a "don't care" status for the outputs of high-byte wake-up address comparators U77 and U78.

As it is described in Chapter 3, the iSBC 215G board also uses the configuration of the wake-up address jumpers to calculate the address of the first byte of the wake-up block, which is the first I/O communications block in system memory.
4.4.1.3.2.2 *Reset and Clear*. The first operation that must be performed during the initialization of the iSBC 215G board is the IOP reset. To reset the IOP, the host CPU writes $02\text{H}$ to the wake-up address. The WAKE-UP* line becomes low and gates the $02\text{H}$ (DAT-$0^*$ high and DAT-$1^*$ low) to the wake-up decoder (U65, 3B7), producing a low on the controller reset (CNTLR RST*) line. A low CNTLR RST* signal resets the IOP (4X4), resets read/write control logic IC U42 (8B1 through 5), clears control register U3 (12B5), and also sets 24-bit addressing PAL U64 (9B3) to its initialized state. When jumper W36-1 -- 2 is not installed, all I/O communication blocks are in the first page of system memory; when jumper W36-1 -- 2 is installed, all I/O communication blocks are in the last page. Once the board has been reset, the host CPU writes $00\text{H}$ (clear interrupt) to the wake-up address, which clears the reset. Wake-up decoder U65 decodes the highs on DAT-$0^*$ and DAT-$1^*$ to drive the CNTLR RST* line high.

4.4.1.3.2.3 *I/O Communications Blocks Links*. Following a power-up event or a software reset ($02\text{H}$ written to the wake-up I/O port), the link between the iSBC 215G board and the I/O communications blocks in system memory must be established. To establish this link, a clear reset ($00\text{H}$) is written to the wake-up I/O port followed by a channel attention ($01\text{H}$). The $01\text{H}$ is gated to wake-up decoder U65, producing a high on the channel attention (CHNL ATTN) line, which, in turn, drives the CA input to the IOP (4C4) high.

Since this is the first channel attention following reset, the IOP starts an internal initialization process. The first step of this process is to do an on-board fetch from address FFFF6H. The board actually gains control of the bus and this address is transmitted on the IOP address/data lines (AD0-AD15) to latches U85 and U86 (5B7). Gates U66, U72, and U76 (5D4) decode the output of these latches. The output of U76 enables U89 (5D3), gating the status configured by system data bus width jumper W30-1 -- 20 through data bit 0 line (DAT-$0^*$) to the IOP. (Jumper W30-1 -- 20 installed indicates that the host memory system supports 16-bit data transfers, jumper W30-1 -- 20 not installed indicates 8-bit data transfers.) Inverter U89 also generates the transfer acknowledge signal (XACK*), which is sent to the IOP (through the clock generator/driver), indicating that the operation has been completed. After determining the width of the system bus (8- or 16-bits), the IOP also performs on-board fetches from the addresses shown in Figure 4-6 as part of the initialization sequence. The XACK signal is generated after each fetch. (Thus, although it appears to the IOP that it is reading from the MULTIBUS interface, the read operation is from the on-board bus.)
Fetching addresses FFFF8/9H gates Ø's into the IOP. Fetching addresses FFFFA/BH causes the GATE SWS* line (5C1) to become low. The GATE SWS* signal gates the settings of the wake-up address jumpers through buffers U93, U94, and U95 (2X3) and into the IOP, which multiplies the configuration settings by $2^4$ to determine the 20-bit address of the wake-up block. The IOP then uses this address to fetch the channel control block address and establish a link with the other I/O communications blocks. On subsequent channel attention operations (when the host CPU writes Ø1H to the wake-up I/O port), the IOP skips the wake-up block and proceeds directly to the channel control block, which had been stored previously in an internal IOP non-programmable register. The IOP uses the channel control block to obtain the starting address of the board ROM-resident I/O transfer program (also called the channel control program). From this point on, this firmware program directs the board 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.

4.4.1.3.3 INTERRUPT PRIORITY. Jumper stake pins W19-C and W19-Ø through W19-7 (3B2) allow the user to select the controller interrupt priority with respect to other system peripherals. To issue an interrupt to the host CPU, the IOP writes Ø1ØØH to local I/O port Ø1ØH. This generates a high on data line BDAT-8 and a low on write decoder line WDCØ*, causing the interrupt latch (U56, 3B5) output to be high and the selected interrupt line to the MULTIBUS interface to be low. A ØØH written to the system I/O port wake-up address by the host CPU clears the interrupt.

4.4.2 BOARD/DRIVE COMMUNICATIONS

The following paragraphs provide a detailed functional description of the sections of the iSBC 215G board that communicate with hard-disk drives through the Winchester drive interface and with flexible-disk and QIC-Ø2 cartridge-tape drives through the iSBX bus interface (via connectors J3 and J4).
FUNCTIONAL DESCRIPTION

4.4.2.1 Board/Hard-Disk Drive Interface

All of the signals that are transmitted between the iSBC 215G board and the hard-disk (Winchester) drives are transmitted through the control cable (J1), the read/write cable (J2), or the ANSI cable (J5). (The physical configuration of these cables is described and illustrated in Chapter 2.) All signals transmitted between the board and the drives (except the read, write, and clock signals) are TTL level. The read, write, and clock signals are transmitted as differential signals.

The interface signals supported by the iSBC 215G board are described in the following paragraphs. Each of the drive interfaces uses the available lines in a unique manner. For the specific use of the lines, refer to Figures 2-1 through 2-5 and the specific drive user's manual.

4.4.2.1.1 CONTROL SIGNALS. Control and status information are exchanged between the iSBC 215G board and the drive through the control cable (J1 for non-ANSI interfaces; J5 for ANSI interfaces). Output signals are defined as those signals that the board transmits; input signals are defined as those that the board receives. The control cable is connected from J1 (or J5) on the iSBC 215G board to the first drive and as many as three subsequent drives in a daisy-chain fashion as shown in Figures 2-13 through 2-17. The functions of the 37 control cable lines can be divided into five classes:

1. Device Select (Output)
2. Head Select (Output)
3. General-Purpose Data Bus (Bidirectional)
4. Command Data (Output)
5. Status Data (Input)

Table 2-13 describes the function of the signals in each of these classes as transmitted through the control cable.

4.4.2.1.2 READ/WRITE SIGNALS. Read data, write data, clocks, and two status lines constitute the information exchanged via the read/write cable. Output signals are defined as those signals that the iSBC 215G board transmits to the disk drives; input signals are defined as those that the board receives. For multiple Shugart SA4000 drives, the read/write cables are connected from the iSBC 215G board to the disk drives in radial fashion; that is, one cable from the board to each drive. Connector J2 provides read, write, and clock signals for two drives; for example, RD0 (+ and -), and RD1 (+ and -). One of these signals selects physical address 0; the other selects physical address 1. For SA1000 drives, only the signals associated with physical address 0 are used. These signals are then daisy-chained between drives, allowing the board to communicate with as many as four drives. Chapter 2 describes the cabling requirements and the physical configuration of the
FUNCTIONAL DESCRIPTION

cables for the various drive types supported by the iSBC 215G board. Table 2-14 describes the function of each of the signals transmitted through the read/write cable. Note that the read, write, and clock signals are differential signals, requiring two lines in the cable; the status lines are TTL-level signals.

4.4.2.2 Board/Flexible-Disk and QIC-02 Cartridge-Tape Drive Interface

All signal and control lines transmitted between the iSBC 215G board and the flexible-disk and QIC-02 cartridge-tape drives via the iSBX bus are transmitted through connectors J3 and J4. These lines are described only in general terms in this manual and only as the lines pertain to the remainder of the description of the board interface with the storage drives. For more detailed information on these lines, refer to the Intel MULTIBUS Handbook.

It should be noted that the iSBC 215G board 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 iSBX interface thus resembles the read/write port, made up of the write buffer and the read buffer, that is used in the iSBC 215G board interface to the hard-disk drives.

The schematic diagram mnemonics for the signal and control lines (from the iSBC 215G board) that are connected to iSBX connectors J3 and J4 often differ from the respective line mnemonic from the iSBX bus specifications. Table 2-11 lists both the iSBX bus mnemonic and the iSBC 215G board mnemonic for each signal, in the iSBX bus, that the board supports.

4.4.2.3 Interface Timing

The following paragraphs provide a detailed description of the inter-circuit timing of formatting a disk, writing to a disk, or reading from a disk. The timing logic is shown on sheet 8 of the schematic diagram; the disk drive interface receivers and drivers are shown on sheets 9 through 12.

4.4.2.3.1 DIRECT MEMORY ACCESS TRANSFERS. In general, when the iSBC 215G board performs a read or a write function, it locates the area of the disk where the read or write is to be performed, then enters the direct memory access (DMA) mode to perform the actual transfer. In the DMA mode, the 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 board controls the transfer of data between the read/write port and the disk.

The ready line (RDY, 8D1) is used for hand shaking between the IOP and the data transfer circuitry. When RDY is low, the IOP is quiescent; when
FUNCTIONAL DESCRIPTION

RDY is high, the IOP performs a DMA transfer of data either from the 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 IOP executes firmware routines 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 (8DB) is low; the MWAIT* line (13D1 and 8D8) is high; the R/W GATE line (8D1) is high (see Figure 4-7), 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 thus keeps RDY activated.

Figure 4-7. Ready Signal Timing Diagram

On its first attempt to transfer data in the DMA mode, the IOP activates either RDC 28* or WDC 28* (8D8), depending on whether a read or a write is being performed, respectively. When RDC 28* or WDC28* is activated, the R/WDC 28 line is activated, lowering RDY and switching the IOP to the quiescent (wait) state. When the board data transfer circuitry locates the area on the disk where the read or write is to begin, it activates ENBL XFER (8D1). On the next occurrence of a bit ring-Ø pulse (bit Ø of each word, BR-Ø, 8D1) following the activation of ENBL XFER, U21-8 (8D3) is activated, activating RDY. The IOP 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 driven high and, on the next IOP attempt to perform a data transfer,
R/WDC 28 is also driven high, lowering RDY. The data transfer does not occur and the IOP switches to the 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-∅ pulse, RDY is again activated and the next DMA data transfer occurs. The IOP 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) allows the IOP to be activated if a sector cannot be located 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. IC U41 (8D3) gates the TIME OUT signal to U12 (8D1) and activates RDY.

The HWAIT* line (8D8) is an iSBX interface control line and is derived from HWAIT∅* and HWAIT1* (13D8). Signal HWAIT* exercises the same control over the RDY line as U4∅ (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 IOP. For more detailed information, refer to the Intel Microsystem Components Handbook.

4.4.2.3.2 DISK FORMATTING. Before the surfaces of a disk can be used for writing and reading data, the disk must be formatted. Formatting is the operation of writing all address fields, gaps, ID headers, etc. for the complete disk. The iSBC 215G board performs this operation under software control. The software routine that controls formatting allows for formatting a single track with each format command to the board until the entire disk is formatted.

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, user-supplied data are written into data fields. The second operation requires two disk revolutions, one to write the odd physical data fields (1, 3, 5, etc.), and one to write the even physical fields (∅, 2, 4, etc.). Three disk revolutions are thus required to format a single track. The hardware execution operation described in the following paragraphs pertains to the formatting of a soft-sectored disk. The iSBC 215G board supports both soft- and hard-sectored disks.

NOTE

A soft-sectored disk (as used in Shugart/Quantum drives) requires that an address mark 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 need not be written.
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 Shugart/Quantum drives. When the format command is issued to the iSBC 215G board, the IOP performs a seek to the desired track (cylinder) to begin the format operation.

When the heads are positioned over the selected track, the IOP writes a C0H (for drive 0), a C8H (for drive 1), a D0H (for drive 2), and a D8H (for drive 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 the WRT GATE line (12C1). (See Figure 4-8.) The WRT GAT-F and FORMAT signals enable the board format control circuitry. With WRT XFER inactive and all 0's applied to the SER/DES, the board then writes all 0's to the drive while the IOP 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. This causes the IOP to poll (read) I/O port 8030H bit F (decoded as RDC 00*). Upon detecting the index, the IOP writes XXXXH to I/O port 8030H (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 allowed by the IOP between the detection of index and the activation of U63 (8B7) is approximately 11 byte times, which is the time that the iSBC 215G board requires to perform a number of firmware steps.
FUNCTIONAL DESCRIPTION

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 IOP 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 (U69, 8A7). (The IOP preset the counters in U69 at the beginning of the format operation.) When counter 1 times out at the end of 11 byte times, it activates the WRT XFER* line through U63-7 (8C3), and starts counter 2. The activation of WRT XFER* initiates the IOP DMA mode, during which the sync byte and the sector ID are written onto the disk. Counter 2 times out at the end of the ID field, starting counter 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. At the end of ECC TIME, the END TIME line is enabled, which lowers the WRT XFER* line and switches the IOP 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.

Counter 0 is set for a time equal to ECC+G3+DATA+G4, which the IOP sets according to the sector size selected for the drive. When counter 0 times out, it activates WRT AM* and counter 1, which begins the formatting of the second sector. This procedure is repeated until the IOP determines that the last ID field has been formatted. The IOP then begins searching for the index pulse. Upon receipt of index, the RST FRMT* line is activated, resetting WRT GAT-F and FORMAT, and inhibiting the writing of the next address mark. The IOP then continues through the format routine to the second operation, which is the writing of user-supplied data into the data fields.

For hard-sectored disks, jumper WL6-1 -- 3 (8B8) is installed. The formatting of the first sector thus begins when the first SECTOR* pulse from the disk (following 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 counter 1 counting. 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 counter 0.

The 8-inch Shugart/Quantum drive sector format differs in two ways from that of the other drive types. In these 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, D9H is used for the sync byte in the data field rather than 19H. When the iSBC 215G board 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), and 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 U1Ø (11C5), indicating to the board the presence of a data field instead of an ID field. In Memorex, 14-inch 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-inch Shugart/Quantum drive and the others is that, with the Shugart/Quantum drives, a 4EH pattern is written
FUNCTIONAL DESCRIPTION

in the gaps rather than Ø's. Inverters U58 and U17 (8D6) and gate U19 (8D5) create the 4EH pattern. Gates U40 and U60 (8A3) apply the pattern to the SER/DES when the SHUGART and WRT GAT-F lines are activated during a format.

4.4.2.3.3 WRITE DATA TRANSFERS. The write operation is divided into two steps: read sector ID and write data. When a write is initiated, the 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 board read circuitry (see Figure 4-9).

---

![Write Data Transfer Timing Diagram](image)

**Figure 4-9. Write Data Transfer Timing Diagram**

---

The IOP 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 (U2, U1, U23, and U22). Note that it has previously written to I/O port 8020H (decoded as WDC20*) to load counters 0, 1, and 2 of U69 (8A7). When the address mark (or sector pulse) is detected, SECTOR* is activated, which activates the AMPND-SECTOR* line (11B1). The low on AMPND-SECTOR* resets U34 (8C7) and de-activates the ID FIELD line, which de-activates the ADMKEN* line and activates the ALW SYNC SRCH,
FUNCTIONAL DESCRIPTION

initiating the search for the sync byte. Note that with the Shugart drives, the sync byte follows the address mark directly. Activating 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 are read into the SER-DES. Sync byte comparator U73 and U54 (7B5) monitors the outputs of the SER-DES and drives the SYNC BYTE* line (7B1) low 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 U2Ø (8D7). The SYNC FND* signal also drives the ENBL XFER line (8C1) high, which enables the ECC generator logic (7AX) and ready latch U21 (8D4) and enables counter Ø of U69 (8A7).

The 32-bit comparator compares the ID read from the disk with the ID of the selected sector. At the end of the ID time, counter Ø times out, driving the ECC TIME* line (7A8) low and initiating the ECC compare. If the ID and the ECC are valid, bit 6 of the board status register (U1Ø, 11C5) is reset. At the end of ECC time, U42-1Ø (8B2) activates the END TIME line, which resets RD GATE. The IOP then checks bit 6 of control status register U1Ø (11C5). If the bit is inactive, the IOP continues with the write operation. If the ID or ECC is not valid (bit 6 active), the AM ENABLE and RD GATE lines are then asserted and the board searches for the next address mark.

To begin the second step of the write operation, the IOP writes Ø1H to I/O port 8ØØØH (decoded WcØØ*) and enables the write gate (WRT GATE), through U24 (12B5), enabling the drive write circuitry. When counter Ø 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*, which enables write buffers U46 and U49 (7C7) and the ECC comparator logic (7AX), and raises the RDY line, indicating to the IOP that the write buffer is ready to receive data.

The IOP then enters DMA mode to write data from local RAM to the disk. The board 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. The END TIME signal then terminates the write operation.

4.4.2.3.4 READ DATA TRANSFERS. The read operation is divided into two steps: read sector ID and read data. Reading the sector ID is performed in the same manner as for the write operation (see Figure 4-1Ø).

When the desired sector is located, the RD GATE is 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 are then transferred from the disk to local RAM for the duration of the count of counter 2. When counter 2 times out, ECC TIME is activated. Following ECC TIME, the END TIME line is driven high, terminating the read operation.
4.4.2.4 Serializer/De-Serializer

The serializer/de-serializer (SER/DES) logic performs two functions: it converts parallel data words into a serial bit string to be sent to the disk drive during a write operation, and it converts a serial bit string into 16-bit words during a read operation. It consists of the write buffer (U46 and U49, 7C7), the serializer/de-serializer (U47 and U50, 7C5), the read buffer (U48 and U51, 7C4), and the selector (U70, 7A7).

During a write operation (WRT XFER* low), the IOP writes to I/O port address 8028H. Write I/O port address decoder U35 (5A2) decodes this address and drives WDC28* low, clocking the data to be written to the disk (BDAT-Ø through BDAT-F) into write buffer U46 and U49 (7C7). A high on load-serial-register line LDSR (7C6), derived from word clock U2Ø (8C7) loads the contents of the write buffer (SR-Ø 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 U5Ø (7D5), and through selector U7Ø (7A7) to the WRT DATA line. The R/W CLK-A signal clocks the serial data string on WRT DATA through U18 (10C3) to the selected drive.

During a read operation, R/W CLK-B (1ØB1) gates the serial data string (RD DATA) from the disk drive through U18 (1ØB4) and selector U7Ø (7A7) and into the SI input of U47 (7C5), creating a 16-bit parallel word. The bit ring-Ø line (BR-Ø, 7B8), which is derived from word clock U2Ø (8C7), then clocks this word into read buffer U48 and U51 (7C4). With the read buffer loaded, the IOP initiates a read to I/O port address 8Ø28H. Read I/O port address decoder U36 (5B2) decodes this address and drives RDC 28* low, which clocks the data word from the read buffer onto internal data bus lines IDAT-Ø through IDAT-F.
4.4.2.5 Sync Byte Comparator

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 before each sector ID and each data field to indicate to the board that data to be read are forthcoming (see Figure 3-2). The sync byte value is always 19H, except for the Shugart/Quantum drives, which use D9H for data fields.

During a read operation, sync byte decoder U54 and U73 (7B5) monitors the output of the SER/OES (U47 and U50, 7C5). When 19H is detected, the SYNC BYTE* signal is driven low, indicating the presence of the sync byte. The SYNC BYTE* signal and the next output of R/W CLK-B set the SYNC FND flip-flop (U57, 9C6), which activates word clock U20 (8C6), and activates the read/write logic (sheet 8).

4.4.2.6 Sector Identity Comparator

The 32-bit sector identity (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 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 IOP writes to I/O ports 8030H and 8038H, enabling the WDC30* and WDC38* lines, respectively. These lines initiate loading 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). The R/W CLK-B signal clocks the ID information, which is stored in the ID comparator, out of U22 (pins 7 and 9) bit-by-bit. Comparator U26 (9D2) compares the serial string of bits with the sector ID from the disk drive (RD-DATA). If the two sector ID's differ, ID no-compare line ID NCMPR* is activated; if the sectors are the same, ID NCMPR* is driven high. Selector U70 (7A7) OR's the ID NCMPR* and the ECC NCMPR* lines. The resulting ID-ECC NCMPR* line is latched into U37 (9B6). The Q output of U37, ID NCMPR-L, is transmitted to bit 6 of status register U10 (11C5). The IOP then reads the contents of the status register and checks the condition of bit 6. Bit 6 being high indicates that the record read from the disk was either not the record being searched for or that it had an ECC error; conversely, bit 6 being low indicates that the ID field compared and that there was no ECC error. The IOP then reads or writes the data portion of the record.

4.4.2.7 Error Checking Code Generator

The error checking code (ECC) logic: 1) generates (during a write operation) a four-byte ECC polynomial check sum that is appended to the ID field (format write) and the data field (normal write) of a record (see Figure 3-2), and 2) re-generates (during a read operation) the ECC polynomial check sum and compares it with the ECC field read from the disk record to ensure that correct data were read from the drive.
During the write operation, serial data (either an ID field or a data field) are transmitted from the SER/DES (7C5) through selector U7Ø (7A7) and into the ECC generator through pins 1 and 2 of U1Ø3 (7A6), where the ECC polynomial check sum 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 U7Ø (7A7) to the WRT DATA line for transmission to the disk. At ECC time (end of data field), the WRT XFER DLYD signal becomes low, inhibiting write data from being transferred through gate U68 (7B4). The ECC TIME* line then becomes low, causing the ECC polynomial check sum to be written onto the disk through U71 (7A3), U7Ø (7A7) and the WRT DATA line.

During a read operation, serial data (again either a sector ID or a data field) are read into the ECC generator through selector U7Ø (7A7) and into the SER/DES through U71 (7A3) and U7Ø. 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 U7Ø to the SER/DES for storage in RAM. If the difference is Ø, the ID-ECC NCMPR* line is driven high, indicating correct data or sector ID. If the result of the comparison is not Ø, the difference, called the "error syndrome", is used by the IOP to correct errors in a sector ID or data field (if correctable).

### 4.4.2.8 Status Register

The status register (U1Ø and U44, 11X5; and U9, 11B3) transmit status information from the selected disk drive, the iSBX interface, and various logic within the board drive interface circuitry to other logic of the board. When the IOP issues a read status command, or checks status as an internal operation, read decode enable lines RDC ØØ* and RDC Ø8* are activated, causing the contents of status registers U1Ø, U44, and U9, respectively, to be transferred onto the internal bus (IDAT-Ø through IDAT-F). The IOP then analyzes the status information for an internal operation or communicates the status of the data transfer operation to the host CPU through system memory (controller invocation block). Table 4-4 lists the status register bits. Refer to Chapter 3 for information on the status information transmitted to the host CPU.
FUNCTIONAL DESCRIPTION

Table 4-4. Status Register Bits

<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8000H (Upper Byte)</td>
<td>8000H (Lower Byte)</td>
<td>8008H (Lower Byte)</td>
</tr>
<tr>
<td></td>
<td>U44 (11D5)</td>
<td>U10 (11C5)</td>
<td>U9 (11B3)</td>
</tr>
<tr>
<td>F</td>
<td>Index</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>Drive Request</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Illegal Request</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Option Bit 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>Option Bit 00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>Interrupt 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Interrupt 00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>iSBX Board on J3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Time Out</td>
<td></td>
<td>Write Protected</td>
</tr>
<tr>
<td>6</td>
<td>ID No Compare</td>
<td></td>
<td>Track Zero</td>
</tr>
<tr>
<td>5</td>
<td>Bus Acknowledge</td>
<td></td>
<td>Vendor</td>
</tr>
<tr>
<td>4</td>
<td>Fault</td>
<td></td>
<td>Option Bit 11</td>
</tr>
<tr>
<td>3</td>
<td>Data Sync</td>
<td></td>
<td>Option Bit 01</td>
</tr>
<tr>
<td>2</td>
<td>Seek Complete</td>
<td></td>
<td>Interrupt 11</td>
</tr>
<tr>
<td>1</td>
<td>Ready</td>
<td></td>
<td>Interrupt 01</td>
</tr>
<tr>
<td>Ø</td>
<td>iSBX Board at J4</td>
<td></td>
<td>iSBX Board at J4</td>
</tr>
</tbody>
</table>

Note: Bit numbers C, B, A, 9, and 8 and 4, 3, 2, 1, and Ø are iSBX bus lines.

4.4.2.9 Line Drivers and Receivers

All serial data and high-speed clock signals transmitted between the iSBG 215G board and the drives use differential pair line drivers and receivers. The polarity on these lines is positive-true logic; that is, when the + side of the line is more positive than the - side of line, a positive logic 1 is being transmitted.

The board differential drivers (U16, 10X3) are referenced to Ø and +5 V. The board receivers that accept differential signals from other than Shugart SA1000 drives (U13, 10X6), are also referenced to Ø and +5 V. The receivers for 8-inch Shugart SA1000 drives (U15, 10X5) accept differential signals that are referenced to -5 and +5 V.

4.4.3 LOCAL MEMORY ORGANIZATION

As was described in the functional overview, the IOP addresses the read-only memory (ROM), random-access memory (RAM), iSBX I/O ports, and the hard-disk communications side of the board circuitry as local memory. Figure 4-11 is a map of this local memory. The following paragraphs describe the ROM, RAM, and I/O ports.

4-25
FUNCTIONAL DESCRIPTION

4.4.3.1 Read-Only Memory

The iSBC 215G board ROM, which contains the IOP disk control program, consists of two (8k x 8-bit) ROM devices (U87 and U88, 6X7). On any read from local memory in the range of 0000H to 3FFFH, chip-select decoder U65 (5B4) decodes address lines IADR–E and IADR–F and drives ROM chip-select line CSROM* low, enabling the ROM devices.

4.4.3.2 Random-Access Memory

The iSBC 215G board 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) drives RAM chip-select line CSRAM* low, enabling the RAM devices.

Figure 4–11. Local Memory Map

4.4.3.3 I/O Port Decoding

The IOP views the control 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 drive control devices, the IOP executes a read or a write to the device address. On any read or write to local memory in the range 8000H through 8038H, chip-select decoder U65 (5B4) pin 10 output is driven low.

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*,
FUNCTIONAL DESCRIPTION

Write I/O port address decoder U35 (5A2) is enabled. Decoder U35 or U36 then decodes local memory address lines IADR-3 through IADR-5 to select the desired drive control device. Table 4-5 shows the address of each local I/O port and its function.

The two iSBX bus connectors on the iSBC 215G board, J3 and J4, provide access to the iSBX bus 16 data lines and 3 address lines. The two iSBX channels provide 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 4-6).

When the IOP executes a read or write to one of these ports, chip-select decoder U6S-9 (SB4) activates the CSMMIO* line. Gate 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.

Table 4-5. Local I/O Ports

<table>
<thead>
<tr>
<th>Address</th>
<th>Read (U33 Enabled)</th>
<th>Write (U32 Enabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enable Line</td>
<td>Function</td>
</tr>
<tr>
<td>8000H</td>
<td>RDC00*</td>
<td>Read status</td>
</tr>
<tr>
<td>8008H</td>
<td>RDC08</td>
<td>Read status</td>
</tr>
<tr>
<td>8010H</td>
<td>RDC10</td>
<td>Read disk data bus</td>
</tr>
<tr>
<td>8018H</td>
<td>RDC18*</td>
<td>Raise IOP Channel 2 Attention</td>
</tr>
<tr>
<td>8020H</td>
<td>RDC20*</td>
<td>Read contents of counter 0</td>
</tr>
<tr>
<td>8022H</td>
<td>RDC20*</td>
<td>Read contents of counter 1</td>
</tr>
<tr>
<td>8024H</td>
<td>RDC20*</td>
<td>Read contents of counter 2</td>
</tr>
<tr>
<td>8026H</td>
<td>Not used</td>
<td></td>
</tr>
<tr>
<td>8028H</td>
<td>RDC28*</td>
<td>Read contents of read buffer</td>
</tr>
</tbody>
</table>
## FUNCTIONAL DESCRIPTION

### Table 4-5. Local I/O Ports (continued)

<table>
<thead>
<tr>
<th>Address</th>
<th>Enable Line</th>
<th>Function</th>
<th>Enable Line</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>80030H</td>
<td>RDC30*</td>
<td>Read vendor bits 3 and 4</td>
<td>WDC30*</td>
<td>Write sector ID to high comparator, start track format operation</td>
</tr>
<tr>
<td>8038H</td>
<td>Not used</td>
<td></td>
<td>WDC38*</td>
<td>Write sector ID to low comparator</td>
</tr>
</tbody>
</table>

### Table 4-6. iSBX™ Bus I/O Port Addresses

<table>
<thead>
<tr>
<th>Port</th>
<th>J3 Channel 0</th>
<th>J3 Channel 1</th>
<th>J4 Channel 0</th>
<th>J4 Channel 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>C070</td>
<td>C0B0</td>
<td>C0D0</td>
<td>C0E0</td>
</tr>
<tr>
<td>1</td>
<td>C072</td>
<td>C0B2</td>
<td>C0D2</td>
<td>C0E2</td>
</tr>
<tr>
<td>2</td>
<td>C074</td>
<td>C0B4</td>
<td>C0D4</td>
<td>C0E4</td>
</tr>
<tr>
<td>3</td>
<td>C076</td>
<td>C0B6</td>
<td>C0D6</td>
<td>C0E6</td>
</tr>
<tr>
<td>4</td>
<td>C078</td>
<td>C0B8</td>
<td>C0D8</td>
<td>C0E8</td>
</tr>
<tr>
<td>5</td>
<td>C07A</td>
<td>C0BA</td>
<td>C0DA</td>
<td>C0EA</td>
</tr>
<tr>
<td>6</td>
<td>C07C</td>
<td>C0BC</td>
<td>C0DC</td>
<td>C0EC</td>
</tr>
<tr>
<td>7</td>
<td>C07E</td>
<td>C0BE</td>
<td>C0DE</td>
<td>C0EE</td>
</tr>
</tbody>
</table>

***

4-28
5.1 INTRODUCTION

This chapter contains the various required diagrams and service and repair instructions for the iSBC 215G Winchester Disk Controller Board.

5.2 SERVICE DIAGRAMS

Figure 5-1 is the parts location diagram; Figure 5-2 shows jumper locations; Figure 5-3 is the schematic diagram. In Figure 5-3 a signal mnemonic that is followed by an asterisk or slash (for example, BHEN*) is active low; a signal mnemonic without a star is active high.

5.3 SERVICE AND REPAIR ASSISTANCE

Customers within the United States can obtain service and repair assistance by contacting the Intel Product Service Center 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 Center, you should have the following information available:

1. Date that you received the product.
2. Complete part number of the product (including the dash number).
3. Serial number of the product (usually stamped on the component side of the board).
4. Shipping and billing address.
5. Purchase order number (for billing purposes if your Intel product warranty has expired).
6. Extended warranty agreement information (if applicable).

Regional Telephone Numbers:

Western Region: 602-869-4951  
Midwest Region: 602-869-4392  
Eastern Region: 602-869-4045  
International: 602-869-4862  
TWX Number: 910-951-1330
SERVICE INFORMATION

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

Use the original factory packing material (if possible) when preparing the product for shipment to the Repair Center. If the original material is not available, wrap the product in cushioning material such as Air Cap TH-24Ø (manufactured by the Sealed Air Corporation, Hawthorne, NJ), enclose it in a heavy-duty corrugated shipping carton, and label it FRAGILE to ensure careful handling. Ship only to the address specified by the Service Center personnel.
Table 5-1. Jumper Connections

<table>
<thead>
<tr>
<th>Jumper Number</th>
<th>Schematic Sheet Number</th>
<th>Default Connection¹</th>
<th>Signal Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>12</td>
<td>W1-1 -- 2</td>
<td>CMD BUS ENB*</td>
</tr>
<tr>
<td>W2</td>
<td>13'</td>
<td>W2-1 -- 2</td>
<td>VENDOR Ø</td>
</tr>
<tr>
<td>W3</td>
<td>11</td>
<td>W3-1 -- 2</td>
<td>EXTRØ</td>
</tr>
<tr>
<td>W4</td>
<td>11</td>
<td>W4-1 -- 2</td>
<td>EXTR1</td>
</tr>
<tr>
<td>W5</td>
<td>1Ø</td>
<td>W5-1 -- 2</td>
<td>RDØ–</td>
</tr>
<tr>
<td>W6</td>
<td>1Ø</td>
<td>W6-1 -- 2</td>
<td>RDØ+</td>
</tr>
<tr>
<td>W7</td>
<td>1Ø</td>
<td>W7-1 -- 2</td>
<td>RDCLØ–</td>
</tr>
<tr>
<td>W8</td>
<td>1Ø</td>
<td>W8-1 -- 2</td>
<td>RDCLØ+</td>
</tr>
<tr>
<td>W9</td>
<td>13</td>
<td>--</td>
<td>TRIPOLAR*</td>
</tr>
<tr>
<td>W1Ø</td>
<td>13</td>
<td>W1Ø-1 -- 2</td>
<td>RADIAL SELECT</td>
</tr>
<tr>
<td>W11</td>
<td>12</td>
<td>W11-1 -- 3</td>
<td>OPØ1</td>
</tr>
<tr>
<td>W12</td>
<td>12</td>
<td>--</td>
<td>OP1Ø/OP11</td>
</tr>
<tr>
<td>W13</td>
<td>12</td>
<td>W13-1 -- 3</td>
<td>RDGATE</td>
</tr>
<tr>
<td>W14</td>
<td>12</td>
<td>W14-1 -- 3</td>
<td>AM CNTRL</td>
</tr>
<tr>
<td>W15</td>
<td>13</td>
<td>W15-1 -- 2</td>
<td>SHUGART</td>
</tr>
<tr>
<td>W16</td>
<td>8</td>
<td>W16-1 -- 3</td>
<td>INDEX SECTOR*</td>
</tr>
<tr>
<td>W17</td>
<td>11</td>
<td>W17-1 -- 2</td>
<td>INDEX B*</td>
</tr>
<tr>
<td>W18</td>
<td>3</td>
<td>W18-1 -- 2</td>
<td>ANYRQST</td>
</tr>
<tr>
<td>W19</td>
<td>3</td>
<td>W19-C -- 5</td>
<td>INT5</td>
</tr>
<tr>
<td>W20</td>
<td>1</td>
<td>--</td>
<td>–12 V</td>
</tr>
<tr>
<td>W21</td>
<td>1</td>
<td>W21-1 -- 3</td>
<td>–5 V</td>
</tr>
<tr>
<td>W22</td>
<td>1Ø</td>
<td>W22-1 -- 3</td>
<td>RDCL</td>
</tr>
<tr>
<td>W23</td>
<td>3</td>
<td>W23-1 -- 2</td>
<td>CBRO*</td>
</tr>
<tr>
<td>W24</td>
<td>13</td>
<td>--</td>
<td>DREQØ</td>
</tr>
<tr>
<td>W25</td>
<td>1</td>
<td>W25-1 -- 2</td>
<td>GND</td>
</tr>
<tr>
<td>W26</td>
<td>12</td>
<td>--</td>
<td>VENDOR 1</td>
</tr>
<tr>
<td>W27</td>
<td>11</td>
<td>--</td>
<td>VENDOR 2</td>
</tr>
<tr>
<td>W28</td>
<td>3</td>
<td>W28-1 -- 2</td>
<td>BPRØ*</td>
</tr>
<tr>
<td>W29</td>
<td>2</td>
<td>W29-5 -- 12</td>
<td>ADRS*</td>
</tr>
<tr>
<td>W29</td>
<td>2</td>
<td>W29-8 -- 9</td>
<td>ADRS*</td>
</tr>
<tr>
<td>W3Ø</td>
<td>2</td>
<td>W3Ø-1 -- 2Ø</td>
<td>16 BIT SYS BUS</td>
</tr>
<tr>
<td>W312</td>
<td>6</td>
<td>W31-1 -- 2</td>
<td>VCC</td>
</tr>
<tr>
<td>W32</td>
<td>4</td>
<td>--</td>
<td>LOCK*</td>
</tr>
<tr>
<td>W33</td>
<td>11</td>
<td>W33-1 -- 3</td>
<td>INDEX*</td>
</tr>
<tr>
<td>W34</td>
<td>11</td>
<td>W34-1 -- 2</td>
<td>SKCOM*</td>
</tr>
<tr>
<td>W35</td>
<td>11</td>
<td>W35-1 -- 2</td>
<td>RDY*</td>
</tr>
<tr>
<td>W36</td>
<td>9</td>
<td>--</td>
<td>ROFØ/F</td>
</tr>
<tr>
<td>W37</td>
<td>9</td>
<td>--</td>
<td>VENDOR 3</td>
</tr>
<tr>
<td>W38</td>
<td>9</td>
<td>--</td>
<td>VENDOR 4</td>
</tr>
</tbody>
</table>

Note: 1. Default connections pertain to the iSBC 215G board used in conjunction with an X3T9/1226 ANSI drive.

2. Default connection for iSBC 215G board.
WAKE-UP ADDRESS
8/16 SYSTEM DATA BUS AND
8/16 I/O PORT BUS JUMPER TABLE

<table>
<thead>
<tr>
<th>JUMPER</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>To</td>
</tr>
<tr>
<td>W29-1</td>
<td>W29-16</td>
</tr>
<tr>
<td>W29-2</td>
<td>W29-15</td>
</tr>
<tr>
<td>W29-3</td>
<td>W29-14</td>
</tr>
<tr>
<td>W29-4</td>
<td>W29-13</td>
</tr>
<tr>
<td>W29-5</td>
<td>W29-12</td>
</tr>
<tr>
<td>W29-6</td>
<td>W29-11</td>
</tr>
<tr>
<td>W29-7</td>
<td>W29-10</td>
</tr>
<tr>
<td>W29-8</td>
<td>W29-9</td>
</tr>
<tr>
<td>W30-1</td>
<td>W30-20</td>
</tr>
<tr>
<td>W30-2</td>
<td>W30-19</td>
</tr>
<tr>
<td>W30-3</td>
<td>W30-18</td>
</tr>
<tr>
<td>W30-4</td>
<td>W30-17</td>
</tr>
<tr>
<td>W30-5</td>
<td>W30-16</td>
</tr>
<tr>
<td>W30-6</td>
<td>W30-15</td>
</tr>
<tr>
<td>W30-7</td>
<td>W30-14</td>
</tr>
<tr>
<td>W30-8</td>
<td>W30-13</td>
</tr>
<tr>
<td>W30-9</td>
<td>W30-12</td>
</tr>
<tr>
<td>W30-10</td>
<td>W30-11</td>
</tr>
</tbody>
</table>
NOTES UNLESS OTHERWISE SPECIFIED

1. RESISTANCE VALUES ARE IN OHMS, 1/4 W, 5%. CAPACITANCE VALUES ARE IN MICROFARADS, ±20% 0.5V, 50V.

2. EVEN PINS ON J1-10 THRU J15, J17, J18, J19, J20, J21 ARE TIED TO GROUND.

3. EVEN PINS ON J2-4, 10, 12, 20, 32, ARE TIED TO GROUND.

4. EVEN PINS ON J3-40-2, J21, J35, J36-4, ARE TIED TO GROUND.

5. RP2-X DENOTES 1/8W, 5% RESISTORS.

6. RP6-X DENOTES 1/16W, 5% RESISTORS.

7. NOT INSTALLED.

8. ANY ASTERISK FOLLOWING A SIGNAL NAME DENOTES UNKNOWNS.

POWER & GROUND LOCATION CHART

REFERENCE DESIGNATIONS

DEVICE TYPE

POWER PINS

REFERENCE DEVICES

POWER PINS

GND - 10V - 12V

-5V 12V

-5V 12V

GND - 10V - 12V
## Vendor Configuration Table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>CMD Bus END</td>
<td>1-3</td>
<td>1-3</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-3</td>
<td>1-2</td>
<td>1-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W2</td>
<td>VENDOR 0</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W5</td>
<td>RD0-</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-3</td>
<td>1-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W6</td>
<td>RD0+</td>
<td>1-3</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W7</td>
<td>RCCL0-</td>
<td>1-3</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W8</td>
<td>RCCL0+</td>
<td>1-3</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W9</td>
<td>RD0 CL</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W10</td>
<td>RD0 CL</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W12</td>
<td>INDEX SELECT</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-3</td>
<td>1-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W13</td>
<td>IN0 GATE</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W14</td>
<td>IN0 CL</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W15</td>
<td>INDEX SELECT</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W17</td>
<td>INDEX SELECT</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td>1-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W20</td>
<td>RD0 OUT</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W21</td>
<td>VENDOR 2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W22</td>
<td>VENDOR 3</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W23</td>
<td>VENDOR 4</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W24</td>
<td>SECTOR</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W25</td>
<td>SKCOMP</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W26</td>
<td>RDY OUT</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W27</td>
<td>VEN0 12V</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W28</td>
<td>VEN0 5V</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td>1-2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
As shown in Table 3-5, bytes 26 through 29 of the I/O parameter block can be set to allow use of the ANSI X3T9/1126 interface. With this feature, present and future ANSI X3T9 drives not fully supported by the iSBC 215G board can still be used by transferring the formatting information to the board at initialization time. This information is contained in the four bytes in the general address pointer of the I/O parameter block and must be there only for initialization. To calculate the values of the four bytes, the user must have available the recommended data format from the disk drive product specification. As an example, assume a data format as shown in Figure A-1.

The four bytes to be calculated \((n_0, n_1, n_2, n_3)\) are shown below:

- \(n_1\)
  
  If the drive is hard-sectored (that is, the drive sends sector pulses), byte 1 will be \(0\). If the drive is soft-sectored (that is, address marks are on the disk), byte 1 is calculated as follows:

  \[
  n_1 = \frac{24 + P2 + PAD}{2}
  \]

  Where \(P2\) = preamble 2 and consists of bytes of \(0\)'s for the purpose of synchronizing the phase lock loop; and \(PAD\) = remaining bytes before sector end, which provides a guard zone between sectors to account for speed and clock tolerances.
ANSI INTERFACE PROGRAMMING

- \(n_2\) For hard-sectored drives, \(n_2\) is calculated as follows:
  \[
  n_2 = \frac{P_1 - 5}{2}
  \]

  For soft-sectored drives, \(n_2\) is calculated as follows:
  \[
  n_2 = \frac{P_1 - 2}{2}
  \]

- \(n_3\) For hard- and soft-sectored drives:
  \[
  n_3 = \frac{P_2 - 14}{2}
  \]

- \(n_\emptyset\) The total overhead per sector \(n_\emptyset\) is thus:
  - Soft Sectored Drive \(n_\emptyset = 2n_2 + 2n_3 + 2\emptyset\)
  - Hard Sectored Drive \(n_\emptyset = 2n_2 + 2n_3 + 23\)

The user can easily calculate the number of sectors per track for a given number of bytes per sector data size.

\[
\frac{\text{number of sectors}}{\text{track}} = \frac{\text{number of bytes per track}}{\text{number of bytes per sector} + n_\emptyset}
\]

Refer to Table 2-4 (the jumper configuration table) for the correct configuration depending on whether the drive is soft-sectored or hard-sectored.
B.1 INTRODUCTION

The information contained in this appendix is provided to illustrate various methods of implementing data transfers between one or more host CPU's and the iSBC 215G board. The flow charts illustrate the handshake procedures required between a host CPU and the board. User sequences are shown for both 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 CPU would run to direct data transfer between the host and the iSBC 215G board. The program is written in MCS-86 Macro Assembler language, and illustrates the data structure that the iSBC 215G board requires and shows a few simple disk operations drivers.

B.2 SINGLE USER SEQUENCE

The flow chart in Figure B-1 shows the handshake sequence between a single host CPU and the iSBC 215G board for basic data transfer operations (with no overlapping seeks). Note that communication between the host and the board is through the status semaphore and operation system bytes of the controller invocation block.

B.3 SINGLE USER SEQUENCE WITH OVERLAPPING SEEKS

The flow chart in Figure B-2 shows the handshake sequence between a single host CPU and the iSBC 215G board for data transfer operations that user overlapping seeks.

B.4 MULTI-USER SEQUENCE

The flow chart in Figure B-3 shows the handshake sequence between a host CPU and the iSBC 215G board when more than one CPU is transferring data between the disk drives through the same board (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.
B.5 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.

Start 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 B-1. Flow Chart for Single User Handshake Sequence Without Overlapping Seeks
Figure B-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 B-2. Flow Chart for Single User Handshake Sequence With Overlapping Seeks (continued)
**LOCK THE MULTIBUS'' INTERFACE**

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

**IS COMM. SEMA. = 0**

Set up command and parameters for desired data transfer operation.

**SET COMM. SEMA. TO NON-ZERO AND UNLOCK MULTIBUS'' INTERFACE**

Initiate data transfer operation.

**SET UP I/O PARAMETER BLOCK**

Continue with other processing tasks.

**WRITE '01' TO WAKE-UP I/O PORT**

**BRANCH TO OTHER PROCESSING TASKS**

**OP COMPLETE INTERRUPT**

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

**IS OP. STATUS OK?**

Relinquish controller of disk controller.

**SET ST. SEMA. = 0**

**SET COMM. SEMA. = 0**

**GO TO ERROR HANDLING ROUTINE**

**FINISH**

---

**Figure B-3. Flow Chart for Multi-User Handshake Sequence**
Figure B-3. Flow Chart for Multi-User Handshake Sequence (continued)
MCS-86 MACRO ASSEMBLER

ISIS-II MCS-86 MACRO ASSEMBLER V2.1 ASSEMBLY OF MODULE EXMPRC
OBJECT MODULE PLACED IN :FI:EXMPRC.OBJ
ASSEMBLER INVOKED BY: ASMR :FI:EXMPRC.MMD DATE(10/27/80) XREF OBJLOC

LOC OBJ

1 5816 :TITLE(ISBC 215 DISK CONTROLLER PROGRAMMING EXAMPLE)
2 5817
3 5818
4 5819
5 5820
6 5821
7 5822
8 5823
9 5824
10 5825
11 5826
12 5827
13 5828
14 5829
15 5830
16 5831
17 5832
18 5833
19 5834
20 5835
21 5836
22 5837
23 5838
24 5839
25 5840
26 5841
27 5842
28 5843
29 5844
30 5845
31 5846
32 5847
33 5848
34 5849
35 5850
36 5851
37 +1 $INCLUDE(:FI:COMBLK.MMD)
38 +1 $DELETE TITLE(ISBC 215 COMMUNICATION BLOCKS)

HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

THIS PROGRAM ILLUSTRATES THE DATA STRUCTURES REQUIRED BY THE ISBC 215
DISK CONTROLLER. A FEW SIMPLE DISK OPERATION DRIVERS ARE ALSO SHOWN.

THE HARDWARE CONFIGURATION SUPPORTED IS:

1. ISBC 86/12A HOST CPU
2. 20 BIT SYSTEM MEMORY ADDRESS WIDTH
3. 16 BIT SYSTEM DATA BUS WIDTH
4. 16 BIT SYSTEM I/O ADDRESS WIDTH
5. ISBC 215

A. WAKE UP ADDRESS (WUA) AT I/O PORT 0635H
B. INTERRUPT 5
C. -12 VOLTS INPUT
D. RELINQUISH BUS CONTROL ON ANY REQUEST

FOR (2), PROGRAMMING OF DATA TRANSFERS MUST TAKE THIS INTO ACCOUNT, e.g., THERE
IS NO WRAPAROUND IN SEGMENTS IF MORE THAN 64K BYTES ARE TRANSFERRED.

FOR (3), SWITCH S2-1 IS CLOSED.
FOR (4), SWITCH S2-2 IS CLOSED.
FOR (5a), SWITCHES S1-6, S1-7, S2-5, S2-6, S2-8, AND S2-10 ARE CLOSED, THE
REMAINING ADDRESS SELECT SWITCHES ARE OPEN.
FOR (5b), W19-C CONNECTS TO W19-D; INTERRUPT VECTORS MUST BE SET UP PROPERLY.
FOR (5c), W21-1 CONNECTS TO W21-3
FOR (5d), W2-1 CONNECTS TO W2-2.

SINCLUDE(:FI:COMBLK.MMD)

## Handshake Sequences and Example Host Processor Disk Control Program

**MCS-86 Macro Assembler**

### ISBC 215 Communication Blocks

<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>46</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>48</td>
<td></td>
</tr>
</tbody>
</table>

**The SCB tells the 8089 on the ISBC 215 the width of the 8089's local bus and points to the CCB.**

**Memory Address of the SCB is equal to the I/O wake-up address (WUA) of the ISBC 215 multiplied by 16.**

```assembly
WUA EQU 0635H ; Wake-up address I/O port number
```

**SCB SEGMENT**

- **SCB** label
- **SOCDB** label
- **DB**
- **CCBPTR** label

```assembly
SCBSEG SEGMENT
SCB LABEL FAR
SOC DB 01H
CCBPTR DD CCB
SCBSEG ENDS
```

**CCB SEGMENT**

- **II. CCB**
- **CH1PC** label
- **CH2PC** label

```assembly
CCBSEG SEGMENT
CCB LABEL FAR
CCW1 DB 01H
BSYFLG1 DB 00H
CCPTR DD CH1PC
CCW2 DB 01H
BSYFLG2 DB 00H
CH2PC
```

**Comments:**

- The SCB must be contiguous.
- The CCB contains the control bytes, busy flags, and pointers to the starting addresses of the channel programs for the 8089.
- The CCB must be contiguous.
- The CCB contains the control bytes, busy flags, and pointers to the starting addresses of the channel programs for the 8089.

---

**0000 0400----**

- **R**
- **80**
- **81**
- **82**
- **83**
- **84**
- **85**
- **86**
- **87**
- **88**
- **89**
- **90**
- **91**
- **92**
- **93**
- **94**
- **95**
---

**CCCSEG ENDS**

$EJECT
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER  ISBC 215 COMMUNICATION BLOCKS

LOC   OBJ

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 95 | *******
| 96 | III. CIB
| 97 | *******
| 98 |
| 99 | THIS BLOCK CONTAINS GENERAL PURPOSE COMMAND AND STATUS BYTES, SEMA-
| 100 | PHORES, AND POINTERS TO ALLOW THE USE OF THE ISBC 215 IN A MULTI-
| 101 | PROCESSOR/MULTI-PROCESSING SYSTEM.
| 102 |
| 103 | CIBSEG SEGMENT ; CIB MUST BE CONTIGUOUS
| 104 |
| 105 | CIB LABEL FAR
| 106 |
| 107 | CIBCMD DB 00H ; CIB COMMAND BYTE NOT USED BY ISBC 215
| 108 | CMDSEN DB 00H ; COMMAND BYTE SEMAPHORE
| 109 | STSSN DB 00H ; STATUS BYTE SEMAPHORE
| 110 |
| 111 | CHIPC LABEL FAR
| 112 |
| 113 | IOPBSEG DW IOPBSEG ; STARTING ADDRESS OF CH. 1 PROGRAM
| 114 |
| 115 |
| 116 | CIBSEG ENDS
| 117 |
| 118 |
| 119 | IV. IOPB
| 120 |
| 121 |
| 122 | THIS BLOCK CONTAINS THE DEVICE DEPENDENT CONTROL INFORMATION FOR
| 123 | THE
| 124 | ISBC 215 CONTROLLER.
| 125 |
| 126 | IOPBSEG SEGMENT ; IOPB MUST BE CONTIGUOUS
| 127 |
| 128 | IOPB LABEL FAR
| 129 |
| 130 | AACTCNT DD 0000H ; ACTUAL TRANSFER COUNT (32 BIT INTEGER)
| 131 | UNIT DB 00H ; UNIT NUMBER (0 <- UNIT <- OFH)
| 132 | FUNC DB 00H ; FUNCTION CODE (0 <- FUNCTION <- OFH)
| 133 | MODIFY DW 0000H ; MODIFIER WORD
| 134 | CYLINDR DW 0000H ; CYLINDER NUMBER
| 135 | HEAD DB 00H ; HEAD NUMBER
| 136 | SECTOR DB 00H ; SECTOR NUMBER
| 137 | BUFOFF DW 0000H ; POINTER TO DATA BUFFER
| 138 | BUFSIZE DW 0000H
| 139 | REQCNT DD 0000H ; REQUESTED TRANSFER COUNT (INTEGER)
| 140 |
| 141 |
| 142 | IOPBSEG ENDS
| 143 |

$INCLUDE(:FI:INITBL.MMD)

$EJECT TITLE (DISK DRIVE INITIALIZATION TABLES)
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER DISK DRIVE INITIALIZATION TABLES

LOC OBJ LINE SOURCE
= 1 146 ;lemn--------------------------------------------
= 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 |
= 1 153 | THAT ARE USED BY THE INITIALIZATION ROUTINE. THEY MUST |
= 1 154 | BE MODIFIED TO REFLECT THE PARTICULAR DRIVES BEING |
= 1 155 | USED WITH THE ISBC 215 DISK CONTROLLER. |
= 1 156 |
= 1 157 |
= 1 158 |
= 1 159 |
= 1 160 |
= 1 161 | BYTES PER SECTOR | MAXIMUM SECTORS PER TRACK |
= 1 162 |
= 1 163 ; | 128 | 54 |
= 1 164 |
= 1 165 ; | 512 | 17 |
= 1 166 |
= 1 167 |
= 1 168 |
= 1 169 |
= 1 170 | INITBSEG 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 01 = 1 177 DB 31 ; NUMBER OF SECTORS PER TRACK |
0005 0001 = 1 178 DW 256 ; NUMBER OF BYTES PER SECTOR |
0007 03 = 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 03 = 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

B-11
**HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR DISK CONTROL PROGRAM**

**MCS-86 MACRO ASSEMBLER DISK DRIVE INITIALIZATION TABLES**

<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>209</td>
<td>+1</td>
<td>210</td>
<td>; $8$ FLEXIBLE DISK DRIVES</td>
</tr>
<tr>
<td>211</td>
<td>+1</td>
<td>212</td>
<td>; MAXIMUM SECTORS PER TRACK</td>
</tr>
<tr>
<td>213</td>
<td>+1</td>
<td>214</td>
<td>; BYTE PER SECTOR</td>
</tr>
</tbody>
</table>
| 215 | +1  | 216 | 128 | 26 (FM)
| 217 | +1  | 218 | 256 | 26 (MFM)
| 219 | +1  | 220 | 512 | 15 (MFM)
| 221 | +1  | 222 | 1024 | 8 (MFM)
| 223 | +1  | 224 | | |
| 225 | +1  | 226 | DB 0 | NUMBER OF FIXED READ/WRITE SURFACES |
| 227 | +1  | 228 | DB 2 | NUMBER OF REMOVABLE R/W SURFACES |
| 229 | +1  | 230 | DB 256 | NUMBER OF SECTORS PER TRACK |
| 231 | +1  | 232 | | |
| 233 | +1  | 234 | DW 77 | NUMBER OF CYLINDERS |
| 235 | +1  | 236 | DB 0 | NUMBER OF FIXED READ/WRITE SURFACES |
| 237 | +1  | 238 | DB 2 | NUMBER OF REMOVABLE R/W SURFACES |
| 239 | +1  | 240 | DB 26 | NUMBER OF SECTORS PER TRACK |
| 241 | +1  | 242 | DW 128 | NUMBER OF SECTORS PER TRACK |
| 243 | +1  | 244 | DW 0 | NUMBER OF CYLINDERS |
| 245 | +1  | 246 | DW 00H | NUMBER OF FIXED READ/WRITE SURFACES |
| 247 | +1  | 248 | DW 00H | NUMBER OF REMOVABLE R/W SURFACES |
| 249 | +1  | 250 | DW 00H | NUMBER OF SECTORS PER TRACK |
| 251 | +1  | 252 | DW 00H | NUMBER OF CYLINDERS |
| 253 | +1  | 254 | DW 00H | NUMBER OF FIXED READ/WRITE SURFACES |
| 255 | +1  | 256 | DW 00H | NUMBER OF REMOVABLE R/W SURFACES |
| 257 | +1  | 258 | DW 00H | NUMBER OF SECTORS PER TRACK |
| 259 | +1  | 260 | DW 00H | NUMBER OF CYLINDERS |
| 261 | +1  | 262 | DW 00H | NUMBER OF FIXED READ/WRITE SURFACES |

---

$\text{INITSSEG ENDS}$

$\text{B-12}$
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER DATA SEGMENT

LOC OBJ LINK 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 ;
+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 ;
+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 ;
+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 ;
+1 307 ERRSTS DW 0000H ; ERROR STATUS WORD
000C 0000 +1 308 SFERST DB 00H ; SOFT ERROR STATUS BYTE
000E 00 +1 309 DESCYL DW 0000H ; DESIRED CYLINDER
000F 0000 +1 310 DESHD DB 00H ; DESIRED HEAD
0111 00 +1 311 DESSEC DB 00H ; DESIRED SECTOR
0112 0000 +1 312 ACTCYL DW 0000H ; ACTUAL CYLINDER + FLAG BITS
0113 00 +1 313 ACTHD DB 00H ; ACTUAL HEAD
0114 00 +1 314 ACTSEC DB 00H ; ACTUAL SECTOR
0117 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 ;
+1 31A 00 +1 320 LSTSTS DB 00H
+1 31B ;
+1 319 90 +1 321 EVEN
+1 322 ;
+1 324 ENDDAT LABEL FAR ; END OF DATA SEGMENT
+1 325 ;
---
+1 326 DATASEG ENDS
+1 327 ;
+1 328 +1 $INCLUDE (:F:USER.HMD)
+1 329 +1 $EJECT TITLE (SYSTEM DEPENDENT INITIALIZATION)
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER

SYSTEM DEPENDENT INITIALIZATION

LOC OBJ

=1 330 ; -----------------------------------------------------------
=1 331 ; | SYSTEM DEPENDENT INITIALIZATION |
=1 332 ; | -----------------------------------------------------------
=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 INITIALIZATIONS ARE PERFORMED |
=1 340 ; |
=1 341 ; |
=1 342 ; |
=1 343 ; |
=1 344 ; | INTERRUPT VECTOR DEFINITION |
=1 345 ; | -----------------------------------------------------------
=1 346 ; |
=1 347 ; | INTRPT EQU 5 ; ISBC 220 INTERRUPT NUMBER |
=1 348 ; |
=1 349 ; | 80000 SEGMENT AT 0000H |
=1 350 ; | INTERRUPT VECTORS ARE FROM ABSOLUTE |
=1 351 ; | ADDRESSES 00000H TO 00FF0H |
=1 352 ; |
=1 353 ; | ORG 80H + 4*INTRPT |
=1 354 ; | LOCATION OF INTERRUPT VECTOR WITH |
=1 355 ; | ISBC 957A Firmware |
=1 356 ; |
=1 357 ; | INTRIP DW 0000H |
=1 358 ; | - INSTRUCTION POINTER |
=1 359 ; |
=1 360 ; | INTRCS DW 0000H |
=1 361 ; | - CODE SEGMENT |
=1 362 ; |
=1 363 ; |
=1 364 ; | STACK SEGMENT |
=1 365 ; |
=1 366 ; | 66H DB 64 DUP(OOH) ; ALLOW 64 BYTES FOR STACK |
=1 367 ; |
=1 368 ; |
=1 369 ; | ENDS |
=1 370 ; |
=1 371 ; | STACK ENDS |
=1 372 ; |
=1 373 ; | STACK AND INTERRUPT CONFIGURATION ROUTINE |
=1 374 ; |
=1 375 ; |
=1 376 ; | USERSEG SEGMENT |
=1 377 ; |
=1 378 ; | PUBLIC CONFIG |
=1 379 ; |
=1 380 ; | ASSUME DS:8000 |
=1 381 ; |
=1 382 ; |
=1 383 ; |
=1 384 ; |
=1 385 ; |
=1 386 ; |
=1 387 ; |
=1 388 ; |
=1 389 ; |
=1 390 ; |
=1 391 ; |
=1 392 ; |
=1 393 ; |
=1 394 ; |
=1 395 ; |
=1 396 ; |
=1 397 ; |
=1 398 ; |
=1 399 ; |
=1 400 ; | SBC215DRIVER SEGMENT |
=1 401 ; |
=1 402 ; | ASSUME CS:SBC215DRIVER |
=1 403 ; |
=1 404 ; |
=1 405 ; | INCLUDE(:FI:RESET.MMD) |
=1 406 ; |
=1 407 ; | INCLUDE(:FI:RESET.MMD) |

B-14
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER

LOC OBJ LINE SOURCE
1 406 ;----------------------------------------------------------
1 407 ; | CONTROLLER RESET ROUTINE |
1 408 ; | 1 |
1 409 ; | 1 |
1 410 ; | 1 |
1 411 ; | 1 |
1 412 ; | 1 |
1 413 ; | RES215 SETS UP THE COMMUNICATION BLOCKS FOR THE 1SCB 215, LINKS THEM |
1 414 ; | TOGETHER AND GIVES A RESET, CLEAR RESET, CHANNEL ATTENTION SEQUENCE TO |
1 415 ; | THE CONTROLLER. THIS CAUSES THE 8089 ON THE CONTROLLER TO SET UP ITS |
1 416 ; | INTERNAL POINTER TO THE CCB BY THREADING DOWN THE LINKS STARTING WITH |
1 417 ; | THE SWITCHES ON THE CONTROLLER. SUBSEQUENT CA"S WILL CAUSE THE 8089 TO |
1 418 ; | FETCH ITS POINTERS STARTING AT THE CCB. |
1 419 ; | IF THE CH. 1 BUSY FLAG IS NOT CLEARED WITHIN A "REASONABLE" AMOUNT OF TIME, |
1 420 ; | THEN THE 1SCB 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 ; | STRORES. |
1 424 ; | THE SYSTEM INTERRUPT LOGIC AND VECTORS FOR THE CONTROLLER ARE ASSUMED TO BE |
1 425 ; | CONFIGURED BY AN EXTERNAL PROGRAM. |
1 426 ; | INPUT DATA: |
1 427 ; | NONE |
1 428 ; | OUTPUT DATA: |
1 429 ; | CARRY FLAG: + 0 IF RESET OKAY |
1 430 ; | 1 IF CH. 1 BUSY FLAG NOT RESET (NOT RESPONDING) |
1 431 ; | -------------------------------------------------------------|
1 432 ; | PUBLIC RES215 |
0000 433 RES215 PROC FAR |
0000 434 |
1 435 ; 0000 50 |
1 436 ; | 1 |
1 437 ; | 1 |
1 438 ; | 1 |
1 439 ; | 1 |
0005 440 | RES215 PROC FAR |
0008 441 |
1 442 ; 0001 53 |
1 443 ; | 1 |
1 444 ; | 1 |
1 445 ; | 1 |
1 446 ; | 1 |
1 447 ; | 1 |
1 448 ; | SCB |
1 449 ; | 1 |
1 450 ; | 1 |
0005 451 | ASSUME DS:SCBSEG |
0008 452 |
1 453 ; 000A C7000000100 |
1 454 ; | 1 |
1 455 ; | 1 |
1 456 ; | 1 |
1 457 ; | 1 |
0010 458 | MOV AX,SCBSEG |
0018 459 |
1 460 ; 001C C5060200 |
1 461 ; | 1 |
1 462 ; | 1 |
1 463 ; | 1 |
1 464 ; | 1 |
1 465 ; | 1 |
1 466 ; | 1 |
1 467 ; | 1 |
1 468 ; | 1 |
1 469 ; | 1 |
1 470 ; | 1 |
0020 471 | GET POINTER TO SCB |
0026 472 |
1 473 ; 002C C7004000040 |
1 474 ; | 1 |
1 475 ; | 1 |
1 476 ; | 1 |
1 477 ; | 1 |
1 478 ; | 1 |
1 479 ; | 1 |
0032 480 | MOV AX,CIBSEG |
0035 481 |
1 482 ; 0055 C7002000000 |
1 483 ; | 1 |
1 484 ; | 1 |
1 485 ; | 1 |
1 486 ; | 1 |
1 487 ; | 1 |
1 488 ; | 1 |
0061 489 | GET POINTER TO CIB |
0067 490 |
1 491 ; 1 $EXIT
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER

LINE SOURCE
LOC OBJ

=1 481 ; CLEAR OUT DATA SEGMENT
=1 482 ; ASSUME DS:DATASEG
=1 483 ; GET POINTER TO DATA SEGMENT

006D B8---- R = 1 484 MOV AX,DATASEG
0070 B8D8 = 1 485 MOV DS,AX
0072 B9D000 = 1 486 MOV CX,(OFFSET ENDDAT)/2
0075 B80000 = 1 487 MOV BX,0000H
0078 C7070000 = 1 488 CLRLP: MOV WORD PTR [BX],0000H
007C 43 = 1 489 INC BX
007D 43 = 1 490 INC BX
007E EOF8 = 1 491 LOOPNE CLRLP
007F 44 = 1 492 ; DONE?
0080 BA3506 = 1 493 ; NO--CLEAR ANOTHER WORD
0081 B002 = 1 494 ; YES--INITIALIZE COMMUNICATION LINKS
0085 EE = 1 495 ; OUTPUT RESET/CLEAR RESET/CHANNEL
0086 B000 = 1 496 ; ATTENTION TO CONTROLLER
0088 EE = 1 497 MOV DX,WUA
0089 B001 = 1 498 MOV AL,02H
008B EE = 1 499 OUT DX,AL
008D B000 = 1 500 MOV AL,00H
; GET CLEAR RESET COMMAND BYTE
008F B000 = 1 501 OUT DX,AL
0091 B90010 = 1 502 MOV AL,01H
0093 EE = 1 503 OUT DX,AL
0095 F6060100FF = 1 504 ASSUME DS:CCBSEG
0097 7403 = 1 505 MOV AX,CCBSEG
0099 F8 = 1 506 MOV DS,AX
; GET POINTER TO CCB

=1 507 ; (OTHER IMPLEMENTATIONS OF RES215 COULD
=1 508 ; INITIALIZE OTHER DEVICES WHILE THE
=1 509 ; ISBC 215 DOES ITS RESET SEQUENCE HERE)
=1 509 ; SET TIME-OUT COUNTER
=1 509 ; CLEAR CARRY FLAG
009E F9 = 1 510 MOV CX,1000H
009F F8 = 1 511 CLC
00A0 F6060100FF = 1 512 RESLP: TEST BSYFLG1,00FFH
00A2 7403 = 1 513 ; CHECK CH. I BUSY FLAG:
00A4 F8 = 1 514 JZ RESDN
00A6 007F = 1 515 ; ZERO FLAG = BSYFLG1 & FFH
00A8 F7 = 1 516 LOOPNE RESLP
00A9 F9 = 1 517 ; BUSY FLAG CLEARED?
00AB 1F = 1 518 STC
00AC 1F = 1 519 RESBN: POP DS
00A2 0A0 = 1 520 POP DX
00A3 0A1 = 1 521 POP CX
00A4 0A2 = 1 522 POP BX
00A5 0A3 = 1 523 POP AX
00A6 0A4 = 1 524 RET
00A7 0A5 = 1 525 ;
00A8 0A6 = 1 526 RES215 ENDP
00A9 0A7 = 1 527 ;
00AA 0A8 = 1 528 +1 $INCLUDE(1FL:INITEX.MMD)
00AB 0A9 = 1 529 +1 $EJECT TITLE(INITIALIZATION ROUTINE)
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER

INITIALIZATION ROUTINE

LOC OBJ  LINE  SOURCE
=1 530 ; -------
=1 531 ;|  INITIALIZATION ROUTINE |
=1 532 ;|-------
=1 533 ; INIT215 INITIALIZES THE ISBC 215 CONTROLLER BY LOADING PERTINENT INFOR-
=1 534 ; MATION ABOUT THE DISK DRIVE(S) ATTACHED.
=1 535 ; IF A DRIVE THAT IS SPECIFIED AS PRESENT WILL NOT RESPOND, INIT215 RETURNS
=1 536 ; IMMEDIATELY WITH THE CARRY FLAG SET.
=1 537 ; INPUT DATA:
=1 538 ; DISK DRIVE INITIALIZATION TABLES, IN SEGMENT "INITBLSEG".
=1 539 ; OUTPUT DATA:
=1 540 ; CARRY FLAG
=1 541 ; = 0 IF CONTROLLER INITIALIZED SUCCESSFULLY
=1 542 ; = 1 IF INITIALIZATION ERROR
=1 543 ;
=1 544 ;
=1 545 ;
=1 546 ;
=1 547 ;
=1 548 ;
=1 549 ;
=1 550 ;
=1 551 ; ASSUME DS:IOFBSEG
=1 552 ;
=1 553 ; INIT215 PROC FAR
=1 554 ;
=1 555 ; PUSH AX
=1 556 ;
=1 557 ; MOV AX,IOFBSEG
=1 558 ; ; SAVE REGISTERS
=1 559 ; GET POINTER TO IOPB
=1 560 ; ; PUT IN DS REGISTER
=1 561 ; WINDSOR DRIVES INITIALIZED FIRST
=1 562 ; ; SET IOFB FUNCTION BYTE = INITIALIZE
=1 563 ; CLEAR MODIFIER (ENABLE RETRIES AND
=1 564 ; INTERRUPT ON COMPLETION)
=1 565 ; ; PUT INITIALIZATION TABLES' SEGMENT IN
=1 566 ; IOPB DATA BUFFER POINTER
=1 567 ; ; START INITIALIZE WITH UNIT 0
=1 568 ; ; CLEAR UNIT COUNTER
=1 569 ; ; POINT TO NEXT DRIVE'S INITIALIZE TABLE
=1 570 ; ; PUT UNIT INTO IOPB
=1 571 ; ; DO INITIALIZE
=1 572 ; ; (RETURNS CARRY FLAG SET OR CLEAR)
=1 573 ; ; UNIT INITIALIZED?
=1 574 ; ; NO--TERMINATE WITH CARRY BIT SET
=1 575 ; ; YES--INCREMENT UNIT COUNTER
=1 576 ; ; CHECK UNIT COUNTER (CLEARS CARRY)
=1 577 ; ; LAST DRIVE INITIALIZED?
=1 578 ; ; NO--INITIALIZE NEXT DRIVE
=1 579 ; ; YES--FLOPPIES INITIALIZED YET?
=1 580 ; ; YES--INITIALIZE FUNCTION FINISHED
=1 581 ; ; NO--INITIALIZE FLOPPY DRIVES
=1 582 ; ; RESTORE REGISTERS
=1 583 ; ; RETURN
=1 584 ;
=1 585 ;
=1 586 ; |INIT215 ENDP |
=1 587 ;
=1 588 ; |INCLUDE(FI:FORMAT.MFD) |
=1 589 ; |SELECT TITLE(FORMAT TRACK ROUTINE) |
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER

LOC OBJ | LINK | SOURCE

---

1 | 590 | ; -----------------------------------------------
1 | 591 | ; FORMAT TRACK ROUTINE
1 | 592 | ; -----------------------------------------------
1 | 593 | ; FMTRK SETS UP THE IOPB FOR A FORMAT TRACK FUNCTION, AND
1 | 594 | ; INVOKES THE ISBC 215 CONTROLLER TO PERFORM THE OPERATION.
1 | 595 | ;
1 | 596 | ; INPUT DATA:
1 | 597 | ; -----------------------------------------------
1 | 598 | ; -----------------------------------------------

; CARRY FLAG = 0 IF TRACK FORMATTED SUCCESSFULLY
; = 1 IF NON-RECOVERABLE ERROR OCCURRED

; INTERLEAVE FACTOR OF 1 IMPLIES SEQUENTIAL SECTOR NUMBERING.
; USER DATA BYTES 0 - 3 ARE REPLICATED THROUGHOUT THE DATA FIELD.

; INTERLEAVE TYPES:
; 00 = NORMAL TRACK (USUALLY FOR FLOPPY)
; 01 = ALTERNATE TRACK (POINTED TO BY EXACTLY ONE DEFECTIVE TRACK, CANNOT SUBSEQUENTLY BE FORMATTED DEFECTIVE)
; 02 = DEFECTIVE TRACK (DATA FIELD POINTS TO ALTERNATE TRACK)
; 03 = TO SET UP A POINTER TO AN ALTERNATE TRACK, SET:

PUBLIC FMTRK
ASSUME OS: IOPBSEG
PROC FAR
PUSH AX
PUSH DS
MOV AX, IOPBSEG
MOV DS, AX
MOV AX, [BP+10]
MOV DEVCOD, AX
MOV AL, [BP]
MOV UNIT, AL
MOV AX, [BP+1]
MOV CYLNDR, AX
MOV AL, [BP+3]
MOV BUFSEG, AX
ADD BUFSEG, 4
MOV BUF, AX
MOV BUFSEG, SS
MOV BUFSEG, AX
MOV MODFF, AL
MOV MODIFY, 0000H
CALL GO215
RET 10
; (RETURNS CARRY FLAG SET OR CLEAR)
; RESTORE REGISTERS
ENDP

$INCLUDE (:FL:RDWRT.MMD)
$EJECT

TITLE (READ DATA ROUTINE)

00EF 50
00F0 1E
00F1 B8---- R
00F4 80DB
00F6 8460A
00F9 A30800
00FC 844600
00FF A20A00
0102 884601
0105 A30E00
0108 844603
010B A21000
010E 892E1200
0112 8306120004
0117 BC161400
011B CB060B0002
0120 C7060C000000
0126 E89900
0129 1F
012A 58
012B C40A00
012F 01
0134 FMTDN: POP DS
0135 POP AX
0137 RET 10
0138 FMT215 ENDP
0139 ; INCLUDE (:FL:RDWRT.MMD)
0140 +1 SELECT TITLE (READ DATA ROUTINE)
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER
READ DATA ROUTINE

LOC OBJ LINE SOURCE

=1 662 ; ---------------------
=1 663 ; READ DATA
=1 664 ; ---------------------
=1 667 ; RD215 SETS UP THE IOPB FOR A READ OPERATION, AND
=1 668 ; INVOKES THE ISSC 215 TO PERFORM THE OPERATION.
=1 669 ;
=1 670 ; INPUT DATA:
=1 671 ; BP + 13 => DEVICE CODE
=1 672 ; BP + 11 => BYTE COUNT HIGH WORD
=1 673 ; BP + 9 => BYTE COUNT LOW WORD
=1 674 ; BP + 7 => DATA BUFFER SEGMENT
=1 675 ; BP + 5 => DATA BUFFER OFFSET
=1 676 ; BP + 4 => SECTOR
=1 677 ; BP + 3 => HEAD
=1 678 ; BP + 1 => CYLINDER
=1 679 ;
=1 680 ;
=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 691 RD215 PROC FAR

=1 692 ;
012E 50 =1 693 PUSH AX ; SAVE REGISTERS
012F 1E =1 694 PUSH DS ;
0130 B8---- =1 695 MOV AX, IOPBSEG ; GET POINTER TO IOPB
0133 8EB8 =1 696 MOV DS, AX
0135 8A660D =1 697 MOV AX, [BP+13]
0138 A30B00 =1 698 MOV DEVCOD,Ax ; GET DEVICE CODE INTO IOPB
013B 8A4600 =1 699 MOV AL, [BP]
013E A20A00 =1 700 MOV UNIT, AL
0141 8B4601 =1 701 MOV AX, [BP+1]
0144 A30E00 =1 702 MOV CYLNDR, AX
0147 8B4603 =1 703 MOV AX, [BP+3]
014A 8B4601 =1 704 MOV AX, [BP+1]
014D A31000 =1 705 MOV WORD PTR HEAD, AX
0150 8A4605 =1 706 MOV AX, [BP+5]
0153 8B4607 =1 707 MOV AX, [BP+7]
0156 A31400 =1 708 MOV BUFSIZE, AX
0159 8B4609 =1 709 MOV AX, [BP+9]
015C A31600 =1 710 MOV WORD PTR REQCNT, AX
015F 8A460B =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 COMPLETION AND RET(KE))
0168 C6060B0004 =1 714 MOV FUNC, 04H ; SET FUNCTION = READ DATA
0170 E84F00 =1 715 CALL GO215 ; START FUNCTION AND WAIT FOR COMPLETION (RETURNS CARRY FLAG SET OR CLEAR)
0173 1F =1 716 POP DS
0174 5A =1 717 POP AX
0175 CA0D00 =1 718 RET 13 ; POP PARAMETERS OFF STACK AND RETURN
=1 719 21 ;
=1 720 RD215 ENDP
=1 721 ;
=1 722 ; EJECT TITLE (WRITE DATA ROUTINE)
=1 723 ;
=1 724 +1 $EJECT TITLE (WRITE DATA ROUTINE)
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER WRITE DATA ROUTINE

LOC OBJ LINK SOURCE

=1 725 ; ----------------------------------
=1 726 ; | WRITE DATA |
=1 727 ; |                  |
=1 728 ; |                  |
=1 729 ; |                  |
=1 730 ; | WRT215 SETS UP THE IOPB FOR A WRITE OPERATION, AND |
=1 731 ; | INVOKES THE ISBC 215 TO PERFORM THE OPERATION. |
=1 732 ; |                  |
=1 733 ; ; INPUT DATA:
=1 734 ; | BP + 13 => DEVICE CODE |
=1 735 ; | BP + 11 => BYTE COUNT HIGH WORD |
=1 736 ; | BP + 9 => BYTE COUNT LOW WORD |
=1 737 ; | BP + 7 => DATA BUFFER SEGMENT |
=1 738 ; | BP + 5 => DATA BUFFER OFFSET |
=1 739 ; | BP + 4 => SECTOR |
=1 740 ; | BP + 3 => HEAD |
=1 741 ; | BP + 1 => CYLINDER |
=1 742 ; | BP => UNIT |
=1 743 ; |
=1 744 ; | DATA BUFFER CONTAINS INFORMATION TO BE WRITTEN TO DISK |
=1 745 ; |
=1 746 ; ; OUTPUT DATA:
=1 747 ; | CARRY FLAG = 0 IF TRANSFER OCCURRED WITH NO OR RECOVERABLE ERROR |
=1 748 ; | = 1 IF UNRECOVERABLE ERROR OCCURRED |
=1 749 ; |
=1 750 ; ; PUBLIC WRT215
=1 751 ; | ASSUME DS:IOPBSEG |
=1 752 ; |
=1 753 ; |
=1 754 ; |
=1 755 ; WRT215 PROC FAR |
=1 756 ; |
=1 757 ; |
=1 758 ; |
=1 759 ; |
=1 760 ; |
=1 761 ; |
=1 762 ; |
=1 763 ; |
=1 764 ; |
=1 765 ; |
=1 766 ; |
=1 767 ; |
=1 768 ; |
=1 769 ; |
=1 770 ; |
=1 771 ; |
=1 772 ; |
=1 773 ; |
=1 774 ; |
=1 775 ; |
=1 776 ; |
=1 777 ; |
=1 778 ; |
=1 779 ; |
=1 780 ; |
=1 781 ; |
=1 782 ; |
=1 783 ; |
=1 784 ; |
=1 785 ; |
=1 786 ; WRT215 ENDP |
=1 787 ; |
=1 788 ; |
=1 789 ; | INCLUDE(:F1:CORE.MMD) |
=1 790 ; |
=1 791 ; | $EJECT |

B-20
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER
START FUNCTION AND WAIT FOR COMPLETION

LOC  OBJ LINE SOURCE

=1 790 ; ---------------------------------------------
=1 791 ; | START FUNCTION AND WAIT FOR COMPLETION |
=1 792 ; | ---------------------------------------------|
=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 E0800 +1 815 CALL WAI215 ; WAIT FOR FUNCTION COMPLETE
  01CD 7303 +1 816 JNC DONE ; ERROR?
  +1 817 |
  01CF E2900 +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 SELECT TITLE(WAIT FOR FUNCTION COMPLETE ROUTINE)
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER  WAIT FOR FUNCTION COMPLETE ROUTINE

LOC OBJ  LINE  SOURCE
1 827  |
1 828  | -- WAIT FOR FUNCTION COMPLETE --
1 829  |
1 830  |
1 831  |
1 832  |
1 833  | NORMALLY, THIS WAIT ROUTINE WOULD TRAP TO THE SYSTEM DISPATCHER/
1 834  | ITS FUNCTION. HOWEVER, FOR THIS EXAMPLE, THE ROUTINE SIMPLY WAITS FOR
1 835  | THE INTERRUPT SERVICE ROUTINE TO LOAD THE OPERATION COMPLETE STATUS
1 836  | FROM THE CIB OPERATION STATUS INTO THE DATA SEGMENT. IF AN ERROR
1 837  | OCCURRED, THE STATUS IS AVAILABLE THERE FOR SUBSEQUENT PROCESSING BY
1 838  | AN ERROR HANDLER.
1 839  |
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  |
1 855  |
1 856  | ASSUME DS:DATASEG
1 857  |
1 858  | 01D5 50  |
1 859  | 01D6 53  |
1 859  | 01D7 1E  |
1 861  |
1 862  |
1 863  | 01D8 BB----  |
1 865  |
1 866  |
1 867  | 01DB 8EDB  |
1 868  | 01DC BBFFFF  |
1 869  | 01DD BBFFFF  |
1 870  | 01DE F8  |
1 871  | 01E1 F4  |
1 873  | 01E2 43  |
1 875  | 01E3 81E30300  |
1 876  | 01E7 F607FF  |
1 877  | 01E8 74F6  |
1 878  | 01E9 7906  |
1 879  | 01EE 8&07  |
1 880  | 01F0 A21800  |
1 881  | 01F3 F9  |
1 882  | 01FA C3  |
1 883  | 01F4 C60700  |
1 884  |
1 885  | 01F7 7F  |
1 886  | 01F8 5B  |
1 887  | 01F9 5B  |
1 888  | 01FA C3  |
1 889  | ; SAVE REGISTERS
1 890  | ; GET POINTER TO DATA SEGMENT
1 891  | ; INITIALIZE INDEX REGISTER
1 892  | ; MAKE SURE INTERRUPT CAN GET THROUGH
1 893  | ; WAIT FOR INTERRUPT *****
1 894  | ; GET INDEX FOR NEXT UNIT
1 895  | ; MASK UPPER BITS
1 896  | ; OPERATION COMPLETE STATUS = 00H?
1 897  | ; (SIGN FLAG = BIT 7 OF OP. STATUS,
1 898  | ; TEST INSTR. CLEARS CARRY FLAG)
1 899  | ; CLEAR OPERATION COMPLETE BYTE
1 900  ; RESTORE REGISTERS ; RETURN

B-22
## Handshake Sequences and Example Host Processor Disk Control Program

### MCS-86 Macro Assembler Error Handler

```assembly
; LOC OBJ LINE SOURCE
-1 888 + ERROR HANDLER +
-1 889 + ERROR HANDLER +
-1 890 + ERROR HANDLER +
-1 891 + ERROR HANDLER +
-1 892 + ERROR HANDLER +
-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 + CIR OPERATION STATUS IN "LSTSTS" IN DATA SEGMENT
-1 906 +
-1 907 + OUTPUT DATA:
-1 908 + ERROR STATUS FROM CONTROLLER IN DATA SEGMENT
-1 909 + CIR 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+ ERROR PROC NEAR +1 916 +
+01FB 50 +1 917 +
+01FC IE +1 918 +
+01FD B8---- R +1 919 +
+0200 B6D8 +1 920 +
+0202 A11200 +1 921 +
+0205 50 +1 922 +
+0206 A11400 +1 923 +
+0209 50 +1 924 +
+020A C70612000C00 +1 925 +
+0210 C70614A00---- R +1 926 +
+0216 C606080001 +1 927 +
+0218 C7060C00000000 +1 928 +
+0221 B8EFFF +1 929 +
+0224 58 +1 930 +
+0225 A13400 +1 931 +
+0228 58 +1 932 +
+0229 A13200 +1 933 +
+022C B8---- R +1 934 +
+022F B6D8 +1 935 +
+0231 F8 +1 936 +
+0232 A01000 +1 937 +
+0235 2440 +1 938 +
+0237 7400 +1 939 +
+0239 F8 +1 940 +
+023A IF +1 941 +
+023B 58 +1 942 +
+023C C3 +1 943 +
+023D 5 +1 944 +
+023E 1 +1 945 +
+023F 1 +1 946 +
+0240 1 +1 947 +
+0241 1 +1 948 +

+0242 1 +1 949 +
+0243 1 +1 950 +
-1 950 +1 $INCLUDE(F1:INTRPT.MMD)
-1 951 +1 $SOURCE ERROR HANDLER
```

### Source Code:

```
LOC OBJ LINE SOURCE
-1 888 ;
-1 889 ;
-1 890 ;
-1 891 ;
-1 892 ;
-1 893 ;
-1 894 ;
-1 895 ;
-1 896 ;
-1 897 ;
-1 898 ;
-1 899 ;
-1 900 ;
-1 901 ;
-1 902 ;
-1 903 ;
-1 904 ;
-1 905 ;
-1 906 ;
-1 907 ;
-1 908 ;
-1 909 ;
-1 910 ;
-1 911 ;
-1 912 ;
-1 913 ;
-1 914 ;
-1 915 ;

01FB

+01FB 50 +1 917 +
+01FC IE +1 918 +
+01FD B8---- R +1 919 +
+0200 B6D8 +1 920 +
+0202 A11200 +1 921 +
+0205 50 +1 922 +
+0206 A11400 +1 923 +
+0209 50 +1 924 +
+020A C70612000C00 +1 925 +
+0210 C70614A00---- R +1 926 +
+0216 C606080001 +1 927 +
+0218 C7060C00000000 +1 928 +
+0221 B8EFFF +1 929 +
+0224 58 +1 930 +
+0225 A13400 +1 931 +
+0228 58 +1 932 +
+0229 A13200 +1 933 +
+022C B8---- R +1 934 +
+022F B6D8 +1 935 +
+0231 F8 +1 936 +
+0232 A01000 +1 937 +
+0235 2440 +1 938 +
+0237 7400 +1 939 +
+0239 F8 +1 940 +
+023A IF +1 941 +
+023B 58 +1 942 +
+023C C3 +1 943 +
+023D 5 +1 944 +
+023E 1 +1 945 +
+023F 1 +1 946 +
+0240 1 +1 947 +
+0241 1 +1 948 +
+0242 1 +1 949 +
+0243 1 +1 950 +
+0244 1 +1 951 +
```

---

B-23

`
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR DISK CONTROL PROGRAM

HCS-86 MACRO ASSEMBLER INTERRUPT SERVICE ROUTINE

LOC OBJ

<table>
<thead>
<tr>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>952</td>
<td>: INTERFRUPT SERVICE ROUTINE :</td>
</tr>
<tr>
<td>953</td>
<td></td>
</tr>
<tr>
<td>954</td>
<td></td>
</tr>
<tr>
<td>955</td>
<td></td>
</tr>
<tr>
<td>956</td>
<td></td>
</tr>
<tr>
<td>957</td>
<td></td>
</tr>
<tr>
<td>958</td>
<td>THIS ROUTINE SERVICES THE INTERRUPT GENERATED BY THE ISBC 215 UPON</td>
</tr>
<tr>
<td>959</td>
<td>OPERATION COMPLETE, SEEK COMPLETE, OR DISK PACK CHANGE. IT COPIES THE</td>
</tr>
<tr>
<td>960</td>
<td>CIB OPERATION STATUS INTO ONE OF FOUR BYTES ASSOCIATED WITH EACH OF</td>
</tr>
<tr>
<td>961</td>
<td>THESE EVENTS. IT IS ASSUMED THAT SYSTEM PROGRAMS MAKE USE OF THE</td>
</tr>
<tr>
<td>962</td>
<td>INFORMATION TO RESUME TASKS, HANDLE ERRORS LOGIC/RECOVERY, AND KEEP</td>
</tr>
<tr>
<td>963</td>
<td>THE OPERATION COMPLETE BYTES ARE USED.</td>
</tr>
<tr>
<td>964</td>
<td></td>
</tr>
<tr>
<td>965</td>
<td>- THE SYSTEM INTERRUPTS ARE CONFIGURED BY EXTERNAL PROGRAMS.</td>
</tr>
<tr>
<td>966</td>
<td></td>
</tr>
<tr>
<td>968</td>
<td></td>
</tr>
<tr>
<td>969</td>
<td>PUBLIC INT215</td>
</tr>
<tr>
<td>970</td>
<td></td>
</tr>
<tr>
<td>971</td>
<td>INT215 PROC FAR</td>
</tr>
<tr>
<td>972</td>
<td>STI</td>
</tr>
<tr>
<td>973</td>
<td>SAVE REGISTERS</td>
</tr>
<tr>
<td>974</td>
<td></td>
</tr>
<tr>
<td>975</td>
<td>PUSH AX</td>
</tr>
<tr>
<td>976</td>
<td></td>
</tr>
<tr>
<td>977</td>
<td></td>
</tr>
<tr>
<td>978</td>
<td>POP DS</td>
</tr>
<tr>
<td>979</td>
<td></td>
</tr>
<tr>
<td>980</td>
<td></td>
</tr>
<tr>
<td>981</td>
<td></td>
</tr>
<tr>
<td>982</td>
<td></td>
</tr>
<tr>
<td>983</td>
<td></td>
</tr>
<tr>
<td>984</td>
<td></td>
</tr>
<tr>
<td>985</td>
<td></td>
</tr>
<tr>
<td>986</td>
<td></td>
</tr>
<tr>
<td>987</td>
<td></td>
</tr>
<tr>
<td>988</td>
<td></td>
</tr>
<tr>
<td>989</td>
<td></td>
</tr>
<tr>
<td>990</td>
<td></td>
</tr>
<tr>
<td>991</td>
<td></td>
</tr>
<tr>
<td>992</td>
<td></td>
</tr>
<tr>
<td>993</td>
<td></td>
</tr>
<tr>
<td>994</td>
<td></td>
</tr>
<tr>
<td>995</td>
<td></td>
</tr>
<tr>
<td>996</td>
<td></td>
</tr>
<tr>
<td>997</td>
<td></td>
</tr>
<tr>
<td>998</td>
<td></td>
</tr>
<tr>
<td>999</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td></td>
</tr>
<tr>
<td>1002</td>
<td></td>
</tr>
<tr>
<td>1003</td>
<td></td>
</tr>
<tr>
<td>1004</td>
<td></td>
</tr>
<tr>
<td>1005</td>
<td></td>
</tr>
<tr>
<td>1006</td>
<td></td>
</tr>
<tr>
<td>1007</td>
<td></td>
</tr>
<tr>
<td>1008</td>
<td></td>
</tr>
<tr>
<td>1009</td>
<td></td>
</tr>
<tr>
<td>1010</td>
<td></td>
</tr>
<tr>
<td>1011</td>
<td></td>
</tr>
<tr>
<td>1012</td>
<td></td>
</tr>
<tr>
<td>1013</td>
<td></td>
</tr>
<tr>
<td>1014</td>
<td></td>
</tr>
<tr>
<td>1015</td>
<td>INT215 ENDP</td>
</tr>
<tr>
<td>1016</td>
<td></td>
</tr>
<tr>
<td>1017</td>
<td>SBC215DRIVER ENDS</td>
</tr>
<tr>
<td>1018</td>
<td>FIND OF SBC 215 DRIVER CODE</td>
</tr>
<tr>
<td>1019</td>
<td></td>
</tr>
<tr>
<td>1020</td>
<td></td>
</tr>
<tr>
<td>1021</td>
<td></td>
</tr>
</tbody>
</table>

B-24
HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR
DISK CONTROL PROGRAM

MCS-86 MACRO ASSEMBLER  SYMBOL TABLE AND CROSS REFERENCE

XREF SYMBOL TABLE LISTING

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>VALUE</th>
<th>ATTRIBUTES, XREFS</th>
</tr>
</thead>
</table>

**HANDSHAKE SEQUENCES**

**DISK CONTROL PROGRAM**

---

B-25
**HANDSHAKE SEQUENCES AND EXAMPLE HOST PROCESSOR DISK CONTROL PROGRAM**

MCS-86 MACRO ASSEMBLER SYMBOL TABLE AND CROSS REFERENCE

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>VALUE</th>
<th>ATTRIBUTES, XREFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFEKST.</td>
<td>V BYTE</td>
<td>000EH</td>
<td>DATASEG 30A#</td>
</tr>
<tr>
<td>SFTRERR.</td>
<td>L NEAR</td>
<td>023AH</td>
<td>SBC215DRIVER 941 944#</td>
</tr>
<tr>
<td>SKCMP</td>
<td>V BYTE</td>
<td>0004H</td>
<td>DATASEG PUBLIC 278 290#</td>
</tr>
<tr>
<td>SKCMPO.</td>
<td>V BYTE</td>
<td>0005H</td>
<td>DATASEG 279#</td>
</tr>
<tr>
<td>SKCMPI.</td>
<td>V BYTE</td>
<td>0006H</td>
<td>DATASEG 279#</td>
</tr>
<tr>
<td>SOC</td>
<td>V BYTE</td>
<td>0007H</td>
<td>DATASEG 294#</td>
</tr>
<tr>
<td>STSSN</td>
<td>V BYTE</td>
<td>0003H</td>
<td>CIBSEG 109#</td>
</tr>
<tr>
<td>UNIT.</td>
<td>V BYTE</td>
<td>000AH</td>
<td>IOPSEG 131# 568 641 700 764</td>
</tr>
<tr>
<td>USERSEG</td>
<td>SEGMENT</td>
<td>SIZE=0022H PARA 375# 398</td>
<td></td>
</tr>
<tr>
<td>WAIT215</td>
<td>L NEAR</td>
<td>01D5H</td>
<td>SBC215DRIVER 815 856# 884</td>
</tr>
<tr>
<td>WAITDN</td>
<td>L NEAR</td>
<td>01F4H</td>
<td>SBC215DRIVER 871 878#</td>
</tr>
<tr>
<td>WAITLP</td>
<td>L NEAR</td>
<td>01E2H</td>
<td>SBC215DRIVER 866# 871</td>
</tr>
<tr>
<td>WRT215</td>
<td>L FAR</td>
<td>0178H</td>
<td>SBC215DRIVER PUBLIC 752 755# 786</td>
</tr>
<tr>
<td>WUA</td>
<td>NUMBER</td>
<td>0635H</td>
<td>57# 59 497 812 999</td>
</tr>
</tbody>
</table>

ASSEMBLY COMPLETE, NO ERRORS FOUND

***

B-26
-5V selection 2-8
14/8-inch drives (with integral data separator) installation 2-36
5 1/4 drives installation 2-37
address 3-28, 3-29, 3-39, 3-49
I/O addresses 3-4
wake up address 3-7
alternate track 3-42
ANSI A-1
Any Bus request signal 2-5

block diagram 5-9
board initialization/resetting 4-10
I/O communications blocks links 4-12
reset and clear 4-12
wake-up address comparator 4-10
board installation 2-35
bus arbiter 4-6
bus priority arbitration 2-5
Any Request signal 2-5
bus priority out signal 2-6
common bus request signal 2-5
I/O communication blocks page selection 2-9
iSBX bus selection 2-6
MULTIBUS LOCK signal 2-6
bus priority out signal 2-6

cabling requirements 2-16
drive interface pin out data 2-17
control cable functions 2-19
read/write cable functions 2-22
interconnecting cable 2-29
5 1/4-inch drive 2-32
8-inch Shugart/Quantum drives 2-29
ANSI drive 2-34
CDC drive 2-33
Fujitsu 2300/Memorex/14-inch drives 2-30
Priam drive 2-31
capacity 1-7
clock circuitry 4-6
commands 3-11, 3-15
buffer I/O 3-30
diagnostic 3-31
erase tape 3-33
execute 3-28
initiate track seek 3-27
load tape 3-34
read sector 3-21
INDEX

read data 3–22
read to buffer and verify 3–24
read tape status 3–35
read/write terminate 3–35
reftension tape 3–34
rewind 3–32
space forward one file mark 3–32
spin down 3–28
tape initialization 3–31
tape reset 3–34
tape write 3–26
transfer 3–28
write data 3–25
write buffer data 3–26
write file mark 3–33
common bus request signal 2–5
communications
  board/disk-drive interface 4–14
    control signals 4–14
    read/write signals 4–14
  board/flexible-disk and cartridge tape drive interface 4–15
    data transmission 4–15
bus arbiter 4–6
bus controller 4–7
  address/data control 4–8
  data and peripheral data signals 4–8
  MULTIBUS data transfer 4–9
  system/local memory access 4–8
disk interface timing 4–15
  direct memory access transfers 4–15
  data transfers 4–16
  performing write or read 4–16
  RDY line 4–15
disk formatting 4–17
  8-inch Shugart/Quantum drive sector format differences 4–19
  procedure 4–18
read data transfers 4–21
write data transfers 4–20
  ID read 4–21
  sync byte search 4–21
error checking code generator 4–23
interrupt priority 4–13
I/O processor 4–5
  bus interface unit 4–6
    internal initialization 4–12
line drivers and receivers 4–25
MULTIBUS interface 4–5
sector identity comparator 4–23
serializer/deserializer 4–22
  function 4–22
  write operation 4–22
  read operation 4–22
status register 4–24
sync byte comparator 4–23
with host 4–3, 4–5
with hard disk 4–4
via iSBX bus 4–5
communications blocks  3-4
  wakeup  3-4, 3-7, 3-13
channel control block  3-5, 3-7, 3-8, 3-9, 3-13
controller invocation block  3-5, 3-9, 3-13
I/O parameter block  3-5, 3-11, 3-12, 3-13, 3-29, 3-31, 3-41
cooling requirements  2-2
direct memory access transfers  4-15
  data transfers  4-16
    performing write or read  4-16
RDY line  4-15
default jumpers  5-5
disk control format  3-1
disk controller communications protocol  3-1
disk formatting  4-17
  8-inch Shugart/Quantum drive sector format differences  4-19
    procedure  4-18
disk interface timing  4-15
disk organization  3-1, 3-3
  cylinders  3-2, 3-12, 3-17, 3-37
  sectors  3-2, 3-12, 3-17
  tracks  3-2, 3-17
drive installation  2-35
  5 1/4-inch drives  2-37
  other drives without data separator  2-37
drive interconnection listings  2-23
  8-inch Shugart/Quantum drives  2-23
  Fujitsu 2300/Memorex/14-inch Shugart drives  2-24
  Priam drive  2-25
  5 1/4-inch drive  2-26
  CDC drive  2-27
  ANSI drive  2-28
drives  1-5
ECC  3-45
error checking code generator  4-23
error detection  1-5
example program  B-2
firmware drivers  3-47
flexible diskette controller  3-1
format command  3-19
general description  1-1
handshake  B-1
hard disk drive interface  2-7
INDEX

I/O communications block 3-5, 3-7
I/O control commands 3-5
I/O processor 3-1, 3-7
I/O processor 4-5
  bus interface unit 4-6
  internal initialization 4-12
ID field 3-45
initialization 3-7, 3-13, 3-31, 3-34
initialize 3-15, 3-16, 3-31, 3-34
installation 2-1
interface connections 2-9
interrupt 3-1, 3-46
interrupt priority 4-13
interrupt priority number 2-4
iSBX 217B/C board installation 2-39
iSBX 218A board installation 2-38
iSBX MULTIMODULE interface 2-14
  iSBX bus/iSBX 215G board mnemonics 2-15
iSBX bus selection 2-6

jumper configuration and installation 2-2
  -5V selection 2-8
  Any Request signal 2-5
  bus priority out signal 2-6
  common bus request 2-5
  configuration 2-2
  installation 2-2
  interrupt priority number 2-4
  I/O communication blocks page 2-9
  iSBX bus selection 2-6
  MULTIBUS LOCK signal 2-6
  system data bus width selection 2-4
  wake-up address 2-3
  wake-up port selection 2-4
jumper connections 5-5
jumper diagram 5-7

line drivers and receivers 4-25
local memory organization 4-25
  I/O port decoding 4-26
  random-access memory 4-26
  read-only memory 4-26

magnetic tape interface 3-1
mounting 1-5
MULTIBUS LOCK signal 2-6
MULTIBUS interface 2-9
  connector P1 pin assignments 2-10
  connector P1 input/output signals 2-12
  connector P2 pin assignments 2-11
  connector P2 input/output signals 2-14
INDEX

physical characteristics 2-2
power 1-5
power requirements 2-1

read 3-11, 3-24
recalibrate 3-31
repair 5-1

schematics 5-11
sector 3-2, 3-12, 3-17, 3-25, 3-43, 3-44
sectors 1-6
seek 3-27, 3-42, 3-46
serializer/deserializer 4-22
function 4-22
read operation 4-22
write operation 4-22
service 5-1
specifications 1-4
status 3-9, 3-10, 3-19, 3-27, 3-35, 3-37, 3-38, 3-39, 3-40, 3-42 to 45
status register 4-24
storage devices 3-2
hard disk 3-2, 3-16, 3-39, 3-40, 3-41, 3-42, 3-43, 3-44
flexible disk 3-2, 3-16, 3-39, 3-40, 3-41, 3-42, 3-43, 3-44
cartridge tape 3-2, 3-16, 3-39, 3-40, 3-41, 3-42, 3-43, 3-44
system data bus width selection 2-4

tape 3-26, 3-31, 3-33, 3-34, 3-35, 3-39 to 45
track sectoring format 3-1

user provided drivers 3-47
wake-up I/O port selection 2-4
wake-up address selection 2-3
write 3-11, 3-25, 3-33
write data transfers 4-20
ID read 4-21
read data transfers 4-21
sync byte search 4-21

***

Index-5
REQUEST FOR READER’S COMMENTS

Intel’s Technical Publications Departments attempt to provide publications that meet the needs of all Intel product users. This form lets you participate directly in the publication process. Your comments will help us correct and improve our publications. Please take a few minutes to respond.

Please restrict your comments to the usability, accuracy, readability, organization, and completeness of this publication. If you have any comments on the product that this publication describes, please contact your Intel representative. If you wish to order publications, contact the Intel Literature Department (see page ii of this manual).

1. Please describe any errors you found in this publication (include page number).

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

2. Does the publication cover the information you expected or required? Please make suggestions for improvement.

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

3. Is this the right type of publication for your needs? Is it at the right level? What other types of publications are needed?

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

4. Did you have any difficulty understanding descriptions or wording? Where?

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

5. Please rate this publication on a scale of 1 to 5 (5 being the best rating).

NAME ________________________________ DATE __________
TITLE ______________________________________________________________________
COMPANY NAME/DEPARTMENT ________________________________________________
ADDRESS ________________________________ STATE ________________ ZIP CODE __
CITY ________________________________ (COUNTRY)

Please check here if you require a written reply. ☐
WE'D LIKE YOUR COMMENTS . . .

This document is one of a series describing Intel products. Your comments on the back of this form will help us produce better manuals. Each reply will be carefully reviewed by the responsible person. All comments and suggestions become the property of Intel Corporation.