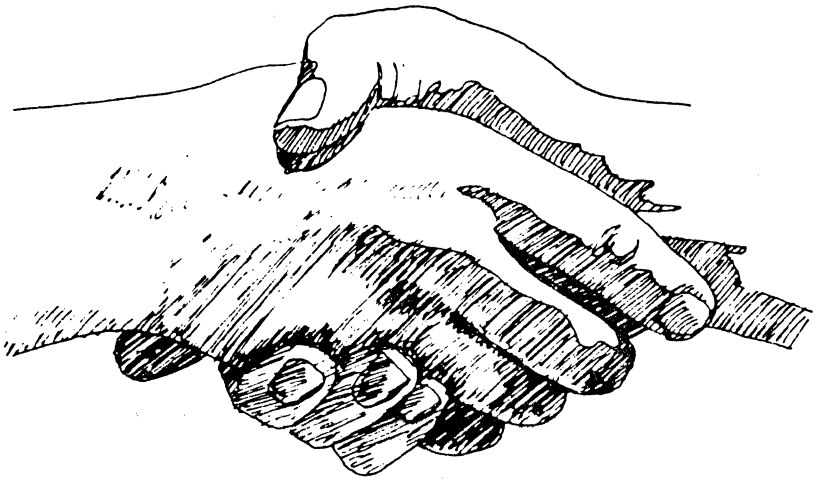
Xylogics

Model 712
User's Manual



YOUR PARTNER FOR PERFORMANCE.

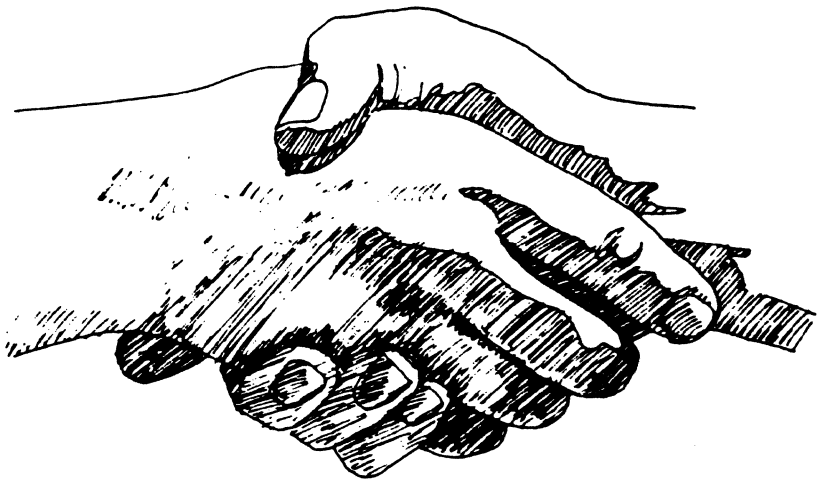
166-712-001
Revision A
September 22, 1986

712 REVISION LEVEL HISTORY

REVISION	DESCRIPTION
A (9/22/86)	Initial release.
A1 (12/1/86)	Added VME compliance number to Section 1.8. Technical addition to Section 4.2 (Controller Parameters IOPB): Byte 0C now contains the 712 Release Level; bit 5 in Byte 08 is now Disable ACFAIL (DACF); added Section 4.2.5. Edited Sections 4.2.3 (ASR), 4.2.8, 4.2.9, 6.1.2.4 (Code 45), 8.4, and 8.4.1. NOTE Only the Table of Contents, Section 4, and pages 6, 54, 58, 80, 89, 91, and 112 are marked Rev. A1.

Xylogics

Model 712
User's Manual



YOUR PARTNER FOR PERFORMANCE.

166-712-001
Revision A
September 22, 1986

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

PAGE

FIGURES

TABLES

SECTION 1: SPECIFICATIONS

1.0 GENERAL

The Xylogics Model 712 Disk Controller accommodates up to four ESDI interface disk drives to VMEbus[1] systems.

1.1 USING THIS MANUAL

This manual provides two Software Reference Cards for fast reference of the IOPB structure and codes (See insert). Section 2 describes how to install and test the 712; Section 3 describes the 712 registers; Section 4 describes the IOPBs; and Section 5 describes the 712 commands. Section 6 describes error processing; Section 7 is a programming tutorial; Section 8 explains the 712's special functions; Section 9 describes the 712 theory of operation; and Section 10 includes maintenance aides.

1.1.1 Abbreviations

This manual uses the following mnemonics:

AFE	Alternate Field Enable
AIO	Add New IOPB
AIOP	AIO Pending
AIOR	AIO Response Time
AM	Address Modifier
ASR	Automatic Seek Retry
AUD	Auto-update
BHT	Black Hole Transfer
CHEN	Chain Enable
CMPL	Command Complete
CRIO	Clear Remove IOPB
COP	Command Optimization
CRBS	Clear Register Busy
CTYP	Controller Type
DFLT	Drive Fault
DMA	Direct Memory Access
DRDY	Drive Ready
DSKCEL	Disk Front End Chip
EC32	32-Bit ECC

- - - - - -

1. VMEbus is a trademark of the VMEbus International Trade Association.

1.1.1 Abbreviations (continued)

ECC	Error Correction Code
ECCM	Error Correction Mode
EDT	Enable DMA Timeout
ERRS	Error Summary
ESD	Embedded Servo Drive
FERR	Fatal Error
FIFO	First In/First Out Disk Data Buffer
H	Notation For Numerical Values Expressed in Hexadecimal
ICS	IOPB Checksum
IEC	Interrupt At End Of Chain
IOPB	Input/Output Parameter Block
MBS	Megabytes Per Second
MMA	Maintenance Mode Active
MM	Maintenance Mode
NPRM	Non-privileged Register Mode
OVS	Overlap Seek Enable
PNUM	Prom Number
PRIO	Priority IOPB
PSEL	Priority Select
RBC	Retry Before Correction
RBS	Register Busy Semaphore
REGCEL	Register Read/Write and Interrupt
RIO	Remove IOPB
RMM	Register Maintenance Mode
ROR	Release On Request
SGM	Scatter/Gather Mode
SKER	Seek Error
TDT	Throttle Dead Time
THRO	Throttle
TMOD	Transfer Mode
VMEDMA	Direct Memory Access Controller Chip
WRPT	Write-protect
ZLR	Zero Latency Reads

1.2 DESIGN RELIABILITY

Xylogics implements the following features to minimize the likelihood of product failure:

- Design for worst case voltage and temperature.
- Extensive evaluation testing.
- Low parts count through extensive use of custom LSI.
- Buffer parity for continuous error checking.

1.2 DESIGN RELIABILITY (continued)

- o Low-stress design on all components.
- o All components burned-in.
- o One card; resides in backplane or expansion chassis.
- o Controller is power-cycled under thermal stress during test.

1.3 PHYSICAL

PACKAGING -- The 712 completely resides on one printed circuit board.

DIMENSIONS — The 712 is a 2 by 2 Eurocard standard; it measures 9.2-inches high by 6.3-inches deep (233.35 mm by 160 mm). The 712 is identical in form-factor to the standard VME (dual high-dual wide) printed circuit board.

SHIPPING WEIGHT — 3 pounds (1.4 kg).

FRONT PANEL — Xylogics offers the 712 with an optional front panel.

CONNECTORS — The ESDI connectors are on the edge of the board facing out; they protrude through the optional front panel. The optional straight connectors do not protrude the front panel.

1.4 ENVIRONMENTAL

The 712 environmental requirements are 0 - 55° C, with a maximum relative humidity of 90% (without condensation). Air flow across the board must maintain a maximum temperature differential of 7° C to prevent hot spots.

1.5 ELECTRICAL

POWER — The 712 uses 4.1 amperes at +5 volts DC (VDC).

TOLERANCE — Voltage must be within plus or minus five percent (4.75 to 5.25).

GROUNDING -- Common earth ground must be established between the disk drives and the CPU chassis, backplane, and expansion cabinets.

1.6 SYSTEM RELATED SPECIFICATIONS

DATA BUFFERING — The 712 has a FIFO buffer that is 8k-bytes long and incorporates parity error detection. Data can be put into one end of the FIFO and simultaneously removed at the other end; there are no delays associated with filling and emptying the buffer.

MULTIPLE IOPBs ON A SINGLE REVOLUTION — The 712 can execute multiple IOPBs on a single revolution. For example, if four IOPBs for four sectors on the same head and cylinder are chained, the 712 can transfer the sectors into the buffer on a single revolution, and transfer each one out to the correct memory location.

PRIORITY IOPBs — The 712 executes priority IOPBs over all IOPBs in its command cache, except for the one in process.

FORMAT — The 712 Format command formats a specified number of tracks. Use the Read/Write Headers commands to incorporate custom interleaving schemes. Standard interleaving is 1:1; 2:1 to 15:1 interleaving is software programmable.

MEDIA DEFECTS — The 712 has several methods for remapping bad blocks. One method leaves spare sectors on each track that can be slipped with Read/Write Track Headers commands. An alternate method has the spare sectors on the last part of the maximum track. The 712 also remaps entire tracks. This lessens the total number of spare sectors required with minimal affect on 712 performance.

READ DEFECT MAP FEATURE — The 712 can read the manufacturer's defect information directly from the disk.

STATUS LEDs — The 712 implements two status LEDs. L1 (BSY) indicates the controller is active; L2 (ERR) indicates the on-board diagnostics did not complete successfully, or a fatal error occurred.

SCATTER/GATHER — The 712 supports Scatter/Gather on Read and Write commands. The controller can gather data from various memory locations and transfer it to the buffer for use in a Write command; it can scatter the data out from the disk drive to the appropriate memory locations with a Read command. To execute a scatter/gather, software issues a normal Read or Write command along with a DMA list that contains a memory address and the number of words to transfer to/from that location. The smallest granularity of scatter/gather is a 16-bit word.

1.6 SYSTEM RELATED SPECIFICATIONS (continued)

ERROR DETECTION AND CORRECTION — The 712 supports a 48-bit data ECC; it optionally supports a 32-bit data ECC. Software controls automatic detection and correction.

The 32-bit ECC detects an error burst up to 22-bits long, and corrects an error burst up to 11-bits long.

The 48-bit ECC detects an error burst up to 28-bits long, and corrects error bursts up to 14-bits long, assuring data integrity.

IMPLIED SEEK CAPABILITY — Data transfer instructions contain an implied seek. Data transfers cross sector, head, and cylinder boundaries as required (spiral read/write).

OVERLAP SEEKS — The 712 supports overlap seeks. When overlap seeks are enabled, the 712 may have both drives simultaneously seeking to the appropriate cylinders.

ELEVATOR SEEKS — When elevator seeks are enabled, the 712 reorders commands in ascending and then descending cylinder order to get the best throughput from the disk subsystem.

ZERO LATENCY READS — When the head arrives over the cylinder, the 712 reads the first sectors it finds that are included in the IOPB; it transfers the data to its own buffer and then out to the correct memory location. The controller finishes the track transfer when the initial sectors arrive under the head.

BLACK HOLE TRANSFERS — The 712 may transfer all the DMA data into the same bus address without incrementing the address at each DMA.

SOFTWARE SUPPORT — Sample software driver supplied for use in UNIX[2] (5.0) based systems (source included).

1.7 DISK DRIVE RELATED SPECIFICATIONS

PHYSICAL DRIVE INTERFACE — The 712 supports the ESDI interface.

INTERFACE DATA RATE AND STANDARD INTERLEAVE FACTOR — The 712 supports a maximum disk data rate of 1.2 Megabytes Per Second (MBS). The 712 supports this data rate at a 1:1 interleave factor.

EMBEDDED SERVO DRIVES — The 712 supports embedded servo drives.

NUMBER OF DISK DRIVES — The 712 supports up to four ESDI disk drives.

DISK SECTOR FORMAT — The 712 sector format includes a header field separated from a data field by a splice area.

- - - - - -

2. UNIX is a trademark of AT&T.

1.7 DISK DRIVE RELATED SPECIFICATIONS (continued)

DISK SECTOR FORMAT — The 712 sector format includes a header field separated from a data field by a splice area.

HEADER FORMAT — Header contains sector, head, cylinder address, and header CRC or redundent header depending on if your using 48 or 32 bit ECC.

CABLING — The 712 uses standard 20- and 34-pin flat ribbon cables. The maximum length of each cable is 3 meters (9.8 feet).

1.8 VMEbus RELATED SPECIFICATIONS

VME COMPLIANCE NUMBER — IEEE P1014/D1.0.

TRANSFER MODE — Direct Memory Access (DMA).

DMA THROTTLE CONTROL -- Each time the 712 becomes bus master, it executes DMA transfers to or from the buffer up to the max throttle limit or the number of bytes/spaces available in the buffer.

DMA DATA TRANSFER RATE — The 712 transfers data at a rate of up to 10 MBS; this rate requires Longword mode transfers and system memory that responds within 200 nanoseconds.

DMA DEAD TIME — The 712 supports a programmable throttle dead time between throttle bursts. This prevents the 712 from taking over the bus and allows time for other DMA devices to access the bus.

DATA TRANSFER LIMIT — Data transfer length, from 1 to 65,535 sectors with a single IOPB.

BUS COMPATIBILITY — The 712 is compatible with the standard VMEbus.

ADDRESSING CAPABILITY — The 712 supports Master A32, and Slave A16, as per the VMEbus Specification Manual. As a slave, the 712 responds to Address Modifiers 29H and 2DH.

DATA WIDTH -- The 712 supports D16 and D32 as per the VMEbus Specification Manual. The 712 transfers one byte, one word, or a byte and a word, until the transfer aligns with a word or longword boundary.

RELEASE ON REQUEST — The 712 releases the bus at the request of other peripheral devices.

RELEASE WHEN DONE — The 712 releases the bus after each bus access.

1.8 VMEbus RELATED SPECIFICATIONS (continued)

BUS REQUEST LEVELS — The 712 supports four bus request levels.

EARLY RELEASE OF BUS BUSY/ — The 712 does not support early release of Bus Busy/.

INTERRUPT PRIORITY — Software programmable interrupt level and vector.

CONTROLLER I/O PARAMETER BLOCK (IOPB) LENGTH — 30 bytes.

CONTROLLER REGISTERS — Seven 8-bit I/O Registers; byte or word addressable. Only eight bits respond during word access.

DIAGNOSTIC SUPPORT — Comprehensive set of stand-alone diagnostics written in 'C' are available.

1.9 SOFTWARE RELATED SPECIFICATIONS

SOFTWARE INTERFACE — The 712 supports a high level software interface that allows host software to use the same method to add IOPBs to a chain while the controller is busy or while it is free.

1.9.1 Software Interface

The software interface includes seven byte-wide registers. Four of these bytes comprise the VME Address Register, the fifth byte is the Address Modifier Register, and the sixth byte is the Control and Status Register (CSR). The CSR includes two bits that are very important to IOPB processing: Add IOPB (AIO) and Remove IOPB (RIO). The last byte is the Fatal Error Register; the 712 returns the fatal error codes in this register.

The IOPB is a block of command and status information; it includes the disk address, the bus address, and the type of operation to be performed. The software driver sets up the IOPB in user memory, sends the IOPB address to the VME Address Registers, and sets AIO. After the 712 receives the IOPB address it resets AIO. The 712 then performs the IOPB function and, upon completion or error, updates the IOPB status and sets RIO. The VME Address Registers point to the complete IOPB; the software driver reads the address and then resets RIO.

Software may add IOPBs to the queue, providing AIO is reset, by writing the IOPB address to the address registers and setting AIO (regardless of the 712's busy status).

1.10 PROGRAMMABLE FEATURES

- o Software Controlled Interrupt or Polled Operations.
- o Software Programmable DMA Parameters.
- o Software Programmable Drive Size Parameters (Including Sector Size).
- o Software Programmable Sector Interleaving — Standard 1:1.
- o Software Controlled Register Response.
- o Software Controlled Transfer Retry/Correction.
- o Software Programmable Hard or Soft Sector Mode.

SECTION 2: INSTALLING AND TESTING THE 712

2.0 GENERAL

This section describes how to unpack, configure, install, and test your 712 controller.

2.1 UNPACKING AND INSPECTION

2.1.1 Inspect the Shipping Carton

Inspect the carton for possible shipping damage. If you determine there is damage, do not unpack the unit. Notify Xylogics and the freight carrier immediately.

If no damage is visible, carefully unpack the 712. Save the carton and other packing material for possible later use.

2.1.2 Contents

The 712 is a single printed circuit board. Optional items include a manual and/or software on a floppy diskette, or 1/2-inch magnetic tape.

If any items are missing or damaged, please contact Xylogics at one of the following telephone numbers.

United States:	(617) 272-8140
United Kingdom (Slough):	44-753-78921

2.1.3 Handling Precautions

Observing proper handling precautions minimizes the risk of damaging the 712 with electrostatic discharge. When transporting the 712, use an antistatic bag, antistatic bin, or the original shipping carton and packing material. Personnel handling the 712 should observe proper grounding methods including, but not limited to, wrist bands, heel straps, and antistatic mats.

The 712 has a non-volatile memory circuit that employs a lithium battery (at location E8). Do not expose this device to excessive heat (greater than 125° C) as it may ignite or explode.

2.1.4 Inspect the 712

Inspect the 712 for socketed parts that may have loosened during shipment. Assure that all parts are firmly seated in their sockets. If any parts must be reinserted, observe proper orientation.

2.2 CONFIGURING THE 712

You can configure the 712 with several jumper options. The following subsections describe these options.

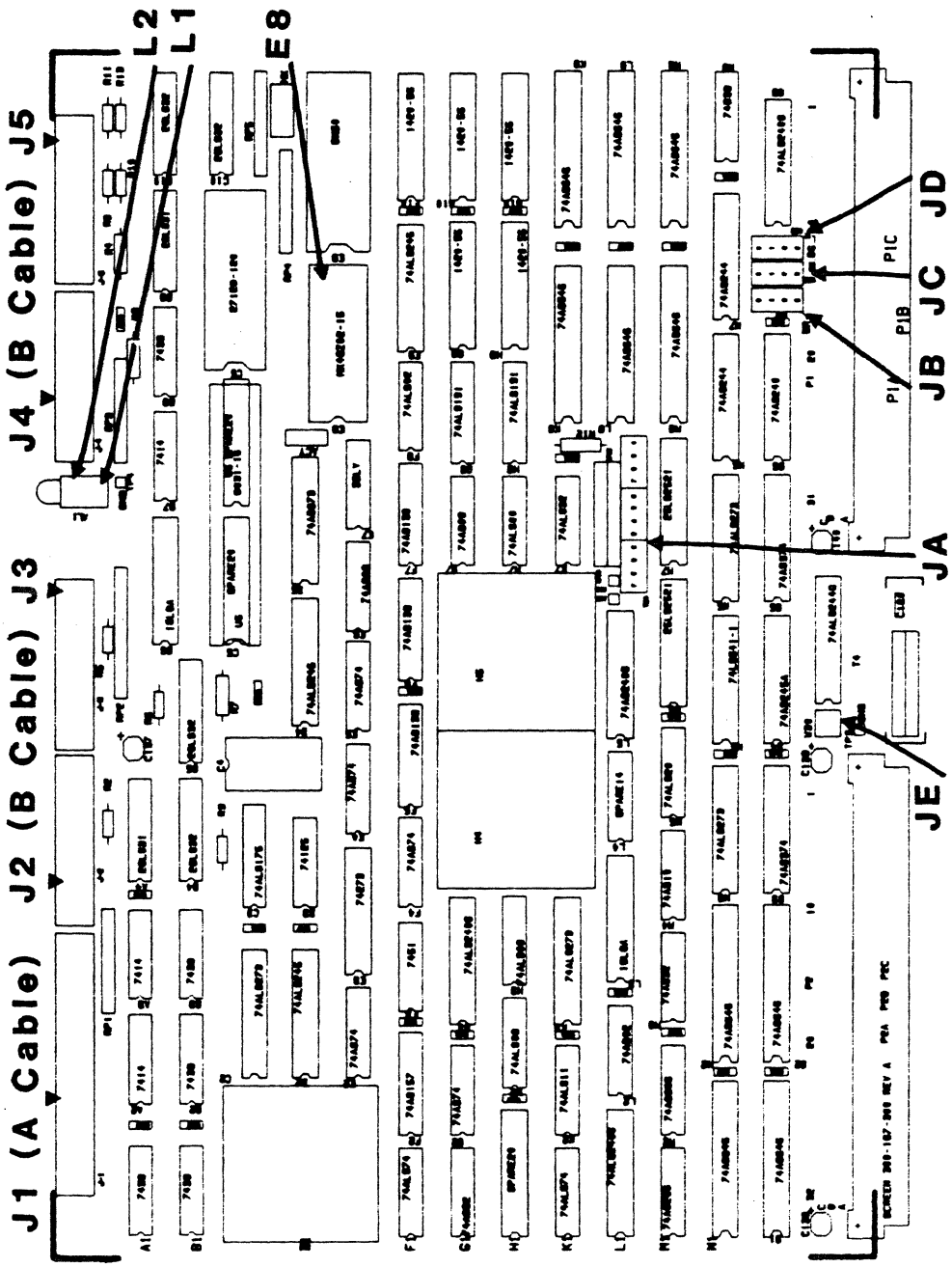
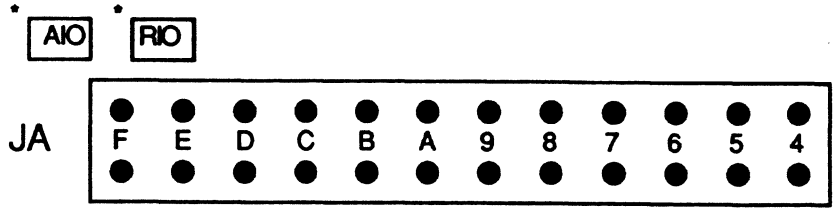


FIGURE 2-1. 712 - COMPONENT LOCATION

2.2.1 Base Address Selection

Jumper block JA controls the base address. Table 2-1 shows how to set the jumpers for commonly used base addresses. Inserting a jumper makes the 712 respond to a 0 on that address line; removing a jumper makes the 712 respond to a 1. Connect the jumper between similar pin numbers on each block. (The 712 uses bits 1 through 3 to determine which register is being accessed.) The 712 is an A16 Slave, and responds to address modifier 02DH, and optionally 029H.



* These two pins are test points, not address jumpers

FIGURE 2-2. BASE ADDRESS JUMPER BLOCK

Screen Label —>	F	E	D	C	B	A	9	8	7	6	5	4
Address:												
EE80*	O	O	O	I	O	O	O	I	O	I	I	I
EE40	O	O	O	I	O	O	O	I	I	O	I	I
0800	I	I	I	I	O	I	I	I	I	I	I	I
0100	I	I	I	I	I	I	I	O	I	I	I	I

O = Out; I = In;

* Standard Factory Configuration

TABLE 2-1. BASE ADDRESS SELECTION

2.2.2 Bus Request and Bus Grant Lines

The 712 uses the Bus Request and Bus Grant lines to become bus master. In VMEbus arbitration, there are four Bus Request/Grant levels: 0 through 3. The 712 drives one Bus Request line according to the jumper scheme you choose. The arbiter drives the four Bus Grant In lines: BG0IN* through BG3IN*. If the 712 receives a Bus Grant, and is not requesting the bus, it passes the grant by driving the appropriate Bus Grant Out line: BG0OUT* through BG3OUT*.

2.2.2 Bus Request and Bus Grant Lines (continued)

Select a request level by jumpering one Bus Request (BR0* through BR3*), one Bus Grant In, and one Bus Grant Out line to match the selected request level. Jumper the remaining Bus Grant In/Out lines so that the incoming signal passes through the board (i.e., jumper BGxIN* to BGxOUT*, where x represents the remaining grant levels).

For example, Figure 2-3 shows the jumpering scheme for level 0 (Figure 2-3A shows the jumper blocks as they actually appear on the board; 2-3B is labeled for this example): jumper JB1 to JB5; then jumper JC1 to JC5, and JD1 to JD5. Jumper the remaining Grant levels from JC6 to JD2, JC7 to JD3, and JC8 to JD4. Factory configuration: Bus Request Level 3.

NOTE

Some VME processors only support Bus Request Level 3.

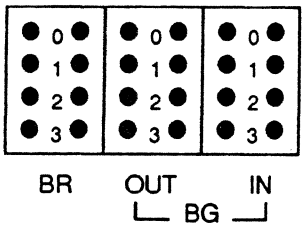


Figure 2-3A. Actual Board Layout

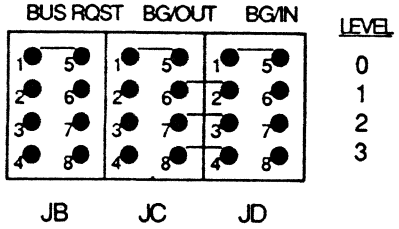


Figure 2-3B. Sample Jumpering Scheme

FIGURE 2-3. JUMPERING BUS REQUEST AND BUS GRANT LEVELS

2.2.3 Parallel Arbitration

If you are using the 712 in parallel arbitration, and the Bus Grant Out lines must be isolated from the next slot's Bus Grant In Lines, remove all jumpers between JC 5-8 and JD 1-4 (See Figure 2-3B).

2.3 FORMAT PARAMETERS AND MAINTENANCE MODE LOCKOUT JUMPER

When jumper JE 1-2 is removed, you may only modify format field lengths for fields 5 and 5A, and you may only execute the diagnostic portion of the Maintenance mode.

2.3 FORMAT PARAMETERS AND MAINTENANCE MODE LOCKOUT JUMPER (continued)

When jumper JE 1-2 is installed, you may set all format parameters and have unrestricted use of the Maintenance mode.

The non-diagnostic portion of the Maintenance mode is proprietary to Xylogics and subject to change without notice.

2.4 SELF TEST DISABLE

When jumper JE 3-4 is installed, the 712 does not execute the Self Test on power-up.

2.5 PROMS AND PALS

LOCATION	PART NUMBER	TYPE
C6	180-002-151	EPROM
B6	181-001-022	PAL
H1	181-001-023	PAL
L3	181-001-024	PAL

TABLE 2-2. PROM / PAL PART NUMBER AND LOCATION

2.6 LIGHT EMITTING DIODES

The 712 has two light emitting diodes (LEDs). L1 (BSY) is the Busy LED (it is located closest to the printed circuit board). L2 (ERR) is the Error LED (it straddles L1).

2.7 BOARD LABELS / REVISION CONTROL

All Xylogics controllers use various revision control labels. This information is important when discussing configuration issues with us. Please familiarize yourself with your board revision levels before contacting us.

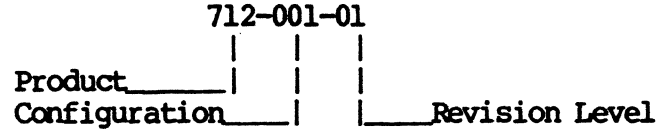


FIGURE 2-4. SAMPLE PART NUMBER

2.8 PREPARING THE COMPUTER SYSTEM FOR INSTALLATION

The backplane of your system must provide a VMEbus slot for the 712. The slot must be capable of handling a bus master, and the power source must handle the power consumption of the entire system, including the 712.

2.8.1 Backplane Jumpers

Remove any jumpers that short, or cause the Interrupt Acknowledge (IACK IN/OUT) and DMA Grants (BG 0-3 IN/OUT) to bypass the slot in which you are installing the 712.

2.8.2 Card Cage Slot

The card cage must have a slot at the proper DMA priority available for the 712. The 712 uses DMA to transfer data and IOPBs. Placement of the 712 in the DMA priority chain may be critical. The amount of bus bandwidth it uses will be high at times; this may affect other boards in the system. Likewise, other boards may not allow enough time for the 712 to DMA enough data to keep up with the disk; consider this when choosing a slot. If the 712 does not get a high enough priority, then its DMA falls behind what the disk requires, and it has to wait until the next revolution before continuing the transfer. If the 712 priority is high, it gets enough DMA time, but other boards having insufficient buffers may starve from lack of DMA time. The priorities must be balanced for your system to work properly.

2.8.3 Power Considerations

The 712 affects the power consumption of the entire computer system. The 712 uses +5 volts (4.75 to 5.25 volts) at 4.1 amps. Be sure the power supplies can handle the entire power load. Readjust the voltages AFTER plugging in the 712. A power supply that is just adequate may cause intermittent and unusual problems due to noise generated by occasionally going into overcurrent protection.

2.9 PREPARING THE DISK DRIVE FOR INSTALLATION

Follow the manufacturer's instructions for unpacking and inspecting the disk drive.

Configure the drive for use with the 712. This entails setting up such parameters as the Unit Select and number of sectors per track. Consult the drive manual for the exact method of configuring your drive.

2.9.1 Drive Unit Select

A set of switches on one of the drive's internal circuit cards usually selects the drive Unit Number. The 712 accesses drives with Unit Numbers ranging from 1 through 7. Set the first drive to Unit 1. The 712 adds one to the specified Unit Number because ESDI drives do not support Unit 0. The 712 defaults to Unit 0.

712 IOPB UNIT NUMBER	PHYSICAL DRIVE NUMBER
0	1
1	2
2	3
:	:
6	7

TABLE 2-3. 712 IOPB UNIT NUMBERS

2.9.2 Number of Sectors Per Track

Switches on one of the drive's internal circuit cards usually select the number of sectors per track. The 712's overhead per sector varies from drive to drive. To calculate the overhead for the drive you are using, look up the section on sector formats in your ESDI Drive Manual. For hard sector drives, the sum of the following bytes equals the 712 overhead:

Intersector Gap (ISG) + Phase Lock Oscillator (PLO) Sync + Address Sync + Address Field + Address CRC + Address Pad + (Gap 2 + Write Splice) + Data PLO Sync + Data Sync + Data ECC + Data Pad. Add your number of data bytes per sector to the sum of these bytes and you will have the total sector size in bytes for your drive.

For soft sector drives, the sum of the following bytes equals the 712 overhead:

ISG + Address Mark + Pad + PLO Sync + Address Sync + Address Field + Address CRC + Address Pad + Write Splice + Data PLO Sync + Data Sync + Data ECC + Data Pad + Format Speed Tolerance Gap. Add your number of data bytes per sector to the sum of these bytes and you will have the total sector size in bytes for your drive.

If you are using the sector slip feature, the number of sectors available to the program is the total number of physical sectors on the drive less the spares.

2.9.3 Sector and Index Pulses

The "A" cable (Control) provides the sector and index pulses. (Hard sector drives use the sector pulse; soft sector drives use the address mark.)

2.10 INSTALL AND CABLE THE 712

2.10.1 Install the 712

Place the 712 into the computer card cage; make sure it is firmly seated. Be careful not to dislodge any socketed ICs. Situate the disk drive and connect it to its power source.

2.10.2 Cable the Subsystem

2.10.2.1 Connect the "A" Cable (Control)

Install the "A" cable, observing "pin 1" markings on both ends. This cable connects to the 34-pin connector on the 712, and to the "A" cable connector on the drive. Use the "in" connector on the drive if there are two 34-pin connectors marked "in" and "out". The other connector should have a terminator, or the terminator should be built into the drive. (Only cable one disk drive for the initial system check. You can connect additional disk drives later.)

2.10.2.2 Connect The "B" Cable (Radial)

Install a "B" cable (20-pin cable) from any "B" cable port on the 712 to the appropriate connector on the disk drive. The 712's "B" cable ports are not keyed to the logical disk drive Unit Number (i.e., Drive 1 can connect to Port 0 of the 712). When installing this cable, make sure the black stripe on the shielded cable lines up with the "pin 1" markings on the controller and drive.

2.10.2.3 Mechanical Restraint

Make sure the "A" and "B" cables are mechanically restrained at both ends to prevent them from accidentally disconnecting. Using "pull tabs" on the cables greatly reduces connector damage.

2.10.2.4 Disk Drive Grounds

Install a ground braid wire between the ground terminal on the disk drive(s) and the computer system ground.

2.11 INITIAL TESTS

This section relies upon your familiarity with your computer system's monitor and diagnostics.

2.11.1 Power-up and Self Test

The 712 initiates a self test upon power-up. The Error LED (L2) lights for a moment, and then goes off. If L2 remains on, and the Fatal Error Register indicates an IRAM Checksum error, then you need to load good parameters into the IRAM. Otherwise, if L2 remains on, the board is not functioning properly (the Fatal Error Register may indicate the nature of the problem). Contact Xylogics for further assistance.

NOTE

Check the power supply voltage to ensure it is within limits (4.75 to 5.25 volts).

2.11.2 Drive Ready

Spin the drive up and wait for it to become ready. Issue a Read Drive Parameters IOPB. The Drive Status byte indicates the drive status at execution time. If DRDY is not set, recheck the drive cable connections and try again. If you are still unable to get the proper status, check the +5V supply on the bus. If the problem persists, check the disk drive for functionality with an off-line tester.

2.12 DIAGNOSTICS

When you run your diagnostics:

- Format the disk with either a diagnostic or format program.
- Run a full pass of your diagnostic (or determine that the system is working properly.
- Cable and test any additional drives.

2.13 CABLING MULTIPLE DRIVES

If you are using multiple drives, make sure the "A" and "B" cables are properly connected; observe the "pin 1" markings on both the cables and the drives.

2.13.1 Terminator

Remove the terminator from the drive currently connected to the controller. Install the terminator in the last drive in the chain.

2.13.2 "A" Cable (Daisy-chain)

Connect the "A" cable directly from the first drive in the chain to the 712; connect the additional drives together, starting with the initial drive. For example, the 712 connects to Drive 1; Drive 1 connects to Drive 2. Be careful; do not reverse the cables. Terminate the "A" cable at the last drive in the chain. The "A" cable's maximum total length is 3 meters (9.8 feet). See Figure 2-5.

2.13.3 "B" Cable (Radial)

The "B" cables connect directly from each drive to a "B" cable port on the 712. A "B" cable may be up to 3 meters (9.8 feet) long.

2.13.4 Unit Select

If you are daisy-chaining drives, assign each drive a unique Unit Select number. The 712 accesses drives with Unit Numbers from 1 through 7.

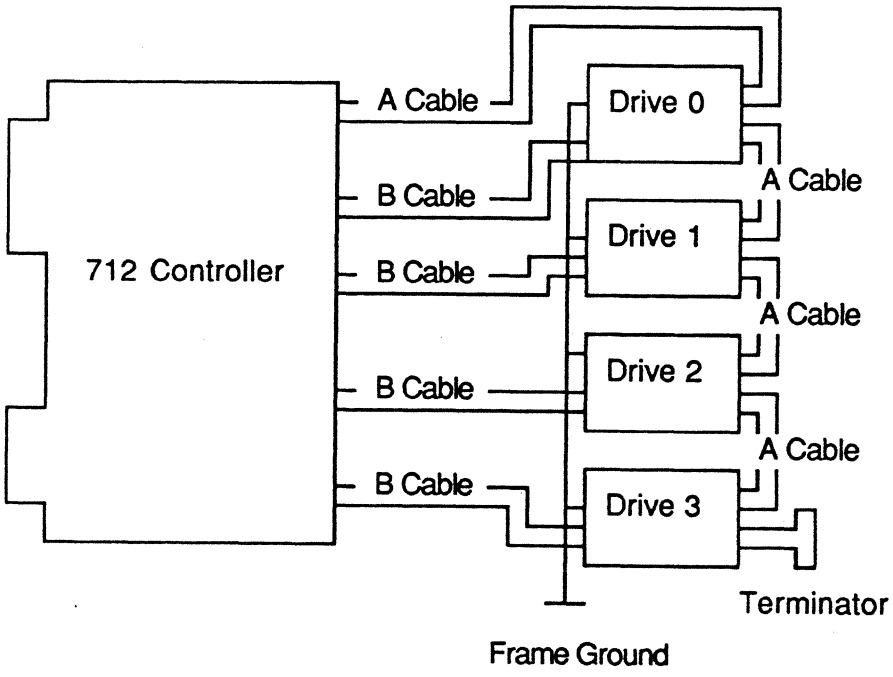


FIGURE 2-5. CABLING MULTIPLE DRIVES

SECTION 3: THE 712 REGISTERS

3.0 GENERAL

The 712 programming interface is based on the use of seven, one-byte long, I/O registers. The bus address jumpers define the base address of the register set. Table 3-1 lists the registers along with the address offset from the base address. The 712 responds to either bytes or 16-bit words; when it responds to words, only 8 bits are valid.

The registers have one function when read, and another when written. The following subsections detail their definitions.

REGISTER	OFFSET
IOPB ADDRESS BYTE 0 (Least Significant Byte)	1
IOPB ADDRESS BYTE 1	3
IOPB ADDRESS BYTE 2	5
IOPB ADDRESS BYTE 3 (Most Significant Byte)	7
IOPB ADDRESS MODIFIER	9
CONTROL AND STATUS REGISTER	B
FATAL ERROR REGISTER	D

TABLE 3-1. REGISTER OFFSETS

3.1 IOPB ADDRESS REGISTERS

The first four registers define the 32-bit address of an IOPB or IOPB chain. When these registers are written, the 712 interprets it as the address of the IOPB or IOPB chain to be executed. When read, and Remove IOPB (RIO) is set, the registers point to the IOPB or IOPB chain completed by the 712.

The protocol for reading and writing this address register is defined by the use of the Add IOPB (AIO) and Remove IOPB (RIO) bits in the Control and Status Register (See Section 3.3).

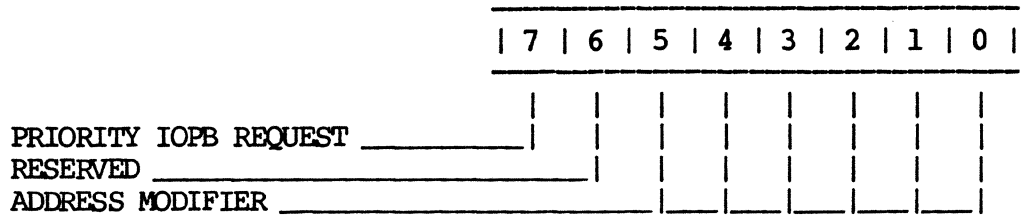
3.2 IOPB ADDRESS MODIFIER / PRIORITY IOPB REGISTER

This register defines the IOPB address modifier. (Address modifiers are used for many purposes, such as memory mapping, privilege levels, and addressing range. Please consult the VMEbus

3.2 IOPB ADDRESS MODIFIER / PRIORITY IOPB REGISTER (continued)

Specification Manual for more information on using address modifiers.) This register also specifies whether an IOPB has priority over the current set of IOPBs in the 712 command queue. Section 3.3 defines the protocol for reading and writing this register.

PRIORITY IOPB REGISTER



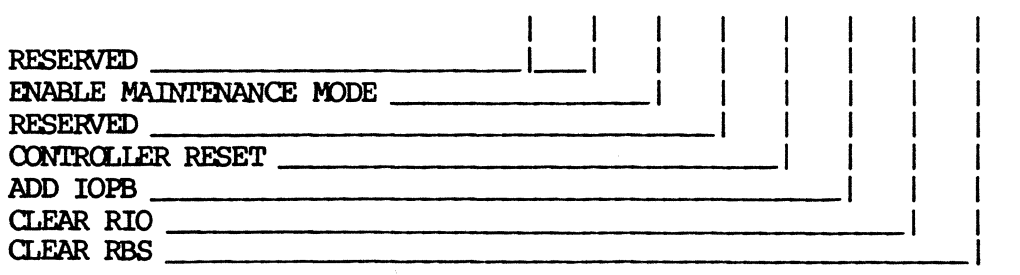
BIT	MNEMONIC	DESCRIPTION
7	PRIO	PRIORITY IOPB REQUEST - When set, the IOPB, or IOPB chain, precedes all others (except the one in process) in the command queue.
6		RESERVED.
5-0	AM	ADDRESS MODIFIER - Most systems use the standard AM code of 3D. See the VMEbus Specification Manual.

3.3 CONTROL AND STATUS REGISTER

When written, this register provides the host with control of the 712 operation; when read, it provides the host with 712 status information. Section 3.3.1 defines the bits in this register when written; Section 3.3.2 defines the bits when read.

3.3.1 Control Register (Write)

CONTROL REGISTER (Write)



3.3.1 Control Register (Write) (continued)

BIT	MNEMONIC	DESCRIPTION
7-6		RESERVED.
5	MM	ENABLE MAINTENANCE MODE - Setting MM and AIO places the 712 in Maintenance mode. This mode supports a different Register protocol and is used as a diagnostic tool. Section 8 outlines the Maintenance mode.
4		RESERVED.
3	CRST	CONTROLLER RESET - This bit signals the 712 microprocessor to perform a "soft" reset; it deselects (releases dual port) all the drives, stops the DMA and Disk Sequencers (potentially during sector transfers), and cancels any IOPBs in the queue. When the Controller Reset completes, the 712 resets the CSR to zero. CRST does not initiate a Power-up Self Test.
2	AIO	ADD IOPB - The host sets AIO to indicate that the 712 should execute the IOPB (chain) at the address pointed to by the IOPB Address and Address Modifier Registers. Essentially, AIO commands the 712 to begin executing a new IOPB (chain). As soon as the host asserts this bit, the 712 asserts the AIO Pending (AIOP) bit in the Status Register; this indicates that the 712 has received the AIO signal but has not yet processed the address of the new chain. AIOP is negated in the Status Register after the 712 internally stores the new (chain) address. The 712 can store up to 47 IOPB addresses in this manner. Reasserting AIO if AIOP is asserted in the Status Register violates the Register protocol.
1	CRIO	CLEAR RIO - The host sets CRIO to clear RIO in the Status Register. Typically, the host sets CRIO after it reads the address of a completed IOPB chain from the IOPB Address and Modifier Registers.

NOTE

A Controller Reset takes up to one second to complete.

3.3.1 Control Register (Write) (continued)

BIT	MNEMONIC	DESCRIPTION
1	CRIO	CLEAR RIO - (continued) Clearing RIO enables the 712 to update the IOPB Address and Modifier Registers with the address and address modifier of a newly completed IOPB chain. Clearing RIO if it is not asserted in the Status Register violates the Register protocol.
0	CRBS	CLEAR RBS - The host sets the Clear Register Busy (CRBS) bit to clear the RBS bit in the Status Register. Clearing RBS effectively releases the registers for use by another host (See Section 8.8.2). (CRBS is only relevant in a multiprocessor environment.)

3.3.2 Status Register (Read)

STATUS REGISTER (Read)

7	6	5	4	3	2	1	0

BUSY
FATAL ERROR
MAINTENANCE MODE ACTIVE
RESERVED
CONTROLLER RESET ACTIVE
AIO PENDING
REMOVE IOPB
REGISTER BUSY SEMAPHORE

BIT	MNEMONIC	DESCRIPTION
7	BUSY	BUSY - The 712 is executing IOPBs. The 712 sets BUSY when it clears AIOP to acknowledge the first IOPB address; it clears BUSY after completing all the IOPBs with no new ones pending (within 500 microseconds of the host clearing RIO on the last IOPB). This bit is redefined when the 712 is in Maintenance mode (See Section 8.6).

3.3.2 Status Register (Read) (continued)

BIT	MNEMONIC	DESCRIPTION
6	FERR	FATAL ERROR - The 712 detected a fatal hardware error. A Controller Reset clears this bit. The 712 asserts FERR under the following conditions: (1) Maintenance Mode Test Failure; (2) Power-up Self Test Failure; (3) IOPB Checksum Miscompare; (4) IOPB DMA Fatal; (5) IOPB Address Alignment Error; (6) Firmware Error; (7) Illegal Maintenance Mode Test Number; and (8) ACFAIL Asserted.
5	MMA	MAINTENANCE MODE ACTIVE - When set, the 712 is in Maintenance mode (See Section 8.6).
4		RESERVED.
3	RSTA	CONTROLLER RESET ACTIVE - The host set Controller Reset in the Control Register and the 712 is currently resetting itself.
2	AIOP	AIO PENDING - When set, AIO has been set in the Control Register, but the 712 has not acknowledged its receipt. When clear, AIO may be set again. After the host reads the address and modifier, it must clear RIO by writing Clear RIO (CRIO) in the Control Register.
1	RIO	REMOVE IOPB - The 712 sets RIO after completing an IOPB, or a chain of IOPBs, and placing the address in the IOPB Address and Address Modifier Registers.
0	RBS	REGISTER BUSY SEMAPHORE - RBS provides a means of allowing multiple hosts to share access to the 712 registers without simultaneous access (See Section 8.8.2). (RBS is only relevant in a multiprocessor environment.)

3.4 FATAL ERROR REGISTER

If a fatal error occurs, the 712 returns the appropriate Completion Code in this register. Table 3-2 lists the fatal error codes; Section 6.5 describes them.

CODE	DESCRIPTION
E0	IRAM Checksum Failure
E1	IRAM Self Test Failure
E2	EPROM Checksum Failure
E3	Maintenance Test 3 Failure (DSKCEL RAM)
E4	Maintenance Test 4 Failure (Header Shift Register)
E5	Maintenance Test 5 Failure (VMEDMA Registers)
E6	Maintenance Test 6 Failure (REGCEL Chip)
E7	Maintenance Test 7 Failure (Buffer Parity)
E8	Maintenance Test 8 Failure (Disk FIFO)
E9-EF	Reserved
F0	IOPB Checksum Miscompare
F1	IOPB DMA Fatal
F2	IOPB Address Alignment Error
F3	Firmware Error
F5	Illegal Maintenance Mode Test Number
F6	ACFAIL Asserted

TABLE 3-2. FATAL ERROR CODES

SECTION 4: IOPB DESCRIPTION

4.0 GENERAL

The Input/Output Parameter Block (IOPB) passes messages between the 712 and host software: software passes the type of transfer, disk address, data address, and count to the 712; the 712 returns the transfer status and possibly the ending addresses upon command completion. This section begins with the standard IOPB for most data transfer commands and follows with variations of the IOPB.

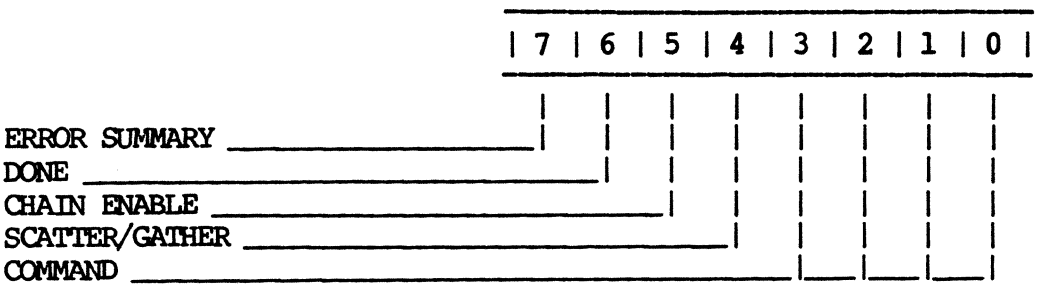
4.1 STANDARD IOPB

The 712 uses the standard IOPB for data transfer commands and some general purpose commands.

STANDARD IOPB

	7	6	5	4	3	2	1	0
00	ERRS	DONE	CHEN	SGM	COMMAND			
01	COMPLETION CODE							
02	0	SR	0		ATTN	0	CMPL	DRDY
03	INTERNAL STATUS							
04	SUBFUNCTION CODE							
05	FIXD	0		BHT	0	UNIT		
06	LINK LIST LENGTH					INTERRUPT LEVEL		
07	INTERRUPT VECTOR							
08	COUNT HIGH							
09	COUNT LOW							
0A	CYLINDER HIGH							
0B	CYLINDER LOW							
0C	HEAD							
0D	SECTOR							
0E	0		DATA OR LINK ADDRESS MODIFIER					
0F	PRIO	0	NEXT IOPB ADDRESS MODIFIER					
10	DATA OR LINK ADDRESS HIGH							
11	DATA OR LINK ADDRESS							
12	DATA OR LINK ADDRESS							
13	DATA OR LINK ADDRESS LOW							
14	NEXT IOPB ADDRESS HIGH							
15	NEXT IOPB ADDRESS							
16	NEXT IOPB ADDRESS							
17	NEXT IOPB ADDRESS LOW							
18	IOPB CHECKSUM HIGH							
19	IOPB CHECKSUM LOW							

4.1.1 IOPB Byte 0 (Command)



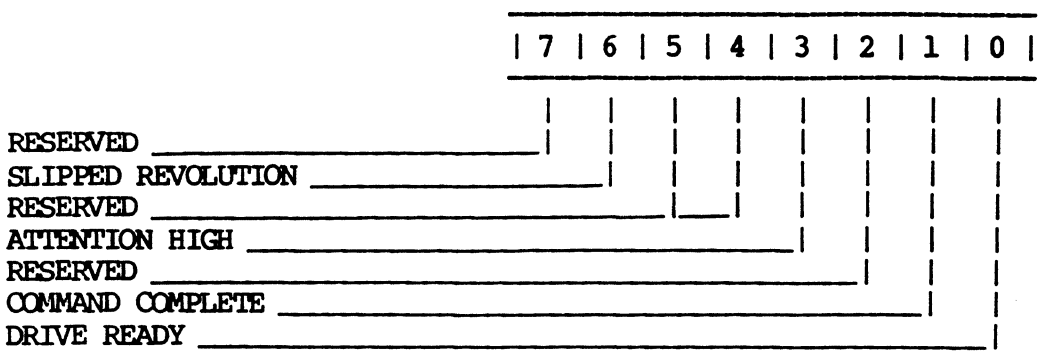
BIT	MNEMONIC	DESCRIPTION
7	ERRS	ERROR SUMMARY - ERRS is only valid if DONE is set. When set, a hard or soft error occurred during IOPB processing. When clear, the 712 successfully completed the IOPB.
6	DONE	DONE - When set, the IOPB is complete; if chained, software may remove the IOPB from the chain. Software must relink the chain in the same order as before, and cannot move IOPBs in memory.
5	CHEN	CHAIN ENABLE - When set, the Next IOPB Address Modifier and Next IOPB Address point to the next chained IOPB. When clear, this IOPB is not chained to another IOPB. If CHEN and IEC are set, the 712 returns the whole chain with one RIO; if CHEN is set and IEC is clear, the 712 returns one IOPB at a time.
4	SGM	SCATTER/GATHER MODE - When set, the IOPB is either a scatter (read) or a gather (write) transfer; a linked list describes the number of 16-bit words and to what address the 712 transfers each section of the data. The link address modifier and the link address specify the link list location. When clear, this IOPB specifies the data transfer address; the data is transferred to/from contiguous memory. SGM is only valid for standard Reads and Writes.
3-0	COMM	COMMAND - See Table 4-2.

4.1.2 IOPB Byte 1 (Status Byte 1)

After the 712 executes the IOPB, it sets DONE and posts a Completion Code in this byte. (Completion Codes are only valid if DONE is set.) A code of 0x indicates a successful completion; any other value indicates an error occurred (See Section 6).

4.1.3 IOPB Byte 2 (Status Byte 2)

IOPB Byte 2 is the Disk Status byte; it is only valid if DONE is set.



BIT	MNEMONIC	DESCRIPTION
7		RESERVED.
6	SR	SLIPPED REVOLUTION - Sets if the 712 is unable to DMA enough data to keep up with the disk; it waits until the sector comes around on the next revolution.
5-4		RESERVED.
3	ATTN	ATTENTION HIGH - The 712 sets ATTN when the drive reports an error.
2		RESERVED.
1	CMPL	COMMAND COMPLETE - The 712 sets CMPL when the currently selected drive is ready to accept a command.
0	DRDY	DRIVE READY - The 712 sets DRDY when the last drive selected is ready.

4.1.4 IOPB Byte 3 (Status Byte 3)

IOPB Byte 3 is reserved. It reflects the 712's internal status and may be a non-zero value.

4.1.5 IOPB Byte 4 (Subfunction)

IOPB Byte 4 is the Subfunction byte. Subfunction Codes follow a convention that indicates whether the code is generic to all VME controllers, generic to a group of controllers (i.e., disk, tape, etc.), or specific to a particular controller (See Table 4-1).

4.1.5 IOPB Byte 4 (Subfunction) (continued)

The 712 combines standard Command Codes with Subfunction Codes to execute commands. The IOPB Command Code and Subfunction Code fields define the required operation. Table 4-2 lists the 712 Command and Subfunction Codes.

SUBFUNCTION CODES	CLASS
00-1F	Generic To All
20-3F	Generic Tape
40-5F	772-Specific
60-7F	Reserved
80-9F	Generic Disk
A0-AF	751-Specific
B0-BF	712-Specific
C0-FF	Reserved

TABLE 4-1. SUBFUNCTION CODE CLASSES

CODE	COMMAND	SUBFUNCTION	DESCRIPTION
0	NOP	00	No Operation
1	WRITE	00	Normal Write
2	READ	00	Normal Read
3	SEEK	00	Report Current Address
		01	Seek and Report Current Address
		02	Seek Start and Report Completion Immediately
4	DRIVE RESET	00	Drive Reset (With RTZ)
		80	Fault Clear
5	WRITE PARAMETERS	00	Write Controller Prmtrs.
		80	Write Drive Parameters
		81	Write Format Parameters
		B0	Store Format Configuration
6	READ PARAMETERS	00	Read Controller Prmtrs.
		80	Read Drive Parameters
		81	Read Format Parameters
		B0	Read Drive Status Extnd.
		B1	Show Drive Configuration

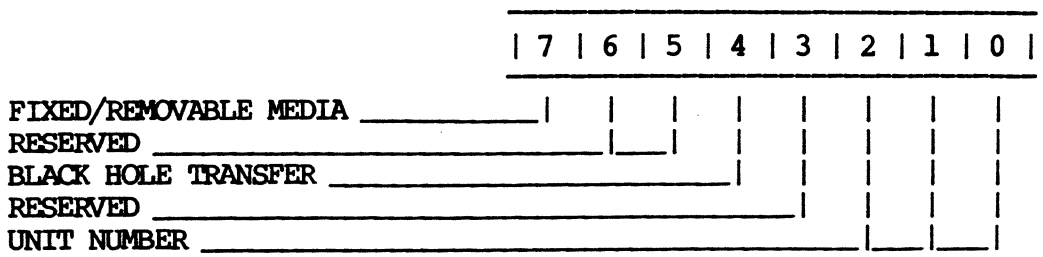
TABLE 4-2. 712 COMMAND/SUBFUNCTION CODES

4.1.5 IOPB Byte 4 (Subfunction) (continued)

CODE	COMMAND	SUBFUNCTION	DESCRIPTION
7	EXTENDED WRITE	80	Write Track Headers
		81	Write Track Format
		82	Write Header, Header Verify, Data, and Data ECC
		B0	Write Defect Map
8	EXTENDED READ	80	Read Track Headers
		81	Verify Data
		82	Read Header, Header Verify, Data, and Data ECC
		B0	Read Defect Map
9	DIAGNOSTICS	00	Self Test
A-B	RESERVED		
C	SEND OPTIONAL	B0	ESDI Command Without Status
		B1	ESDI Command With Status
D-F	RESERVED		

TABLE 4-2. 712 COMMAND/SUBFUNCTION CODES (continued)

4.1.6 IOPB Byte 5 (Unit)

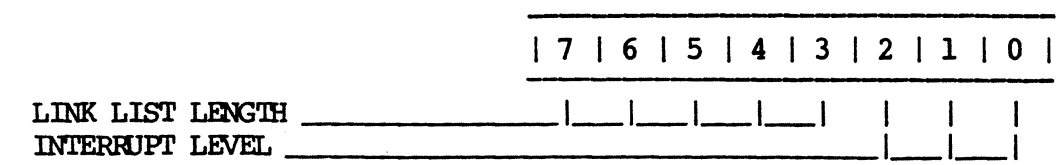


BIT	MNEMONIC	DESCRIPTION
7	FIXD	FIXED/REMOVABLE MEDIA - When set, the 712 is accessing the fixed media portion of a disk drive. When clear, the 712 is accessing the removable media portion of a drive. This bit allows you to treat a fixed/removable drive as two separate disk drives. See Section 8.16.

4.1.6 IOPB Byte 5 (Unit) (continued)

BIT	MNEMONIC	DESCRIPTION
6-5		RESERVED.
4	BHT	BLACK HOLE TRANSFER - When set, the 712 does not increment the bus address during a data transfer; IOPB transfers occur normally. When clear, the 712 does increment the bus address.
3		RESERVED.
2-0	UNIT	UNIT NUMBER - This value specifies the Unit Number of the attached drive to which the transfer is directed. ESDI drives support Unit Numbers 1 through 7; the 712 adds 1 to the Unit Number specified in the IOPB, i.e., Unit Number 0 equals physical drive 1, Unit Number 1 equals physical drive 2...Unit Number 6 equals physical drive number 7.

4.1.7 IOPB Byte 6 (Interrupt Level)



BIT	MNEMONIC	DESCRIPTION
7-3	LLL	LINK LIST LENGTH - Bits 7-3 specify the length, in elements, of a linked list for Scatter/ Gather commands. Each element refers to an 8-byte block in the linked list. See Table 8-2.
2-0	INTL	INTERRUPT LEVEL - The 712 uses these bits as the VMEbus hardware interrupt level when it completes the IOPB. The 712 will not interrupt if bits 0 through 2 are clear.

NOTE

Depending on the command, Bytes 6 through 13 have different definitions (See Sections 4.2 through 4.4).

4.1.8 IOPB Byte 7 (Interrupt Vector)

IOPB Byte 7 determines the interrupt vector that the 712 uses upon command completion. This byte is not valid if the interrupt level is zero.

4.1.9 IOPB Bytes 8 and 9 (Count)

Byte 8 is Count High; Byte 9 is Count Low. These bytes specify the number of sectors to be transferred in a data transfer IOPB. The Format command uses this count to determine the number of tracks to format.

4.1.10 IOPB Bytes A and B (Cylinder)

Byte A is Cylinder High; Byte B is Cylinder Low. These bytes specify the starting cylinder address for a transfer.

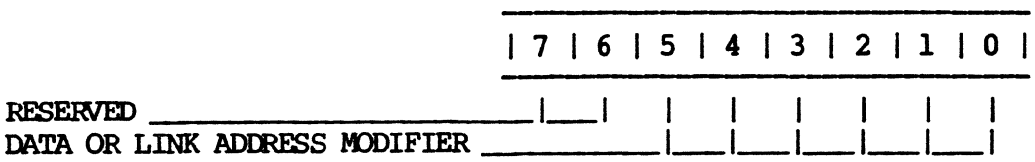
4.1.11 IOPB Byte C (Head)

IOPB Byte C specifies the starting head number for a transfer.

4.1.12 IOPB Byte D (Sector)

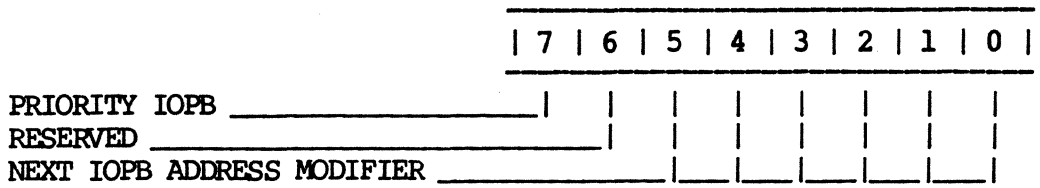
IOPB Byte D specifies the starting sector number for a transfer.

4.1.13 IOPB Byte E (Data or Link Address Modifier)



BIT	DESCRIPTION
7-6	RESERVED.
5-0	DATA OR LINK ADDRESS MODIFIER - If SGM is set, bits 5 through 0 specify the Link List Address Modifier; if SGM is clear, this field specifies the Data Address Modifier. The 712 uses these modifiers to complete the address.

4.1.14 IOPB Byte F (Next IOPB Address Modifier)



BIT	MNEMONIC	DESCRIPTION
7	PRIO	PRIORITY IOPB - If PRIO was set in the Address Modifier Register when AIO was set, setting PRIO in Byte F indicates that this is a priority IOPB.
6		RESERVED.
5-0		NEXT IOPB ADDRESS MODIFIER - The Next IOPB Address Modifier, along with the Next IOPB Address, point to the next IOPB in the chain.

4.1.15 IOPB Bytes 10 through 13 (DMA Data Address)

IOPB Byte 10 is DMA Data Address High; Byte 13 is DMA Data Address Low. These bytes comprise the data or link list address pointers. The 712 uses these bytes with the data or link list address modifier to point to the data or linked list address. If SGM is set, this address points to the linked list; if SGM is clear, this address points to the data address.

4.1.16 IOPB Bytes 14 through 17 (Next IOPB Address)

IOPB Byte 14 is Next IOPB Address High; Byte 17 is Next IOPB Address Low. These bytes comprise the Next IOPB Address pointers. The 712 uses these bytes with the Next IOPB Address modifier to point to the next IOPB in the chain (if CHEN is set in Byte 0).

4.1.17 IOPB Bytes 18 and 19 (IOPB Checksum)

Byte 18 is IOPB Checksum High; Byte 19 is IOPB Checksum Low. The 712 calculates the checksum by adding the IOPB bytes. See Section 8.15.

4.1.18 IOPB Bytes 1A and 1B (ECC Pattern Word)

Byte 1A is ECC Pattern Word High; Byte 1B is ECC Pattern Word Low. These bytes are required for ECC Mode 0 and may be required for Mode 2 (See Section 6.4).

4.1.19 IOPB Bytes 1C and 1D (ECC Offset Word)

Byte 1C is ECC Offset Word High; Byte 1D is ECC Offset Word Low. These bytes are required for ECC Mode 0 and may be required for Mode 2 (See Section 6.4).

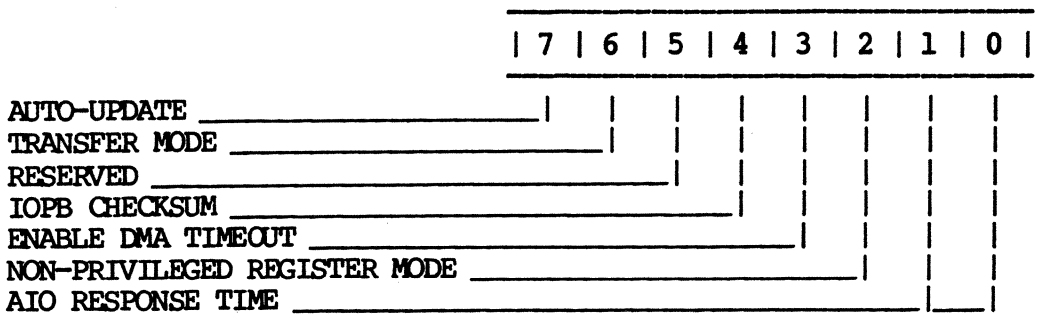
4.2 CONTROLLER PARAMETERS IOPB

This IOPB sets and reads various controller parameters. The 712 uses the standard IOPB, but redefines bits in Bytes 8, 9, A, B, C, and E.

CONTROLLER PARAMETERS

	7	6	5	4	3	2	1	0
00	ERRS	DONE	CHEN	SGM	COMMAND			
01	COMPLETION CODE							
02	0	SR	0		ATTN	0	CMPL	DRDY
03	INTERNAL STATUS							
04	SUBFUNCTION CODE							
05	FIXD	0				UNIT		
06	0					INTERRUPT LEVEL		
07	INTERRUPT VECTOR							
08	AUD	TMOD	DACF	ICS	EDT	NPRM	AIOR	
09	TDT		0	ROR	0			
0A	OVS	COP	IEC	ASR	ZLR	RBC	ECCM	
0B	THROTTLE							
0C	RELEASE LEVEL							
0D	0							
0E	CONTROLLER TYPE							
0F	PRIO	0	NEXT IOPB ADDRESS MODIFIER					
10	PROM PART NUMBER HIGH							
11	PROM PART NUMBER LOW							
12	REVISION							
13	SUBREVISION							
14	NEXT IOPB ADDRESS HIGH							
15	NEXT IOPB ADDRESS							
16	NEXT IOPB ADDRESS							
17	NEXT IOPB ADDRESS LOW							
18	IOPB CHECKSUM HIGH							
19	IOPB CHECKSUM LOW							

4.2.1 IOPB Byte 8 (Controller Parameters A)



BIT	MNEMONIC	DESCRIPTION
7	AUD	AUTO-UPDATE - When set, the 712 updates the IOPB to the transfer's correct ending parameters; it updates the disk address, the sector count, and the data address after completing the transfer or detecting an error. When clear, the 712 only updates the IOPB if an error occurs. The values are then set up so that host software can tell the 712 to continue (the values should point to the sector in error, the correct remaining sector count, and proper data address).
6	TMOD	TRANSFER MODE - When set, the 712 executes data transfers in Longword mode. When clear, it executes transfers in Word mode. (IOPB transfers are always in Word mode.) If a transfer starts on an improper address boundary, the 712 first transfers a byte and/or a word, as necessary, to align boundaries and continues the transfer in the selected mode. The 712 may end the transfer with a byte and/or a word.
5	DACF	DISABLE ACFAIL - When set, the 712 ignores the asserting ACFAIL line on the VMEbus. Normally, ACFAIL causes a fatal error.
4	ICS	IOPB CHECKSUM - When set, the 712 reads the IOPB, compares the checksum it generated during the read with the checksum the software driver appended to the IOPB. The 712 also updates the Checksum bytes in any IOPB if AUD is set. Clearing ICS disables this feature. See Section 8.15.

NOTE

Since this feature adds 50 us to each transfer, it effects the 712's performance.

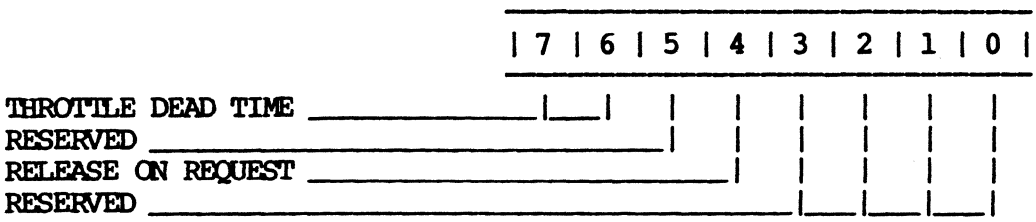
4.2.1 IOPB Byte 8 (Controller Parameters A) (continued)

BIT	MNEMONIC	DESCRIPTION
3	EDT	ENABLE DMA TIMEOUT - When set, the 712 enables a DMA bus error timer. When clear, the 712 relies on the VMEbus transfer timer.
2	NPRM	NON-PRIVILEGED REGISTER MODE - When set, the 712 responds to address modifiers 2DH and 29H. When clear, the 712 only responds to 2DH. (See the VMEbus Specification Manual for more information on address modifiers.)
1-0	AIOR	AIO RESPONSE TIME - These bits respond to the four values that indicate the maximum AIO response time. This is the time from setting AIO to the time the 712 clears it. The shorter the response time, the greater the 712 overhead.

VALUE	TIME
00	100 us (Factory Default)
01	75 us
02	62 us
03	50 us

TABLE 4-3. AIO RESPONSE TIMES

4.2.2 IOPB Byte 9 (Controller Parameters B)

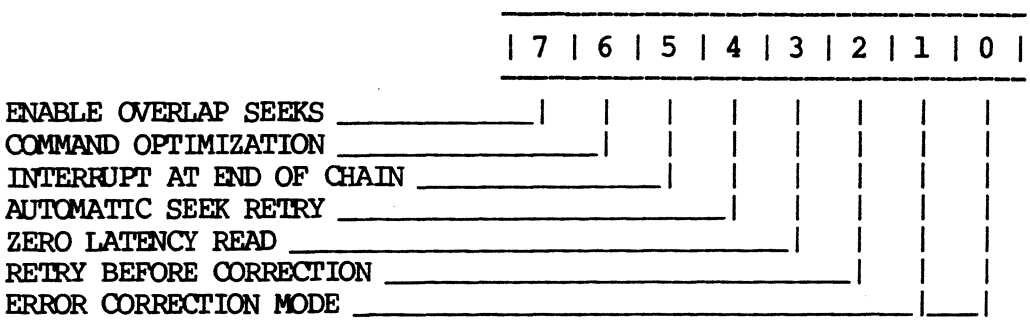


BIT	MNEMONIC	DESCRIPTION
7-6	TDT	THROTTLE DEAD TIME - TDT selects one of four minimum time periods that determines the time the 712 remains off the bus between throttle bursts (See Section 8.12).
5		RESERVED.

4.2.2 IOPB Byte 9 (Controller Parameters B) (continued)

BIT	MNEMONIC	DESCRIPTION
4	ROR	RELEASE ON REQUEST - When set, the 712 releases the bus at the request of other bus masters; otherwise, it continues with the next throttle burst. The 712 monitors the bus request lines and releases bus busy only if another bus request is pending. It completes its specified throttle burst before releasing the bus due to a pending request. When clear, the 712 releases the bus at the end of each throttle burst and rearbitrates if more data transfers are pending.
3-0		RESERVED.

4.2.3 IOPB Byte A (Controller Parameters C)



BIT	MNEMONIC	DESCRIPTION
7	OVS	ENABLE OVERLAP SEEKS - When set, the 712 initiates overlap seeks if more than one drive is present. When clear, the controller does not initiate Overlap Seek operations.
6	COP	COMMAND OPTIMIZATION - When set, the 712 initiates elevator seeks and command optimization. See Section 8.9.
5	IEC	INTERRUPT AT END OF CHAIN — When set, the 712 returns all IOPB chains with one RIO and one interrupt; it does not relink or unlink IOPBs. The RIO address of a completed chain is the address of the first IOPB in the chain. The 712 also uses the interrupt level and vector of the first IOPB in the chain. Clearing IEC disables this feature. (Do not set or clear IEC while the 712 is processing an IOPB chain.)

4.2.3 IOPB Byte A (Controller Parameters C) (continued)

BIT	MNEMONIC	DESCRIPTION
4	ASR	AUTOMATIC SEEK RETRY - When set, the 712 resets the drive, seeks to the commanded cylinder and retries the transfer on Seek and Header Error/Cylinder errors.
3	ZLR	ZERO LATENCY READ - When set, zero latency reads are enabled; when clear, zero latency reads are disabled. See Section 8.7.
2	RBC	RETRY BEFORE CORRECTION - When set, the 712 retries the operation once on an ECC error without calculating the error syndrome. If an error occurs, on the second try, the 712 reverts to the specified Error Correction mode.
1-0	ECCM	ERROR CORRECTION MODE - There are three Error Correction modes. Mode 0 stops a transfer and provides the driver with the error's offset and pattern. The driver performs the actual correction. Mode 1 flags an error and continues the transfer. Mode 2 performs the correction in host memory, flags a soft error, and continues the transfer.

4.2.4 IOPB Byte B (Controller Parameters D)

Bits 0 through 7 are the Throttle (THRO) bits. The throttle is the maximum number of transfers allowed each time the 712 becomes bus master. The throttle value determines the maximum DMA burst length for both data and IOPB DMA transfers. Each bit position represents a binary weight, allowing a throttle from 1 to 256.

VALUE	WEIGHT
0	256
1	1
2	2
3	3
:	:
255	255

TABLE 4-4. THROTTLE VALUES

4.2.5 IOPB Byte C (Release Level)

The 712 returns its release level on a Read Controller Parameters command.

4.2.6 IOPB Byte E (Controller Type)

IOPB Byte E is the Controller Type byte. Xylogics assigns each VME controller a unique controller type code.

CONTROLLER	CODE (H)	
712	12	(ESDI Disk Controller)
751	51	(SMD/SMD-E Controller)
772	72	(Pertec Tape Controller)

TABLE 4-5. CONTROLLER TYPE CODES

4.2.7 IOPB Bytes 10 and 11 (EPROM Part Number)

The 712 returns a portion of the EPROM part number on a Read Controller Parameters command. The 4 nibbles in these 2 bytes refer to the part number's last 4 digits. For example, if the part number is 180-002-151, Byte 10 holds 21H and Byte 11 holds 51H.

4.2.8 IOPB Byte 12 (Revision)

This byte contains the revision level of the EPROM plugged into the board.

4.2.9 IOPB Byte 13 (Subrevision)

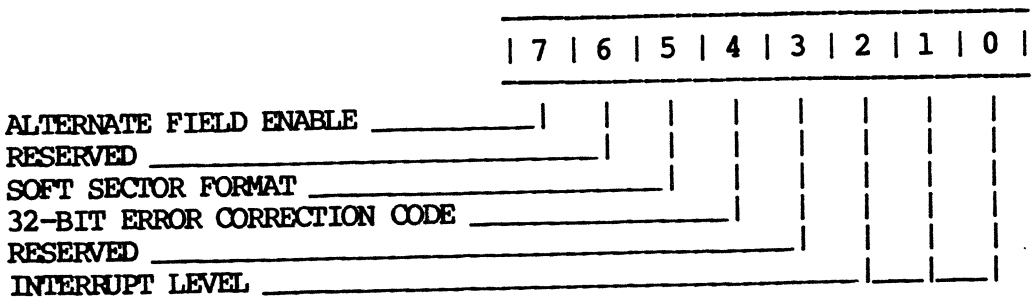
This byte contains the subrevision level of the EPROM plugged into the board.

4.3 DRIVE PARAMETERS IOPB

DRIVE PARAMETERS

	7	6	5	4	3	2	1	0
00	ERRS	DONE	CHEN	0	COMMAND			
01	COMPLETION CODE							
02	0	SR	0		ATTN	0	CMPL	DRDY
03	INTERNAL STATUS							
04	SUBFUNCTION CODE							
05	FIXD	0		0		UNIT		
06	AFE	0	SSF	EC32	0	INTERRUPT LEVEL		
07	INTERRUPT VECTOR							
08	MAX SECTOR LH							
09	HEAD OFFSET							
0A	MAX CYLINDER HIGH							
0B	MAX CYLINDER LOW							
0C	MAX HEAD							
0D	MAX SECTOR							
0E	SECTORS PER TRACK							
0F	PRIO	0	NEXT IOPB ADDRESS MODIFIER					
10	0							
11	0							
12	0							
13	0							
14	NEXT IOPB ADDRESS HIGH							
15	NEXT IOPB ADDRESS							
16	NEXT IOPB ADDRESS							
17	NEXT IOPB ADDRESS LOW							
18	IOPB CHECKSUM HIGH							
19	IOPB CHECKSUM LOW							

4.3.1 IOPB Byte 6 (Drive Parameters)



4.3.1 IOPB Byte 6 (Drive Parameters) (continued)

BIT	MNEMONIC	DESCRIPTION
7	AFE	ALTERNATE FIELD ENABLE - When set, the 712 uses Field 5A to determine sector size. When clear, it uses Field 5. See Section 8.3.2.

NOTE

Using AFE, SSF, or EC32 in different modes on the four connecting disks will have a detrimental effect on the disk subsystem's performance. The 712 must modify the DSKCEL code each time it switches drives.

6		RESERVED.
5	SSF	SOFT SECTOR FORMAT - When set, the 712 reads and writes in Soft Sector mode. When clear, the 712 reads and writes in Hard Sector mode.
4	EC32	32-BIT ECC - When set, the 712 uses a 32-bit ECC on the data. When clear, it uses a redundant header, and a 48-bit ECC on the data. (See the note below bit 7.)
3		RESERVED.
2-0	INTL	INTERRUPT LEVEL - Bits 0 through 2 are the VMEbus Interrupt Level bits. The 712 does not interrupt if the interrupt level is set to zero. See Section 4.1.7.

4.3.2 IOPB Byte 8 (Max Sector/Last Head)

IOPB Byte 8 specifies the max sector value on the last head for use in cylinder sparing. Bytes 0DH and 08H must be equal if cylinder sparing is not used. See Section 8.1.2.

4.3.3 IOPB Byte 9 (Head Offset)

IOPB Byte 9 specifies the drive's head offset value. Use zero for non-fixed/removable drives. Section 8.16 explains using the head offset to access fixed/removable drives.

4.3.4 IOPB Bytes A and B (Max Cylinder)

IOPB Byte A is Max Cylinder High; Byte B is Max Cylinder Low. These bytes specify the drive's max cylinder value. This value is zero-based, i.e., the max cylinder on an 823 cylinder drive is 822.

4.3.5 IOPB Byte C (Max Head)

IOPB Byte C specifies the drive's max head value. This value is zero-based.

4.3.6 IOPB Byte D (Max Sector)

IOPB Byte D specifies the drive's max sector value. This value is zero-based. See Section 8.1.

4.3.7 IOPB Byte E (Sectors Per Track)

IOPB Byte E returns the number of sectors per track (the 712 determines this value by counting the sector pulses from a hard-sectored drive) on a Read Drive Parameters command. This is the actual number of sectors; it has not been modified to be zero-based. For soft-sectored drives, you must supply the number of sectors per track for a Write Drive Parameters command. This value should take into account the Format Speed Tolerance Gap (if required).

The Format command uses this count to determine the number of sectors to format. Normal Read and Write commands use this count to limit the number of header compares before a Header Not Found error occurs.

4.4 FORMAT PARAMETERS IOPB

FORMAT PARAMETERS

	7	6	5	4	3	2	1	0
00	ERRS	DONE	CHEN	SGM	COMMAND			
01	COMPLETION CODE							
02	0	SR	0		ATTN	0	CMPL	DRDY
03	INTERNAL STATUS							
04	SUBFUNCTION CODE							
05	FIXD	0				UNIT		
06	INTERLEAVE				0	INTERRUPT LEVEL		
07	INTERRUPT VECTOR							
08	FIELD 1							
09	FIELD 2							
0A	FIELD 3							
0B	FIELD 4							
0C	FIELD 5 HIGH							
0D	FIELD 5 LOW							
0E	FIELD 12 (SOFT SECTOR ONLY)							
0F	PRIO	0	NEXT IOPB ADDRESS MODIFIER					
10	FIELD 6							
11	FIELD 7							
12	FIELD 5 ALT. HIGH							
13	FIELD 5 ALT. LOW							
14	NEXT IOPB ADDRESS HIGH							
15	NEXT IOPB ADDRESS							
16	NEXT IOPB ADDRESS							
17	NEXT IOPB ADDRESS LOW							
18	IOPB CHECKSUM HIGH							
19	IOPB CHECKSUM LOW							

4.4.1 IOPB Byte 6 (Interleave)

7	6	5	4	3	2	1	0

INTERLEAVE FACTOR

INTERRUPT LEVEL

4.4.1 IOPB Byte 6 (Interleave) (continued)

BIT	MNEMONIC	DESCRIPTION
7-4	INTF	INTERLEAVE FACTOR - The 712 uses INTF during Format operations. For 1:1 interleaving, the interleave factor is zero. The interleave factor for other ratios is (n+1):1, where n is the interleave factor.

INTERLEAVE FACTOR BITS 7-4	RATIO
0	1:1
1	2:1
2	3:1
:	:
F	16:1

TABLE 4-6. 712 INTERLEAVE FACTORS

3-0	INTL	INTERRUPT LEVEL - See Section 4.1.7.

4.4.2 IOPB Byte 8 (Field 1)

Field 1 is the number of bytes from the index or sector pulse to when the 712 enables the Read Gate for headers; this value must be larger than one. The 712 also uses this field to skip over the Intersector Gap (ISG).

4.4.3 IOPB Byte 9 (Field 2)

Field 2 is the number of bytes from when the 712 enables the Read Gate to when it starts looking for the Header Sync byte; this value must be larger than one.

4.4.4 IOPB Byte A (Field 3)

Field 3 is the number of bytes from the sector pulse to the Header Sync byte; this value must be larger than two.

4.4.5 IOPB Byte B (Field 4)

Field 4 is the number of bytes between the Header Pad and the Data Sync byte; this value must be larger than two.

4.4.6 IOPB Bytes C and D (Field 5 High/Low)

Byte C specifies Field 5 High; Byte D specifies Field 5 Low. The sector size in bytes must be larger than 254, even, and smaller than 2050. (Xylogics ships units set to 200H.)

4.4.7 IOPB Byte E (Field 12)

Field 12 includes the Speed Tolerance Gap, if required, and the Pre-index ISG. Only soft-sectored drives require Field 12.

4.4.8 IOPB Byte 10 (Field 6)

Field 6 is the number of bytes from enabling Read Gate to when the 712 starts looking for data sync; this value must be larger than one.

4.4.9 IOPB Byte 11 (Field 7)

Field 7 is the number of bytes the Write Gate remains on after the Data ECC; this value must be greater than or equal to one. At format time, the 712 uses this field for both the Header Pad and the Data Pad.

4.4.10 IOPB Bytes 12 and 13 (Alternate Field 5 High/Low)

Byte 12 specifies Alternate Field 5 High; Byte 13 specifies Alternate Field 5 Low. This field defines the number of data bytes per sector when AFE is set for this unit; see Section 4.4.6 for limits. See Section 8.18.

SECTION 5: COMMANDS

5.0 GENERAL

Each disk command begins a new page. An IOPB diagram follows each command description. The diagrams are highlighted to indicate which bytes the 712 requires for command execution, and which bytes return after execution.

Each 712 IOPB is 30-bytes long. Generally, all commands use Bytes 0 through 19H (Bytes 1AH through 1DH are reserved). Reserving all 30 bytes maintains IOPB integrity.

5.0.1 Setting Up the Command

Each IOPB diagram indicates the bytes or fields that must be set for each operation. Certain parameters are essential; others are optional. All commands require the Command, Unit, and Interrupt Level fields to contain valid information. (This is also true for the Interrupt Vector field if the Interrupt Level is not zero.)

5.0.2 Completing the Command

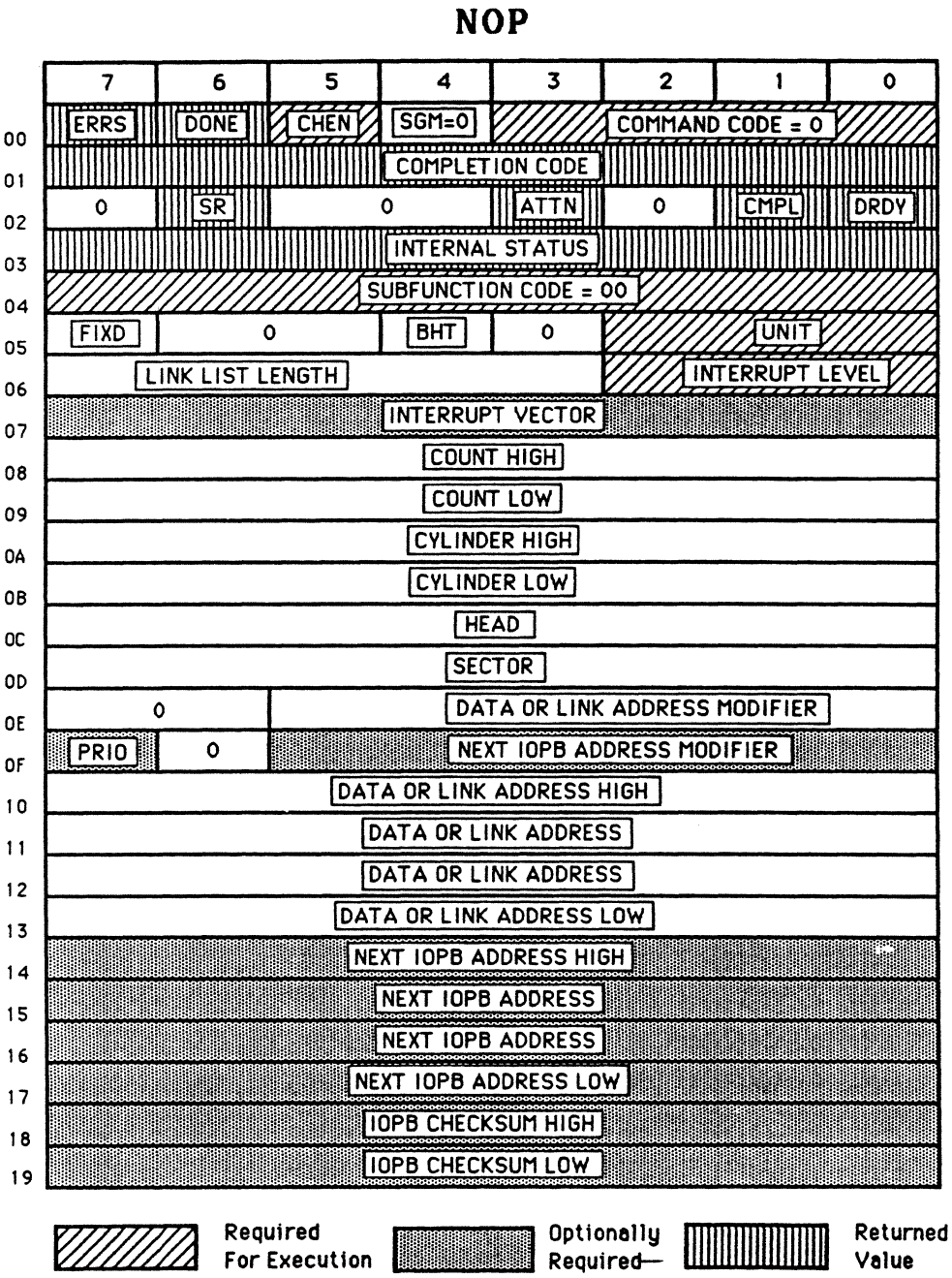
After the 712 completes the command, it updates IOPB Bytes 0 through 3 with ERRS, DONE, a Completion Code, and an internal status. The 712 only updates the entire IOPB if Auto-update (AUD) is enabled, an error occurs, or if Read Parameters or Read Extended Status commands are executed. If AUD is set, and no errors occur, the 712 sets DONE, posts a Completion Code of zero in Byte 1, and disk drive status information in Byte 2; for any command that DMAs data to/from memory, the 712 updates the data address to point to the last address plus one of the transfer. See Table 5-1.

STATUS	ACTION
AUD Clear/No Error Occurs	712 updates Bytes 0-3 with ERRS, DONE, Completion Code, and internal status
AUD Set/No Error Occurs	712 updates entire IOPB
AUD Clear/Error Occurs	712 updates the entire IOPB
AUD Clear/A Read Parameters or Read Extended Status Command Is Executed	712 updates the entire IOPB

TABLE 5-1. 712 COMMAND COMPLETION

5.1 NO OPERATION

The No Operation (NOP) command is a diagnostic tool. The 712 reads the IOPB and marks it complete.



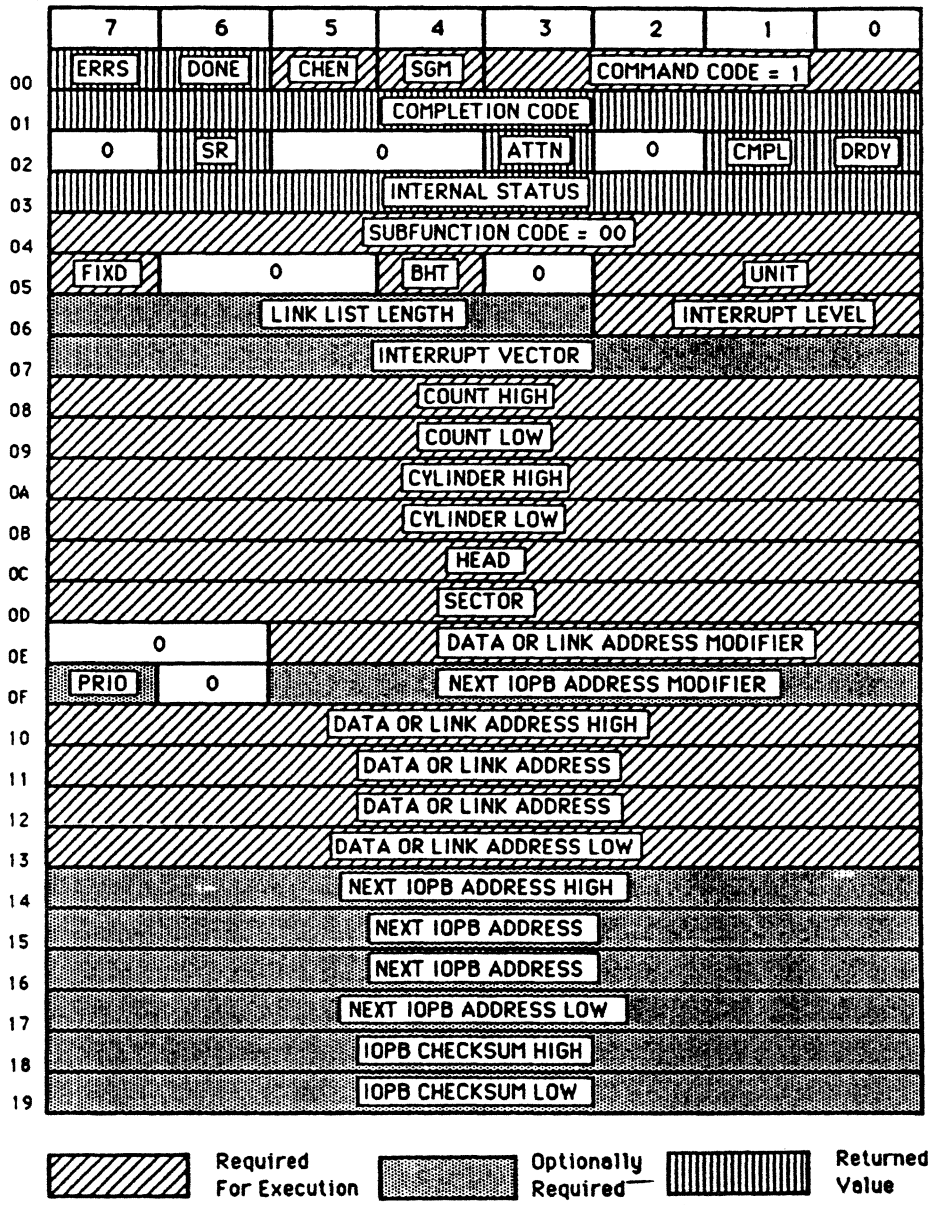
5.2 WRITE DATA

The 712, after reading and decoding the IOPB, positions the disk drive heads at the target cylinder; it then reads in the data from the host (indicated by the IOPB) and writes the data contiguously to the disk's sequential sectors.

Write Data has two IOPB formats: Normal and Scatter/Gather. A Normal IOPB specifies one contiguous block of host memory to write to the disk. A Gather Write IOPB specifies up to 32 different blocks of host memory to be placed in contiguous sectors on the disk (See Section 8.11).

The 712 stores IOPBs in a command queue that holds up to fourteen full IOPBs. This queue allows the 712 to optimize the commands for both elevator/overlap seeks and processing multiple IOPBs per revolution (See Section 8.9).

WRITE DATA



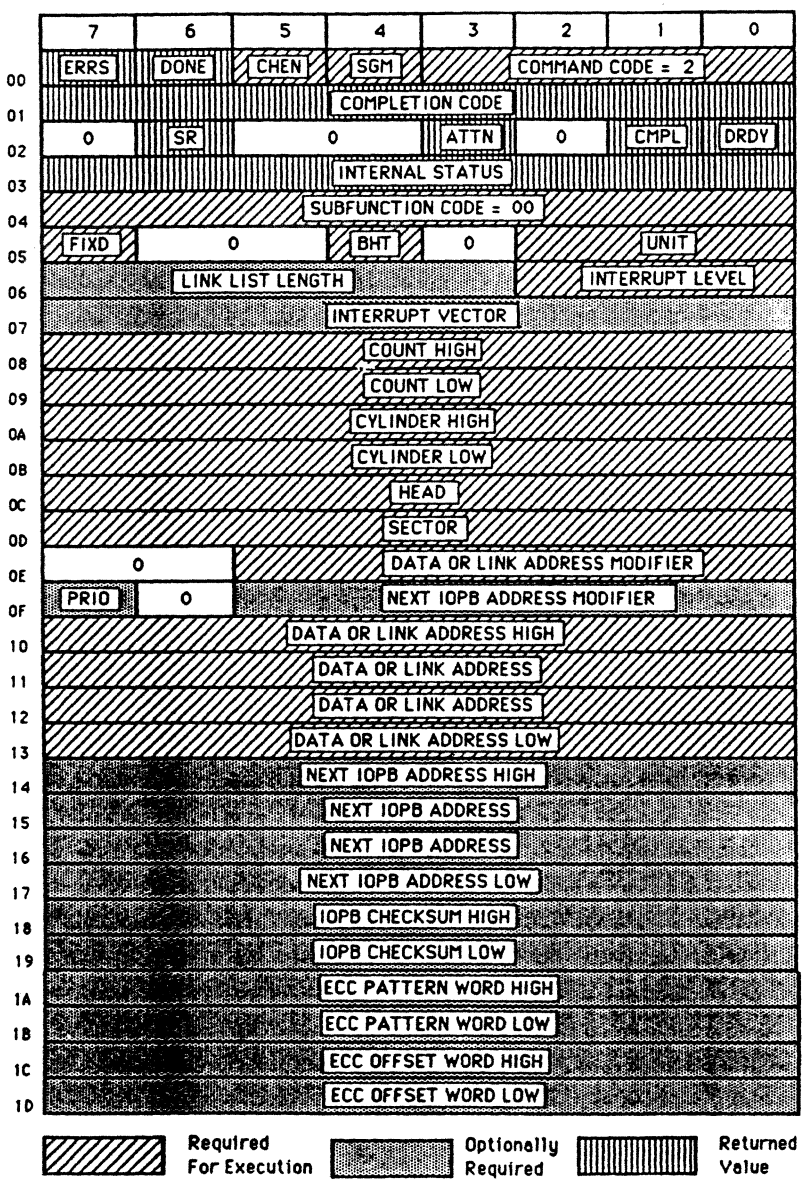
5.3 READ DATA

The 712, after reading and decoding the IOPB, positions the disk drive heads at the target cylinder, then reads the disk data indicated by the IOPB, and writes the data in host memory.

Read Data has two IOPB formats: Normal and Scatter/Gather. A Normal IOPB specifies one contiguous block of host memory that is used when placing the data from the disk. A Scatter Read IOPB specifies up to 32 different blocks of host memory where the disk data will be placed (See Section 8.11).

The 712 stores IOPBs in a command queue that holds up to fourteen full IOPBs. This queue allows the 712 to optimize the commands for both elevator/overlap seeks and processing multiple IOPBs per revolution (See Section 8.9).

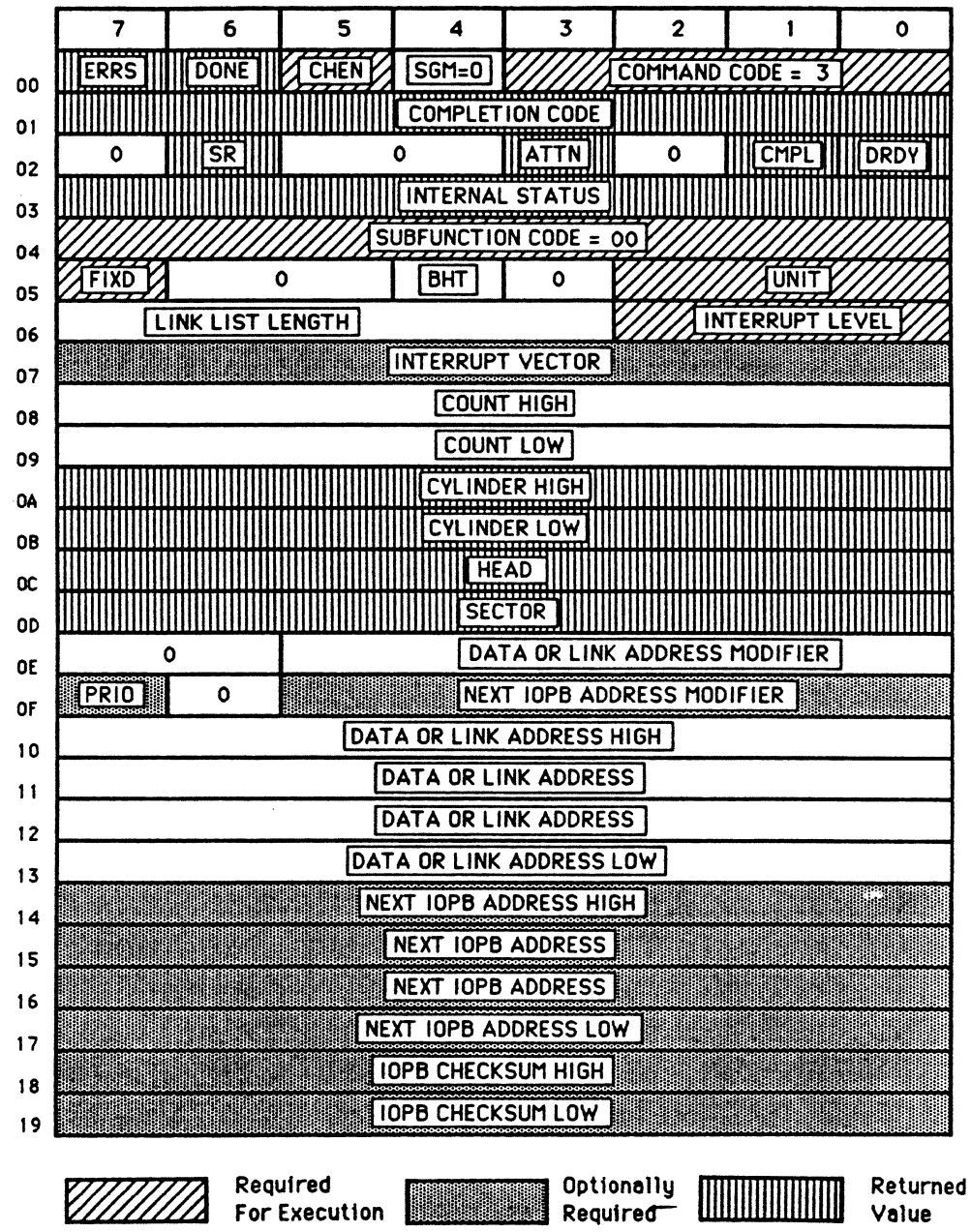
READ DATA



5.4 REPORT CURRENT ADDRESS

The 712 selects the disk drive, reads the first good header field, and returns the address to the host via the IOPB; it updates the IOPB regardless of AUD's status.

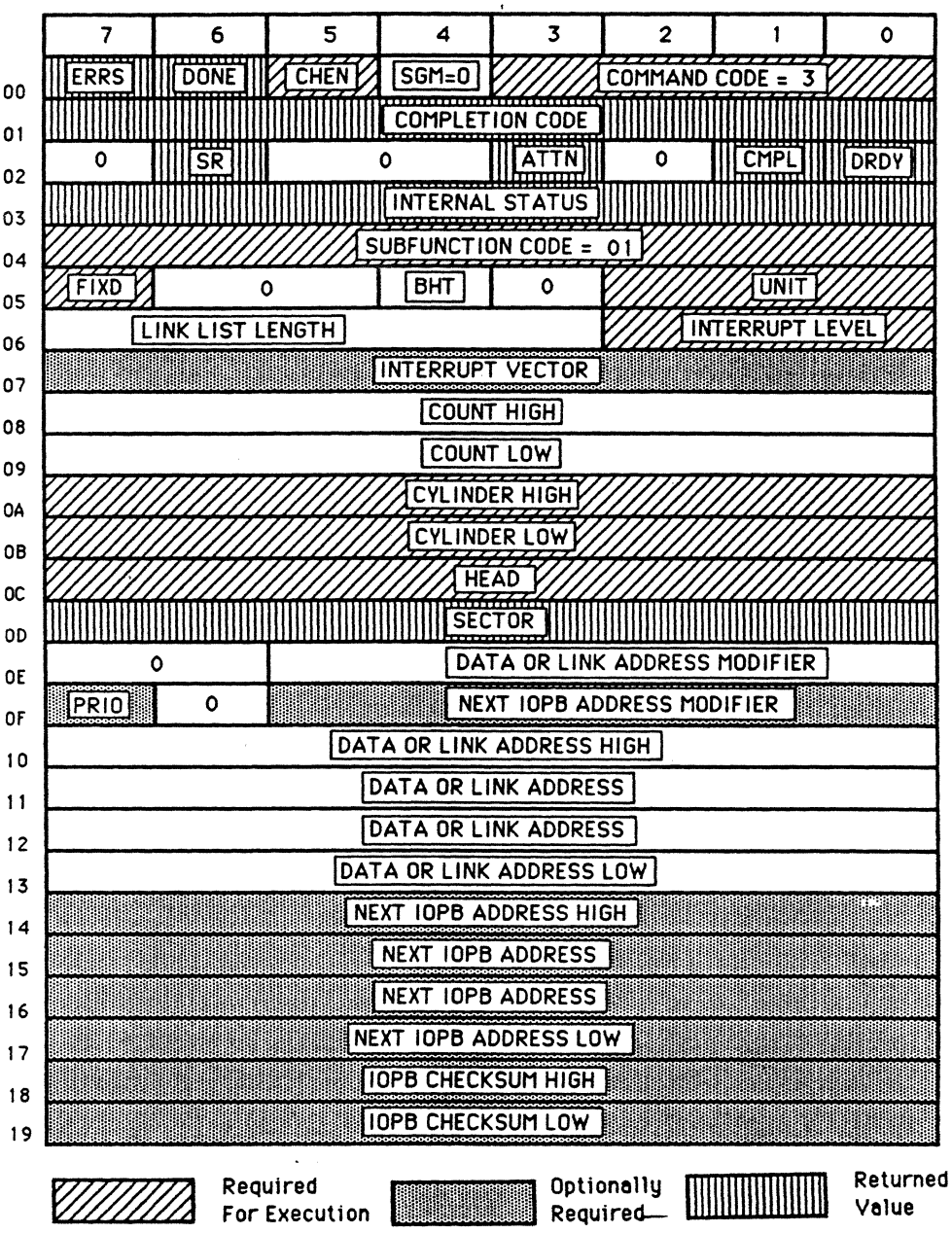
REPORT CURRENT ADDRESS



5.5 SEEK AND REPORT CURRENT ADDRESS

The 712 issues a seek to the selected disk drive for the target cylinder. After the drive completes the seek, the 712 reads the first good header field it encounters and reports it to the host via the completed IOPB. The 712 updates the IOPB regardless of AUD's status.

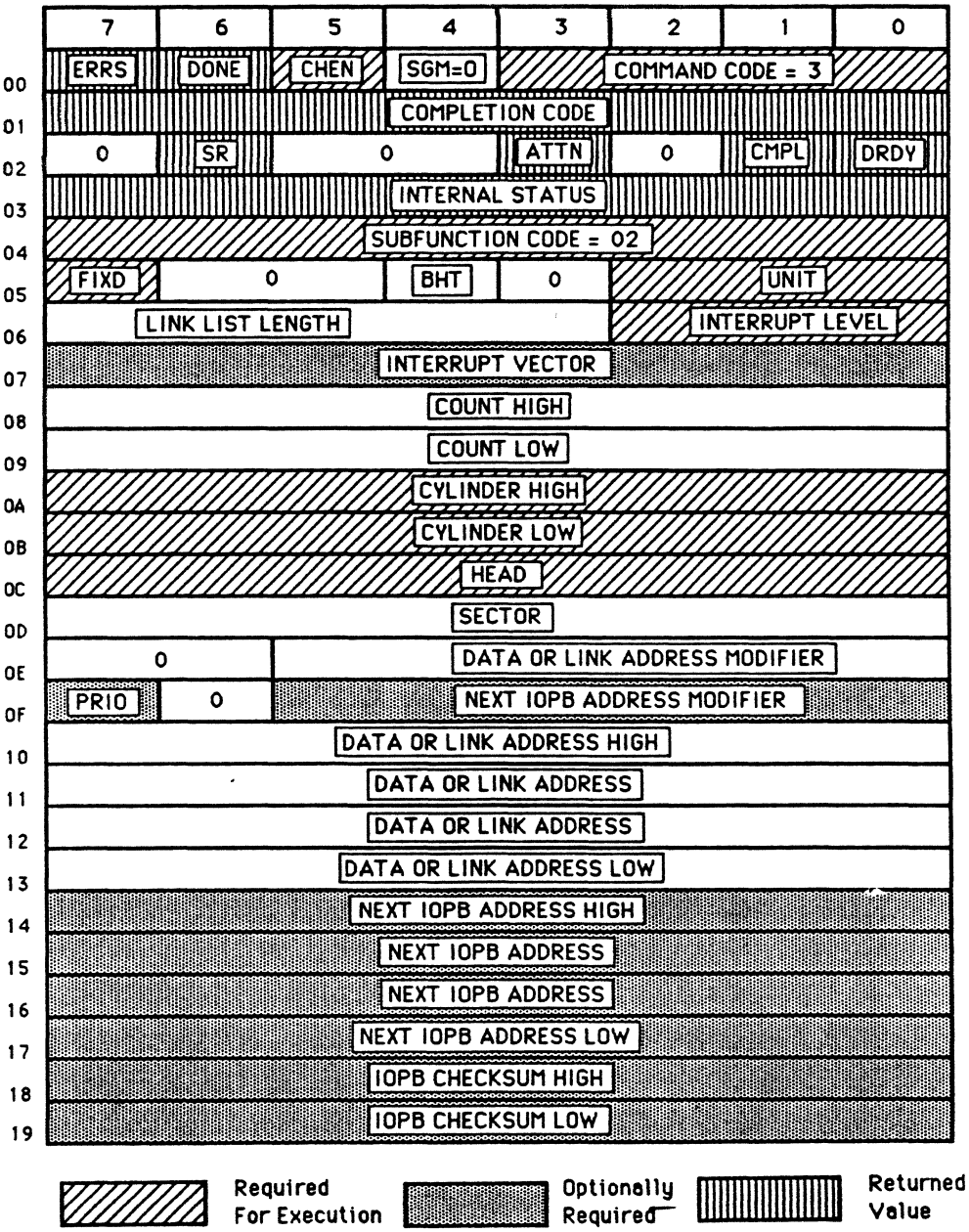
SEEK AND REPORT CURRENT ADDRESS



5.6 START SEEK AND REPORT COMPLETION IMMEDIATELY

The 712 issues a seek to the selected disk drive for the target cylinder, and reports a completion to the host without waiting for the seek to complete.

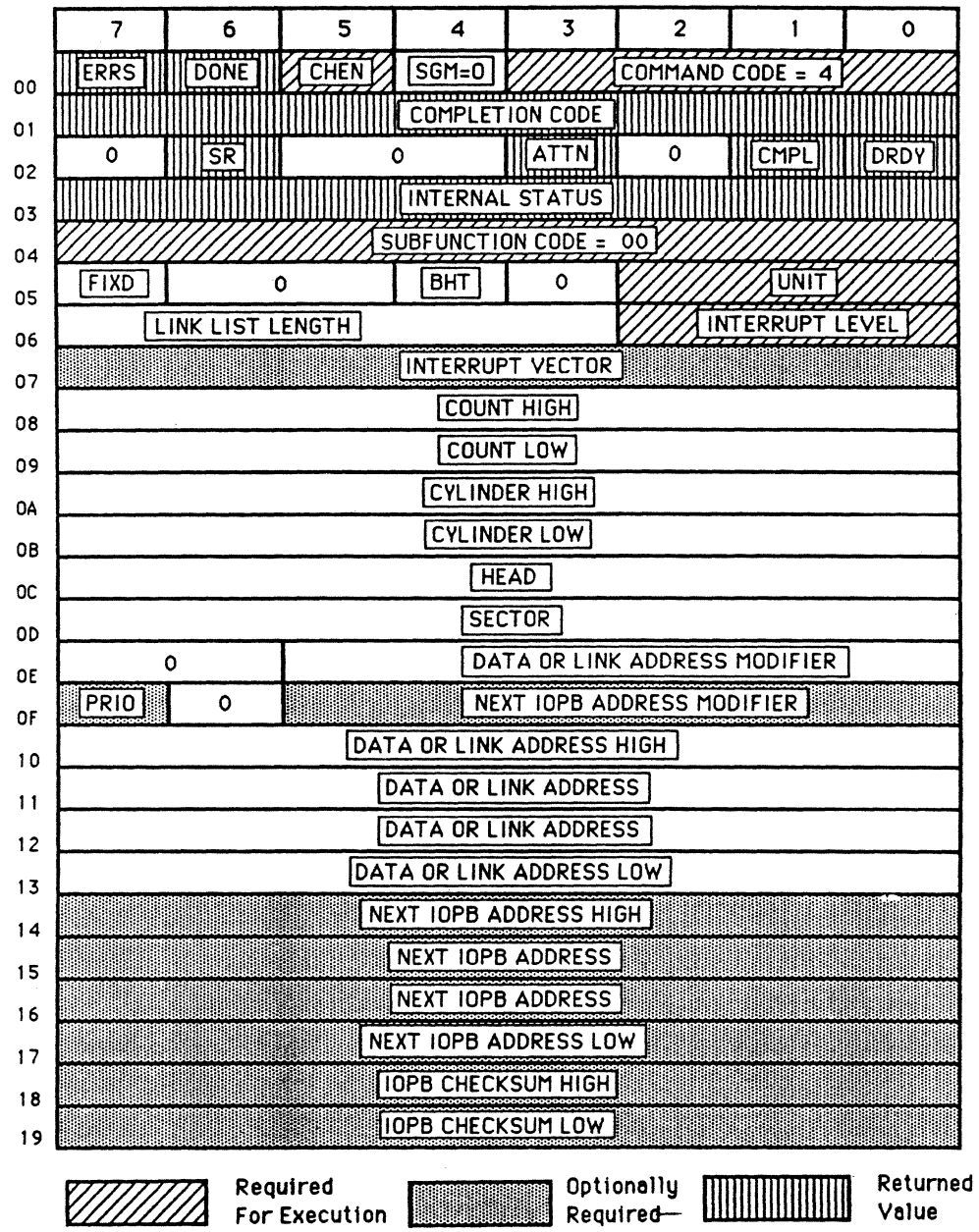
START SEEK AND REPORT COMPLETION IMMEDIATELY



5.7 DRIVE RESET WITH RETURN TO ZERO

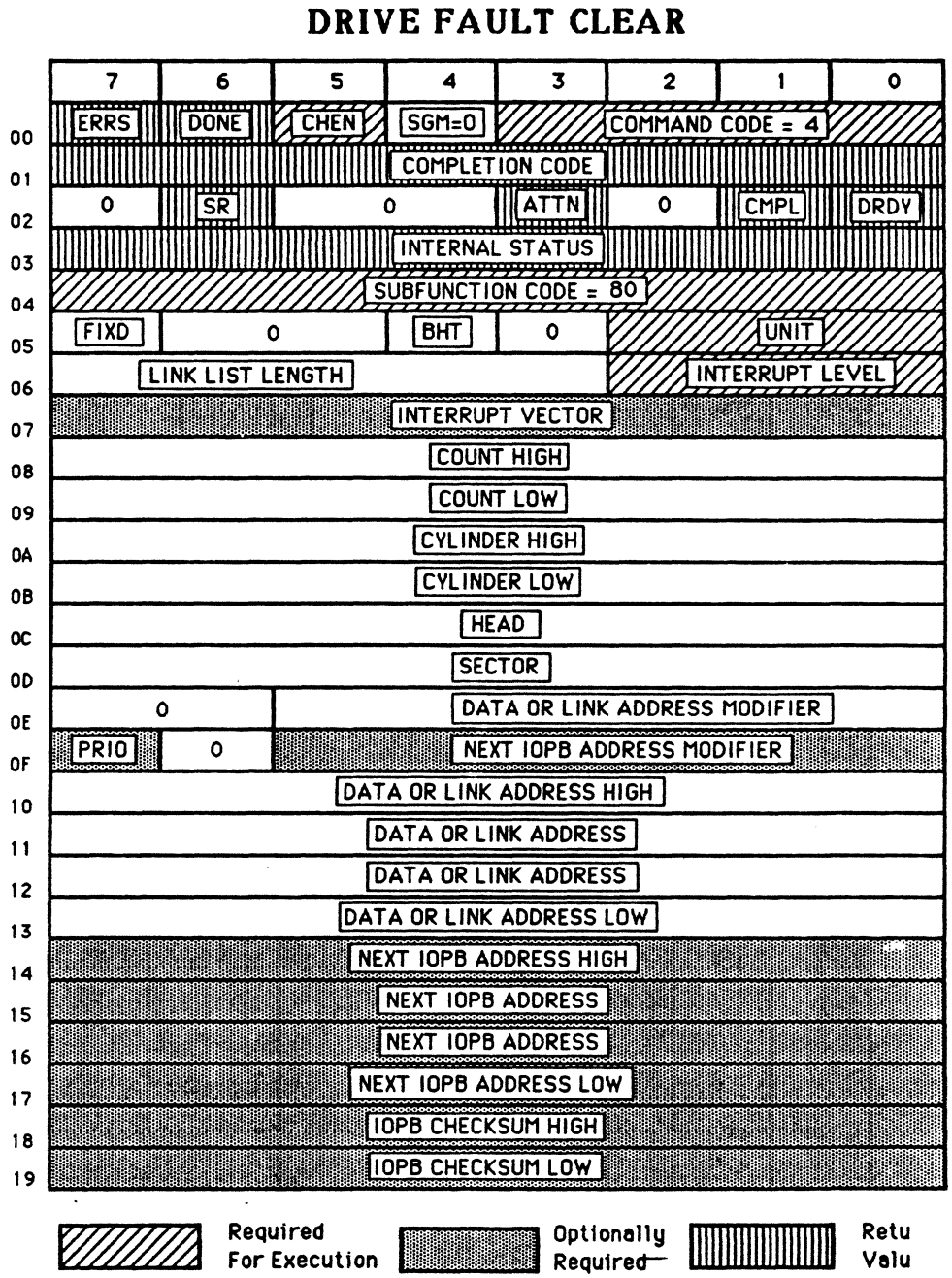
The 712 issues commands to the disk drive to reset. First it issues a fault clear, and then a recalibrate (return to zero). The IOPB is complete when the recalibrate completes or times out on drives that are ready. The 712 does not wait for the recalibrate to complete on drives that are not ready.

DRIVE RESET (WITH RTZ)



5.8 DRIVE FAULT CLEAR

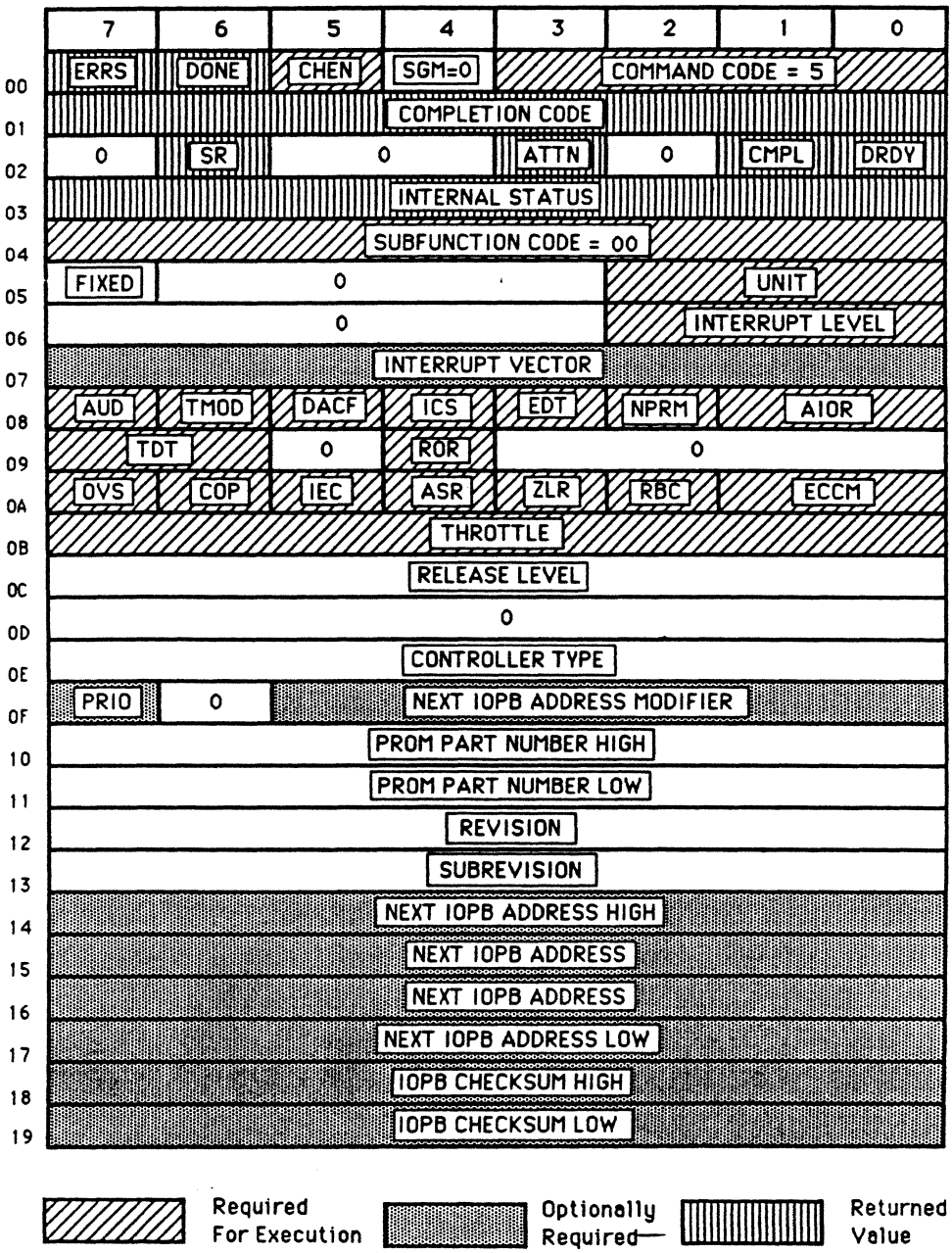
The 712 resets the Attention High signal. The 712 does not recalibrate the drive.



5.9 WRITE CONTROLLER PARAMETERS

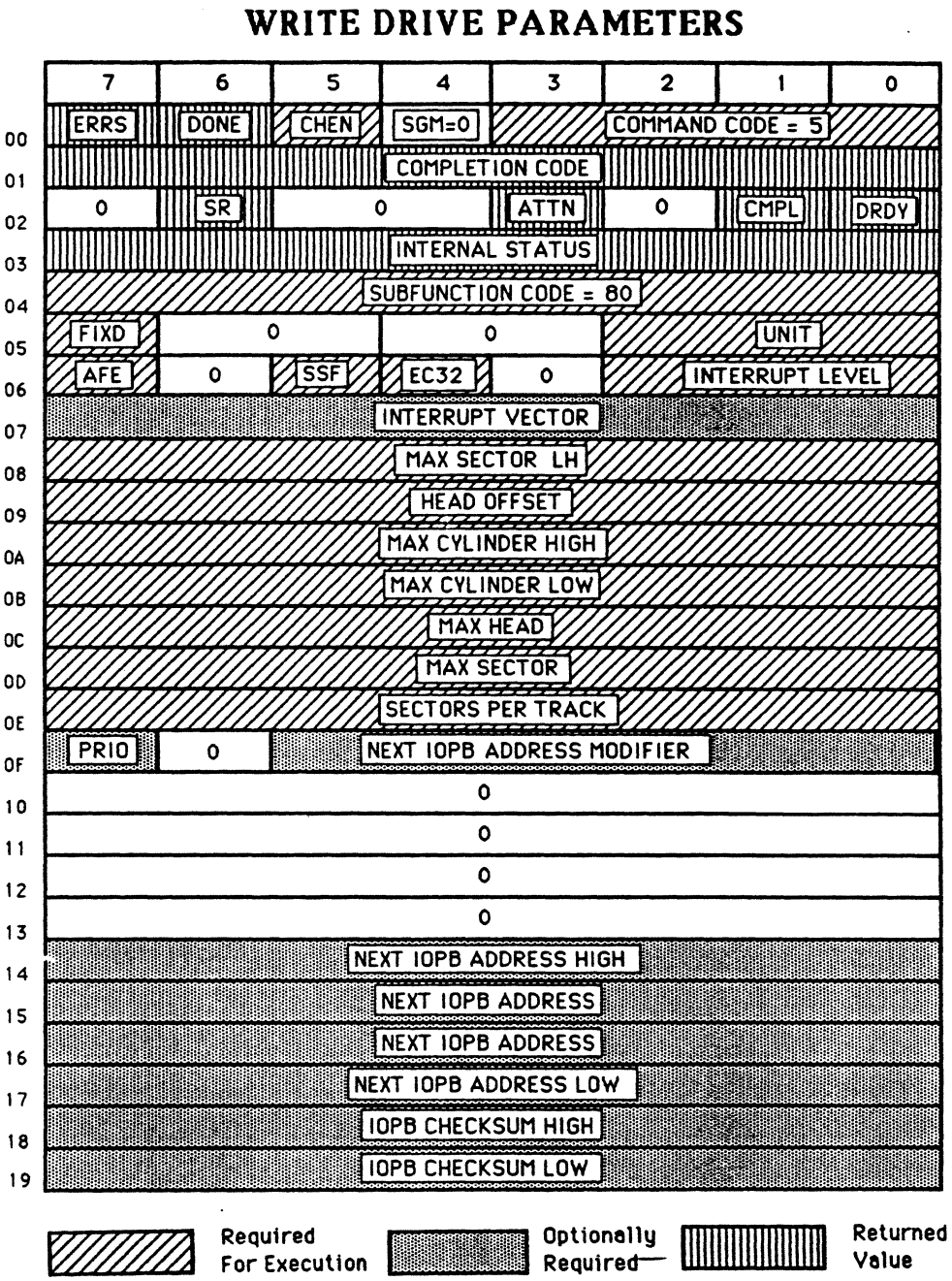
This command initializes the 712 with its operational parameters. No default parameters are assumed, but once written, the parameters remain in the 712 non-volatile memory. Section 4.2 defines how to change the parameters for individual applications; Section 6.6 explains the IRAM checksum.

WRITE CONTROLLER PARAMETERS



5.10 WRITE DRIVE PARAMETERS

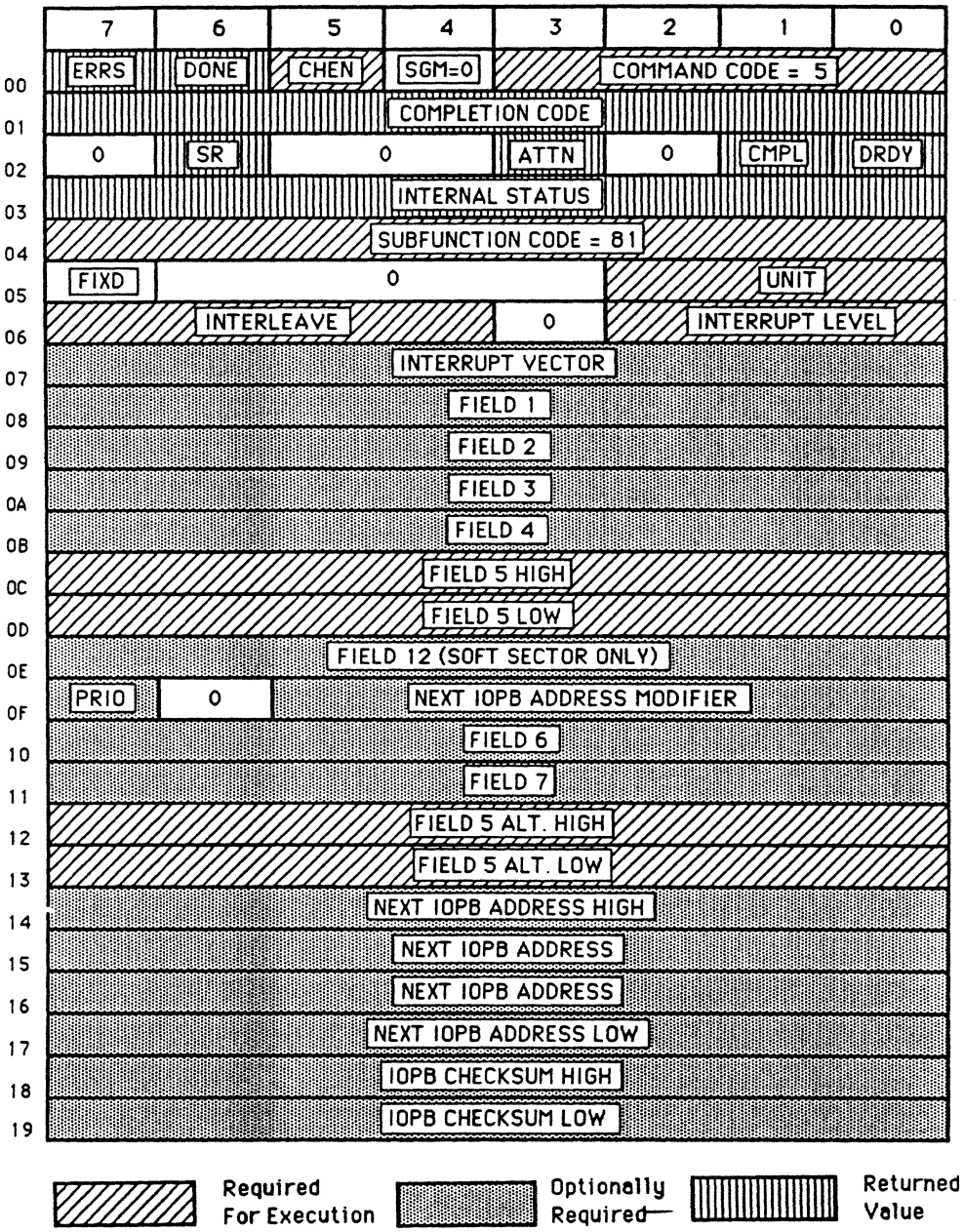
This command informs the 712 of the disk drive's physical characteristics. No default values are assumed, but once loaded, the parameters remain stored in the 712 non-volatile memory. See Section 4.3.



5.11 WRITE FORMAT PARAMETERS

This command informs the 712 of the disk drive's media format. No default values are assumed, but once loaded, the values remain stored in the 712 non-volatile memory. See Section 4.4; Section 8.1 defines how to change the media format for individual applications.

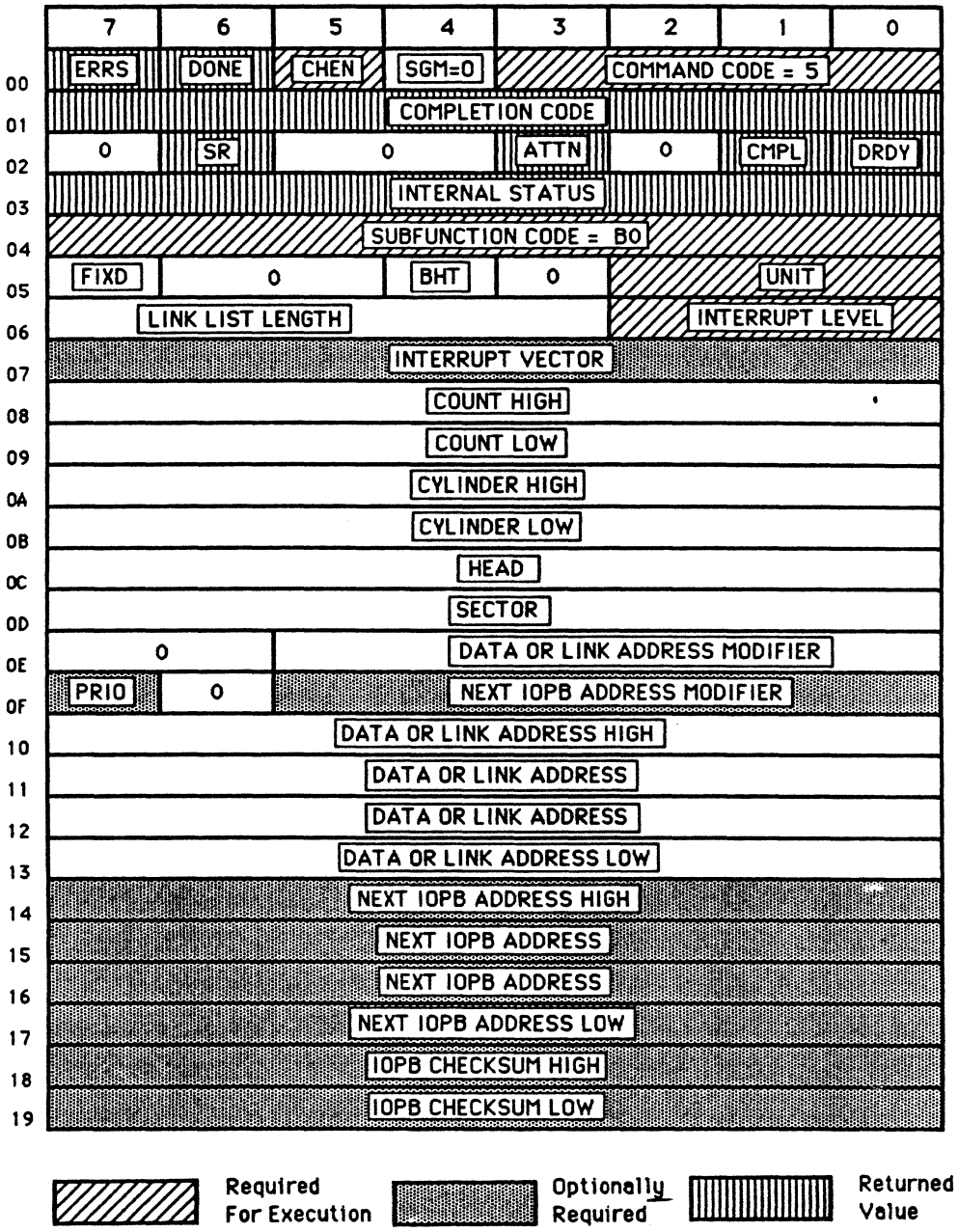
WRITE FORMAT PARAMETERS



5.12 STORE FORMAT CONFIGURATION

The 712 reads the minimum number of bytes in the Intersector Gap (ISG) and Phase Lock Oscillator (PLO) sync fields from the drive, computes fields 1 through 4 and 6, and stores them in a non-volatile RAM. A Read Format Parameters command returns the computed fields.

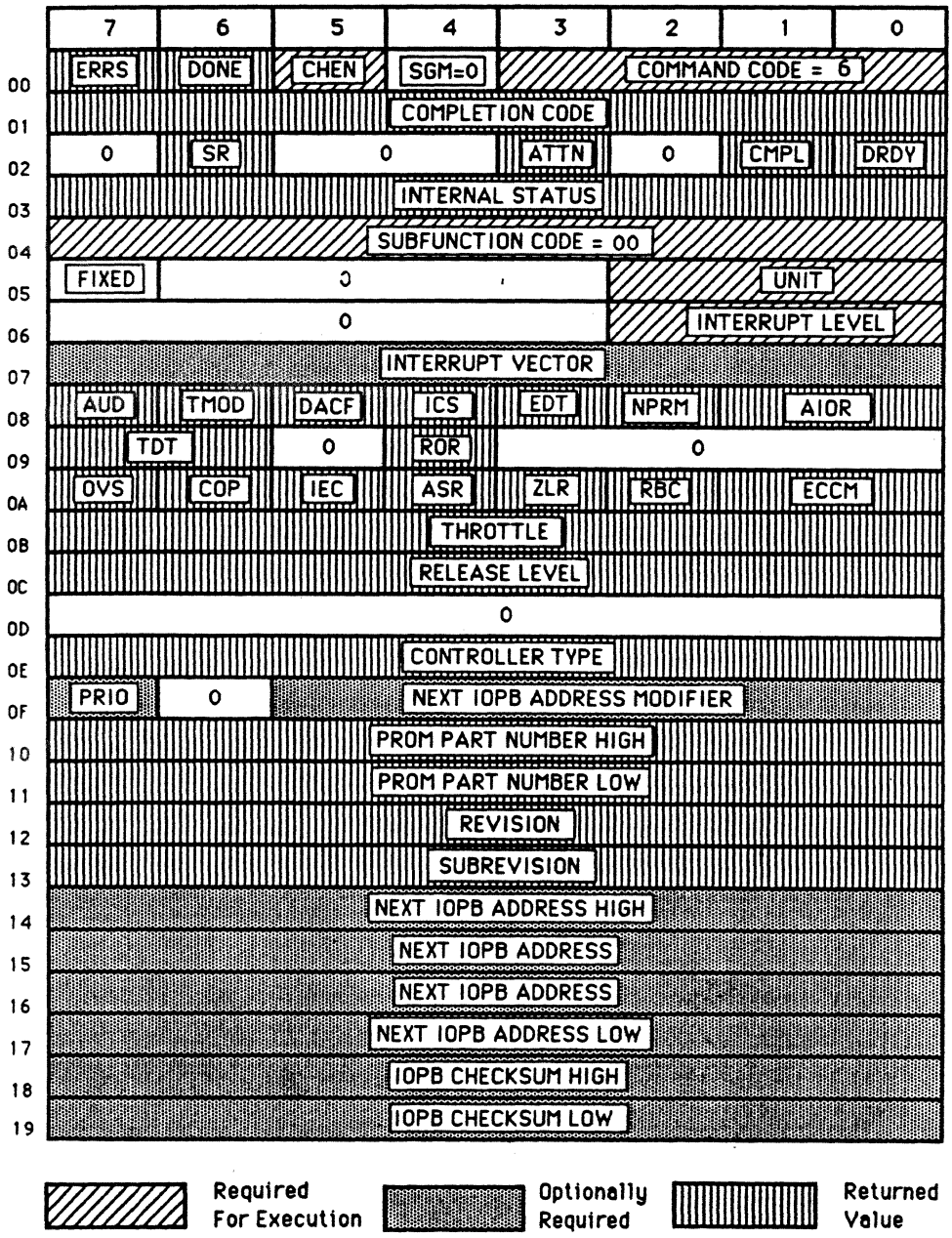
STORE FORMAT CONFIGURATION



5.13 READ CONTROLLER PARAMETERS

The 712 returns the current 712 operational parameters to the host via the IOPB; it verifies the IRAM checksum before completing the transfer regardless of AUD's status. See Section 4.2.

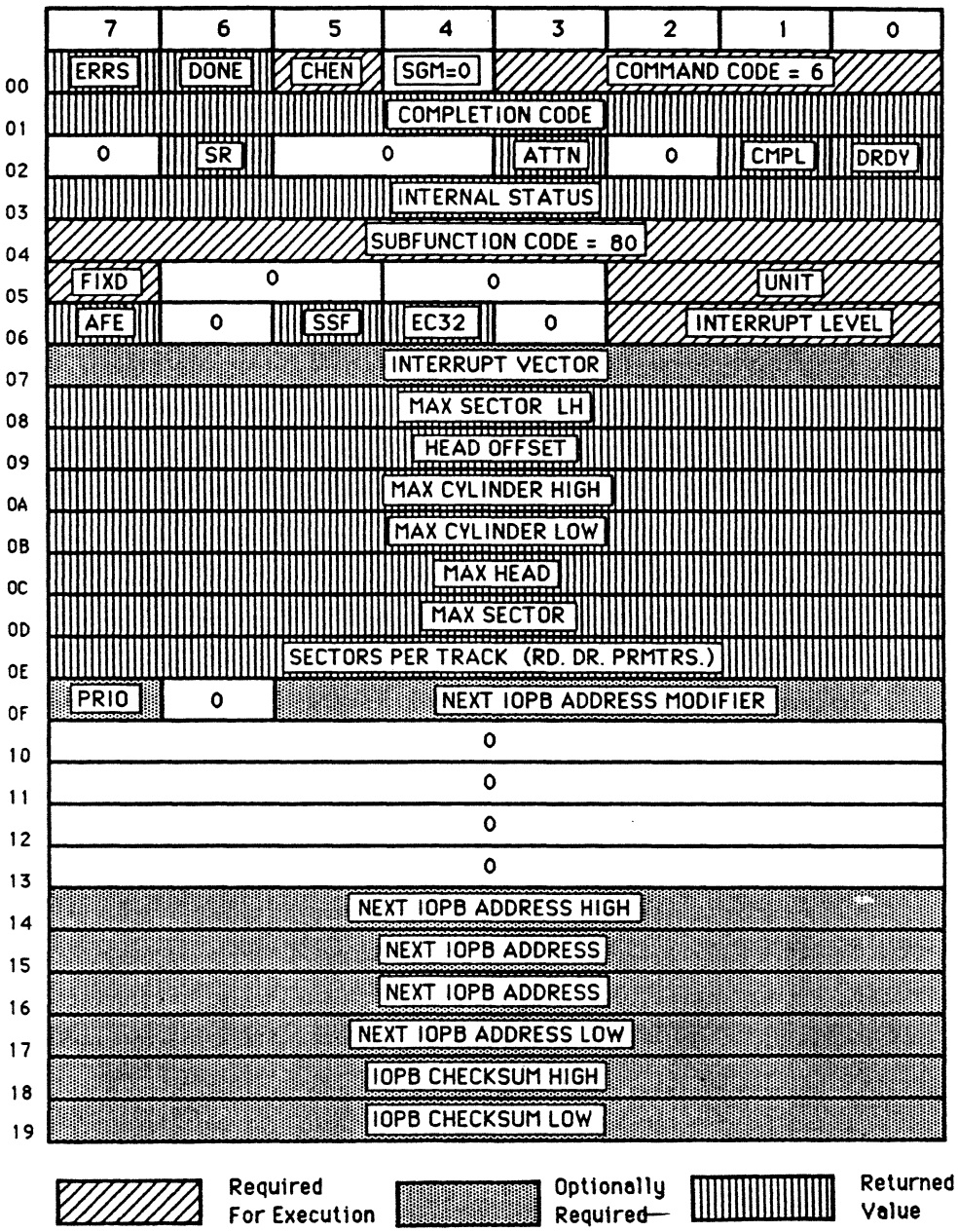
READ CONTROLLER PARAMETERS



5.14 READ DRIVE PARAMETERS

The 712 returns the disk drive's physical characteristics to the host via the IOPB; it returns the specified drive's actual number of sectors per track in Byte 0EH. The 712 verifies the IRAM checksum before completing the transfer; it updates the IOPB regardless of AUD's status. See Section 4.3.

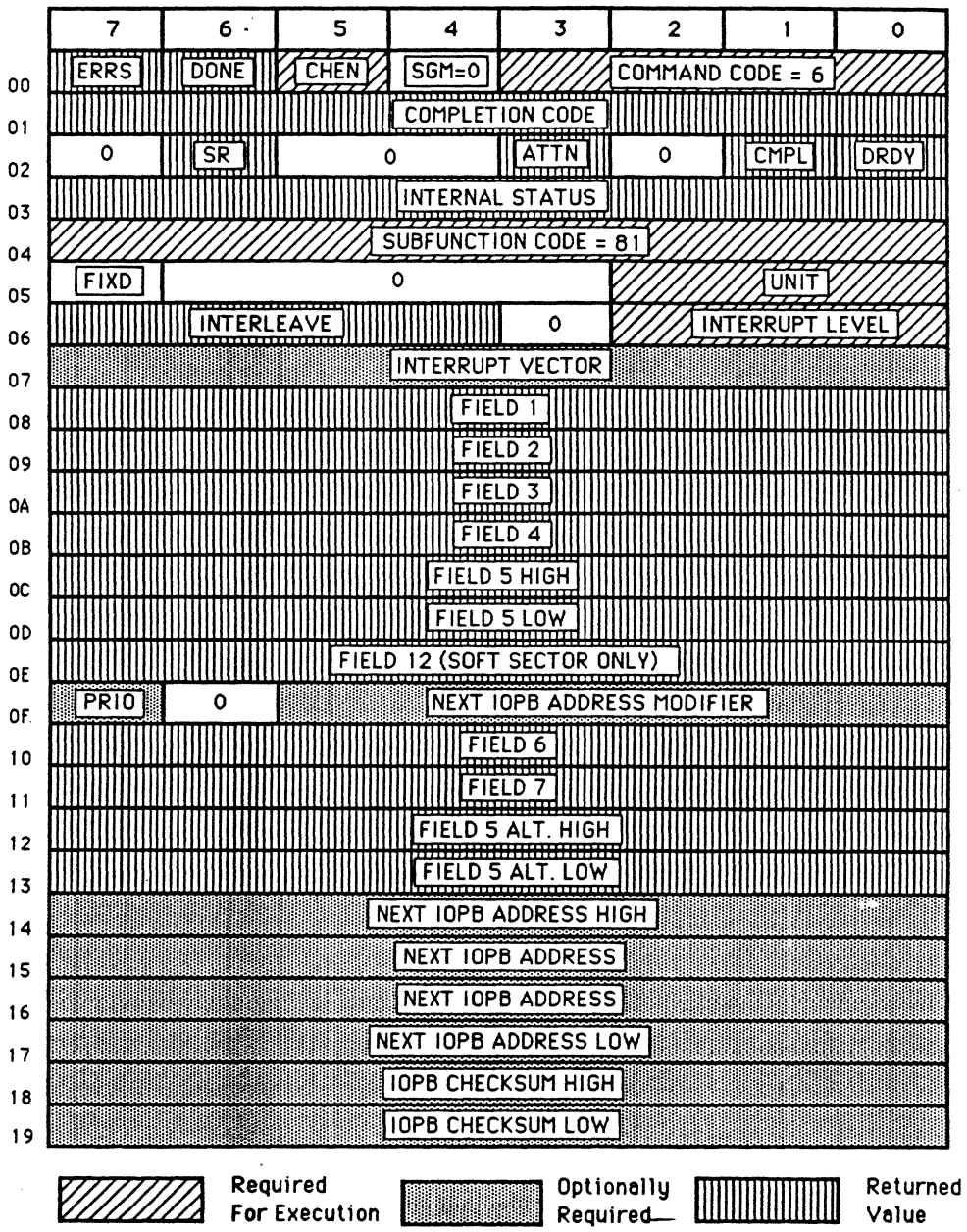
READ DRIVE PARAMETERS



5.15 READ FORMAT PARAMETERS

The 712 returns the current disk drive's format parameters to the host via the IOPB; it verifies the IRAM checksum before completing the transfer, and updates the IOPB regardless of AUD's status. See Section 4.4.

READ FORMAT PARAMETERS

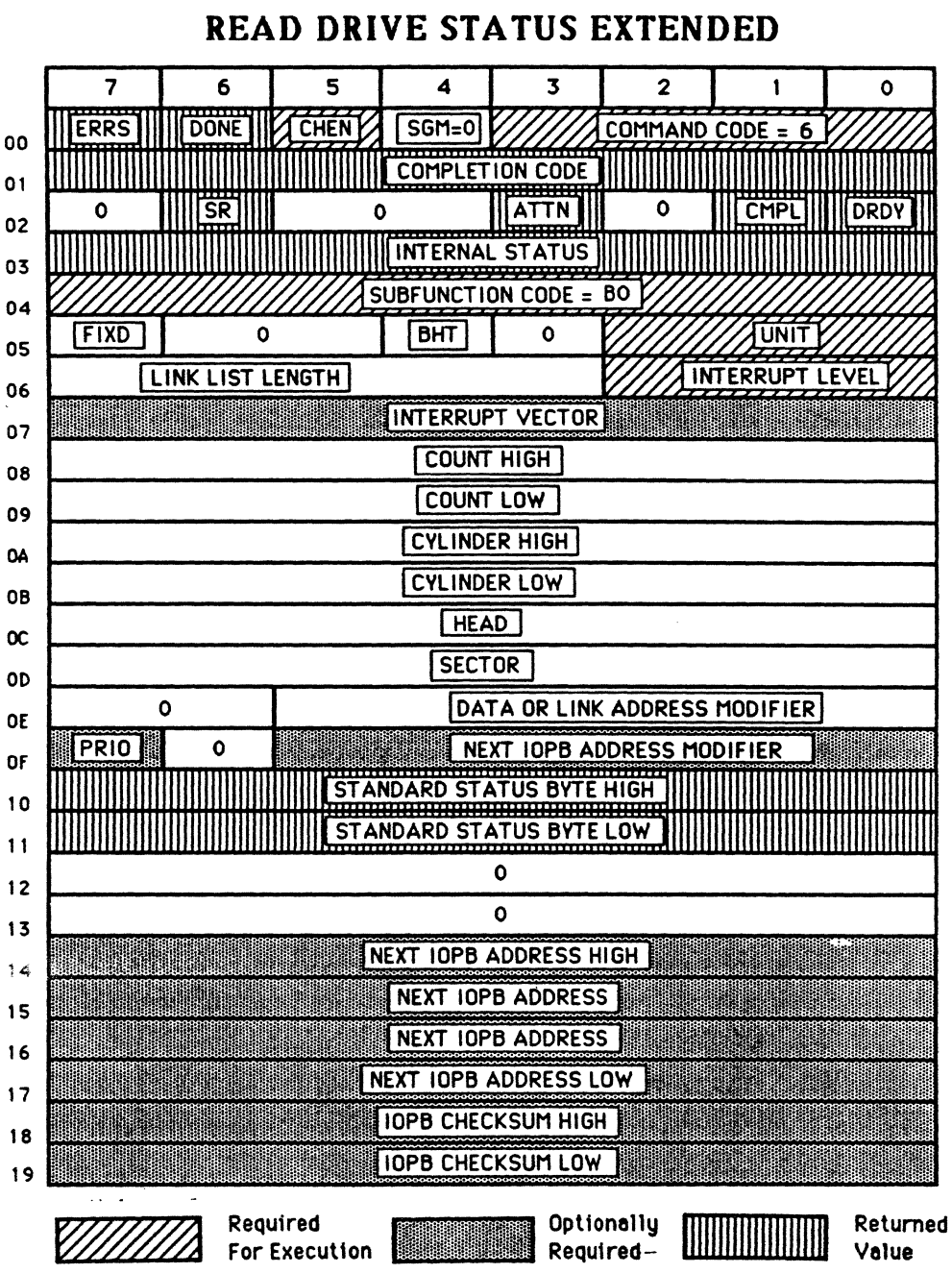


5.16 READ DRIVE STATUS EXTENDED

The 712 reads the ESDI drive interface's extended status. It returns the drive's status in the IOPB's Data Address bytes regardless of AUD's status (See Table 5-2). The drive-specific bytes follow the same bit alignment as the standard Status byte.

BYTE	BIT: 7	6	5	4	3	2	1	0
10	MSB	Standard Status High Byte						LSB
11	MSB	Standard Status Low Byte						LSB

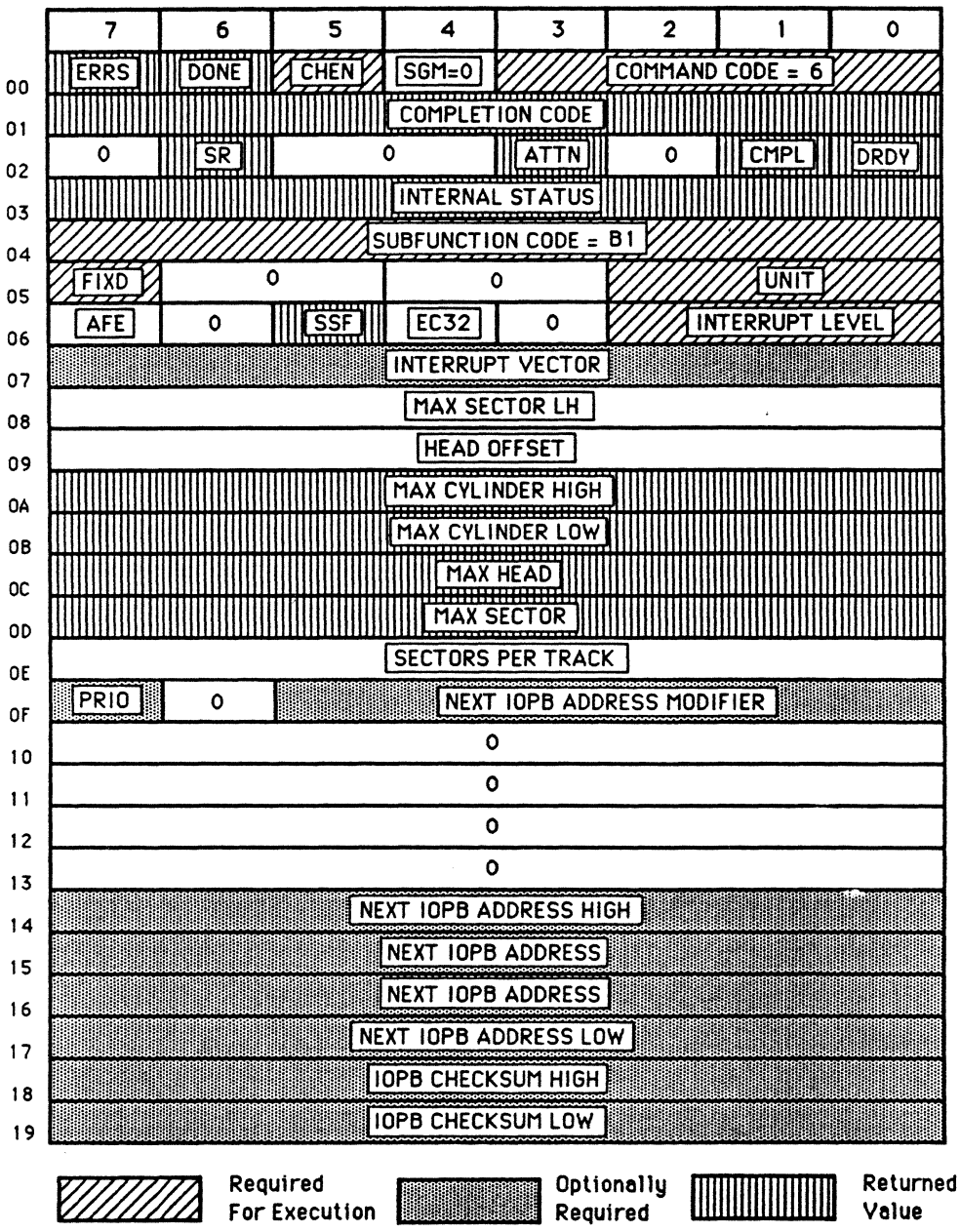
TABLE 5-2. EXTENDED DRIVE STATUS



5.17 SHOW DRIVE CONFIGURATION

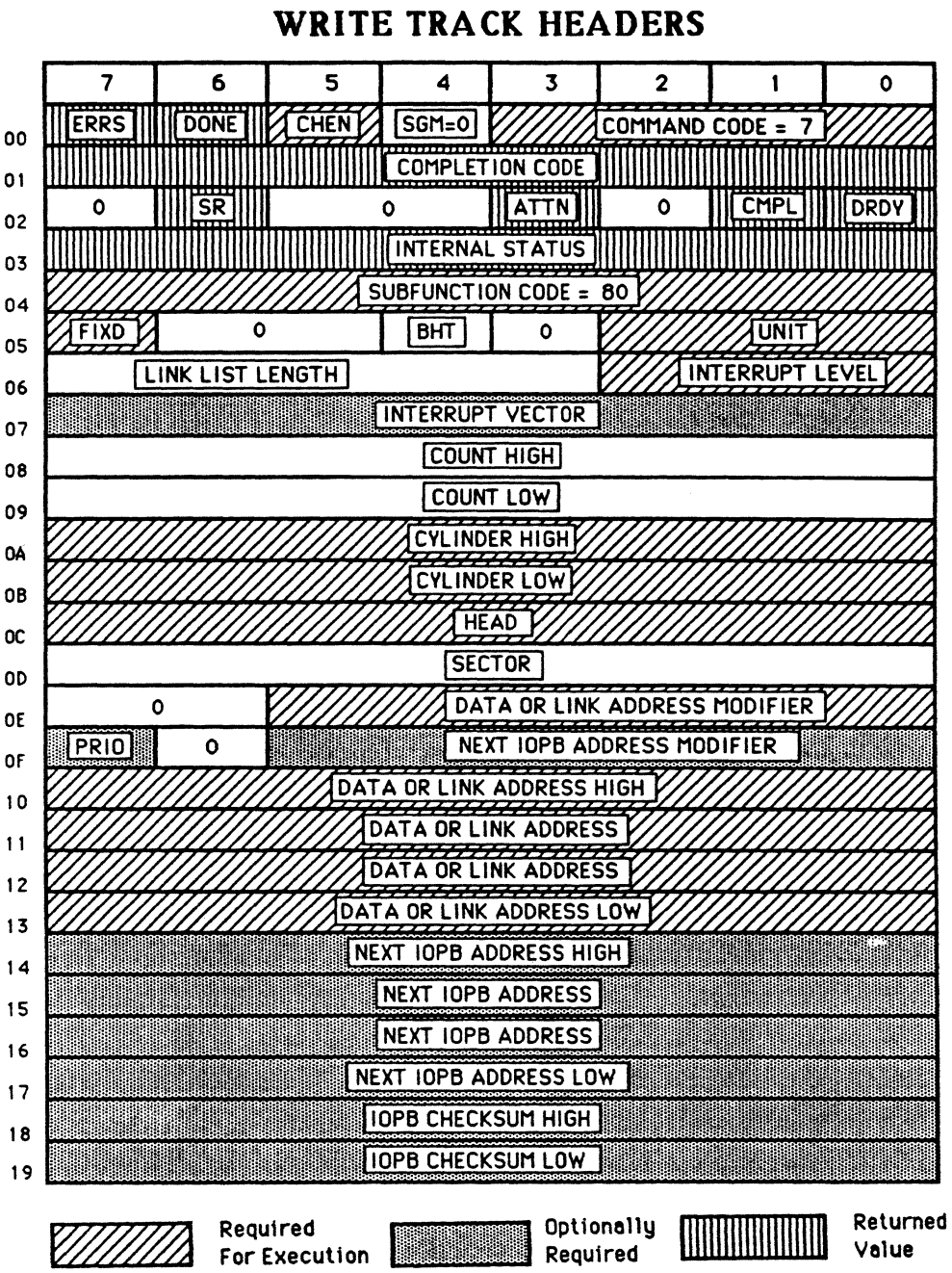
The 712 reads the max cylinder, head, and sector from the drive and displays them in the completed IOPB. It also reads the general configuration from the drive to set up the SSF and ESD bits.

SHOW DRIVE CONFIGURATION



5.18 WRITE TRACK HEADERS

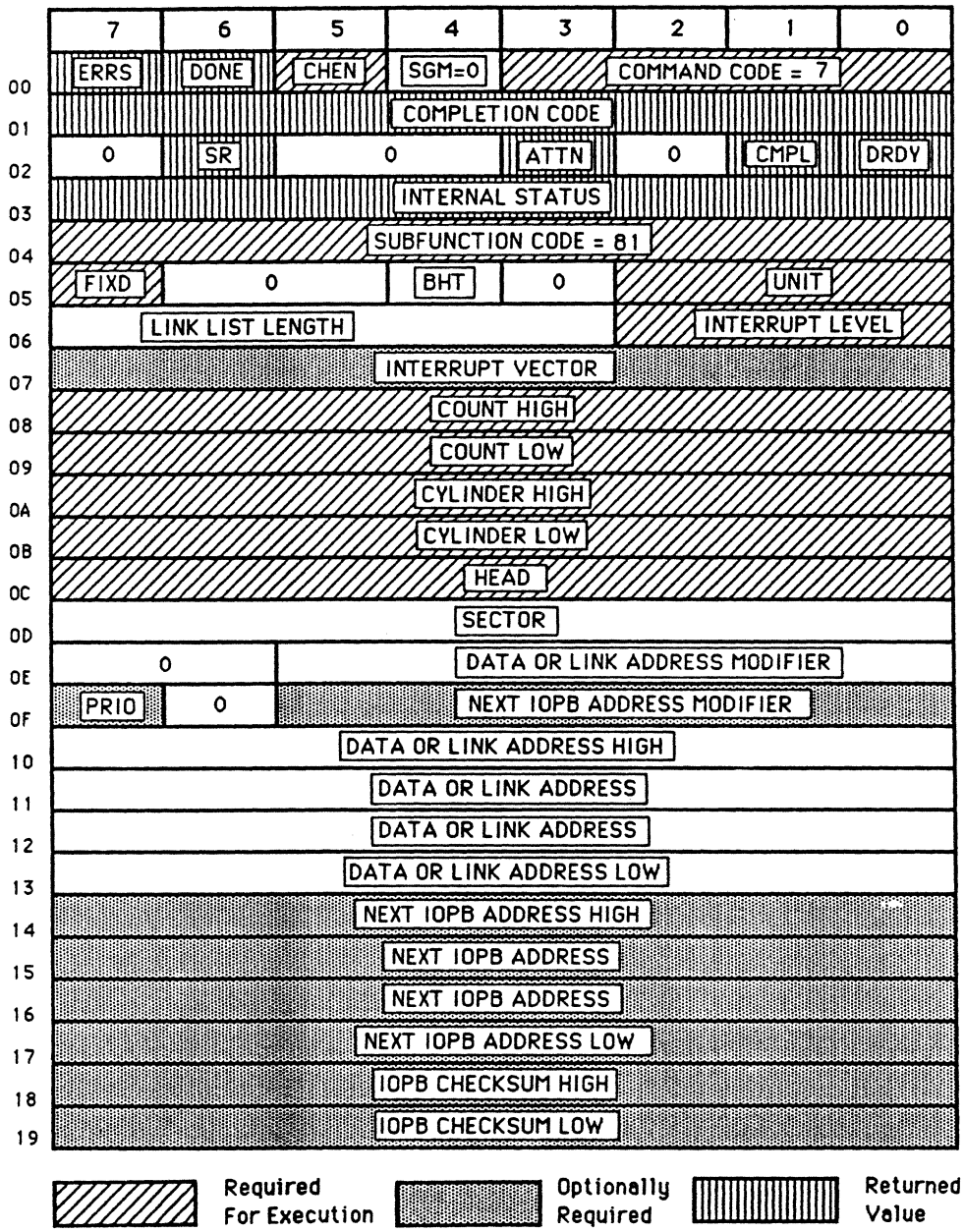
This command enables the host to write the sector header fields on a track (only one track per IOPB). The 712 takes the data in the header fields from host memory: four bytes per header; one header for each sector on the track. The data fields are not preserved. It places the data on the track starting from index. Section 8.1 defines the data format in each header.



5.19 WRITE TRACK FORMAT

The Write Track Format command directs the 712 to format the drive, writing the header of all sectors with the appropriate sector ID. The data field contains zeros and a valid ECC. The Count bytes in this command refer to the number of tracks to be formatted. See Section 8.3.

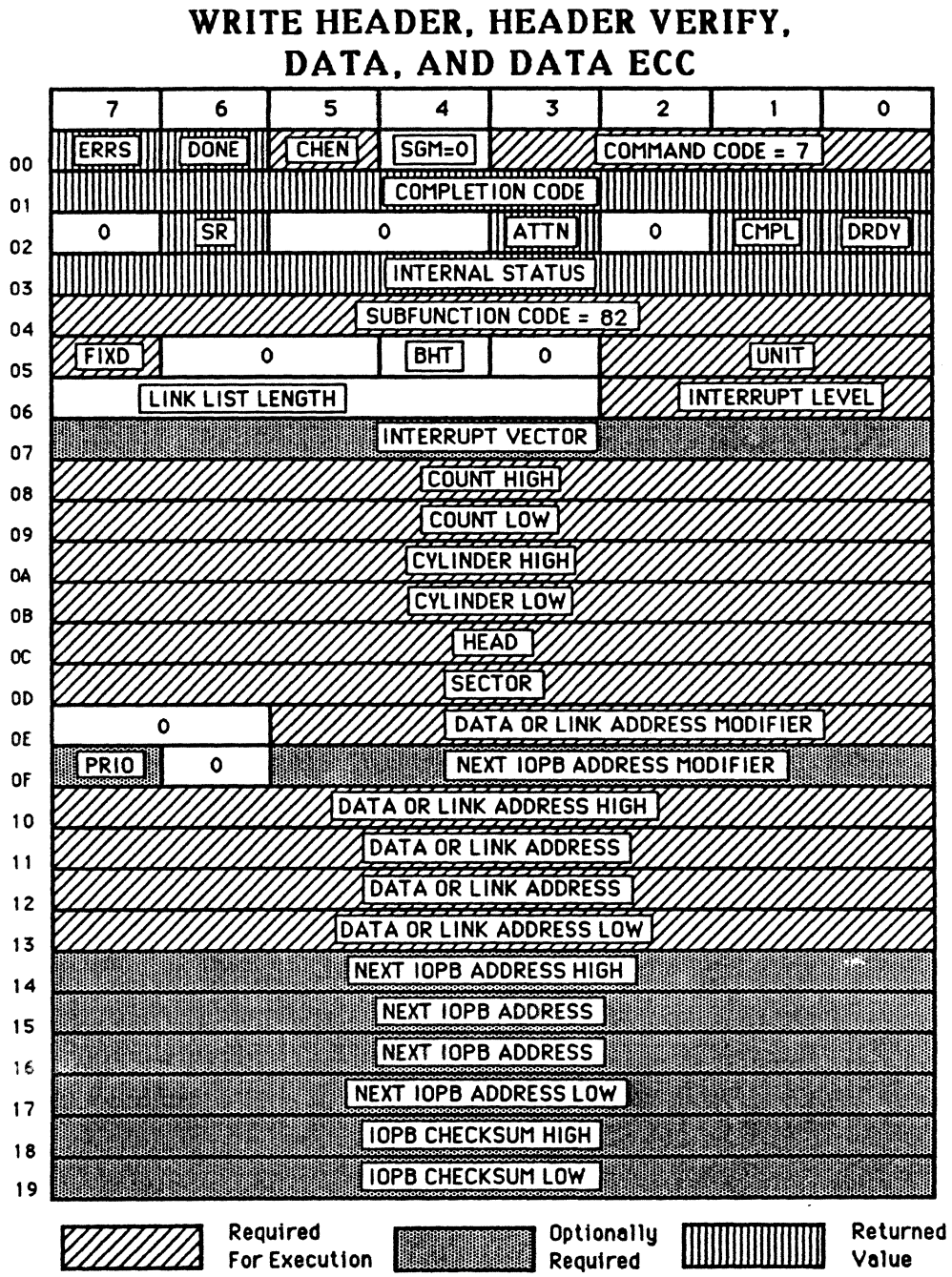
WRITE TRACK FORMAT



5.20 WRITE HEADER, HEADER VERIFY, DATA, AND DATA ECC

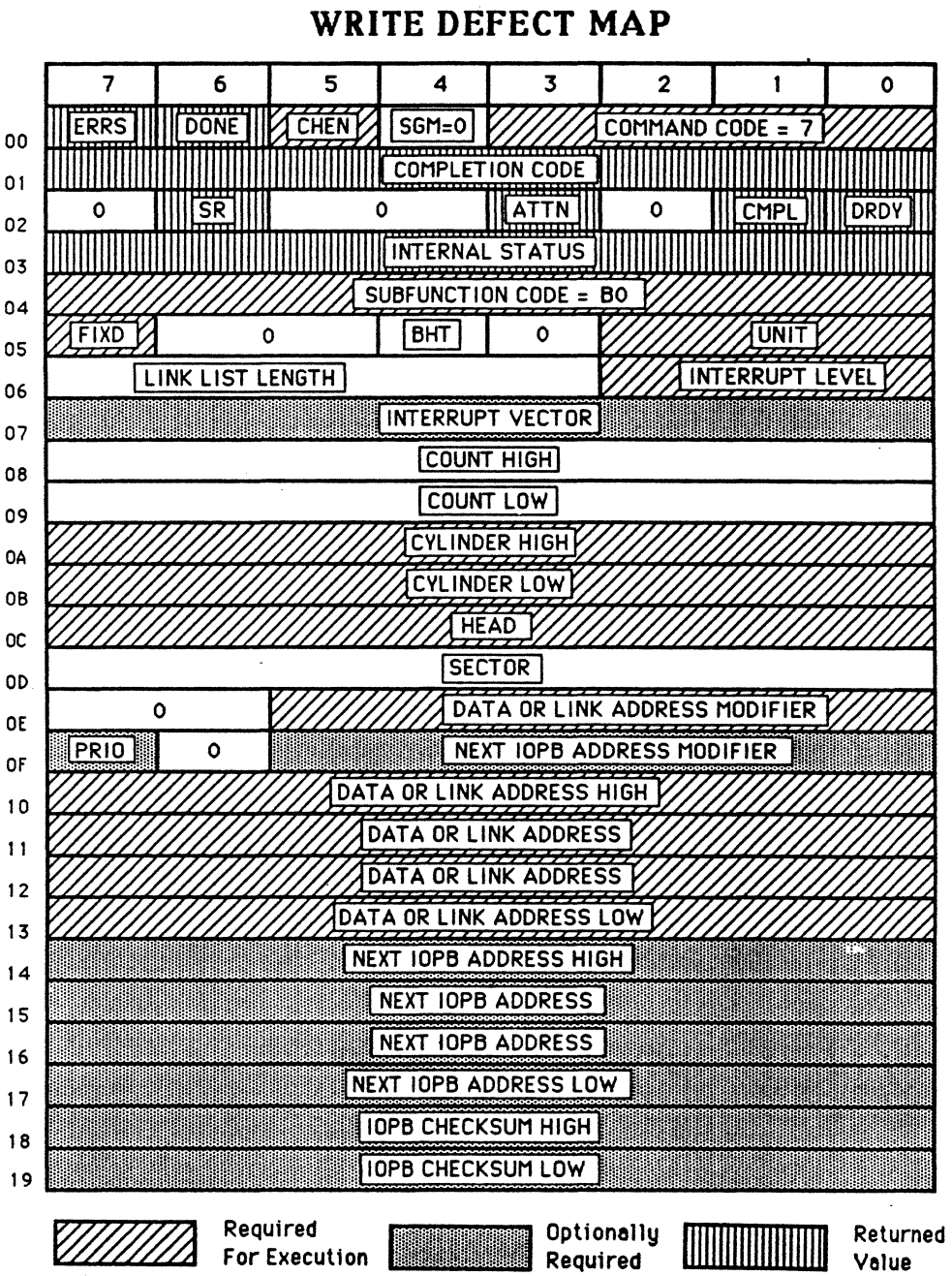
This command directs the 712 to write a sector header, header verify, data, and data ECC. There are always four bytes in the header, but the other fields vary according to the initial 712 set-up. The 712 does not cross head or cylinder boundaries while executing this command.

The host must calculate the ECC in all ECC fields since the 712 does not calculate any ECC fields for this command.



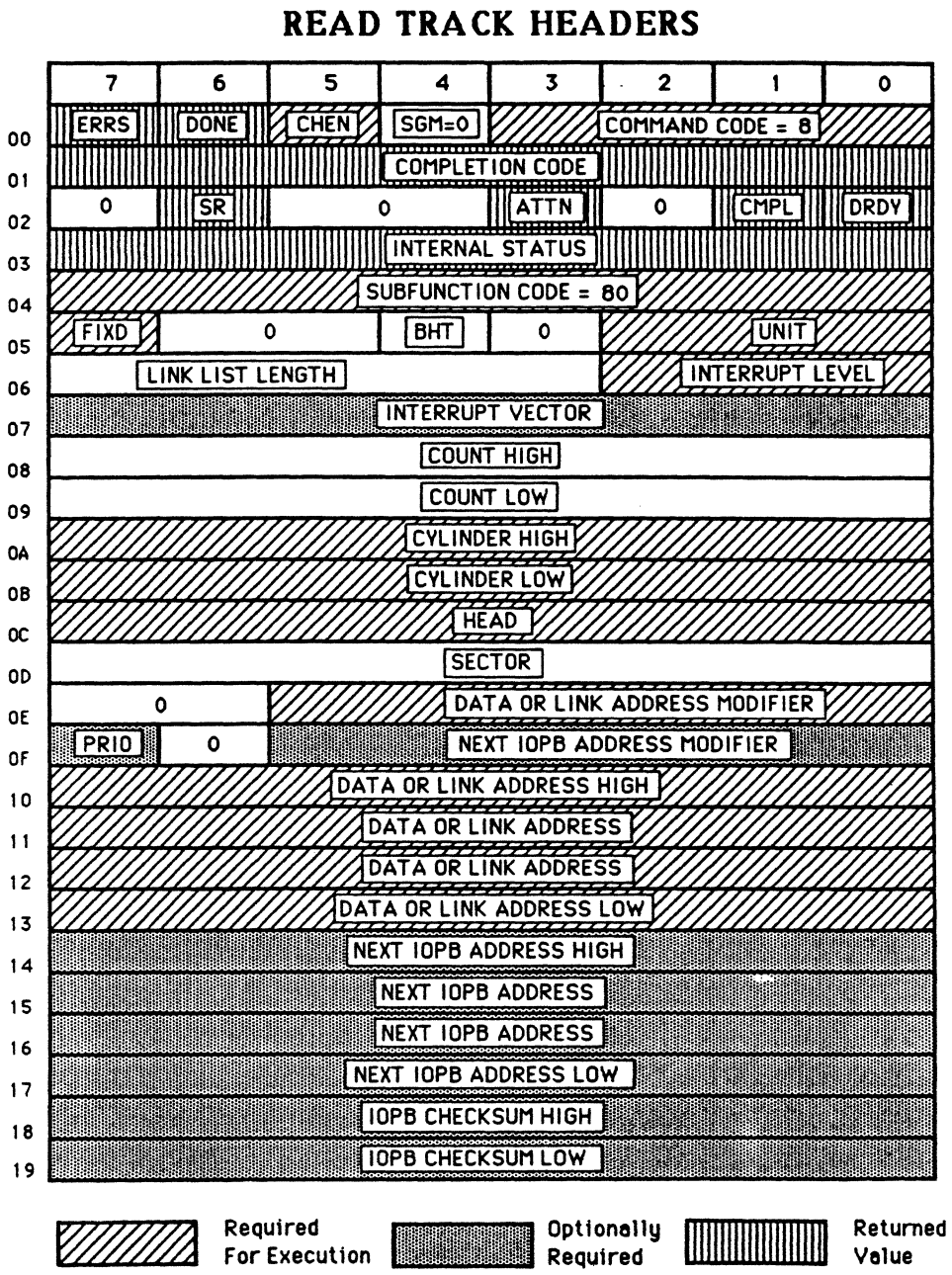
5.21 WRITE DEFECT MAP

Write Defect Map is a useful maintenance command for debugging software. The 712 uses data from host memory and writes a manufacturer's defect map to the disk. See Section 8.5.



5.22 READ TRACK HEADERS

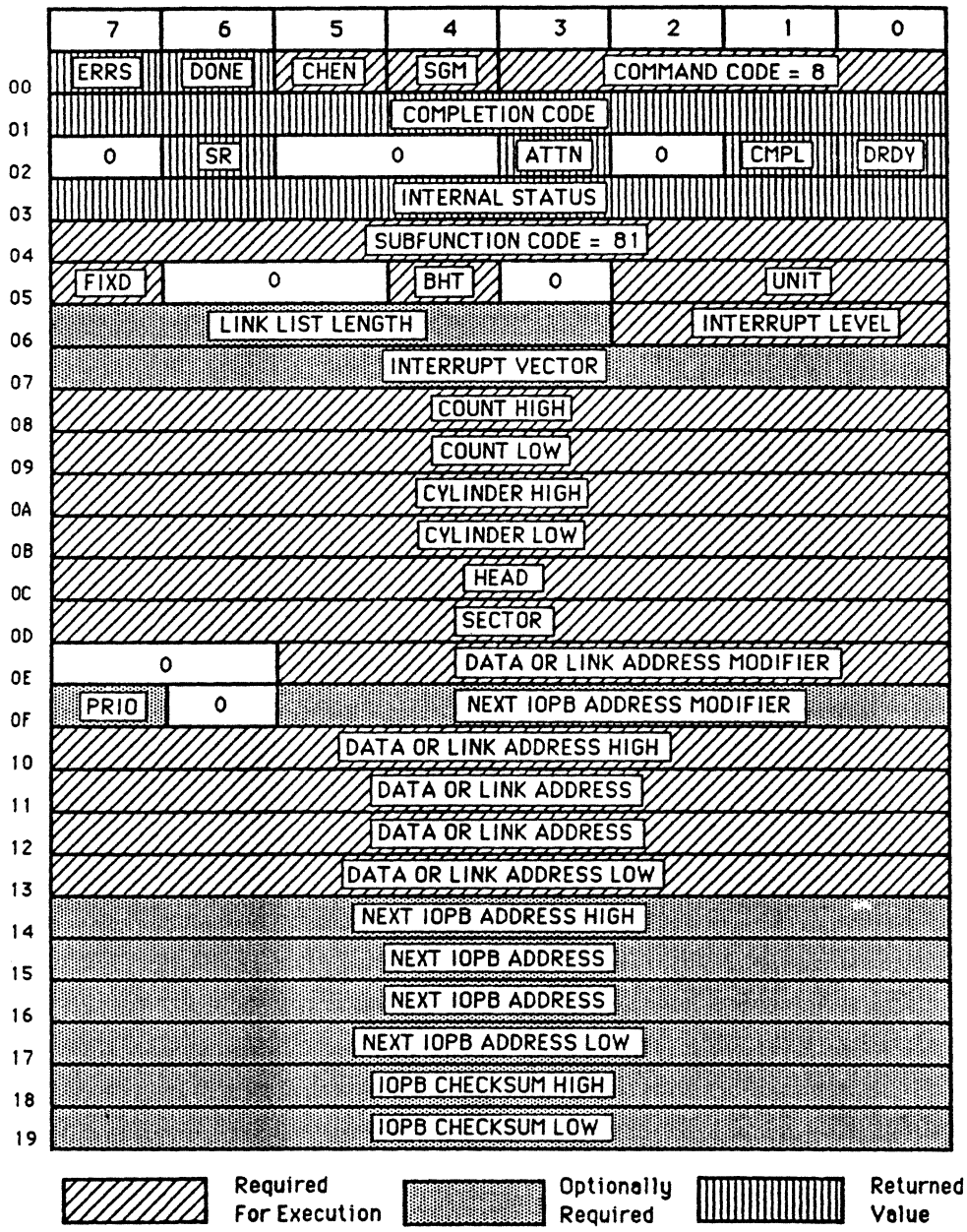
This command enables the host to read the sector header fields on a track. The 712 places the data in the header fields in host memory: four bytes per header; one header for each sector on the track. Section 8.1 defines the data format in each header.



5.23 VERIFY DATA

This command verifies the data on the disk. The 712 reads the data from the host and the disk simultaneously, and compares them on a bit-by-bit basis. The granularity of the mismatch reporting is one sector. The ending data address does not indicate where a mismatch error occurred.

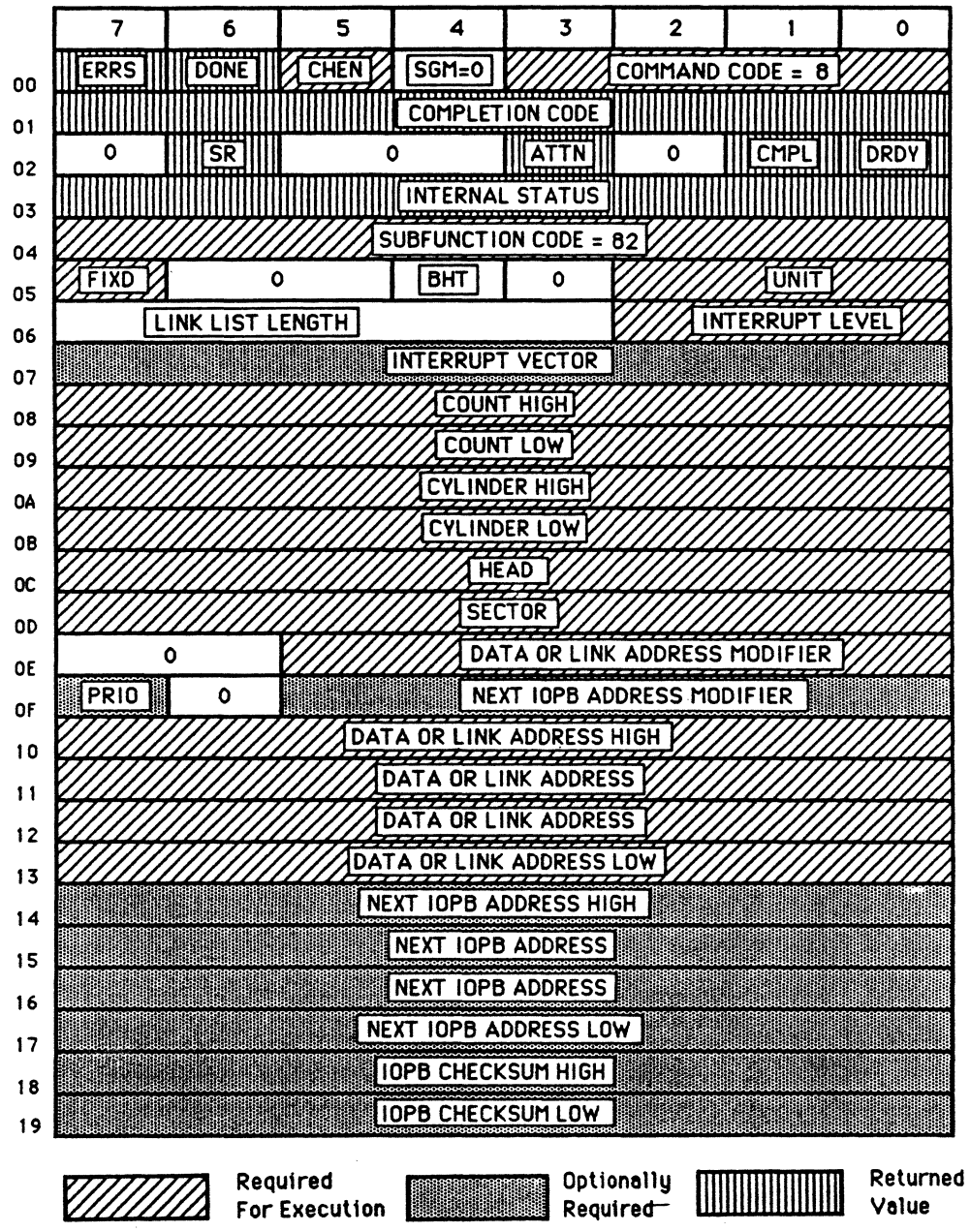
VERIFY DATA



5.24 READ HEADER, HEADER VERIFY, DATA, AND DATA ECC

This command directs the 712 to read a sector header, header verify, data, and data ECC. There are always four bytes in the header, but the other fields vary according to the initial 712 set-up. The 712 talks to physical sectors regardless of the interleave factor; it does not cross head or cylinder boundaries. See Section 8.

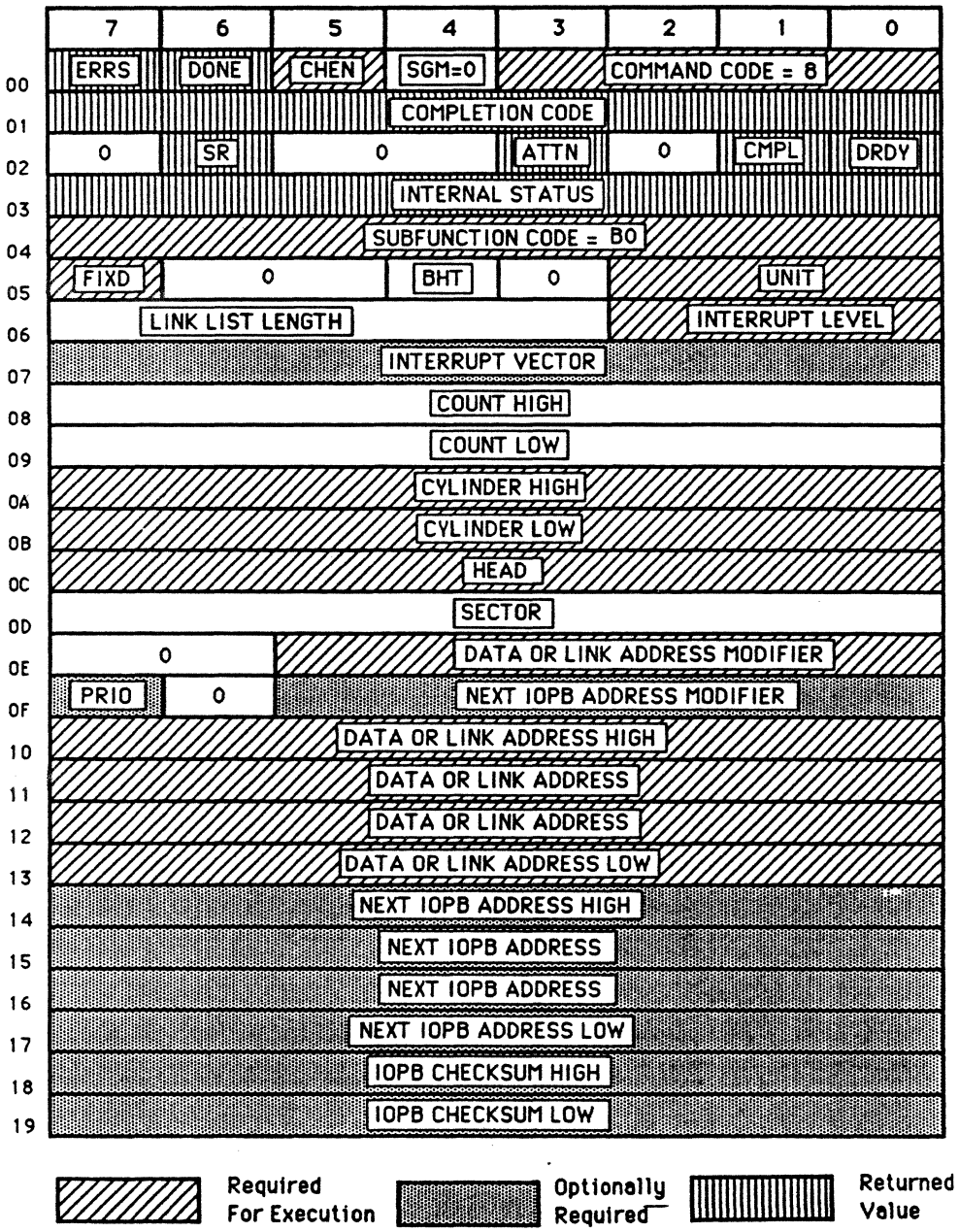
READ HEADER, HEADER VERIFY, DATA, AND DATA ECC



5.25 READ DEFECT MAP

The 712 reads the manufacturer's defect map and returns the data to memory in the correct bit order. See Section 8.5.

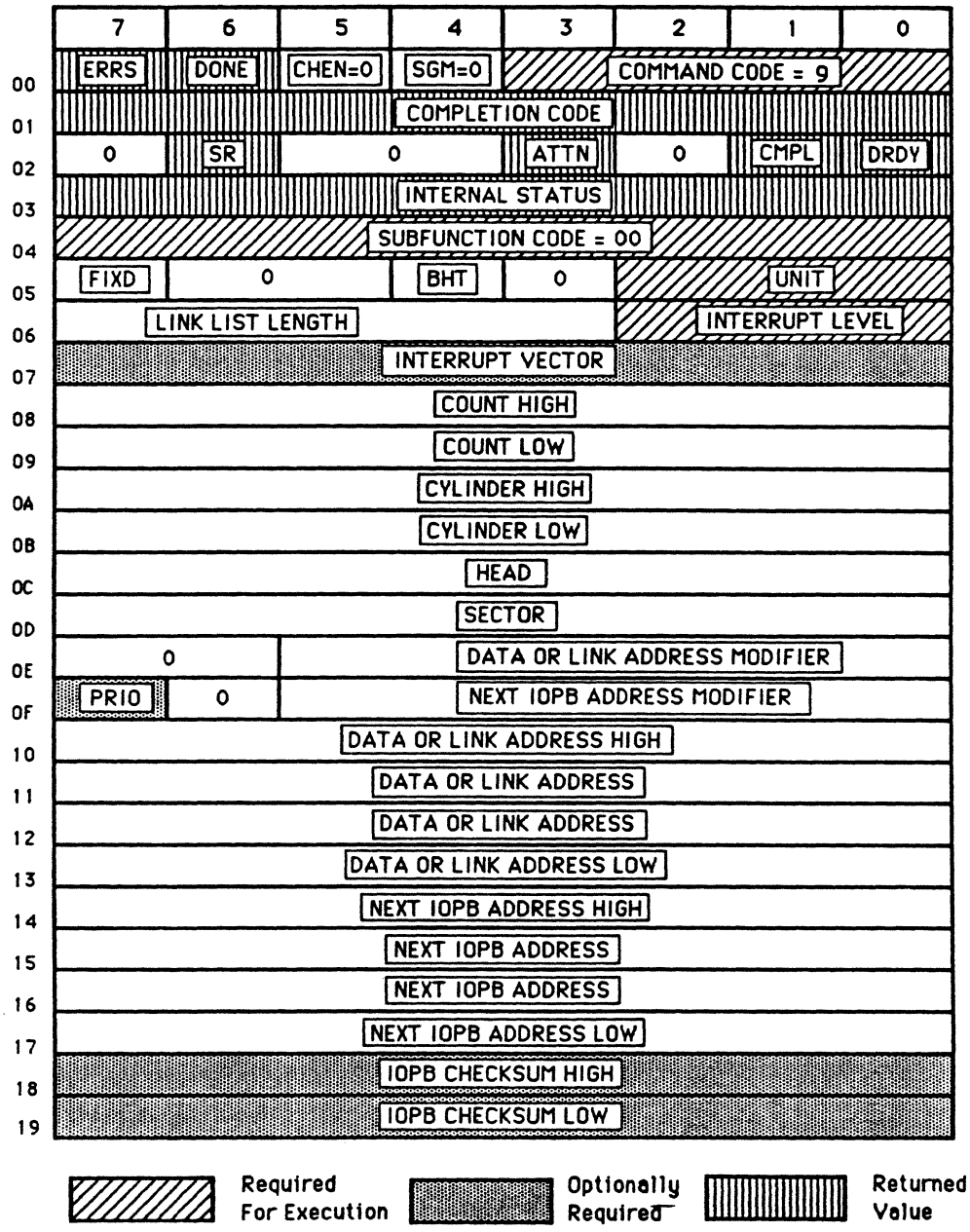
READ DEFECT MAP



5.26 DIAGNOSTICS

The 712 executes the on-board self test diagnostics. Do not chain this IOPB to another IOPB. It cannot be used in conjunction with other IOPBs in the command queue.

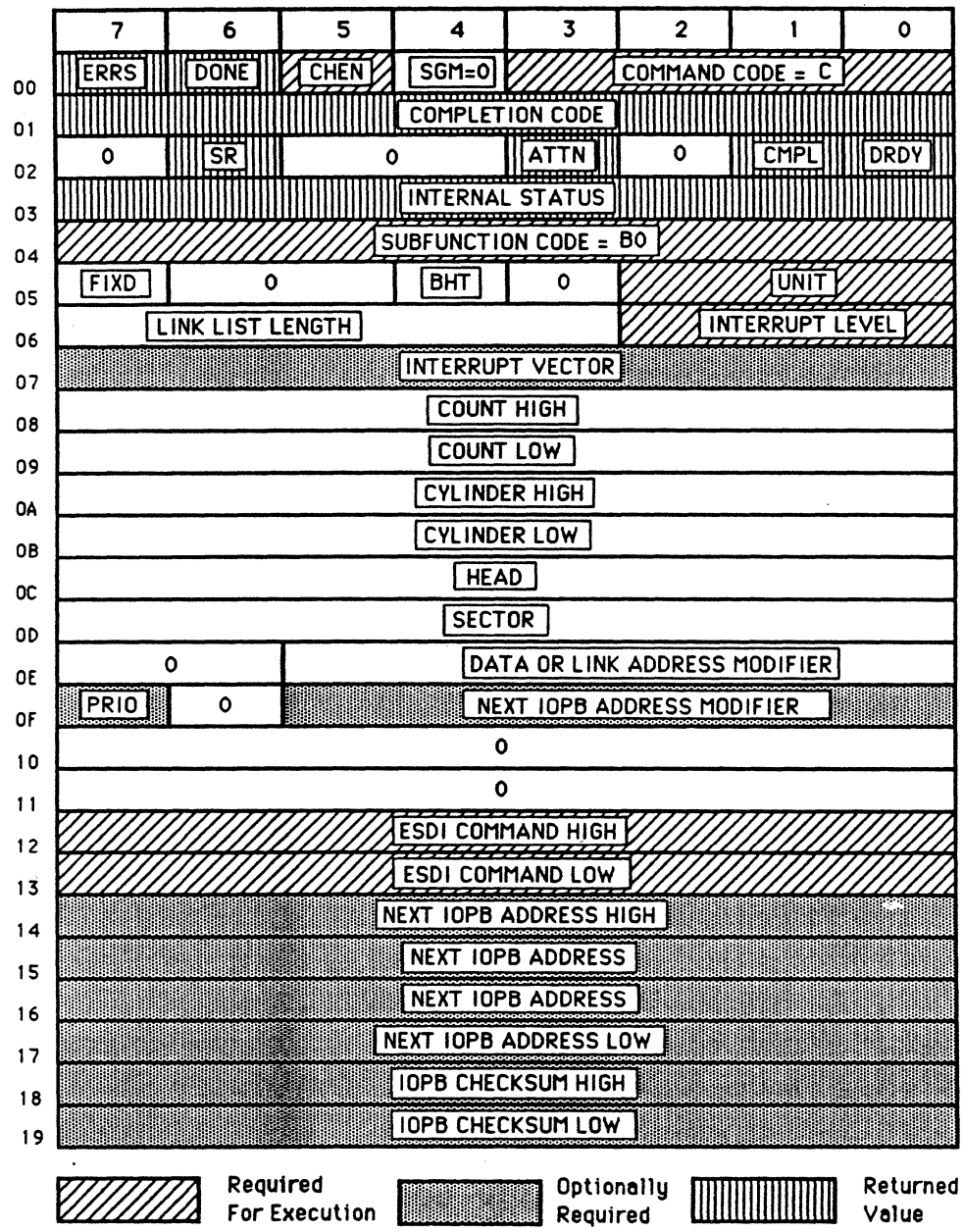
DIAGNOSTICS



5.27 ESDI COMMAND WITHOUT STATUS

The 712 sends the ESDI command specified in Bytes 12 and 13 to the drive.

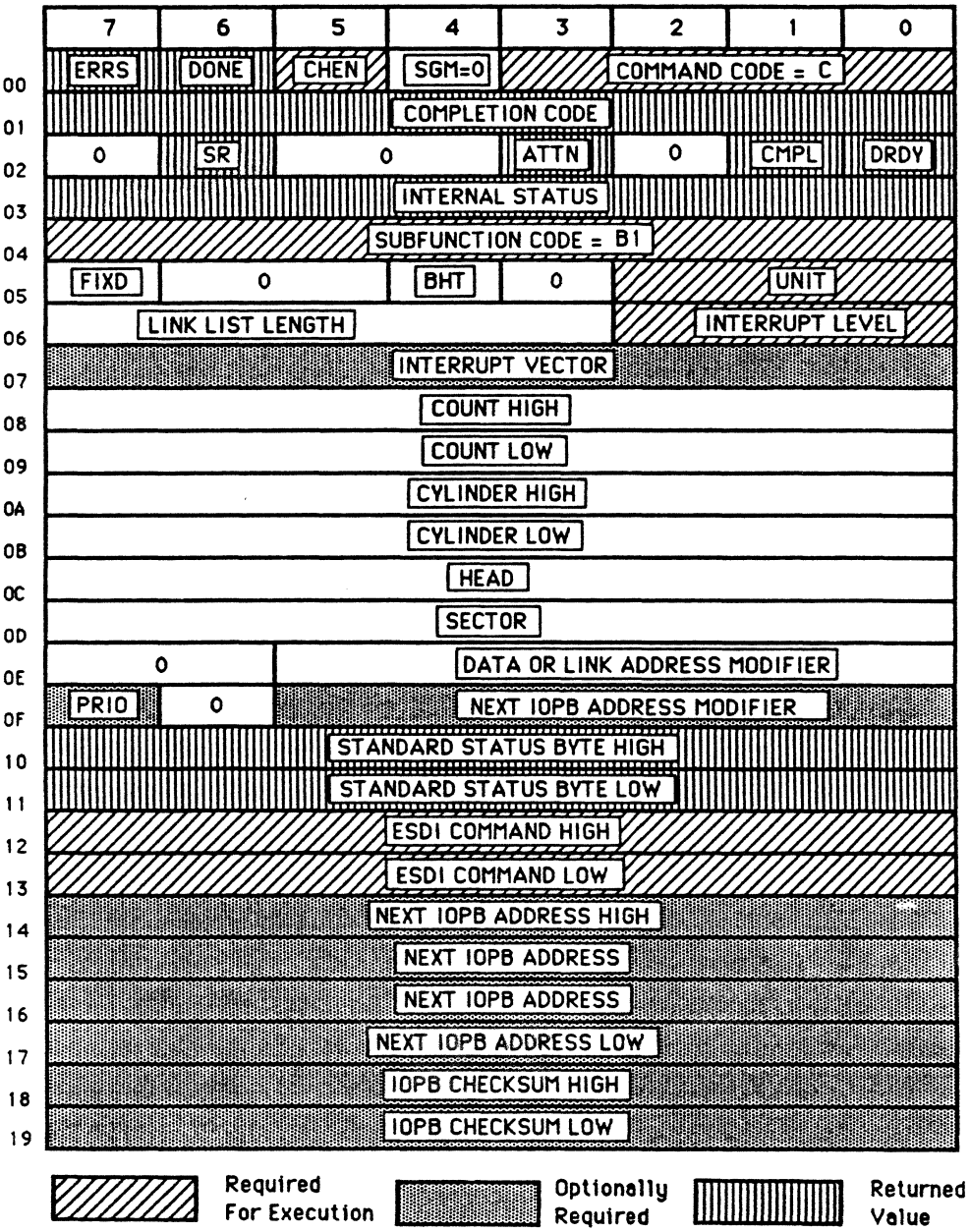
ESDI COMMAND WITHOUT STATUS



5.28 ESDI COMMAND WITH STATUS

The 712 sends the ESDI command specified in Bytes 12 and 13 to the drive. The 712 reads the ESDI status or configuration from the drive and puts it into Bytes 10 and 11.

ESDI COMMAND WITH STATUS



SECTION 6: ERROR PROCESSING

6.0 GENERAL

The Error Summary (ERRS) bit, Fatal Error (FERR) bit, and Completion Code represent the 712's status after executing a command. FERR indicates the transfer failed and the 712 requires a Controller Reset before continuing. ERRS only affects the specific IOPB and may be tested in lieu of checking the Completion Code; the 712 does not require a Controller Reset before continuing. The Completion Code informs software that the 712 successfully completed a command, failed to complete a command, or encountered and corrected a problem with one of several internal recovery procedures.

6.1 THE COMPLETION CODE

The 712 posts a Completion Code in IOPB Byte 1 (Status Byte 1); a Completion Code is only valid if DONE is set. Table 6-2 lists the Completion Codes (all codes not listed in the table are reserved). The following subsections describe these codes, along with any required corrective action.

6.1.1 Completion Code Convention

Completion Codes follow a convention that indicates the action required by either the software driver or manual intervention. The byte's upper nibble is the recovery code, and the lower nibble is the actual error code.

RECOVERY CODE	RECOVERY PROCEDURE
0	No Action / Status Only
1	Non-retryable Programming Error
3	Successfully Recovered Soft Error
4	Hard Error / Retry
6	Hard Error / Reset and Retry
7	Fatal Hardware Error
8	Miscellaneous Error
9	Requires Manual Intervention

TABLE 6-1. RECOVERY CODE

6.1.1 Completion Code Convention (continued)

ACTION	CODE (HEX)	DESCRIPTION
No Action / Status Only	00	Successful Completion
	01	Set Format Fields 5/5A Only
Non-retryable Programming Errors	10	Illegal Cylinder Address
	11	Illegal Head Address
	12	Illegal Sector Address
	13	Count Zero
	14	Unimplemented 712 Command
	15-1B	Illegal Field Lengths 1-7
	1C	Illegal Scatter/Gather Length
	1D	Not Enough Sectors/Track
	1E	Next IOPB Alignment Error
	1F	Scatter/Gather Addr. Alignment
	20	Scatter/Gather With Auto ECC
	21	Illegal Black Hole Address
	22	Illegal Field 12 Length
Successfully Recovered Soft Errors	30	Soft ECC Corrected
	31	ECC Ignored
	32	Auto Seek Retry Recovered
	33	Soft Retry Recovered
Hard Errors/Retry	40	Hard Data ECC
	41	Header Not Found
	42	Drive Not Ready
	43	Operation Timeout
	44	VMEDMA Timeout
	45	Disk Sequencer Error
	46	FIFO Parity Error
	48	Header ECC Error
	49	Read Verify
	4A	Fatal VMEDMA Error
	4B	VMEbus Error
	4C	Interface Parity Error
Hard Errors - Reset/Retry	60	Drive Faulted/Write Fault
	61	Header Error/Cylinder
	62	Header Error/Head
	64	Seek Error
	65	Miscellaneous Attention High
	66	Command not Complete
	67	Drive Interface Fault
	68	Write Gate With Track Offset
	69	Unimplemented Drive Command
	6A	Vender-unique Status Avail.
	6B	Command Data Parity Fault
	6C	Drive Power Fault

TABLE 6-2. SUMMARY OF COMPLETION CODES

.1.1 Completion Code Convention (continued)

ACTION	CODE (HEX)	DESCRIPTION
Fatal Hardware Errors	70	Illegal Sector Size
	71	Firmware Failure
Miscellaneous Errors	80	Soft ECC
	81	IRAM Checksum Failure
Requires Manual Intervention	90	Write-protect Error

TABLE 6-2. SUMMARY OF COMPLETION CODES (continued)

6.1.2 Completion Code Descriptions

6.1.2.1 No Action / Status Only

Typically, the following Completion Codes require no action; the 712 returns the codes for status only.

CODE(H)	DESCRIPTION
00	SUCCESSFUL COMPLETION — Not an error; indicates the command is complete and the IOPB may be removed from the queue.
01	SET FORMAT FIELDS 5 AND 5A ONLY — Jumper JE 1-2 is removed; therefore only Fields 5 and 5A were set with this Write Format Parameters command.

6.1.2.2 Non-retryable Programming Errors

This group of errors usually occurs while debugging drivers; they should not occur in a normal operating system environment.

CODE(H)	DESCRIPTION
10	ILLEGAL CYLINDER ADDRESS — Host software specified a cylinder address greater than the maximum cylinder number specified in the last Set Drive Parameters command for this drive. Correct the cylinder address, and retry the IOPB operation.

6.1.2.2 Non-retryable Programming Errors (continued)

CODE(H)	DESCRIPTION
11	ILLEGAL HEAD ADDRESS — Host software specified a head address greater than the maximum head address specified in the last Set Drive Parameters command for this drive.
12	ILLEGAL SECTOR ADDRESS — Host software specified a sector address greater than the maximum sector number specified in the last Set Drive Parameters command for this drive.
13	COUNT ZERO — Host software issued the 712 an IOPB that required a count, but the count was zero. Read, Write, and Format commands require a valid count.
14	UNIMPLEMENTED CONTROLLER COMMAND — This error occurs on all reserved 712 commands.
15	ILLEGAL FIELD LENGTH 1 — See Section 8.3.3.1.
16	ILLEGAL FIELD LENGTH 2 — See Section 8.3.3.2.
17	ILLEGAL FIELD LENGTH 3 — See Section 8.3.3.3.
18	ILLEGAL FIELD LENGTH 4 — See Section 8.3.3.4.
19	ILLEGAL FIELD LENGTH 5 / 5 ALTERNATE — See Section 8.3.3.5.
1A	ILLEGAL FIELD LENGTH 6 — See Section 8.3.3.6.
1B	ILLEGAL FIELD LENGTH 7 — See Section 8.3.3.7.
1C	ILLEGAL SCATTER/GATHER LENGTH — The linked list specified a number of words to transfer that does not agree with the amount of data contained in the requested number of sectors for transfer.
1D	NOT ENOUGH SECTORS PER TRACK — The format routine was unable to format since too few sectors were actually available on the track.
1E	NEXT IOPB ALIGNMENT ERROR — The Next IOPB Address did not start on a 16-bit boundary; the 712 does not execute the NIOPB.

6.1.2.2 Non-retryable Programming Errors (continued)

CODE(H)	DESCRIPTION
1F	SCATTER/GATHER ADDRESS ALIGNMENT ERROR — A Scatter/Gather address started on a byte boundary.
20	SCATTER/GATHER WITH AUTO ECC ERROR — A Scatter/Gather operation resulted in a correctable ECC error. Due to Scatter/Gather boundaries, the 712 did not automatically correct the error, but reverted to ECC Mode 0.
21	ILLEGAL BLACK HOLE ADDRESS — During a Black Hole Transfer, the data address did not start on a word boundary when the 712 was in Word mode, or it did not start on a longword boundary when the 712 was in Longword mode.
22	ILLEGAL FIELD 12 LENGTH — See Section 8.3.3.8.

6.1.2.3 Successfully Recovered Soft Errors

This group of errors is for status only. If some errors recur often, the operating system should try to map out the sectors involved. Allowing these errors to recur degrades performance.

CODE(H)	DESCRIPTION
30	SOFT ECC CORRECTED — The 712 detected and corrected one or more ECC errors, during a disk read, in ECC Mode 2.
31	ECC ERROR IGNORED — The 712 detected an ECC error, during a Read command, in ECC Mode 1.
32	AUTO SEEK RETRY RECOVERED — This is a soft error. The 712 completed the transfer successfully but, during the transfer, it had to reset the drive to recover from an error.
33	SOFT RETRY RECOVERED — The 712 encountered an error while executing this command. A retry due to RBC being set or a zero latency read was successful.

6.1.2.4 Hard Errors/Retry

These errors indicate the transfer failed; retry the operation. If several retries fail, manual intervention is required or the operating system may crash.

CODE(H)	DESCRIPTION
40	HARD DATA ECC ERROR — The 712 detected a hard data ECC error in the data field (longer than 11 bits) during a Read command. Retry the previous Read operation.
41	HEADER NOT FOUND — The 712 cannot find the requested sector. The controller searches for a match for at least one disk revolution plus five sectors to locate the header. See Section 9.3.2.
42	DRIVE NOT READY — The selected drive is not ready, but not faulted; issue a Drive Reset. Causes include: o Drive not up-to-speed. o Drive hardware error. o Bad or improperly connected cable(s). o No drive of the specified Unit Number is connected to the 712.
43	OPERATION TIMEOUT — The 712 did not complete the IOPB within a two second timeout period.
44	VMEDMA TIMEOUT — The DMA controller did not complete within its timeout. One reason could be that memory did not respond in time.
45	DISK SEQUENCER ERROR — The disk sequencer did not complete its task within the allotted time limit. The 712 cannot send or receive the appropriate signals from the selected drive. Causes include: o Drive is not connected. o Improper or defective cabling. o Unformatted drive. o Interface fault (detected by the 712)
46	FIFO PARITY — The transfer failed; the 712 detected a FIFO parity error.
48	HEADER ECC ERROR — The 712 found a header match, but the Header ECC did not compare.

6.1.2.4 Hard Errors/Retry (continued)

CODE(H)	DESCRIPTION
49	READ VERIFY — The data read from the disk did not match the data read from memory.
4A	FATAL VMEDMA ERROR — The VMEDMA stopped for no apparent reason. The count nor the address overflowed, and there was no bus error.
4B	VMEBUS ERROR — The VME BERR* signal was asserted while the 712 was bus master (See the VMEbus Specification Manual).
4C	INTERFACE PARITY ERROR — The 712 detected a parity error on the data comming from the drive.

6.1.2.5 Hard Errors - Reset/Retry

This group of errors indicate the transfer failed. Software should issue a Drive Reset command to the drive in use before retrying the operation.

CODE(H)	DESCRIPTION
60	DRIVE FAULTED / WRITE FAULT — The selected drive is faulted. Issue a Drive Reset. If the fault persists, you must intervene.
61	HEADER ERROR/CYLINDER — The cylinder address did not match during a sector search. Check the cylinder address and retry the operation.
62	HEADER ERROR/HEAD — The head address did not match during a sector search.
64	SEEK ERROR — The disk drive reported a seek error.
65	MISCELLANEOUS ATTENTION HIGH — The drive asserted Attention High during the transfer. The 712 did not decode the drive status. Issue a read parameters command with subfunction B0, and decode the drive status.
66	COMMAND NOT COMPLETE — The drive deasserted Command Complete. Issue a Drive Reset, and retry the command.

6.1.2.5 Hard Errors - Reset/Retry (continued)

CODE(H)	DESCRIPTION
67	DRIVE INTERFACE FAULT — The 712 decoded a drive interface fault on the drive's status lines. Issue a Drive Reset and retry the command. If the error persists, you must intervene.
68	WRITE GATE WITH TRACK OFFSET FAULT — The 712 decoded a Write Gate with track offset fault on the drive's status lines. Issue a Drive Reset and retry the command.
69	UNIMPLEMENTED DRIVE COMMAND — The drive does not support the last command issued.
6A	VENDOR-UNIQUE STATUS AVAILABLE — Vendor-unique error. Issue a Read Drive Status command (see your drive manual for specific error information).
6B	COMMAND DATA PARITY FAULT — The disk drive detected a parity error in the command data. Issue a Drive Reset and retry the command.
6C	DRIVE POWER FAULT — The drive issued a Power-on/Reset fault. Read the drive status and reconfigure the drive. If the error persists, you must intervene.

6.1.2.6 Fatal Hardware Errors

These errors indicate the hardware failed. Manual intervention or a Controller Reset may be the only recovery approach.

CODE(H)	DESCRIPTION
70	ILLEGAL SECTOR SIZE — The disk drive's sector size is not large enough to hold the header, data, and specified field lengths.
71	FIRMWARE FAILURE — Flag settings or counter values are inconsistent with the firmware routines being executed. Document the conditions and call Xylogics.

6.1.2.7 Miscellaneous Errors

CODE(H)	DESCRIPTION
80	SOFT ECC ERROR — The 712 detected a correctable 11-bit or less error in the data field of the current sector, during a Read operation, in ECC Mode 0. Software must perform the final correction. See Section 6.4.
81	IRAM CHECKSUM FAILURE — The calculated checksum from the IRAM and its stored value did not match during the Self Test or read parameters command. The parameters that are in error are not necessarily in the parameters read by this IOPB; they may be elsewhere in the IRAM. Recheck all the programmable parameters. Any write parameters command resets the checksum, and any subsequent read parameters will be error free. A soft bit in the IRAM, static, or probing the board with the power on can cause this error. See Section 6.6.

6.1.2.8 Requires Manual Intervention

The write-protect error requires you to manually remove the write-protection.

CODE(H)	DESCRIPTION
90	WRITE-PROTECT ERROR — A command that writes to the disk (e.g., Write, Format, Write Track Headers) is issued, but the drive is write-protected.

6.2 ERRORS AND ZERO LATENCY READS

If a disk error occurs during a zero latency read, the 712 may retry the operation as it finishes the command. The controller posts a retry successful code if the retry succeeds. If the retry fails, the disk address and sector count reflect the error point, and all previous sectors will be complete and without error.

6.3 SOFT ERROR COMPLETION CODES

The 712 updates the IOPB with the last error it encounters; it may overwrite previous soft errors with a new soft error status or a hard error status.

6.4 ERROR CORRECTION CODE

Most ECC algorithms require retrying the operation at least once before attempting the correction. When RBC is set, the 712 automatically retries the operation once before applying the correction algorithm.

6.4.1 Error Correction Code - Mode 0

When utilizing Mode 0, use the following procedure to correct a soft ECC error. The 712 provides a pattern and offset for the correction process.

1. Reserve 32-bits of storage for the shifted ECC pattern, and initialize them to zero. Take the ECC Pattern word from the IOPB and put it in the lowest 16 bits of the reserved space.

2. Get the offset from the IOPB and decrement by one. This makes the count zero-based instead of one-based.

3. Use the three low order bits of the offset as a count to shift the pattern the number of count bits left.

4. Divide the bit address by eight (by performing three logical shifts to the right). The result is the word offset into the bad sector. Adding this offset to the starting memory address of the sector in error creates a pointer to the first word to be corrected.

5. Exclusive-OR the two Memory words at the pointer and the two Pattern words generated in step 1.

6.4.2 Error Correction Code - Mode 1

The 712 does not correct any detected errors in Mode 1. After completing the operation, it posts a Completion Code indicating that at least one ECC error occurred during the transfer.

6.4.3 Error Correction Code - Mode 2

The 712 automatically corrects a soft ECC error in this mode. The 712 determines the pattern and offset, completes the DMA, and goes to host memory to fetch the data in error; it corrects the data, and returns it to memory.

6.5 FATAL ERROR CODES

If a fatal error occurs, the 712 sets FERR in the Status Register and posts the error code in the Fatal Error Register. (The following error codes appear only in the Fatal Error Register.) The only way to clear a fatal error is by issuing a Controller Reset (CRST).

CODE	DESCRIPTION
E0	IRAM CHECKSUM FAILURE — The IRAM checksum did not match the expected checksum following bus initialization.
E1	IRAM SELF TEST FAILURE — The 712 writes the IRAM with an incrementing data pattern then reads it with a decrementing pattern. An error indicates a bad IRAM.
E2	EPROM CHECKSUM FAILURE — At power-up, the EPROM checksum did not match the IRAM checksum. Either the EPROM is degraded, or the IRAM changed during power-down.
E3	MAINTENANCE TEST 3 FAILURE — The 712 writes the Writable Control Store in the DSKCEL with an incrementing data pattern then reads it with a decrementing pattern. An error indicates a bad DSKCEL.
E4	MAINTENANCE TEST 4 FAILURE — The 712 shifts a pattern of 0's and 1's through the Header Shift Register (HSR). An error indicates a bad HSR.
E5	MAINTENANCE TEST 5 FAILURE — The 712 writes the VMEDMA Registers and then reads them. An error indicates a bad VMEDMA.
E6	MAINTENANCE TEST 6 FAILURE — There is a problem with the REGCEL chip.
E7	MAINTENANCE TEST 7 FAILURE — The FIFO parity circuit failed its diagnostic.
E8	MAINTENANCE TEST 8 FAILURE — The 712 fills the Disk FIFO with sequential data and then reads it. An error indicates a problem with the DSKCEL or FIFO.
F0	IOPB CHECKSUM MISCOMPARE — The generated checksum did not match the appended checksum. This error can only occur while IOPB checksum feature is active. ICS is controlled via controller parameters. See Section 8.15.

6.5 FATAL ERROR CODES (continued)

CODE	DESCRIPTION
F1	IOPB DMA FATAL — The 712 did not complete the DMA within the prescribed timeout period. The memory could be defective or not present; the 712 may not have been able to become bus master.
F2	IOPB ADDRESS ALIGNMENT ERROR — The IOPB address did not start on a 16-bit boundary. Change the address of the IOPB and retry.
F3	FIRMWARE ERROR — Flag settings or counter values are inconsistent with the firmware routines being executed; the IOPB cannot DMA the appropriate error status. The 712's state is indeterminate; you must issue a Controller Reset.
F5	ILLEGAL MAINTENANCE MODE TEST NUMBER — The command is invalid, or the Maintenance mode jumper is not in.
F6	ACFAIL ASSERTED — The VMEbus signal ACFAIL is asserted, causing the 712 to stop. Correct the problem asserting ACFAIL and then reset the 712.

6.6 IRAM CHECKSUM

Each time the 712 executes a read parameters command, it compares a generated checksum with the stored checksum. This checksum encompasses the area that contains all the parameters, not just the ones being read. When this error occurs, the checksums did not match; rewrite or check all the parameters. Any write parameters command generates and stores a new checksum.

SECTION 7: A TUTORIAL IN PROGRAMMING THE 712

7.0 GENERAL

This section describes programming the 712 for basic use. This tutorial programming procedure begins with a single NOP IOPB and progresses to normal Read and Write commands. Each section builds on the previous section's information. (In the Sent/Returned portion of each sample IOPB, the x represents an indeterminate value that depends on the external conditions.)

7.1 NO OPERATION (NOP)

The NOP command allows you to become familiar with the 712 programming interface.

NOP

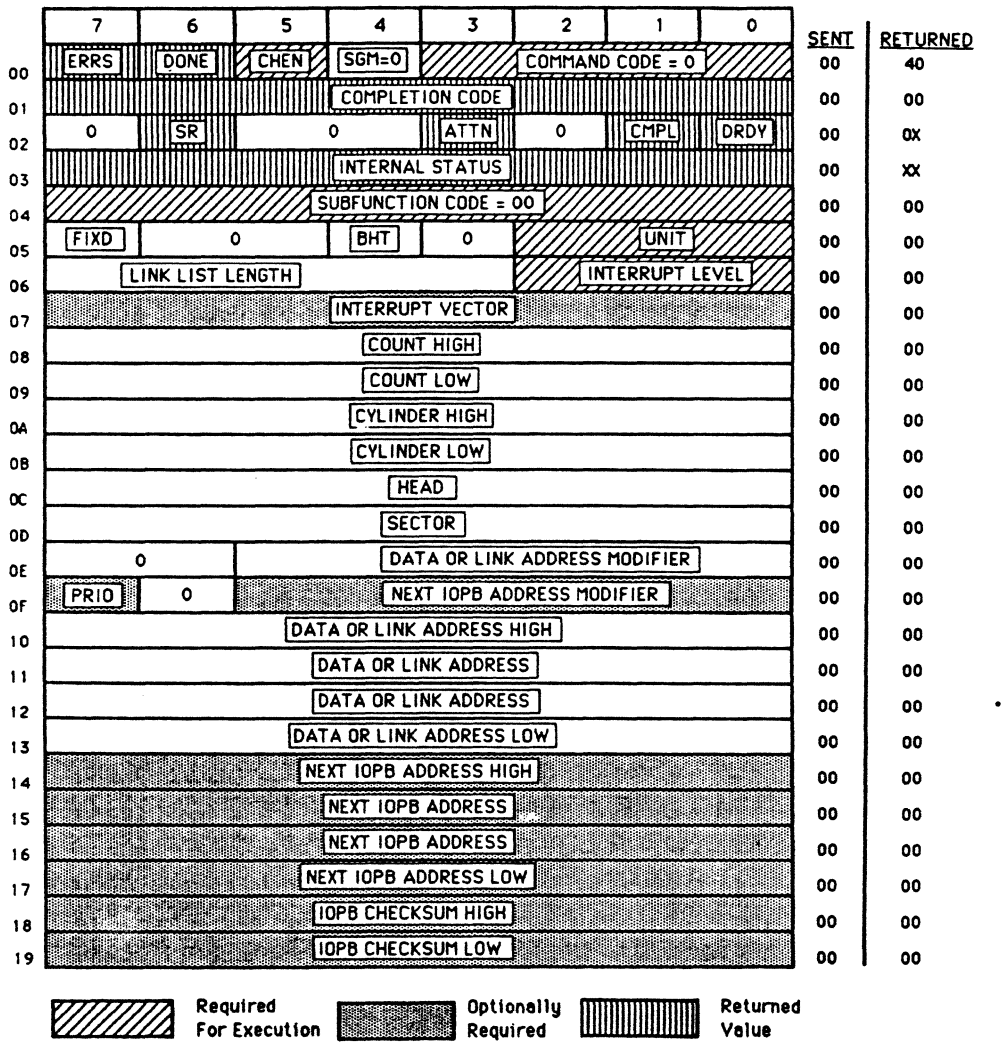


FIGURE 7-1. SAMPLE NOP IOPB

7.1.1 Allocating Memory for an IOPB

First, allocate space in host memory to store the IOPB. This allocation is a function of the operating system or the program that is currently executing. Next, set up the IOPB to execute a simple NOP command.

7.1.2 Point the 712 to the IOPB

The IOPB is now in host memory. Point the 712 to the IOPB by loading the IOPB address and address modifier into the appropriate 712 registers. Make sure the address compensates for any memory mapping that may be done between virtual and physical addressing in your system. The 712 looks for the IOPB at the physical address to which the registers point.

7.1.3 Starting the Operation

The 712 now points to the IOPB in host memory. Writing the AIO bit in the CSR directs the 712 to process the IOPB.

7.1.4 712 Operation

At this point, the 712 performs the following functions:

1. Clears AIOP and sets BUSY.
2. Reads the IOPB from host memory.
3. Decodes the command.
4. Performs the operation (NOP).
5. Sets the DONE bit.
6. Updates the IOPB.
7. Puts the completed IOPB's address into the registers.
8. Sets RIO.
9. Clears BUSY.

7.1.5 Command Completion

Software has been polling RIO (since interrupts are not enabled [Interrupt Level = 0]). Software knows that the 712 sets RIO when it is done. Software should get the completed IOPB's address from the registers, and then clear RIO. This completes the NOP command.

NOTE

Do not poll the DONE bit in the IOPB. The 712 sets DONE while the rest of the IOPB is still updating.

7.1.6 Returned Values

DONE is set in the returned IOPB. Status Byte 2 reflects the status of Disk Drive 0. Status Byte 3 reflects the 712's internal status.

NOTE

Status Byte 3 is proprietary to Xylogics and may change definition without notice.

7.2 READ CONTROLLER PARAMETERS

Next, implement a Read Parameters command with a Controller Parameters subfunction (See Section 4.2). This command returns several controller parameters in the returned IOPB.

READ CONTROLLER PARAMETERS

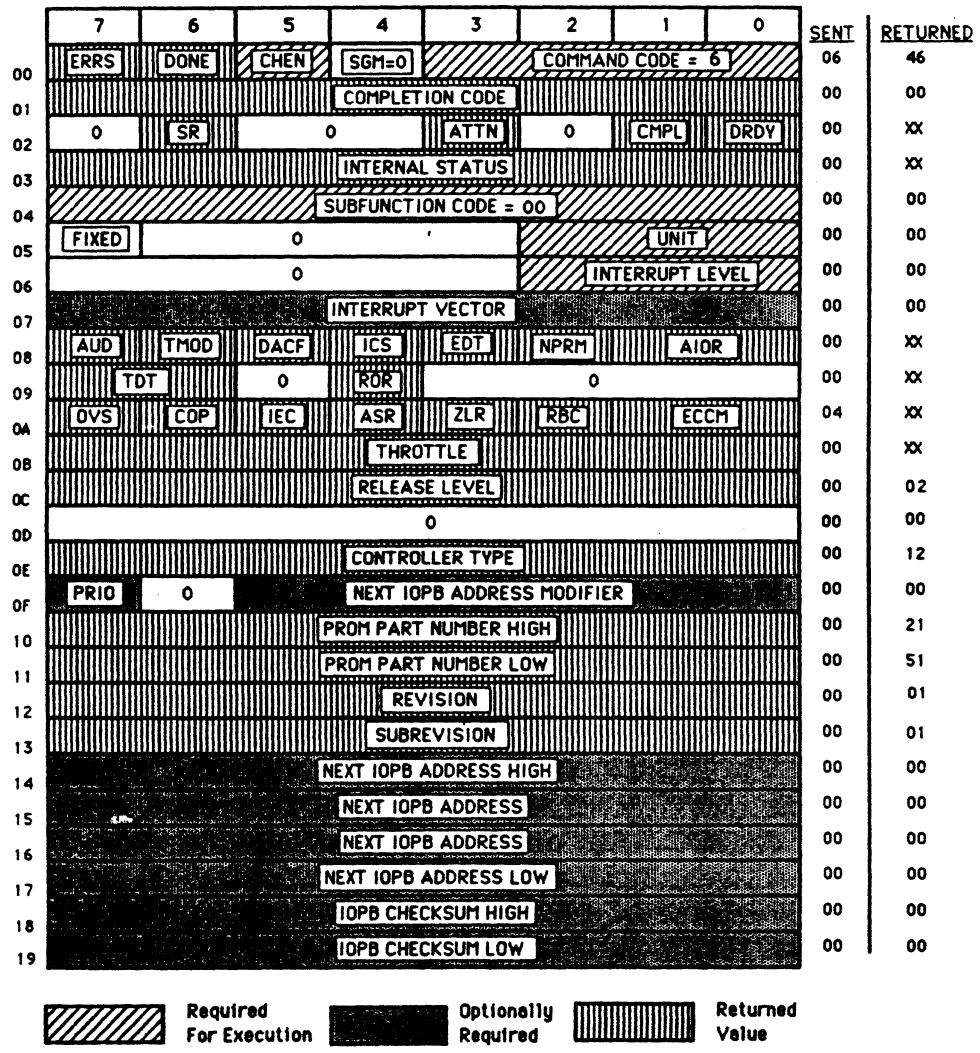


FIGURE 7-2. SAMPLE READ CONTROLLER PARAMETERS IOPB

7.2.1 Execute the IOPB

Set up the IOPB in host memory; point the 712 to the IOPB. Set AIO and the 712 executes this IOPB.

7.2.2 712 Operation

The controller operation changes slightly from the example in Section 7.1.4.

The 712 performs the Read Controller Parameters operation instead of the NOP. The controller gets the parameters from its internal store, and puts them in the proper IOPB locations. The 712 fully updates the IOPB, including the returned values.

While reading the controller parameters, the 712 calculates a new internal RAM (IRAM) checksum and compares it to the previous value. The 712 returns the appropriate Completion Code if the values do not match.

7.2.3 The Returned IOPB

The values in the returned IOPB describe the last setting of the software-programmable parameters. Determine if each value works for your application. After making any necessary changes, write the parameters back to the 712.

Specific bytes have known values. The Controller Type byte contains a 12H; the PROM Part Number byte contains 21H and 51H. See Section 4.2 for more information.

7.3 WRITE CONTROLLER PARAMETERS

Next, write the controller parameters. Xylogics recommends reading the current parameters, modifying the ones in question, and then writing them back to the 712. This method allows you to change only those parameters that affect your system.

7.3 WRITE CONTROLLER PARAMETERS (continued)

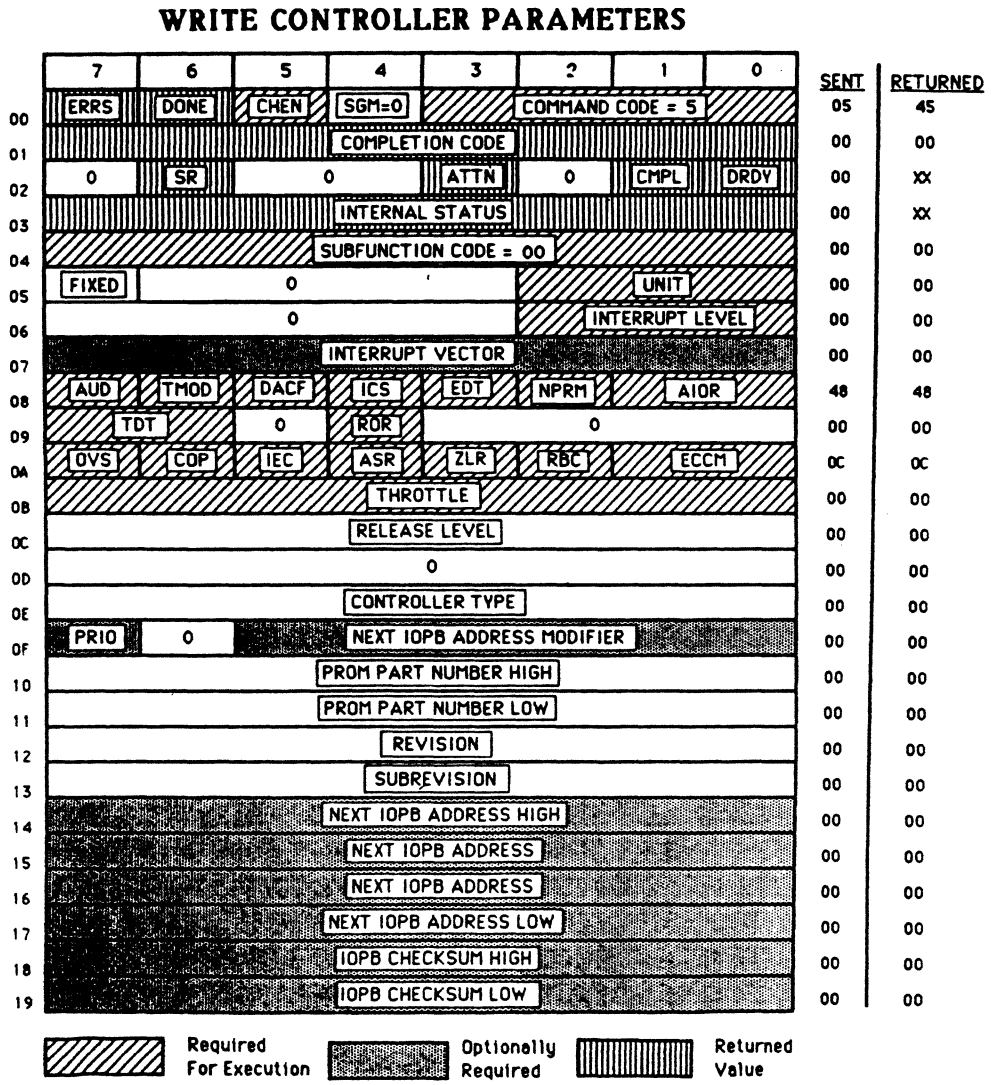


FIGURE 7-3. SAMPLE WRITE CONTROLLER PARAMETERS IOPB

7.3.1 712 Operation

The 712 executes the IOPB slightly different than in Sections 7.1.4 and 7.2.2: it performs this function by taking the values of all programmable parameters out of the IOPB and setting the appropriate flags and variables in its internal code. It also calculates a new checksum in the IRAM and stores it for use in the next reading of any parameters.

7.4 READ/WRITE DRIVE PARAMETERS

The Drive Parameters commands allow you to configure the 712 to your drive's size and parameters. Section 4.3 describes the size and configuration variables that may be modified with these commands. The operation is similar to controller parameters.

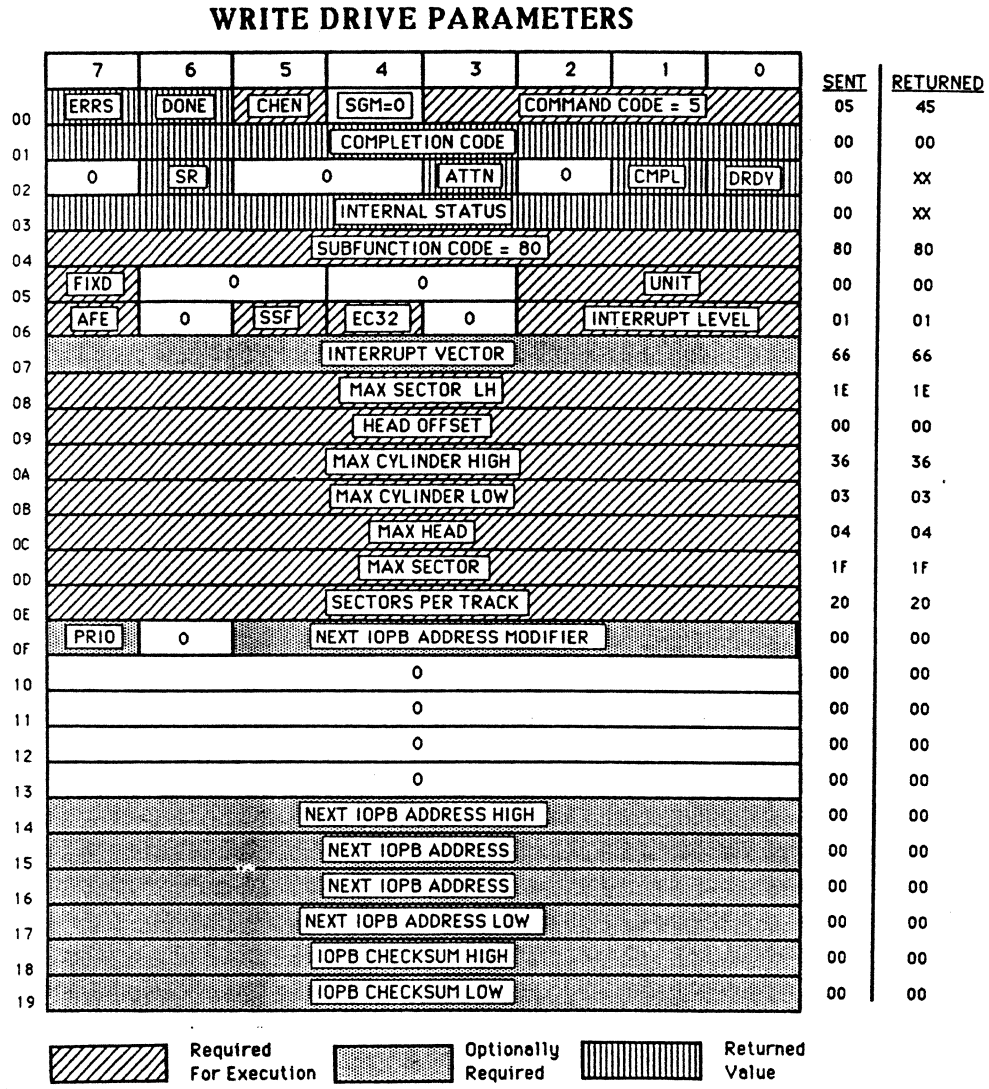


FIGURE 7-4. SAMPLE WRITE DRIVE PARAMETERS IOPB

7.4.1 712 Operation

On a Write Drive Parameters command, the 712 performs an operation similar to that of controller parameters. The Read Drive Parameters function differs in that the 712 returns the number of physical sectors on a hard-sectored drive.

7.4.1 712 Operation (continued)

The 712 selects the disk drive specified in the Unit bits. It times the interval between index pulses and, using this time value, counts the number of drive-generated sector pulses. The 712 puts this count into Byte E of the IOPB. This count is the number of sectors per track.

7.5 READ/WRITE FORMAT PARAMETERS

The format parameters are handled similarly to the controller and drive parameters. Use extra caution when modifying the format parameters as improper selection may cause data corruption and/or unreliable operation. The data field size is the only parameter Xylogics recommends changing.

READ FORMAT PARAMETERS

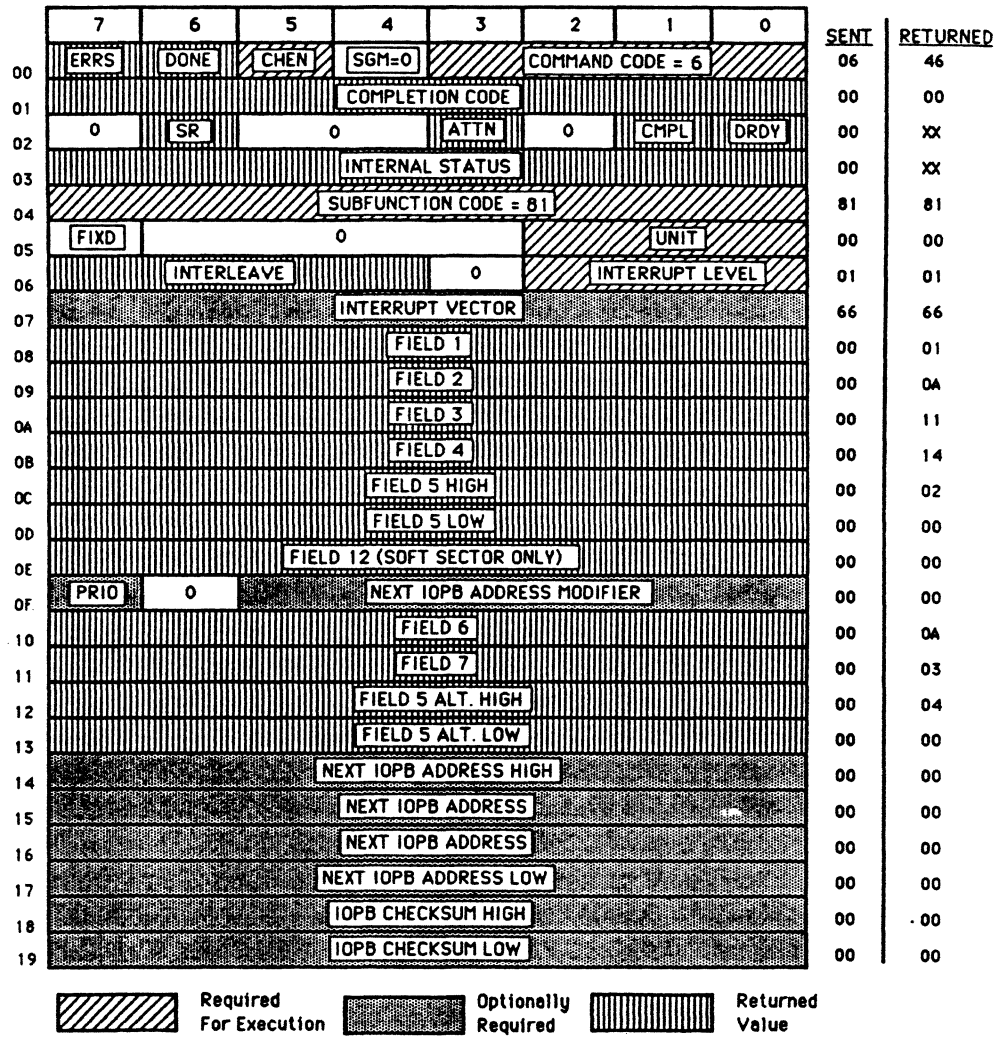


FIGURE 7-5. SAMPLE READ FORMAT PARAMETERS IOPB

7.5.1 Execute the IOPB with Interrupts

To build on 712 functionality, enable interrupts for this example by specifying an interrupt level and vector.

7.5.2 712 Operation

The 712 performs the operation almost identically to the example in Sections 7.2.2 and 7.3.1, but with two additional steps. It makes sure the new format parameters are within the valid ranges. After the 712 sets RIO, it performs an interrupt sequence.

7.5.3 Command Completion

Enabling interrupts modifies the command completion. Software does not poll RIO when it is set, but may be off doing something else (probably waiting for an interrupt). When the interrupt occurs, hardware and software execute an Interrupt Service Routine (ISR) and process the interrupt. Hardware resets the actual hardware interrupt when the ISR is called.

The ISR reads the address of the completed IOPB from the registers, and clears RIO. This completes the Read/Write Format Parameters operation.

7.5.4 Returned Values

Figure 7-5 illustrates the read portion of this subsection. The 712 returns the data for which it was last programmed. The sector size is set to 512 (200H) and the alternate sector size is set to 1024 (400H). The other fields are all set to the recommended values.

7.6 FORMAT A TRACK

Up to this point we have been initializing the 712. Initialization informs the 712 of the drive size, and parameters it requires before it can properly function. Now, let's format one track of the drive. The 712 can only execute Read and Write commands on a formatted track. (Typically, formatting is done only once in the lifetime of the media.)

7.6 FORMAT A TRACK (continued)

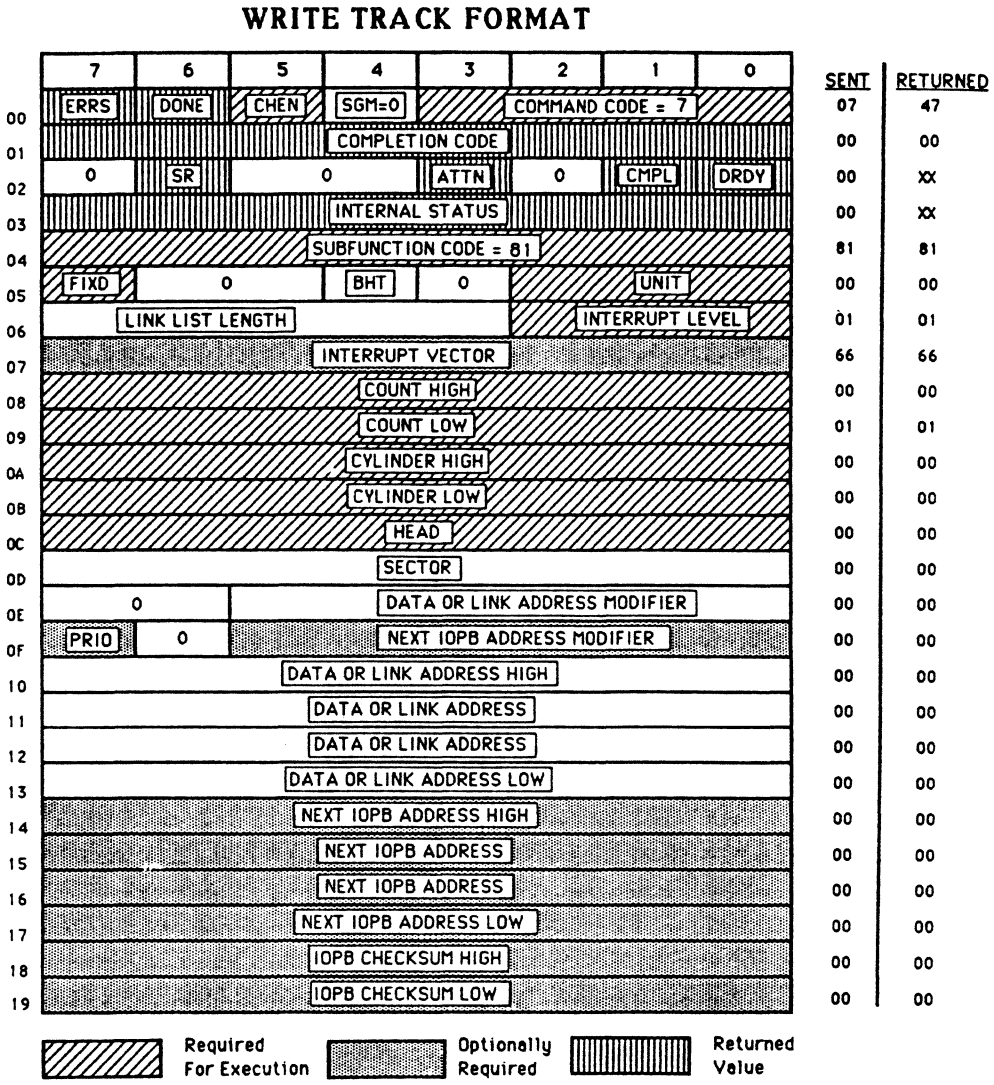


FIGURE 7-6. SAMPLE WRITE TRACK FORMAT IOPB

7.6.1 712 Operation

The Format command is the first command in this tutorial that transfers data from the controller to the disk. The 712 operation for data transfer commands differs greatly from initialization commands.

7.6.1 712 Operation (continued)

To format a track:

1. The 712 still clears AIOP, sets BUSY, and reads the IOPB from memory. The next step occurs after the IOPB is in the 712.

2. The 712 decodes the function, and determines if a seek is required. All Write, Read, Write Extended, and Read Extended functions require the drive to seek to the commanded cylinder. Format is a Write Extended function; it requires the drive to seek. The 712 issues a seek to the drive by sending it the commanded cylinder number.

3. The 712 waits for the drive to complete the seek; the drive indicates it's done by returning command complete.

4. When the drive is on-cylinder, the 712 determines if the current physical sector count for that drive is valid. If the count is not valid, it repeats the steps in Section 7.4.1 to determine the actual physical sector. The 712 proceeds to Step 5 when the count is valid.

5. The 712 loads the data for the new sector header into the FIFO, waits for index, and writes the new header on Sector 0; it writes the data field with zeros, and writes the data field ECC.

6. The 712 repeats Step 5 for each sector on the track, except it uses sector pulse instead of index pulse to start the operation.

7. The 712 updates the IOPB with the ending values, and completes the command.

7.7 READ TRACK HEADERS

Now that the track is formatted, read the headers back and verify they are correct. This command requires a data buffer; allocate space in host memory just as you did for the IOPB. The buffer length must be four bytes (per sector) times the number of sectors per track. (The Read Drive Parameters command will give you the number of sectors per track.) Make sure software passes the 712 the physical buffer address, not the virtual address.

7.7 READ TRACK HEADERS (continued)

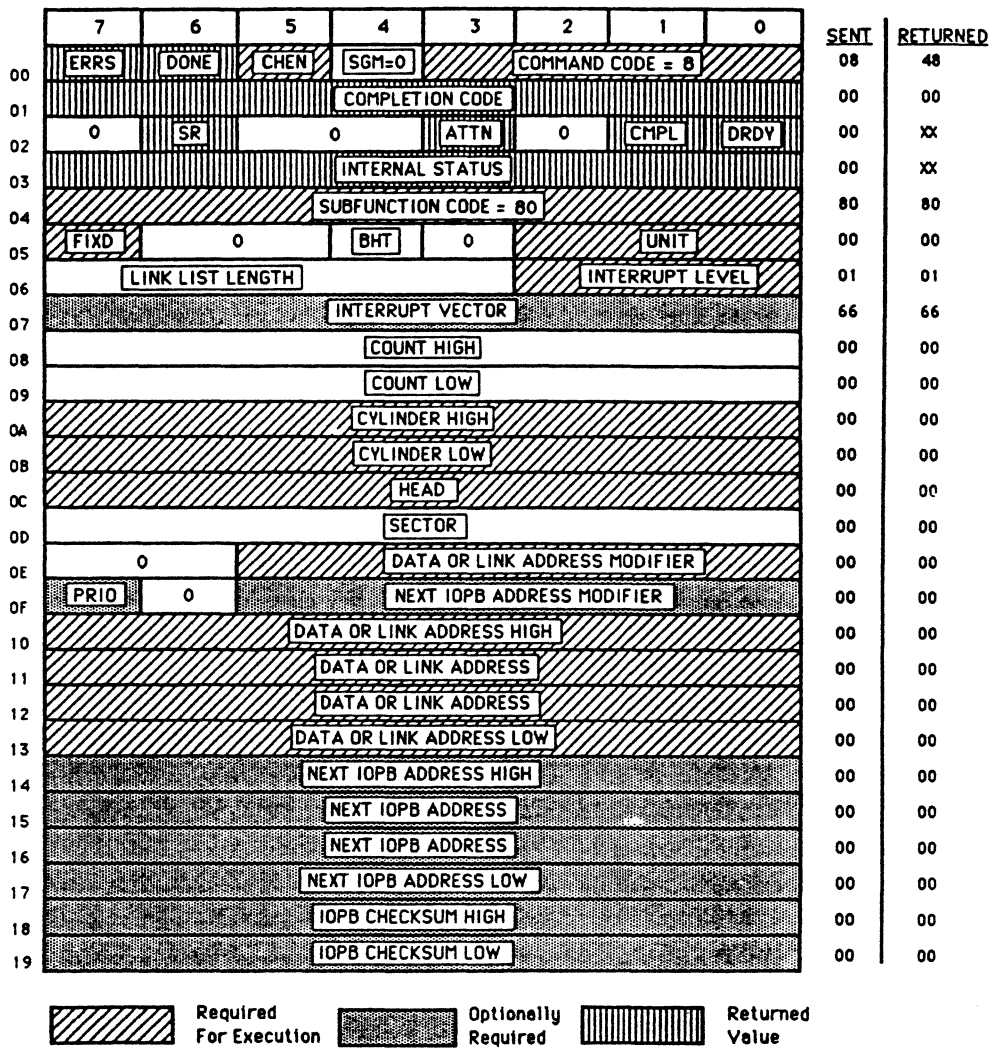


FIGURE 7-7. SAMPLE READ TRACK HEADERS IOPB

7.7.1 712 Operation

The Read Track Headers is the first command in this tutorial that transfers data to or from host memory. Data transfers to or from memory modify the 712 operation as follows:

1. The 712 reads the IOPB, issues a seek, and decodes the command identically to the previous examples.

7.7.1 712 Operation (continued)

2. The 712 waits for index from the drive. It tests the physical sector count to determine if it is valid. If the count is not valid, the 712 determines the actual count (See Section 7.4.1).

3. After index arrives, the 712 synchronizes itself with the data in the header, and reads the data into its on-board FIFO.

4. The 712 repeats Step 3 for each sector on the track, except it waits for the sector pulse instead of index. The actual physical sector count determines the number of sectors the 712 transfers into its FIFO.

5. After the 712 transfers all sector headers into the FIFO, it begins the DMA. The controller transfers the data from the FIFO to host memory.

6. When the 712 completes the DMA transfer, it places the updated information into the IOPB, and completes the command.

7.7.2 Verifying the Data

After the 712 completes the transfer, verify the data against what is expected. The data should be divided into groups of four bytes each. Each group describes a sector header. The first sector header should be four bytes of 00. The second header should be three bytes of 00 and one byte equal to 01. The third header should have the last byte equal to 02, etc. (See Section 8.1.2 for more information on headers).

HEADER 1	HEADER 2	HEADER 3
00 00 00 00	00 00 00 01	00 00 00 02

FIGURE 7-8. SAMPLE SECTOR HEADERS

7.8 WRITE DATA

The format is valid after verifying the headers. This subsection describes a Write operation, and the following subsection describes reading back the data. Allocate space in host memory for the buffer, and set up a data pattern in this buffer; an incrementing count in the buffer will suffice.

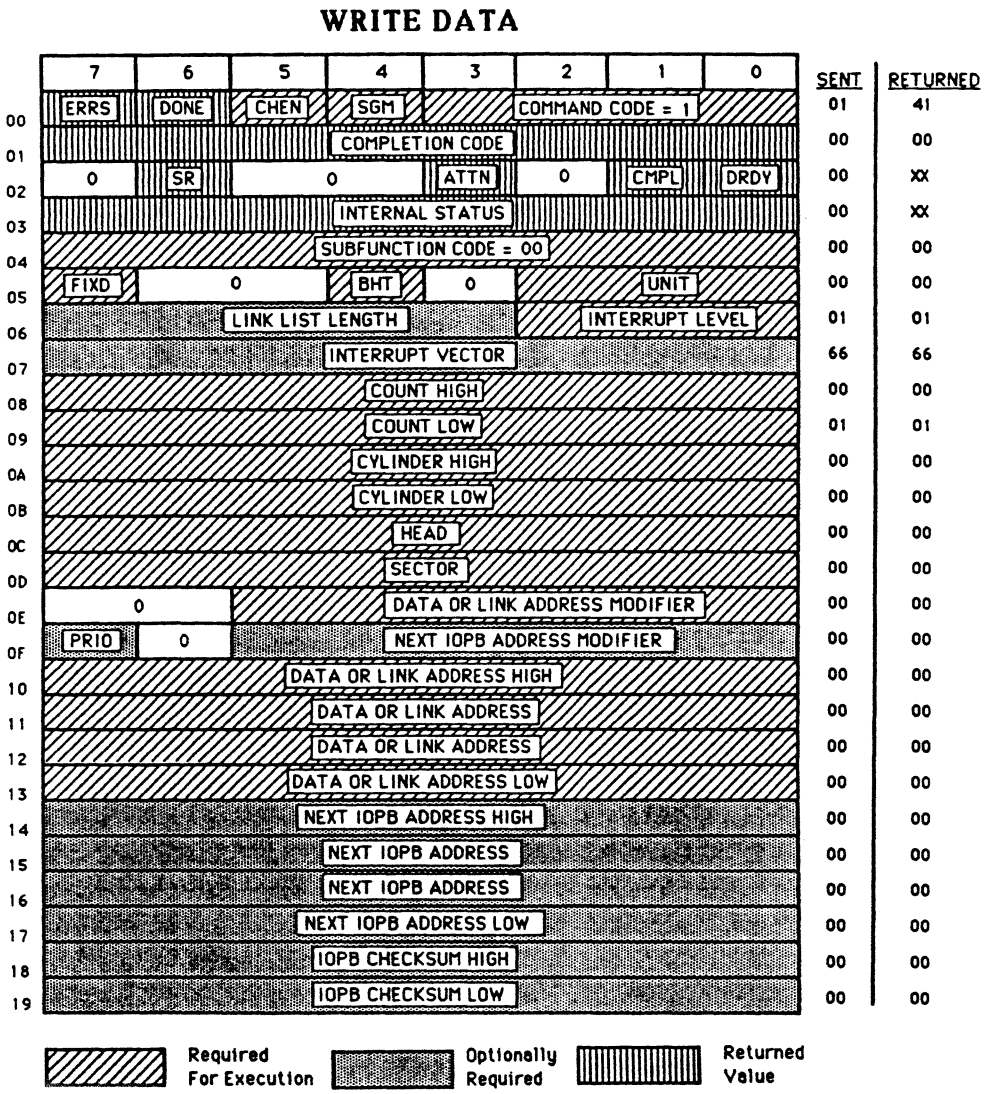


FIGURE 7-9. SAMPLE WRITE DATA IOPB

7.8.1 712 Operation

The 712 operation is similar to the previous examples; the differences are in DMAing data into the FIFO, and writing data to the disk.

The 712 starts the DMA from host memory to the FIFO after determining the drive is on-cylinder; it enables the disk sequencer when the FIFO contains one full sector of data.

The 712 compares and verifies the header: the disk sequencer tests all the headers as they pass under the head, until it finds the sector designated for transfer. At the proper point in the sector, the 712 writes a new Sync byte, and then the data it read from memory. The 712, using the data to be written, generates and appends an ECC on the end of the sector.

7.8.2 Command Completion

The command is complete as soon as the disk sequencer completes its operation. The 712 puts the ending values into the internal IOPB, and performs an appropriate update.

7.9 READ DATA

The 712 writes the data to the drive on Sector 0, Head 0, and Track 0. This subsection describes reading back the data and verifying it. You must allocate a data buffer for the 712 to write the data in memory. After allocation, it is a good idea to fill the buffer with a known pattern that differs from the expected data.

7.9.1 712 Operation

The 712 treats this command like the previous operations, except in the way it reads the data from the disk, and DMAs the data from the FIFO.

The 712 enables the disk sequencer as soon as the drive is on-cylinder. After the controller finds the correct header, it transfers the data from the disk to the FIFO. As soon as the first word of data is available from the buffer, the DMA controller DMAs the data from the FIFO to host memory. The transfer is done when the DMA controller completes the DMA.

7.9.2 Command Completion

The 712 completes the command when the DMA to memory is complete. The next subsection describes how to verify the data.

7.9.2 Command Completion (continued)

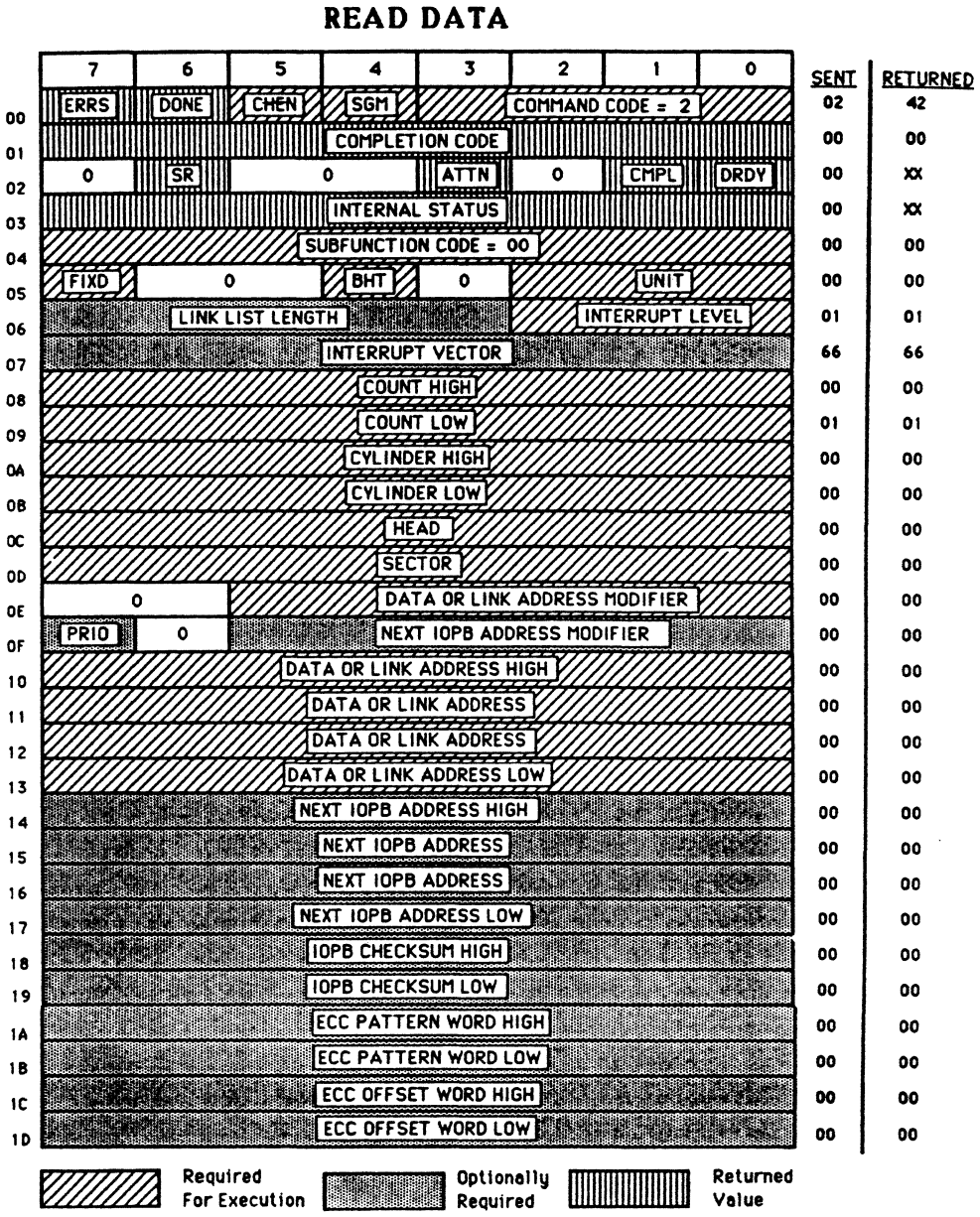


FIGURE 7-9. SAMPLE READ DATA IOPB

7.9.3 Verify Data

First, make sure the buffer was modified. If it was not modified, either an error occurred, or software specified the wrong buffer address. Next, compare the data written with the data read; they should match.

7.10 MULTIPLE SECTOR TRANSFERS

You can repeat the steps in Sections 7.8 and 7.9 using a larger sector count. The 712 crosses head and cylinder boundaries, as required, to complete the required number of sectors. Be sure to allocate enough buffer space for the increased sector count.

7.11 SUMMARY

This section was an exercise in testing the 712's functionality in your system. The steps are basically the same when the software driver controls the 712. (Operating systems always allocate the buffers.)

SECTION 8: 712 SPECIAL FUNCTIONS

8.0 GENERAL

This section describes how to implement the various 712 special functions. Each subsection describes how minor functions implement a given major function.

8.1 MEDIA DEFECT HANDLING

There are three methods for handling media defects: 1) slipping a sector; 2) remapping a sector to a new sector on the last head of that cylinder; and 3) remapping the entire track to a different track on the disk.

Each of these methods has a dedicated subsection (including information pertaining to its performance difference). Section 8.3 describes how to allocate spare sectors. Section 8.5 describes how to read the manufacturer's defect information.

8.1.1 Slipping a Sector

Slipping a sector requires using the Read and Write Track Headers commands to mark the bad sector and slip the rest of the sectors into the next position on the disk. Figure 8-1 shows an 8-sector track before and after slipping Sector 3.

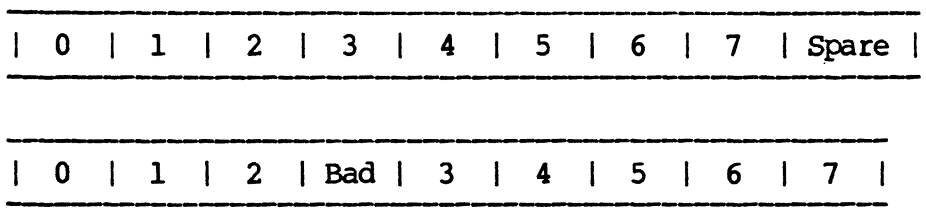


FIGURE 8-1. SECTOR SLIP

8.1.1.1 Sector Slip Procedure

1. Determine which sector is bad by writing and reading the track with several patterns.
2. Read the track headers into a buffer in host memory.
3. Compare each header with the bad sector's header.

8.1.1.1 Sector Slip Procedure (continued)

4. After locating the bad sector, mark it by writing EEH into each of the four header bytes (See Figure 8-3).

5. Test the last sector to determine if it is a spare. If it is a spare, continue; if not, you must find an alternate method of sparing.

6. Move each sector header into the next location, "slipping" the sectors down the track (See Figure 8-1).

7. Write the track headers back to the disk.

The following figures depict 712 headers:

Sync	Cylinder Low	Cylinder High	Head	Sector

FIGURE 8-2. NORMAL 712 HEADER

Sync	EE	EE	EE	EE

FIGURE 8-3. 712 HEADER MARKED BAD

Sync	DD	DD	DD	DD

FIGURE 8-4. 712 HEADER MARKED SPARE

Sync	New Cyl Low	CC	New Cyl High	Head

FIGURE 8-5. 712 TRACK REMAP HEADER

8.1.1.2 Advantages of Sector Slipping

A full track of information is still transferred in one revolution of the disk. Other methods of sector slipping require two or more revolutions to transfer one track of information.

8.1.1.3 Disadvantages of Sector Slipping

Having one or more spares on each track uses a lot of disk space.

8.1.2 Cylinder Sparing

Cylinder sparing is similar to sector slipping, except the spares are on the maximum head of the cylinder.

8.1.2.1 Cylinder Sparing Procedure

1. Determine the defective sector.
2. Read the track headers and mark the defective sector bad (See Figure 8-3).
3. Write the track headers back to the disk.
4. Read the track headers on the maximum track of that cylinder.
5. Find a spare sector; it contains four bytes of 0DDH. (See Figure 8-4.)
6. Put the bad sector's header into this sector. The header should contain the head and sector values for the bad sector on the original track.
7. Write the track headers back to the drive.

8.1.2.2 Advantages of Cylinder Sparing

Cylinder sparing uses less disk space for remapping bad sectors. You may decide to allow only ten spares for a drive with twenty heads. Sector slipping is less efficient as it requires a minimum of 20 sectors per cylinder for this drive (one per head).

8.1.2.3 Disadvantages of Cylinder Sparing

Cylinder sparing is slower than sector slipping. The 712 looks for the commanded sector for one revolution plus one sector on the original track; then it switches to max head and looks for another revolution plus one sector. This method takes up to three revolutions to transfer one track of information (assuming only one bad sector).

8.1.3 Track Remapping

Track remapping allows remapping an entire track to another location on the same disk drive: the 712 writes the defective track's headers with a code and new disk address for the transfer to continue.

8.1.3.1 Track Remapping Procedure

1. Read and save the defective track's headers.

2. Allocate space for a write track headers buffer. Each header has 0CCH in the second byte. The first, third, and fourth bytes contain the new Head, Cylinder High, and Cylinder Low. Write this information to the defective track with a Write Track Headers command. (See Figure 8-5.)

3. Write the track headers that were read in Step 1 to the destination track with a Write Track Headers command.

8.1.4 Recommended Remapping Procedure

Xylogics recommends using all three methods of defect mapping. Allowing one spare sector per track takes care of 95% of the media defects. An additional 0.2% of the sectors on the cylinder to be spared on the last head provide up to 99% remapping. Allowing three or four tracks for remapping should provide a defect-free media. Allocating this amount of disk space for defect handling totals 2% of the media. Having two spares per track on the same drive uses 4% of the media.

8.2 CHAINING AND MULTIPLE I/O REQUESTS

The 712 has two ways of speeding up multiple IOPB execution. One method allows the driver to chain IOPBs together, and then give the 712 a command-chain. The second method allows the driver to add IOPBs to the 712's queue by the same procedure as starting the first IOPB.

8.2.1 Chaining

Each IOPB has a Chain Enable (CHEN) bit and a Next IOPB pointer. IOPBs can be chained together by setting CHEN and having the Next IOPB pointer point to the next IOPB to be executed. Each IOPB in the chain points to the next, and, in order to stop the chain, CHEN is not set in the last IOPB.

NOTE

The Next IOPB Address is the physical address, not the virtual address.

8.2.2 Multiple I/O Requests

The following procedure allows you to add IOPBs to the 712 queue:

1. AIOP must be clear. If it is not clear, wait; it normally clears within 100 microseconds.
2. Write the five IOPB address registers to point to the beginning of the IOPB or IOPB chain.
3. Write the AIO bit.

8.2.3 712 Operation

The 712 treats IOPBs the same, regardless of how they were added to the queue. Overlap seeks, and IOPB reordering function only when they are enabled and the 712 is working with a queue or chain of IOPBs.

8.3 FORMATTING

This subsection describes formatting, including how to set the number of spares for use in media defect handling and setting the sector size.

8.3.1 Allocating Spare Sectors

You must allocate spare sectors at format time. The Write Drive Parameters command allows setting the maximum size parameters for the drive. Any sectors in excess of the drive size parameters are marked as spares. For example, a drive with 35 physical sectors is specified as having only 34 in the Write Drive Parameters command. The 712 formats 34 normal sectors and 1 spare sector.

8.3.1 Allocating Spare Sectors (continued)

A separate variable (max sector/last head) specifies the number of sectors on max head. This allows extra spares on max head for use with cylinder sparing. Given the above example, max head/max sector is set to 29. The 712 formats the last head of the same drive with 29 normal sectors, and 6 spare sectors.

8.3.2 Specify Sector Data Size

The 712 operates with drives having different sector sizes. Field 5 in the format parameters commands specifies the standard sector size. This value must be greater than 255 and less than 4097, and have an even number of bytes. Field 5 Alternate specifies an alternate sector size. The alternate size has the same limitations as the standard size. Clearing AFE with a Write Drive Parameters command selects the standard sector size; setting AFE selects the alternate sector size.

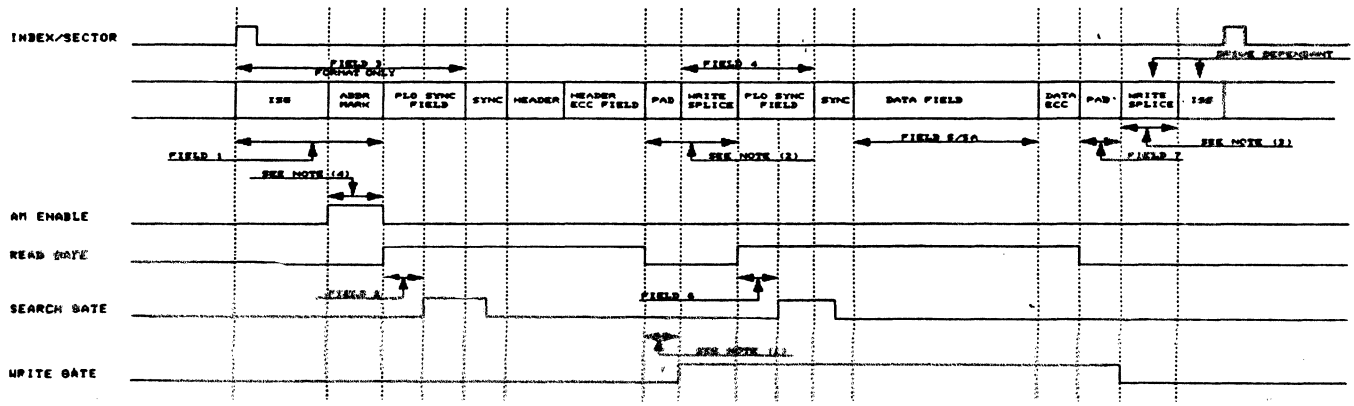
NOTE

Using four drives with different sector sizes will have a detrimental effect on the disk subsystem's performance. The 712 must modify the DSKCEL code each time it selects a disk with a different sector size.

8.3.3 Specify Sector Gap Size

This subsection outlines the parameters that affect each of the gap sizes. Many of the parameters that affect the gap sizes are specifications from the drive in use. Drives from different manufacturers, and even drives in one manufacturer's line, may have very different specifications (See Figure 8-6).

8.3.3 Specify Sector Gap Size (continued)



Key:

(1) This pad is Field 7
(2) This is Field 7 plus 1 byte
(3) For soft-sectored drives, this is the speed tolerance gap
(4) The address mark is written at format time for 3 bytes following the ISG

FIGURE 8-6. SECTOR GAP SIZES

8.3.3.1 Field 1 - Read Gate Delay - Gap 1

Field 1 specifies the time (in bytes) from the leading edge of sector or index to when Read Gate is asserted. The applicable drive specification is the head settling time. This field also provides for skipping over the ISG area.

8.3.3.1 Field 1 - Read Gate Delay - Gap 1

The head settling time is the time required for the heads to settle after the drive completes a seek. We do not know how long before the sector or index pulse the seek completed, therefore this field must be large enough to encompass the head settling time.

When Write Gate is deasserted, a minimum time must be allowed for the read heads and amplifiers to stabilize. In a multisector transfer, Write Gate is deasserted after the last sector, and Read Gate is asserted for reading the next header.

8.3.3.2 Field 2 - Sync Search Delay

Field 2 is the delay from asserting Read Gate to when data is compared for the Sync byte. Field 2 masks any read data from being detected as a sync until the data is stable.

8.3.3.3 Field 3 - Header Preamble

Field 3 is the sum of the ISG after index, the address mark, and PLO Sync fields. It is used at format time.

8.3.3.4 Field 4 - Gap 2

Field 4 is equivalent to Gap 2. It is the time (in bytes) between the end of the Header ECC Pad and the Data Sync byte. This field includes the time required to allow the Phase Lock Oscillator (PLO) in the drive to sync to the data and to skip the write splice.

8.3.3.5 Field 5, Field 5 Alternate - Sector Size

Section 8.3.2 describes sector size.

8.3.3.6 Field 6 - Search Gate Delay - Gap 2

Field 6 asserts Read Gate after the write splice during Gap 2; it is the time allowed for the PLO in the drive to lock up. The 712 searches for the Data Sync byte after reading across Field 6.

8.3.3.7 Field 7 - Write Continuation

The write continuation field is necessary so that when Write Gate is deasserted, the collapsing magnetic field does not splash over the ECC that was just written. The end of Field 7 is the beginning of the minimum Write Gate to Read Gate period described in Section 8.3.3.1. The 712 uses this field after both the Header ECC and Data ECC.

8.3.3.8 Field 12 - Head Switch Time

The 712 requires 6 bytes of time after Field 7 to switch heads. If this 6-byte field is not possible because the sector size is too short, the 712 will miss revolutions on every head switch. This should not be a problem; typically, the minimum Write Gate to Read Gate time allows for this field. For soft-sectored drives, this field must include the speed tolerance gap.

8.3.4 Format Interleave

The 712 can optionally format with an interleave pattern from 2:1 to 16:1. Specify the interleave factor when writing the format parameters; it is invisible to the operating system.

Interleaving can increase the throughput of a disk subsystem on a fully loaded system by effectively cutting the disk speed in half. In a contiguously formatted pack (1:1 interleave), the sectors increase by one each time. As the disk spins, the sectors arrive under the head in the following order for a 32-sector disk:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 ...31

If you interleaved the same disk with a 2:1 interleave factor, it would look like this:

0 16 1 17 2 18 3 19 4 20 5 21 6 22 7 23 8 24 9 25 10 ...31

The 2:1 interleave allows the 712 two sector times to transfer a sector to memory. For example, if you are transferring sectors 0 and 1 to memory, the following occurs in each case.

In 1:1 interleaving, when the first sector completes reading from the disk, the next sector is almost under the head and ready for reading. At this time there must be enough room in the buffer or a data late occurs.

In 2:1 interleaving, when the first sector completes reading from the disk, the next sector is still a full sector time away, thereby giving the 712 twice the time to empty the buffer to memory. The diagram above shows the extra sector time as sector 16.

8.3.4 Format Interleave (continued)

Interleaving schemes from 2:1 to 15:1 are software programmable. This can be a great advantage on a fully loaded system. The current method for handling a data late is to drop a revolution; this means waiting until the next time the sector comes under the head. If you are transferring 16 sectors and drop a revolution on each sector, the transfer takes 16 x 16.6 milliseconds (ms) after the seek or 265.6 ms. If the disk had been interleaved 2:1, it may not have dropped a revolution, and taken only one revolution after the seek, or 17 ms. Interleaving is advantageous, but it doubles the transfer time on a lightly loaded system.

Since the 712 determines which sector the drive is at by comparing headers, you can use the Write Track Headers command to customize the interleaving scheme to your application. For example, if your system transfers data in 4K-blocks, then the most effective interleaving scheme may be 2:4 or: 0 1 2 3 16 17 18 19 4 5 6 7 20 21 22 23 8 9 10 11 ...

8.4 ERROR RECOVERY

The 712 may automatically retry operations that have errored. Two options are available. One option deals with retrying seek-type errors. The other option involves the retry algorithms for ECC recoveries. The Read and Write Controller Parameters commands enable or disable these options.

8.4.1 Automatic Operation Retry

The 712 automatically retries an operation if the reason for the initial failure is a seek error. Setting ASR with a Write Controller Parameters command enables this option.

8.4.2 ECC Error Recovery

ECC algorithms have a much better chance of recovery if the 712 retries the operation before using the ECC correction. The 712 retries the Read operation once before applying the ECC correction if RBC is set. Set RBC with a Write Controller Parameters command.

There are three options for applying ECC correction. Using Mode 0, the 712 provides the correcting pattern and offset for the driver to go and correct the actual error in memory. Mode 1 flags the fact that an error occurred, but does not stop the transfer to calculate any correction information. Mode 2 calculates the correction information and applies it to the data in host memory.

8.4.3 Using the Error Recovery Options

Changing the appropriate controller parameter enables or disables the error recovery options. Once set, the 712 takes care of applying the retries as requested. The 712 provides a completion status to indicate a recovery operation took place. The driver applies the ECC correction if ECC Mode 0 is utilized. If a retry fails, the Completion Code reflects the fatal error.

8.5 READ DEFECT MAP

8.5.1 Manufacturer's Defect Map

The 712 reads the data recorded on the media by the manufacturer that describes the location and length of factory-detected media defects. The defect map's format must conform to the Enhanced Small Device Interface Specification, Revision F, January 31, 1986.

8.5.2 Read the Defect Map

Perform an Extended Read with a subfunction of Read Defect Map. The 712 transfers the header, header CRC, defect map, and data CRC to memory. Allocate 266 bytes to accommodate this information.

8.6 MAINTENANCE MODE

Firmware supports a non-IOPB driven Maintenance mode. It allows you to perform basic testing within the 712 by setting Control bits in the CSR and entering the desired test number and data through the address registers. This firmware also provides a window through which internal registers may be examined or modified.

8.6.1 Register Use in Maintenance Mode

The function code in the Test Number Register determines whether or not the 712 uses the Input Data Byte and Output Data Byte Registers (See Table 8-1). You should be familiar with the Control and Status Register before reading this section (See Section 3.3).

8.6.1 Register Use in Maintenance Mode (continued)

REGISTER	DESCRIPTION
1	Test Number or Function Code
3	Input Address Low
5	Input Address High
7	Input Data Byte (If Required)
9	Output Data Byte (If Required)
B	Control and Status Register
D	Fatal Error Register

TABLE 8-1. REGISTER USE IN MAINTENANCE MODE

8.6.2 Maintenance Mode Protocol

8.6.2.1 Executing a Maintenance Command or Entering the Maintenance Mode

First, set the Maintenance Mode (MM) and AIO bits. This forces entry into the maintenance kernel. The kernel initializes the CSR and Poll mask and sets the Remove IOPB (RIO) bit; then clear RIO.

The kernel expects the Input Address Low Registers to contain a maintenance test number or function code for execution. Data may be expected or may be returned (see register layout).

BUSY and AIO are configured for polling. Setting BUSY selects the register image test; clearing BUSY returns control to the maintenance kernel.

AIO causes the maintenance firmware to read and decode the command string from the Input Address Registers. After successfully decoding the command string, the firmware echoes it (command, address, and data) to the Output Address Registers and clears AIO. This acknowledges receipt of and attempts to execute the requested command. After completing the requested command, the 712 updates the Output Address Registers with test-pertinent data and sets the RIO bit. The AIO/RIO protocol is identical to Normal mode. (RIO indicates the end of firmware involvement and valid contents in the Output Address Registers.)

Since each test and its expected results are different in nature, the Output Address Registers hold the test result information (address, data, etc.). In any case, the firmware sets RIO upon command completion; it sets the Fatal Error bit if a failure occurs or if host software issues an illegal command.

8.6.2.2 Exiting the Maintenance Mode

To exit the Maintenance mode, clear MM and RIO, and set AIO. This returns control to the Normal mode kernel. The 712 acknowledges by setting RIO.

8.6.2.3 Diagnostic Considerations

The Input/Output Address Register Verify is the first test the diagnostic should execute.

Firmware flags the Power-up Test failures by setting the Fatal Error bit while leaving the Maintenance mode bit set. Firmware saves the Self Test error numbers internally until it verifies the Input and Output Registers.

8.6.2.4 Register Tests

You must request entry into the Maintenance mode to invoke the Register test. After the firmware acknowledges the request, you should set the BUSY bit. BUSY remains set during this test.

NOTE

You must enter the Maintenance mode as a separate step because the Normal mode firmware does not allow setting BUSY (defined as RMM when Maintenance mode is enabled).

Writing the Input Address Registers, followed by AIO, signals the firmware to copy the data to the Output Address Registers. Firmware sets RIO when it completes the copy. Host software should then clear RIO.

Clearing the BUSY bit exits this test and returns the 712 to Maintenance mode.

8.6.2.5 Test Variables

Some of the internal tests require the address and data to perform their particular function. On-board memory has space allocated for this data. These locations are loaded with default values for initial use. However, you may alter these variables through the Manual mode. (As the internal tests are defined, the protocol and results expected will be made available.)

8.7 ZERO LATENCY READS

To implement Zero Latency Reads, enable the function when executing a Write Controller Parameters command. Zero Latency Reads cause the 712 to examine the first sector header on the target cylinder. If this sector is within the transfer's boundaries, the 712 starts transferring data to the FIFO. It continues this transfer until it reads the last sector of the transfer on that track. The 712 then searches for the first sector of the transfer on that track and transfers that sector and the remaining sectors (up to the point where it began) into the FIFO. The 712 keeps track of the data, ensuring that it gets transferred to the proper memory location.

If the 712 encounters an error during the initial part of the transfer, it sets the Completion Code to retry successful, marks the next sector as the one it started from, and continues the operation. the 712 automatically retries the errored sector after completing the transfer on this track. If the error recurs on the second try, all previous sectors will have been transferred, and the Completion Code reflects the hard error.

8.8 MULTIPROCESSOR SUPPORT

The 712 has several options that make multiprocessor environments easier to support: the programmable interrupt vector, interrupt level, register address modifiers, and busy semaphore.

8.8.1 Interrupts

Each IOPB specifies the interrupt level and vector for that command. In a multiprocessor environment, each processor can have its own assigned interrupt level and vector.

8.8.2 Register Busy Semaphore

RBS allows multiple processors to share the registers without colliding. Hardware supports the RBS bit. The register access protocol involves reading the CSR. If RBS is clear, the host has control of the register, and retains control until it clears RBS in the Control Register. If the first read to the Status Register indicates that RBS is set, then another host has control of the register and this host must wait until RBS clears.

The 712 sets RBS immediately after a host reads the CSR. If a host attempts a read, and RBS is clear, then the 712 sets RBS; any successive reads by other hosts will "see" that RBS is set. When the host using the registers is done, it must clear RBS. Clearing RBS and setting AIO can occur in the same register write. Clearing RBS without having control of the registers violates the register protocol.

8.8.3 Address Modifiers

The address modifiers can be used to assign separate address space for each of the processors.

8.9 COMMAND OPTIMIZATION

To implement Command Optimization, enable the function when executing a Write Controller Parameters command.

Command Optimization is the reordering of IOPBs in the 712's command queue. The reordering causes the 712 to enable elevator seeks and process several IOPBs within one revolution. The 712 starts the first IOPB it receives, and then reorders IOPBs as they are DMAed into the queue. This feature is most effective when there are more than two IOPBs in the 712's queue.

The 712 places the IOPB in a position relative to the other IOPBs in its internal queue. Then it tests the IOPBs to determine if any are contiguous on the disk. The 712 links together any contiguous IOPBs it finds, and executes them as one disk operation; it does not link the DMA portion of the transfer as this is not necessary.

The 712 first links the IOPBs with respect to an ascending cylinder order. If the 712 receives an IOPB with a cylinder number lower than the current cylinder, it positions the IOPB at the end of the queue in descending cylinder order. Conversely, if the 712 is operating in descending cylinder order, it positions an IOPB with a cylinder number higher than the current cylinder at the end of the queue in ascending cylinder order.

8.10 SOFTWARE CONTROL

The 712 has many parameters that can be modified by software control. The parameters can be set in bulk with three write parameters commands. The Write Format Parameters command modifies the format parameters. The Write Drive Parameters command modifies the drive parameters. The Write Controller Parameters command modifies the controller parameters.

8.10.1 Modifying a Single Parameter

The best method for modifying a single parameter is to first do a read parameters of the associated parameter block, modify the single parameter, and then write the parameter block back to the controller.

8.10.2 Modifying a Group of Parameters

Use the same method as in Section 8.10.1, or set all the parameters in the specific IOPB and execute the appropriate write parameters command. For example, Fields 1 through 7, 5A, and the interleave factor must be set to the appropriate values before issuing the Write Format Parameters command. The 712 sets all parameters to the new values contained in the IOPB.

8.10.3 Parameter Reference Point

After the 712 is working as intended, read the parameters and save the information for future use.

8.10.4 Setting Parameters at Boot Time

It is not necessary to reload the parameters at each boot since the parameters are stored in a battery backed-up RAM. It is a good idea to reload them each time if you are not sure how the board was last used.

8.10.5 Validate Current Parameters

The parameters are all protected by a checksum, and any read parameters command performs a checksum test. Any read parameters terminates with an error if the generated parameter checksum is different than the stored checksum (See Section 6.6).

8.11 SCATTER/GATHER

The Scatter/Gather feature is used in conjunction with standard Read and Write commands. In a Scatter Read, the 712 transfers the data to up to 32 blocks of memory. Gather Writes gathers data from up to 32 blocks of memory and writes it to the disk. The size of each memory block must be an even byte count and less than 64K-bytes long. The blocks may be scattered throughout memory.

8.11.1 Scatter/Gather Link List

You can determine the length of the linked list by multiplying the number of elements in the list by eight (each element is eight-bytes long). All data addresses must be on word boundaries and the byte count must be even. For Read and Write operations, enter the number of elements in the linked list into IOPB Byte 6, bits 3 through 7. A zero in this field indicates the linked list has 32 elements. See Tables 8-2 and 8-3.

8.11.1 Scatter/Gather Link List (continued)

LINK NUMBER	BYTE	DESCRIPTION
1	00-01	Byte Count (Multiples of 2)
	02	Reserved
	03	Data Address Modifier
	04-07	Data Address (Word Boundaries Only)
2	08-09	Byte Count
	:	
	:	
n	xx	

TABLE 8-2. SCATTER/GATHER LINK LIST

LINK FIELD VALUE	DECIMAL EQUIVALENT
0	32
1	1
2	2
:	:
9	9
A	10
B	11
:	:
1E	30
1F	31

TABLE 8-3. LINK LIST FIELD VALUES

8.11.2 Setting Up a Scatter/Gather Transfer

The Data Address and Modifier bytes in the IOPB should now point to the start of the linked list. The linked list length field should give the total number of element descriptors on the list.

Elements of memory descriptors comprise the linked list. Each element describes the starting address and the length in bytes of the memory block. The sum of the byte count of all the elements in the linked list must equal the sector count times the sector size in bytes.

8.11.2 Setting Up a Scatter/Gather Transfer (continued)

The IOPB and Linked List in Figure 8-7 illustrate a Read transfer to 6 blocks of memory. The sector size in this case is 528-bytes per sector; we are transferring 3 sectors of information. The 712 transfers the first 16 bytes of data from each sector to a separate data buffer. It scatters the bulk of the data, 512-bytes per sector, into memory as 3 blocks having 512 bytes each.

Set SGM and execute this IOPB.

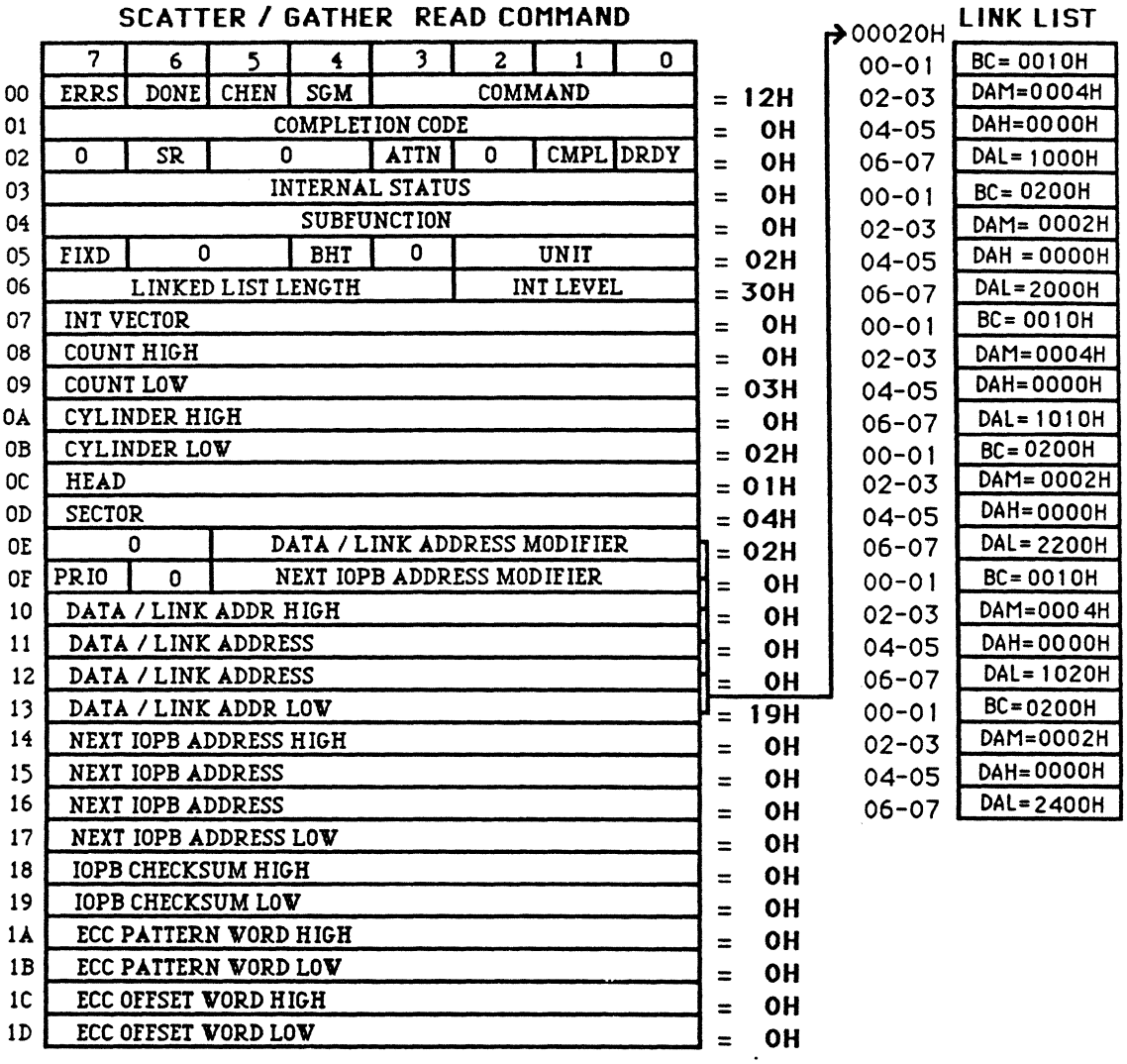


FIGURE 8-7. SCATTER/GATHER TRANSFERS

8.11.3 712 Operation

The 712 proceeds as if doing a normal read until it starts the data transfer into memory. The contents of the linked list now controls the DMA processor; it gives the processor the byte count and address for each element on the list. The processor takes the data out of the FIFO and transfers it to memory as decribed in each element on the list.

8.11.4 Zero Latency Reads and Scatter/Gather

Due to the flexibility allowed in scatter/gather, the linked list elements may not align with sector boundaries. Thus, zero latency reads may not function properly. Xylogics does not recommend mixing these two options.

8.12 DMA THROTTLE / THROTTLE DEAD TIME

The 712 always transfers IOPBs in Word mode; it uses the last specified values for the throttle and throttle dead time.

Host software can set the Throttle Dead Time (TDT) field in the Controller Parameters IOPB. This value defines the time that the 712 waits before attempting to regain control of the bus between throttle bursts. There are four valid TDT values.

TDT VALUE	TIME
0	0 microseconds
1	3.2 "
2	6.4 "
3	12.8 "

TABLE 8-4. THROTTLE DEAD TIME VALUES

8.13 BLACK HOLE TRANSFERS

Sometimes the data to be transferred has to go to a single memory location. This single location is usually a graphics controller with a single port on the bus. The normal DMA mode increments the bus address on each transfer so the data is put into contiguous memory space. When Black Hole Transfers are implemented, the 712 does not increment the bus address between each data transfer.

8.13 BLACK HOLE TRANSFERS (continued)

Any transfer that includes a DMA to a single location should have BHT set in Byte 5 of the IOPB. This causes only the data transfer portion of the command to not have its bus address incremented. The IOPB DMA still occurs in Normal mode (i.e., the 712 increments the address).

The data address must be properly aligned: word aligned for word transfers, and longword aligned for longword transfers. The 712 cannot do dynamic mode switching with this option.

8.14 PRIORITY IOPBs

The 712 processes Priority IOPBs in advance of any other IOPBs in its queue. This feature works on both a single IOPB and an IOPB chain.

8.14.1 Executing a Priority IOPB

To execute a priority IOPB, set PRIO in both the IOPB Address Modifier Register and the Next IOPB Address Modifier byte in the IOPB. Set the rest of the IOPB Address Registers to point to the IOPB (do not reset PRIO in the Address Modifier Register when loading it). Set AIO as you normally would.

8.14.2 Executing a Priority Chain

To execute a Priority chain, follow the directions in 8.14.1. All IOPBs in the chain must have PRIO set in the Next IOPB Address Modifier byte.

8.14.3 712 Response to a Priority IOPB (Chain)

The 712 finishes executing the IOPB that is currently active (if any). The next IOPB to execute is the priority IOPB (or the first in the priority chain). If the 712 starts a chain of priority IOPBs, it completes one at a time until it completes the chain, and then goes back to processing the IOPBs in its queue. All IOPBs in a priority chain must have PRIO set.

8.15 IOPB CHECKSUM

While debugging the driver, you may choose to append the checksum to the IOPB. The checksum is the sum of Bytes 0 through 17 in the IOPB, and is expressed as a 16-bit quantity. The 712 generates a checksum with the data from the IOPB and compares it to the appended checksum; a miscompare causes a fatal error. If AUD and ICS are set, the 712 appends a new checksum as it updates the IOPB. If you want to disable the checksum, the Write Controller Parameters IOPB must have a valid checksum.

8.16 FIXED/REMOVABLE MEDIA

Any physical drive with fixed and removable media is accessed as one logical unit (FIXD is clear for the removable media of Unit 0, and FIXD is set for the fixed media of Unit 0).

8.16.1 Head Offset

The head offset refers to the bit(s) that must be set during a drive head select sequence to select between the fixed and removable portions of the drive. Host software must specify a head offset value for fixed/removable drives: one for the removable portion of the drive, and one for the fixed portion. The offset value is a hexadecimal number that the 712 adds to the head number in order to select either the fixed or removable portion of the disk. Reference the appropriate vendor manual to determine the head offset values for the fixed and removable portions of the disk.

8.17 EMBEDDED SERVO DRIVES

The 712 waits for command complete after every head change (required by embedded servo disk drives). The drive requires this to lock onto the new track.

8.18 SUPPORTING FOUR DRIVES WITH DIFFERENT SECTOR SIZES

The 712 can support four drives having different sector sizes. For example, Drive 0 can have 512-bytes per sector, Drive 1 can have 1024-bytes per sector, etc.

8.18.1 Setting the 712 Format Parameters

When executing the format parameters command, field 5 should be set to the standard sector size (200H [512]) and Field 5A should be set to the alternate sector size (400H [1024]).

8.18.2 Setting the 712 Drive Parameters

The Write Drive Parameters IOPB has an Alternate Field Enable (AFE) bit. If AFE is set, the drive specified by this command uses the alternate field size (1024). If AFE is clear, then the drive uses the standard sector size. For example, Unit 0 has 512-bytes per sector and AFE is clear in the Write Drive Parameters command; Unit 1 has 1024-bytes per sector and AFE is set in the Write Drive Parameters command.

8.18.3 Accessing Drives With Alternate Fields

The 712 automatically sets the sector size to the specified drive. When allocating buffers, host software must remember, and take into account, the sector size of the drive being accessed.

8.19 READ/WRITE HEADER, HEADER VERIFY, DATA, AND DATA ECC

This maintenance command is used to test the controller and software driver. It enables simulating ECC errors to verify the ECC is working. The operation includes reading a sector with its header into memory, modifying the data and then writing the sector back to the disk. The 712 does not recalculate the ECC for this command.

The data read back is either 12 or 14 bytes larger than the data sector size (depending on EC32's status). For example, given a 512-byte sector size, the 712 returns 524 bytes if EC32 is set, and 526 bytes if EC32 is clear. Table 8-5 shows the relationship between EC32's status and the returned data.

BYTE	EC32 CLEAR	BYTE	EC32 SET
0-3	Header*	0-3	Header*
4-7	Redundant Header*	4-7	Header ECC
8-(n+8)	Data	8-(n+8)	Data
(n+8)-(n+14)	48-bit ECC	(n+8)-(n+12)	32-bit ECC

* Figures 8-2 through 8-6 describe the header information

TABLE 8-5. EC32 VS. RETURNED DATA

8.19.1 Simulating an ECC Error

To simulate an ECC error, read a sector by issuing a Read Header, Header Verify, Data, and Data ECC command. Then change a data byte or bit, and write the sector back by issuing a Write Header, Header Verify, Data, and Data ECC command. Reading this sector with a normal Read command should return an ECC error.

There are two common problems associated with simulating an ECC error. First, the corrected data byte may be next to the one in error. Second, the data may not be serially written to the disk as you see it on your terminal screen. Thus, a "2-bit error" crossing a byte boundary may be uncorrectable.

The 712 usually accesses memory in Word or Longword mode, but corrects data in ECC Mode 2 via byte transfers. Since some bus adapters reverse the byte addressing scheme within a word, the 712 corrects the wrong data. The only solution for this situation is to either correct the adapter or use ECC Mode 0.

The serial data is placed on the disk with bit 0 of each byte first. Table 8-6 shows a simulated 2-bit error crossing byte boundaries. Since the two bits in error are really fifteen bits apart, they may be uncorrectable. This can only occur when testing because a 2-bit adjacent error refers to two adjacent bits on the media.

BEFORE SIMULATED ERROR

MEMORY DATA:	45	67		
SERIAL DISK DATA:	5	4	7	6
	1010	0010	1110	0110

SIMULATED ERROR

MEMORY DATA:	44	E7		
SERIAL DISK DATA:	4	4	7	E
	0010	0010	1110	0111

TABLE 8-6. SIMULATED 2-BIT ERROR CROSSING BYTE BOUNDARIES

8.20 INTERRUPT AT END OF CHAIN

IEC prevents the 712 from interrupting after completing each IOPB in a chain. The 712 executes the entire chain and then interrupts (using the interrupt level and vector from the first IOPB in the chain). When IEC is clear, the 712 interrupts after completing each IOPB (providing the interrupt level is not zero).

8.21 RELEASE ON REQUEST

When ROR is enabled, the 712 tests the VMEbus between each throttle for other pending bus requests. If another request is pending, the 712 releases the bus. If there are no bus requests, the 712 remains bus master. The throttle value determines how often the 712 tests the bus. Using lower throttle values causes the DMA to slow down; using higher throttle values causes the 712 to test the bus less frequently.

SECTION 9: THEORY OF OPERATION

9.0 GENERAL

This section is an overview of how the controller works. It deals with the functional blocks of the hardware and microcode, and how the code affects 712 operation.

9.1 The Hardware

The 712 interfaces the VMEbus to up to four ESDI disks and includes these logic blocks:

VMEbus Interface
Register Read/Write and Interrupt (REGCEL)
Microcontroller
Direct Memory Access Controller (VMEDMA)
Disk Data Buffer (FIFO)
Disk Front End (DSKCEL)
ESDI Disk Interface

9.1.1 VMEbus Interface

This block contains interface logic for the signals on the VMEbus. The 712 is a VME slave for programming purposes, i.e., the register file. The 712 is also a slave when it responds to an interrupt acknowlege with the interrupt vector. The REGCEL performs both of these functions. The address modifier transceiver is used by the REGCEL (receive mode) and VMEDMA (driving mode). The 712 is a VME master for DMA purposes; it uses DMA to read and update IOPBs and also to read and write disk data from host memory. The VMEDMA chip performs this function. The VME data bus is 8-, 16-, or 32-bits wide; the VME address bus is 32-bits wide.

NOTE

The 712 uses only 16 address bits for decoding its register addresses (it only compares 12 bits). It drives all 32 address bits as a master. The VMEDMA updates the lower 16 address bits while the microcontroller updates the upper 16 address bits.

9.1.2 Register Read/Write and Interrupt

The REGCEL provides the program interface for the 712. The 712 uses registers to point to an IOPB to be executed, to point to a completed IOPB, and to perform various control functions.

The VMEbus accesses the registers via a bus that is shared between the REGCEL and VMEDMA. There is no contention because the VMEDMA cannot acquire the bus while a slave (register) access is in progress. The REGCEL answers a register access by the VMEbus with the signal DTACK. The upper address bits are decoded and the proper address modifier is required to do a register access.

The microcontroller (micro) accesses the registers via the internal data bus. It programs the REGCEL to interrupt it if certain bits are set in the CSR or if a timer overflows. Other conditions are programmed to assert another interrupt to the micro.

The REGCEL also supports the VMEbus interrupt protocol. It is programmed by the micro to assert the request line to start an interrupt sequence. When the system responds with IACK and DS0 the REGCEL drives the interrupt vector onto the bus and asserts DTACK. This process causes the system to execute an interrupt routine. The interrupt service routine should clear RIO in the CSR. In this way, the 712 passes the completed IOPB back to the system.

The REGCEL has a register for storing the address modifier that the VMEDMA enables onto the bus. Like the VME address, the address modifier is pipelined. A new modifier can be loaded into the REGCEL while a different modifier is being driven onto the VMEbus.

The REGCEL also has two timers. Timer 0 is a watchdog timer: it expires and interrupts the microcontroller if it is not periodically reset by the firmware. Timer 1 is a counter for header errors. A clock comes from the DSKCEL and is asserted when a disk header does not match the expected header. If Timer 1 overflows before the desired header is found, a Header Not Found error results.

9.1.3 The Microcontroller

The 712 uses a 16-MHz 8031 microcontroller. It fetches instructions by asserting an address on Port 0, latching the address with ALE-L, and reading the data from the EPROM; it reads data into Port 0. Many of the instructions cause the micro to access an external byte using strobe decoders and a transceiver. Many of these external bytes are in the Xylogics LSI chips: REGCEL, VMEDMA, and DSKCEL. External bytes are accessed through Port 0. Port 1 is used for DSKCEL related outputs. Port 2 is used for the upper byte of the EPROM address.

9.1.3 The Microcontroller (continued)

Port 3 is used for miscellaneous control signals. Inputs can all be considered micro interrupts, although most are actually polled. Two inputs are from the REGCEL, one from the VMEDMA, and two from the DSKCEL.

This block also includes the internal RAM (IRAM) logic. The micro really uses the IRAM as a scratch pad RAM; it stores the controller, drive, and format parameters as well as IOPBs. Since the RAM device is non-volatile, the 712 remembers its configuration. However, a board that has never been programmed has garbage for parameters. If in doubt, reprogram the board with the parameters for the drive you are using before using the 712.

The IRAM is single ported, but is used by both the VMEDMA and the micro. Since the microcontroller starts the VMEDMA, it knows when the IRAM is off limits for micro access. IOPBs are always DMAed in full words for better performance.

9.1.4 Direct Memory Access Controller

The VMEDMA controls the transfer of data between the disk buffer or IRAM, and the VMEbus. The micro programs the VMEDMA to transfer a certain amount of data to/from a specified area in system memory. To properly handle odd starting addresses, the amount of data is always a sector's worth or less. In this way, the DSKCEL can be instructed to do an even or odd address transfer on a per sector basis. Before the VMEDMA can transfer data, it must acquire the bus by sending out BUSREQ (which must be jumpered to one of the Bus Request lines). The system arbiter sends back BGIN via another jumper. The VMEDMA then asserts BUSY on the VMEbus and enables the 712 to control the VME address, data, and control lines.

A transfer involves asserting a valid address, asserting DS0 and/or DS1 and valid write data or read data, waiting for DTACK, and proper buffer control. The order of the buffer request and data strobe is reversed, depending on the direction of the transfer.

The DMA circuitry pipelines data to increase performance. The pipeline allows one word of data to be transferred on the bus while another is transferred to the buffer. In this way, the access times of the buffer and the bus can be overlapped (except for the first and last transfers of a burst). A prefetch primes the pipeline, and at the end of the burst the pipeline is emptied.

The disk buffer is word wide and uses a longword wide pipeline. The interface logic turns a VMEDMA longword request into two buffer (word) requests. The IRAM is byte wide and uses a word wide pipeline. The interface logic turns the VMEDMA word request into two IRAM requests.

9.1.5 Disk Data Buffer

The 8K-byte buffer for disk data has byte parity. The parity bits are stored in a RAM. The buffer is organized word wide. DMA longwords must be written using two buffer requests. If an odd byte has been DMAed, the DSKCEL can be programmed to ignore the dummy byte in the other half of the buffer word. The buffer control logic is an 8x60 and accepts requests to read and write the buffer from the VMEDMA and DSKCEL. One device at a time is allowed to access the buffer and the micro tracks the buffer's full/empty status. The VMEDMA waits for the buffer to become ready before initiating a transfer. The DSKCEL will not start a write unless a sector's worth of data is in the buffer. The DSKCEL will not start a read unless the buffer has room for a sector's worth of data. The micro tracks buffer use and starts/stops the VMEDMA and DSKCEL as necessary.

9.1.6 Disk Front End

The DSKCEL is a downloadable disk sequencer. The micro loads the disk read, write, and format programs into the DSKCEL on power-up, and modifies the programs when new format parameters are loaded or an alternate command (such as Read Header, Data, ECC) is received. The DSKCEL issues some ESDI control signals, such as Read Gate and Write Gate. Generally, if timing is critical, the DSKCEL issues the signal since it runs off the disk bit clock. The DSKCEL has serial registers for FIFO data, the Header, and ECC. It performs sync bit search, header check, and ECC check and provides status bits to the micro. The DSKCEL interrupts the micro when done. DSKCEL Done may mean Header Found, End of Sector, or Bad Spot Found. Generally, the DSKCEL runs on a sector-by-sector basis, with the micro controlling how many sectors are transferred. The micro allows the DSKCEL to run. The program starts running when a sector or index pulse comes in from the disk. The micro informs the DSKCEL when the next sector involves an odd address DMA and when the current sector is the last.

9.1.7 Enhanced Small Device Interface

This block contains interface logic for ESDI connectors. The micro controls the Head, Control (read/write), and Unit Select lines. The micro and the disk sequencer receive and use various disk status lines, and the sector and index pulses. The micro controls whether the pulses to the DSKCEL include sector pulses (hard sector), sector by address mark found (soft sector) or just the index. You can configure a drive with any Unit Number, from 1 through 7, and connect it to any of the four ports.

SECTION 10: MAINTENANCE AIDS

10.0 GENERAL

This section provides useful information for installing and maintaining your Xylogics Model 712 Disk Controller.

10.1 VMEbus INTERFACE SIGNALS

MNEMONIC	CONN.	PIN	USED BY 712	DESCRIPTION
A01	P1A	30	Y	
A02	P1A	29	Y	
A03	P1A	28	Y	
A04	P1A	27	Y	
A05	P1A	26	Y	
A06	P1A	25	Y	
A07	P1A	24	Y	
A08	P1C	30	Y	
A09	P1C	29	Y	
A10	P1C	28	Y	
A11	P1C	27	Y	
A12	P1C	26	Y	
A13	P1C	25	Y	
A14	P1C	24	Y	
A15	P1C	23	Y	Address Bus
A16	P1C	22	Y	
A17	P1C	21	Y	
A18	P1C	20	Y	
A19	P1C	19	Y	
A20	P1C	18	Y	
A21	P1C	17	Y	
A22	P1C	16	Y	
A23	P1C	15	Y	
A24	P2B	4	Y	
A25	P2B	5	Y	
A26	P2B	6	Y	
A27	P2B	7	Y	
A28	P2B	8	Y	
A29	P2B	9	Y	
A30	P2B	10	Y	
A31	P2B	11	Y	
AM0	P1B	16	Y	
AM1	P1B	17	Y	
AM2	P1B	18	Y	Address Modifier
AM3	P1B	19	Y	
AM4	P1A	23	Y	
AM5	P1C	14	Y	

10.1 VMEbus INTERFACE SIGNALS (continued)

MNEMONIC	CONN.	PIN	USED BY 712	DESCRIPTION
D00	P1A	1	Y	
D01	P1A	2	Y	
D02	P1A	3	Y	
D03	P1A	4	Y	
D04	P1A	5	Y	
D05	P1A	6	Y	
D06	P1A	7	Y	
D07	P1A	8	Y	
D08	P1C	1	Y	
D09	P1C	2	Y	
D10	P1C	3	Y	
D11	P1C	4	Y	
D12	P1C	5	Y	
D13	P1C	6	Y	
D14	P1C	7	Y	
D15	P1C	8	Y	Data Bus
D16	P2B	14	Y	
D17	P2B	15	Y	
D18	P2B	16	Y	
D19	P2B	17	Y	
D20	P2B	18	Y	
D21	P2B	19	Y	
D22	P2B	20	Y	
D23	P2B	21	Y	
D24	P2B	23	Y	
D25	P2B	24	Y	
D26	P2B	25	Y	
D27	P2B	26	Y	
D28	P2B	27	Y	
D29	P2B	28	Y	
D30	P2B	29	Y	
D31	P2B	30	Y	
STROBES				
AS*	P1A	18	Y	Address Strobe
DS0*	P1A	13	Y	Data Strobe Zero
DS1*	P1A	12	Y	Data Strobe One
DTACK*	P1A	16	Y	Data Transfer Acknowledge
CLOCKS				
SERCLK	P1B	21	N	Serial Clock
SYSCLK	P1A	10	N	System Clock

9.3.6 Extended Read and Write Commands

This section is similar to Section 9.3.2; the following subsections detail their differences.

9.3.6.1 Track Headers Commands

The disk sequencer waits for index before determining where to start the transfer. Track headers commands always start at index. The number of sector headers returned equals the pysical sector count (read with a Read Drive Parameters command).

9.3.6.2 Header, Header Verify, Data, and Data ECC Commands

The disk sequencer waits for index before determining where to start the transfer. Header, Header Verify, Data, and Data ECC commands use the sector address as an offset count from index to determine where to start. The 712 increments the sector address during a multisector transfer, but does not not clear it if it reaches maximum sector. Illegal Sector Address error is inhibited for this command. The 712 does not increment the head and cylinder address.

9.3.6.3 Read Verify

The disk sequencer does a Read command, and the DMA sequencer performs as if it were doing a disk write. The 712 compares the serial data from the FIFO/SERDES with the data from the disk. A miscompare causes a verify failure. The 712 returns the failing disk address in the IOPB.

9.3.7 Diagnostics

The 712 executes the power-up self test.

9.4 COMPLETING A FUNCTION

The 712 completes the transfer when both the DMA and disk transfers are complete. The 712 updates the IOPB in host memory, interrupts, and clears BUSY if all IOPBs in its queue are complete.

If an error occurs, the 712 completes the errored IOPB and continues processing the other IOPBs.

If a fatal error occurs, the 712 finishes the IOPB(s) in process, sets the appropriate fatal error code, and sets FERR. The host must execute a Controller Reset before sending any IOPBs to the 712.

9.2.4 Is Any IOPB Ready for Completion?

The 712 checks for an IOPB in the third group, seek complete or not required. It does this by first checking if an IOPB has a seek done set; if not, it selects each drive in the second group, seek started, to determine if a seek is complete. The 712 executes the first seek done IOPB. When the 712 completes the function, it updates the IOPB and issues an interrupt. See Figure 9-1.

9.2.5 Queuing IOPBs for Execution

The 712 command queue accommodates 14 IOPBs. The first AIOs have their respective IOPBs (IOPB chains) read directly into the queue, until the queue is full. If COP is set, the 712 reorders the IOPBs inside the queue, so that it executes them in proper order.

As the 712 completes an IOPB, it frees up a slot in the internal queue, and reads in a new IOPB. If COP is set, the 712 inserts the new IOPB at the proper point in the reordered IOPBs.

9.3 PERFORMING A FUNCTION

The 712 performs each function differently. If a function requires a seek, the 712 issues the seek and waits for it to complete before performing the function. The following subsections group similar functions together and explains their differences.

9.3.1 NOP

The function of a NOP is to do no operation, so the 712 goes on to complete the function.

9.3.2 Normal Reads and Writes

Normal reads and writes are very similar commands regarding 712 processing. The 712 is optimized to perform these functions as fast as possible. The main difference between reads and writes is which way the data moves and when. On writes, the data DMA begins and the 712 begins the disk transfer when one sector's worth of data is in the FIFO. On reads, the disk transfer begins immediately and the DMA begins as soon as the first word is available from the FIFO.

9.3.2 Normal Reads and Writes (continued)

When enabled, the disk sequencer starts to compare every header that arrives under the heads with the target sector. When a header compare is successful, the 712 also tests the header verify. If both tests are successful, the transfer occurs on that sector. To continue the transfer, the 712 loads the next target header so it can do a comparison on the next sector that arrives under the heads. The 712 does not wait for index before comparing headers.

If more than one sector is specified, the 712 increments the disk address on successive sectors. First, the controller increments the sector number until it reaches the maximum sector address. When the 712 reaches the maximum sector address, it clears the sector address and increments the head address. When the 712 reaches the maximum head and sector addresses, it clears them and increments the cylinder address. When the 712 reaches the maximum sector, head, and clinder addresses, the next sector causes an Illegal Cylinder Address error.

The 712 continues, and completes the IOPB as soon as it completes the DMA and disk transfers, or an uncorrectable error occurs.

9.3.3 Seeks

Depending on the subfunction, the 712 may select the drive and read the first header that arrives under the heads. If it is a spare or bad header, the 712 reads the next header until it gets a good one. The 712 returns the data read in the header in the IOPB.

9.3.4 Drive Reset

This command issues a Fault Clear and then a Recalibrate (Return To Zero) command to the drive. The 712 waits for the Recalibrate to complete before completing the IOPB. The wait for recalibrate done is a background task, and the contoller continues processing other IOPBs (not on the drive being recalibrated).

9.3.5 Write and Read Parameters

Section 7 describes these functions in detail.

9.1.8 Power-up

During power-up the bus signal SYSRESET sets the error LED (L2) and asserts SYSFAIL on the bus. The micro runs its diagnostics and then clears L2 and SYSFAIL.

9.1.9 Power-down

During power-down, the 712 responds to ACFAIL by turning off DSKCEL (it allows any writes to finish the current sector). This sequence generates a fatal error which cannot be reset until ACFAIL is deasserted.

9.1.10 System Reset

When the 712 detects SYSRESET, it resets its internal micro and the three custom integrated circuits. This sequence immediately terminates any writes to the disk (possibly leaving it with an unreadable sector). The 712 then executes the power-up diagnostics.

9.2 THE MICROCODE

9.2.1 The Kernel

Figure 9-1 illustrates the kernel. It is entered after the power-up self test and initialization. It has four major functions, of which three will be discussed. The fourth function is scheduling DMA, and would needlessly complicate this discussion.

9.2.2 Is AIO Set?

Each time around the kernel, the 712 tests to see if the host set AIO. If AIO is set, the 712 must process the AIO. This processing entails reading the IOPB and placing it in its internal command queue. If the queue is full, the 712 saves the address of the IOPB so that it can later read it into the queue. The queue can accommodate 14 IOPBs, plus a priority IOPB; the 712 saves the next 32 IOPB (chain) addresses. This function is really tested several times in the kernel, but for simplicity Figure 9-1 shows it as a single function.

9.2.3 Is Start Queue Empty?

The IOPBs in the internal queue are divided into three groups: seek not started queue, seek started, and seek complete or not required. When first entering the queue, the IOPBs belong to the first group, seek not started queue. This step examines the IOPBs to determine if a seek is required. If a seek is required, and the drive is not busy, the 712 issues the Seek command to the drive and assigns the IOPB to the second queue, seek started. If a seek is not required, the 712 marks the IOPB as having its seek complete and assigns it to the third qroup, seek done or not required.

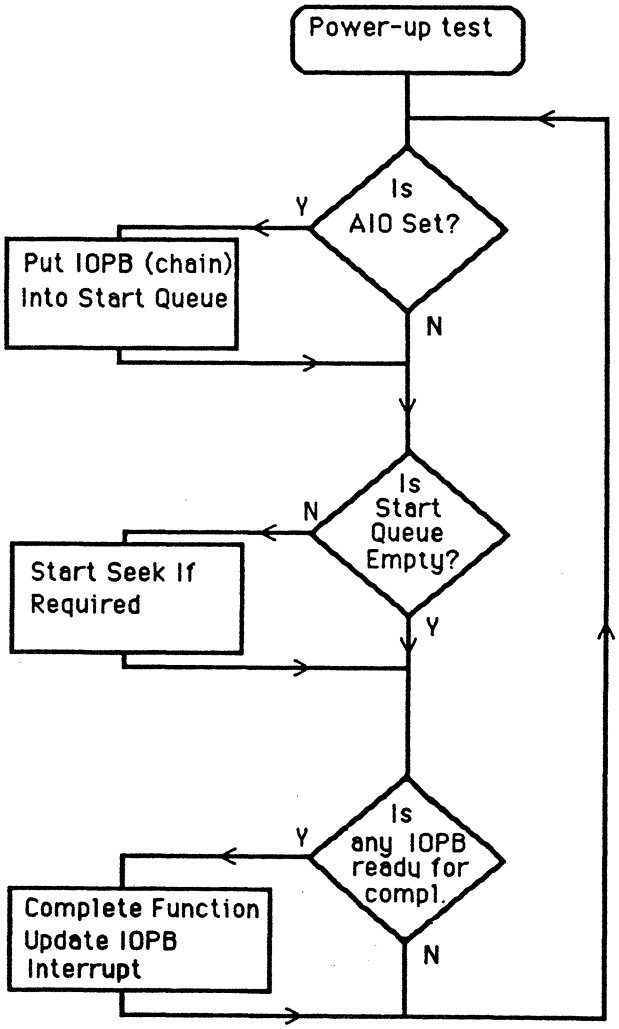


FIGURE 9-1. THE MICROCODE KERNEL

10.1 VMEbus INTERFACE SIGNALS (continued)

MNEMONIC	CONN.	PIN	USED BY 712	DESCRIPTION
DMA				
BBSY*	P1B	1	Y	Bus Busy
BCLR*	P1B	2	N	Bus Clear
BERR*	P1C	11	Y	Bus Error
BG0IN*	P1B	4	Y	
BG1IN*	P1B	6	Y	Bus Grant In
BG2IN*	P1B	8	Y	
BG3IN*	P1B	10	Y	
BG0OUT*	P1B	5	Y	
BG1OUT*	P1B	7	Y	Bus Grant Out
BG2OUT*	P1B	9	Y	
BG3OUT*	P1B	11	Y	
BR0*	P1B	12	Y	
BR1*	P1B	13	Y	Bus Request
BR2*	P1B	14	Y	
BR3*	P1B	15	Y	
INTERRUPTS				
IRQ1*	P1B	30	Y	
IRQ2*	P1B	29	Y	
IRQ3*	P1B	28	Y	Interrupt Request Levels
IRQ4*	P1B	27	Y	
IRQ5*	P1B	26	Y	
IRQ6*	P1B	25	Y	
IRQ7*	P1B	24	Y	
IACK*	P1A	20	Y	Interrupt Acknowledge
IACKIN*	P1A	21	Y	Interrupt Acknowledge In
IACKOUT*	P1A	22	Y	Interrupt Acknowledge Out
MISCELLANEOUS				
ACFAIL*	P1B	3	Y	AC Failure
LWORD*	P1C	13	Y	Longword
RESERVED	P2B	3	N	Reserved
SERDAT*	P1B	22	N	Serial Data
SYSRESET*	P1C	12	Y	System Reset
WRITE*	P1A	14	Y	Write

10.1 VMEbus INTERFACE SIGNALS (continued)

MNEMONIC	CONN.	PIN	USED BY 712	DESCRIPTION
POWER				
+5V	P1A,P1B,P1C	32	Y	+5 VDC
+5V	P2B	1,13,32	Y	+5 VDC
+5V STDBY	P1B	31	N	+5 VDC Standby
+12V	P1C	31	N	+12 VDC
-12V	P1A	31	N	-12 VDC
GND	P1A	9,11,15,17,19	Y	Signal Ground
GND	P1B	20,23	Y	Signal Ground
GND	P2B	2,12,22,31	Y	Signal Ground
GND	P1C	9	Y	Signal Ground

10.2 ENHANCED SMALL DEVICE INTERFACE

The ESDI interface is compatible with the standard ESDI Interface Specification.

NOTE

Xylogics follows the industry standard for pin-numbering. (A Cable=34-pin control cable; B Cable=20-pin data cable.)

NAME	CABLE	PIN	GND	DESCRIPTION
Head Sel. Bit 0	A	14	13	These binary weighted
Head Sel. Bit 1	A	18	17	signals determine which
Head Sel. Bit 2	A	4	3	head the 712 selects.
Head Sel. Bit 3	A	2	1	
Drive Sel. 1	A	26	25	These binary weighted
Drive Sel. 2	A	28	27	signals determine which
Drive sel. 3	A	30	29	drive the 712 selects.
Write Gate	A	6	5	The 712 asserts Write Gate when writing to the drive.
Read Gate	A	32	31	The 712 asserts Read Gate when reading from the drive.
Index	A	20	19	The 712 detects index from the drive at the beginning of each track.

10.2 ENHANCED SMALL DEVICE INTERFACE (continued)

NAME	CABLE	PIN	GND	DESCRIPTION
Sector/Address Mark Found	A	16	15	The 712 detects sector pulse at the beginning of each sector (in Hard Sector mode). In Soft Sector mode, the 712 detects Address Mark Found at the beginning of each sector.
Ready	A	22	21	The 712 detects Ready when the drive spindle is up to speed. When true, Ready and Command Complete indicate the drive is ready to read, write or seek.
Attention	A	12	11	The 712 detects Attention when the drive has a fault condition.
Transfer Acknowledge	A	10	9	This signal functions as a handshake along with Transfer Request during a command sequence.
Transfer Request	A	24	23	The 712 issues this signal as a handshake along with Transfer Acknowledge.
Command Data	A	34	33	When the 712 issues a command, 16 bits of serial data, plus 1 bit of parity are presented on this line.
Configuration/ Status	A	8	7	The ESDI drive sends its Configuration/Status Data serially across this line.
Drive Selected	B	1		The drive asserts this signal after the 712 selects the appropriate drive.
Reserved	B	2		
Command Complete	B	3		When true, the drive is ready to accept commands.
Address Mark Enable	B	4		The 712 uses this line to write the address mark when using soft sector.

NAME	CABLE	PIN	GND	DESCRIPTION
Gnd	B	5		Ground.
Gnd	B	6		Ground.
Write Clk+	B	7		The 712 supplies Write Clock
Write Clk-	B	8		to the drive.
Gnd	B	9		Ground.
Read/Ref Clk+	B	10		Read Clock from the drive.
Read/Ref Clk-	B	11		
Gnd	B	12		Ground.
NRZ Write Data+	B	13		The 712 uses these lines to
NRZ Write Data-	B	14		send write data to the drive.
Gnd	B	15		Ground.
Gnd	B	16		Ground.
NRZ Read Data+	B	17		The 712 reads the data from
NRZ Read Data-	B	18		the drive.
Gnd	B	19		Ground.
Reserved	B	20		

INDEX

PAGE

INDEX

PAGE

INDEX

PAGE

INDEX

PAGE

INDEX

PAGE

PAGE

144 MIDDLESEX TURNPIKE
BURLINGTON, MA 01803