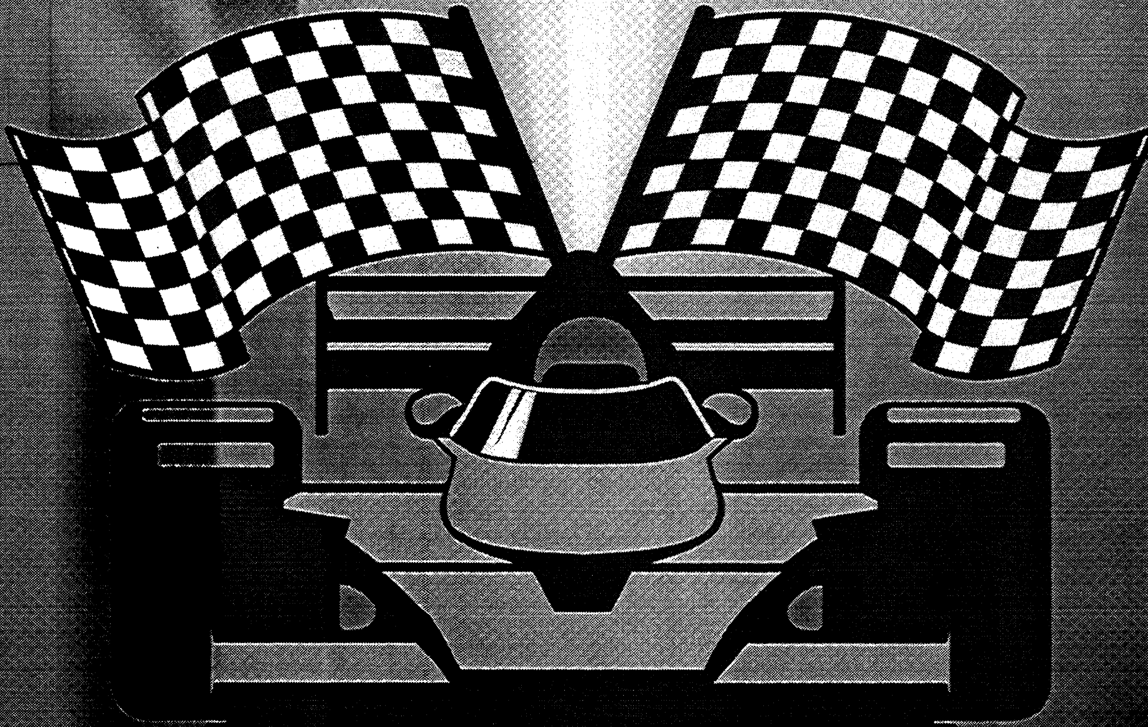
7/21/94



DAYTONA

Firmware Manual

Masspro Version

Table of Contents

General Information

Revision History

Rev	Date	Description	Author
P1	10/12/92	Preliminary Release	Mark Sompel
MP	1/10/93	Masspro Release	Mark Sompel
1	3/29/94	Update the manual	Mark Sompel
2	7/21/94	Update	Mark Sompel

Scope

The purpose of this manual is to document the Daytona Series Firmware. This manual documents deviations from the SCSI and AT specifications and describes the superset (diagnostic) commands. In addition to documenting the external interface, certain internal features of the firmware and its architecture are described.

Applicable Documents

SCSI-II Specification
CAM ATA Specification
Konishiki Specification.
Apple Diagnostic Specification.

Note: The latest revisions of these manuals should be used.

This manual was designed by Mark Sompel using Corel Ventura version 4.1.

cim

Superset Commands

Introduction

The superset or diagnostic commands are accessible as opcode FFH along with a one-byte subcode specifying the particular diagnostic function.

Note: This manual only describes the commands implemented for the Daytona product line.

The actual superset commands are the same for both SCSI and AT. The method of enabling them and sending the bytes over the interfaces differs between them. The following sections will detail those differences.

SCSI Superset Commands

The diagnostic mode must be enabled prior to executing any diagnostic command with one exception:

Write Configuration to page 6 (customer name) is allowed at all times.

Executing a diagnostic command in normal mode will result in an ILLEGAL COMMAND condition. The SCSI diagnostic commands are enabled by sending a SEND DIAGNOSTIC command to the drive. The diagnostic mode is always disabled on power-up or after a host Reset. Once enabled, the diagnostic commands are issued by sending a ten byte command with CDB0 (byte 0) set to 0FFh and CDB6 set to the appropriate subcode. Other bytes may be used as necessary and are described in the Command Description section in this chapter.

SCSI Send Diagnostic Command

	7	6	5	4	3	2	1	0
0	Opcode = 01Dh							
1	LUN = 0			Reserved = 0		SLFTST	DEVOFL	UNTOFL
2	Reserved = 0							
3	Reserved = 0							
4	Parameter List Length = 7							
5	VU = 0		Reserved = 0				F	L

This is the standard SCSI SEND DIAGNOSTIC command. This is used to enable and disable the diagnostic (a.k.a. Super) mode. The diagnostic mode is always disabled on power up or after a host reset.

SLFTST - Self test is supported
DEVOFL - Device off-line is not supported
UNTOFL - Unit off-line is not supported

The parameter list length specifies the length in bytes of the parameter list that shall be transferred during the DATA OUT phase. A value other than 7 will result in an ILLEGAL REQUEST sense error.

SCSI Send Diagnostic Data Format

Byte	Enable Quantum Diagnostic Mode	Enable Apple Basic Diagnostic Mode	Enable Apple Advanced Diagnostic Mode	Disable Diagnostic Modes
0	C1h	70h	C1h	00h
1	F9h	49h	F9h	xx
2	37h	6Fh	37h	xx
3	CFh	4Eh	CFh	xx
4	8Eh	64h	8Eh	xx
5	59h	52h	60h	xx

A value of 0 in byte 0 of the parameter list will disable Diagnostic Mode. Bytes 1 through 6 are ignored when byte 0 has a value of 0.

A value of 1 in byte 0 of the parameter list will enable Diagnostic Mode, provided that bytes 1 through 6 are as specified. Other values in bytes 1 through 6 will result in an ILLEGAL REQUEST sense error.

A value other than 0 or 1 in byte 0 of the parameter list will result in an ILLEGAL REQUEST sense error.

Self Test Description

When the **selftest** bit is set in CDB1, Lethal performs the following diagnostic functions:

1. Perform a checksum on the internal rom.
2. Write and read the sector buffer with FF's and 0's.
3. Verify sequencer ram.
4. Test sequencer rollover registers.
5. Perform a butterfly seek test.
6. Verify that the interface is functioning correctly (SCSI).

AT Superset Commands

The diagnostic (superset) mode must be enabled prior to executing any diagnostic commands. Executing a diagnostic command in normal mode will set the Error and Abort bits. The AT diagnostic commands are enabled by sending an AT extended command (command register = F0H) with the Sector Count register set to 9. A 9 byte password is sent to the drive using the special handshake method described below. The diagnostic mode is always disabled on power-up or after a host Reset. The diagnostic commands are sent by setting the Command register to FFH. (see the diagram on the next page)

After the drive has decoded the command and is ready to accept the command bytes, it resets Busy. The host then puts the first command byte (CDB0) into the Command register causing Busy to be set. After the drive receives this byte, it again resets Busy signalling that it is ready for the another byte. All ten CDB bytes are communicated through the command register in this manner. In general, input parameters for the diagnostic commands are sent to the drive using the following handshake:

1. wait for not Busy.
2. write a CDB to the Command register.
3. repeat from step 1 until all CDBs have been sent.

Information is returned from the drive with the following handshake:

1. wait for DRQ.
2. read data from the Error register.
3. write a zero to the Command register.
4. wait for not Busy.
5. if DRQ is set, repeat from step 2 for another byte.

AT Superset Enable Data Format

The password used to enable the Quantum diagnostics is as follows:

Ascii "HBNJMKRH".

AT Extended Command

	7	6	5	4	3	2	1	0
1F1								
1F2	Sector Count = 9							
1F3	Sector = 0							
1F4	Cylinder Low = 0							
1F5	Cylinder High = 0							
1F6	N/A			Drive	Head = 0			
1F7	Command = 0F0h							

To **enable** the superset mode, the task file registers must be set to the values in the figure above. Then the password is sent byte by byte through the command register, register 1F7, using the handshake method mentioned on the previous page.

To **send** a super command, the command register must be set to 0FFh. (all other bytes will be ignored.) Then the CDB bytes are sent through the command register using the handshake method mentioned on the previous page. Again note if the superset command mode has not been enabled and a superset command is sent to the drive, an error will occur.

Ack!

Superset Command Descriptions

Command Summary

Note: All commands marked with a **dw** are implemented in diskware.

Sub Opcode		Command
00h		Read Micro Memory
01h		Write Micro Memory
02h		Read Configuration
03h		Write Configuration
04h	dw	Call Subroutine
05h	dw	Convert LBA To CHS
06h	dw	Compute Starting Sector
07h	dw	Read Command History
08h	dw	Read Cache Tables
09h	dw	Read ECC Results
0Ah		Seek Physical
0Bh	dw	Seek Verify
0Ch		Read Physical
0Dh		Read Long Physical
0Eh		Write Physical
0Fh		Write Long Physical
10h	dw	Reassign Physical
11h	dw	Read Index Time
12h		Read ID
14h	dw	Microstep
15h	dw	Recalibrate
16h	dw	Erase Track
17h	dw	Erase Track Data
18h		Format Track
19h	dw	Seek and Write (Write Immediate)
1Ah	dw	Read Sequencer WCS
1Bh	dw	Write Sequencer WCS
1Ch		Peek Ram
1Dh		Poke Ram
1Eh	dw	Read Variables
1Fh		Factory Format
20h	dw	Start / Stop
21h	dw	Convert CHS To LBA
22h	dw	Do Wiggle
24h	dw	Sectors from Index
25h		Read Short Physical
26h		Write Short Physical
27h		Switch Wedge Set
60h	dw	Read PES (see command 86h)
61h	dw	Read NULLI Table
80h	dw	Write Arizona
81h	dw	Servo Verify
82h	dw	Read Current Cylinder
84h	dw	Load / Unload Heads
85h	dw	Self Scan
86h	dw	Read Perr Variables
87h		Set Mux

Sub	Opcode	Command
C6h	dw	Read Usage
C8h	dw	Read Operating System Information
CAh	dw	Write Operating System Information
D0h	dw	OTC - Offtrack Read Test
D1h	dw	Move Data Strobe
D2h	dw	Window Margin
D4h	dw	Read Offset Units
E0h	dw	Apple Burn In Test
E1h	dw	Read Apple Burn In Test Results
E6h	dw	Read Recovery Information
E7h	dw	Modify Recovery Information

Apple Burn In Test

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			0	LP	ND	SF	Fmt D
2	LBA - msb							
3	LBA							
4	LBA							
5	LBA - lsb							
6	Sub Opcode = E0h							
7	Reserved = 0							
8	Test Parameter Length = 4							
9	Reserved = 0						F	L

This command causes the drive to do one of the following:

Perform a series of Apple specified burn in tests.

Download burn-in parameters. (see the Apple Diagnostic Command Specification for details)

LBA - specifies the starting block number that destructive testing can occur. If this number is zero, the whole drive can be tested destructively. If LBA is non-zero, the block numbers < LBA will be tested read only. Block numbers > LBA will be destructively tested.

FmtD - if set, indicates that the drive will be formatted to erase the G list. **LBA** must be zero for the formatting to occur, or an error will occur.

SF - Stop - Fail - Stop at the end of a test if a fatal error occurs.

ND - Non-destructive - All tests are read only.

LP - Load parameters - Save 512 bytes of parameters sent by the host.

There are four types of tests that can be performed. They include seek tests, defect scans, margin measurements, and soft error rate measurements. The seek tests and defect scans are executed once at the beginning of testing. The margin and soft error tests are executed repeatedly until the drive is reset, powered off, or a read burn in test results command with the stop bit set is received.

Each test is enabled or disabled by a control bit in the test parameter block. Test parameters are sent during the data out phase and consist of a 4 byte block. The following is a description of that block:

Byte	Bit	Test	Description
0	**Bit**	**Test**	**Description**
	0	E_T010	Fixed length seek.
	1	E_T020	Random Seek.
	2	E_T030	
	3	E_T040	Forward scan, write/read.
	4	E_T050	Reverse scan, read only
	5	E_T060	Reverse scan, write/read.
	6	E_T070	Forward scan, read only.
	7	E_T080	
1	**Bit**	**Test**	**Description**
	0	E_T090	Canned OTC test.
	1	E_T100	Canned window margin test.
	2	E_T110	
	3	E_T120	Forward scan, write/read.
	4	E_T130	Random multi-block, read only.
	5	E_T140	Random multi-block, write/read.
	6	E_T150	
	7	E_T160	
2	**Bit**	**Description - data pattern enable byte**	
	0	High frequency pattern enabled	tests 040 - 080.
	1	Low frequency pattern enabled	tests 040 - 080.
	2	Bit shift pattern enabled	tests 040 - 080.
	3	Random pattern enabled	tests 040 - 080.
	4	High frequency pattern enabled	tests 120 - 160.
	5	Low frequency pattern enabled	tests 120 - 160.
	6	Bit shift pattern enabled	tests 120 - 160.
	7	Random pattern enabled	tests 120 - 160.
3		**Reserved**	

The data pattern enable byte specifies the data patterns used in the various tests. For enabled data patterns, they are used in turn for each pass of the test starting from the higher order bit. Since there is only one pass of the tests 040-080, only one bit should be set out of bits 0-3.

See the Apple Diagnostic Command Specification for a detailed description of this command and the definition of the command parameters.

Call Subroutine

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Subroutine Address - msb							
3	Subroutine Address - lsb							
4	Unused							
5	Unused							
6	Sub Opcode = 04h							
7	R1 - A							
8	R0 - X							
9	Reserved = 0						F	L

This command causes the drive to call the subroutine whose address is specified in **subroutine address.** There are two purposes for the command. One is to be able to programmatically execute individual subroutines in the firmware to see that they work or to set up certain initial conditions. This command also allows for the implementation of quick and simple commands without the need to modify the command decode tables and the firmware documentation. This will also helps to reduce the proliferation of commands.

R1, **R0** contain values that are loaded into the cpu's **AX** register before the routine is called.

CHS to LBA

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 21h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Convert the given CHS to an LBA. This command uses defect management to assign a correct LBA to the CHS. An Invalid LBA error is returned if the given CHS is not associated with an LBA. This can occur if the given CHS is an alternate sector with no defect assigned to it. The 3 byte LBA is returned msb first.

Compute Starting Sector

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 06h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to calculate and return the wedge skew of the first sector on the track.

The returned data format is:

Byte	**Contents**
0	Wedge skew number for specified track.

Convert LBA to CHS

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0			Phys	Next
2	Logical Block Address - msb							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address - lsb							
6	Sub Opcode = 05h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return the physical address of the block specified by the **logical block address,** and the number of contiguous sectors that can be read starting at that address. The number of contiguous sectors is determined by the lesser of the number of sectors remaining on the track or the number of sectors until a spare or an offline spared sector is encountered.

Next = 0 The INIT_LBA_TO_CHS routine is invoked.
Next = 1 The NEXT_LBA_TO_CHS routine is invoked.

Phys = 0 The formatted sector is returned in all cases.

The returned data format is:

Byte	**Contents**
0	Cylinder - lsb.
1	Cylinder - msb.
2	Head.
3	Sector.
4	Number of contiguous sectors.

Erase Track

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 16h							
7	Count - msb							
8	Count - lsb							
9	Reserved = 0						F	L

This command causes the drive to erase a track at the specified head of the current cylinder.

Sync bytes are written from the beginning of the wedge for the length of the number of bytes per wedge - count.

The servo information is preserved.

Erase Track Data

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				Inf
2	Reserved = 0							
3	Step Size							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 17h							
7	Count - msb							
8	Count - lsb							
9	Reserved = 0						F	L

This command causes the drive to AC erase a track data area using the following method:

1. AC erase track data on the current position.
2. Move the number of steps specified by **step size.**
3. AC erase track data.
4. Move the number of steps specified by **step size**.
5. AC erase track data.

The move direction is inward if **inf** (IN Flag) is 1, and outward if **inf** is 0. The head number of the area to erase is specified by the **head** parameter.

The main difference between this command and the **erase track** command is that this command erases at the current head position and at two positions offset from the current position.

Count = number of unerased bytes at the end of the wedge.

Factory Format

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Format Type				
2	Data Pattern							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 1Fh							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to perform a "high level" drive format operation. It is included primarily to allow diagnostic software to factory format the drive.

If **Format Type** = 1, the drive is formatted using the existing P list.
If **Format Type** = 0, this format command performs the following functions:

1. Create the primary defect list (P list) with incoming physical defect descriptors.
2. Store the new P list to the disk, **erasing the old P list.**
3. Create a new W list from the new P list information.
4. Unformat the old inlines, if any.
5. Create new inlines using the new W list.
6. Store the new W list to the disk, **erasing the old W list** - **Note:** Grown defect information is lost.

The defect descriptor data format is as follows:

Byte	Description	
0-1	0000h	
2-3	Transfer length - msb first = number of defects * 8	
	Wedge type descriptor	**CHS type descriptor**
4	Cylinder - msb	Cylinder - msb
5	Cylinder - lsb	Cylinder - lsb
6	Head	Head
7	Wedge	0FFh
8	Bytes from wedge - msb	00h
9	Bytes from wedge - lsb	00h
10	Defect length - msb	00h
11	Defect Length - lsb	Sector
12-...	Repeat for each defect descriptor.	

Format Track

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			0	Long	Neg	FD	SD
2	Reserved = 0							
3	Data Field Byte Value							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 18h							
7	Parameter List Length - msb							
8	Parameter List Length - lsb							
9	Reserved = 0						F	L

Format the desired cylinder either with IDs provided by the HOST or IDs provided internally. The value provided in CDB3 will be written throughout the data field in each sector.

Long bit - If this bit = 1, a format long operation will be performed. Format long writes the id field count bytes, head and sector numbers, AND the CRC bytes using data from the **Read ID** command with the **Long** bit set. (See the **read id** command in this chapter for details.)

SD bit - If this bit is set = 1, data is sent from the host. If = 0, the data is read off of the system cylinder. **Note:** The system cylinders CANNOT be formatted. The reason for this is due to the fact that the count byte info is stored in this area. If you format this area, you lose the information. If **SD** is set to 1, the ID field data format for each wedge is as follows:

Byte	Content
0	Flag. (see explanation on the next page.)
1	Count byte 2.
2	Count byte 1.
3	Count byte 0.
4	Sector.
5	Cylinder/Head. (bits 7-3 = lower 5 bits of the cylinder number, bits 0-2 head number)

Count byte = (number of bytes in data segment / 4) - 1

FD bit - If this bit is = 1, the whole drive is formatted from cylinder 0 to the last cylinder using the count byte information stored on the system cylinder. The system cylinders are excluded.

Neg bit - If this bit is set, the negative cylinders will be formatted.

NOTE: Formatting individual tracks with this command may alter certain tracks on the drive which may have been formatted with inline sparing (factory format). This will confuse defect management. Be careful.

ID Field Flag Byte

The flag byte in the ID field performs several functions. The following is a list of those functions.

If there are 3 data fields in the wedge:

Bit	Description
7	Continuing sector - the first count byte is a partial data field from the previous wedge.
6	Second continuing bit - the first count byte is a partial data field from a sector that started 2 wedges previous.
5	0
4	Next header bad - the next ID field is defective, a read late ID is required.
3	ECC is on the last non-zero sector count. (i.e. end of a sector)
2	Skip the 3'rd data field if set. (defective sector).
1	Skip the 2'nd data field if set (defective sector).
0	Skip the 1'st data field if set (defective sector).

If there are 2 data fields in the wedge:

Bit	Description
7	Continuing sector - the first count byte is a partial data field from the previous wedge.
6	Second continuing bit - the first count byte is a partial data field from a sector that started 2 wedges previous.
5	0
4	0
3	Next header bad - the next ID field is defective, a read late ID is required.
2	ECC is on the last non-zero sector count. (i.e. end of a sector)
1	Skip the 2'nd data field if set (defective sector).
0	Skip the 1'st data field if set (defective sector).

If there is 1 data field in the wedge:

Bit	Description
7	Continuing sector - the first count byte is a partial data field from the previous wedge.
6	Second continuing bit - the first count byte is a partial data field from a sector that started 2 wedges previous.
5	0
4	0
3	0
2	Next header bad - the next ID field is defective, a read late ID is required.
1	ECC is on the last non-zero sector count. (i.e. end of a sector)
0	Skip the 1'st data field if set (defective sector).

Load / Unload Heads

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							LD
5	Reserved = 0							
6	Sub Opcode = 84h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

If LD = 1, this command will start up the drive normally.
If LD = 0, the heads will be parked.

Microstep

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				Inf
2	Micro Step Count - msb							
3	Micro Step Count - lsb							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 14h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to move the head position by the number of microsteps specified in the parameter **micro step count**. If **inf** (In Flag) is 1, then the drive microsteps inwards, otherwise, it microsteps outwards.

1 microstep = 1/2048 of 1 track.

Modify Recovery Configuration

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = E7h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to modify the current recovery parameters in RAM only. The drive will then use these new parameters until another modify command comes in or the drive is reset.

See the **Read Recovery Configuration** command for the data description.

Refer to the Apple Diagnostic Command Spec.

Move Data Strobe

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				Dir
2	Reserved = 0							
3	Reserved = 0							
4	Offset - msb							
5	Offset - lsb							
6	Sub Opcode = D1h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Sets the window margin. It only can be cleared when there is a bus reset, power up, another **Move Data Strobe** command, or power cycle occurs.

Dir = 0, window is early.
Dir = 1, window is late.

Refer to the Apple Diagnostic Command Spec.

Off Track Read Test

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			DM	Head			
2	Cylinder - msb							
3	Cylinder - lsb							
4	Worst Case Data Pattern - msb							
5	Worst Case Data Pattern - lsb							
6	Sub Opcode = D0h							
7	Low Frequency Data Pattern - msb							
8	Low Frequency Data Pattern - lsb							
9	Reserved = 0						F	L

Perform a series of write and offtrack read tests.

DM = 0 Perform the test according to the specifications. (?? what are the specs ??)
DM = 1 Perform the test until it fails.

Refer to the Apple Diagnostic Command Spec.

Peek Ram

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							BMA 16
4	Buffer Memory Address 15-8							
5	Buffer Memory Address 7-0							
6	Sub Opcode = 1Ch							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

This command is used to read the contents of the cache buffer RAM. The starting buffer location is specified by the **Buffer Memory Address. Transfer length** specifies the number of bytes to be read.

The address range is from 00000h thru 1FFFFh.

Poke Ram

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							BMA 16
4	Buffer Memory Address 15-8							
5	Buffer Memory Address 7-0							
6	Sub Opcode = 1Dh							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

This command is used to write to the cache buffer RAM. The starting buffer location is specified by the **Buffer Memory Address. Transfer length** specifies the number of bytes to be written.

Pulse Write Gate

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 22h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Wiggle recovery command.

Seek to unused cylinder 1704, search for a wedge pulse, wait 62 byte times, then pulse write gate.

Read Apple Burn In Test Results

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			0	RP	0	Clr	Sbit
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = E1h							
7	Allocation Length - msb							
8	Allocation Length - lsb							
9	Reserved = 0						F	L

This command causes the drive to return the results of the burn in test to this point.

If **Sbit** is set, the test is first stopped and the results are sent back, otherwise the test continues.

If the **Clr** bit is set, the results are cleared after the command is completed.

If the **RP** bit is set, the drive returns the the previously saved burn-in parameters.

The allocation length specifies the number of bytes the initiator has allocated for the returned test results.

See the Apple DiagnosticCommand Specification for a detailed description of this command and the test parameters.

Read Cache Tables

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 08h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return the status of the cache and the cache tables. It returns the segment number of the segment last accessed, the status of the segment (whether it was a hit or a miss) and the cache table entries for all segments.

The number of bytes returned depends on the number of segments in the cache. There are 10 bytes of data for each cache segment. The returned data format is:

Cache Header:

Byte	Content
0-1	Number of bytes in the cache table (lsb first).
2	Segment number of last segment accessed.
3	Cache hit/miss flag: 0 = miss, 1 = hit.
4-n	Cache tables for all cache segments.

Cache Table:

Byte	Content (one per cache segment)
0-2	Lower cache LBA (lsb first).
3	Number of blocks in the entry.
4-5	Upper word of rollover register.
6-7	Upper word of rollunder register.
8	Cache read flag.
9	Cache segment pointer - gives the offset in sectors from the start of the segment to the start of the valid cache data.

Read Command History

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 07h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return 512 bytes. The first two bytes returned indicates the length of the data to be returned (507 for Lethal). The next byte is the number of bytes per entry (13 for Lethal), and the next two bytes point to the next available entry in the command stack. The rest of the buffer (507 bytes) is the command stack data. (39 entries @ 13 bytes per entry)

In order to ease the host decoding logic, the last command entry in the buffer is not allowed to wrap around. Thus the first byte in the command buffer is always the first byte of a CDB for a command.

The returned data format is:

Byte	**Content**
0-1	1FBh. (507)
2	Bytes per entry. (13)
3-4	Offset within buffer.
5-511	Command history data.
	Data format is as follows per command:

Byte	**AT 6 byte**		**AT 10 byte**	**SCSI 6 byte**	**SCSI 10 byte**
0	CDB0	(cmd)	CDB0	CDB0	CDB0
1	CDB1	(head)	CDB1	CDB1	CDB1
2	CDB2	(cyl h)	CDB2	CDB2	CDB2
3	CDB3	(cyl l)	CDB3	CDB3	CDB3
4	CDB4	(sector #)	CDB4	CDB4	CDB4
5	CDB5	(sec cnt)	CDB5	CDB5	CDB5
6	A0		CDB6	S0	CDB6
7	F0		CDB7	F0	CDB7
8	F1		CDB8	F1	CDB8
9	F2		CDB9	F2	CDB9
10	Error code		Error code	Error code	Error code
11	A1		A1	Initiator ID	Initiator ID
12	Cache status (CF)		Cache status	Cache status	Cache status

Read Configuration

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				D/M
2	Page Number							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 02h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command is used to get drive configuration information, such as the drive model and serial number, from the drive. The information is organized in pages similar in concept to the Mode Select/Sense pages. The page to be returned is specified in **page number**. The **write configuration** command may be used to modify these pages.

Upon completion of the command, the number of bytes specified in the requested page is transferred to the host. ASCII fields are left-justified and filled with spaces on the right. Numeric fields are organized with the LSB being sent first.

IF the D/M bit = 1, the configuration information is read from the disk. If 0, the information is read from memory.

Configuration data format

Page	Lengh	Contents
0	1	**Customer number (binary data)** 0 Generic. 1 Apple. 2 Sun. 3 Nixdorf. 4 Nippon. (MEI, Fujitsu) 5 Hp. 6 Olivetti. 7 Acer.
1	2	**Jumper settings.**
2	16	**Vendor = ASCII "QUANTUM ".**
3	16	**Product identification = ASCII characters.**
4	8	**Drive revision level = ASCII characters.**
5	12	**Drive serial number = ASCII characters.**
6	32	**Customer name = ASCII characters.**

Page	Length	Contents				
7	6	**AT configuration flags.**				
		Byte	**Bit**	**Description**		
		0	1	1 = disable courtesy retry.		
			2	1 = disable wiggle retry.		
				0 = enable wiggle retry.		
			3	Auto idle.		
			4	Save command history.		
			5	Auto transfer.		
			6	Auto read.		
		1	0-1	Park mode		
				00	ID park	
				01	MD park	
				10	MD park	
				11	OD park	
			2	Recal record mode (1 = enable)		
			3	Synthesizer power down mode (1 = enable)		
			4-5	I/O read delay.		
				00	0 ns.	
				01	10 ns.	
				10	20 ns. (default)	
				11	30 ns.	
			6-7	DMA modes.		
				00	Single mode - Multiword DMA	transfer one word at a time.
				01	Demand -1 mode -	always 1 word left in fifo until end of transfer.
				10	Demand mode -	no word left in the fifo de-asserts request on last word.
				11	- unused -	
		2-3		AT logical cylinders per drive.		
		4		AT logical heads per cylinder.		
		5		AT logical sectors per track.		
8	1	**Number of heads.**				
9	16	**Configuration validation field.**				
		0,1,0FF,2,0FE,3,0FD,4,0FC,5,0FB,6,0FA,7,0F9,8				
10	167	**Software Zone Table.**				
		Byte	**Description**			
		0-1	Zone 0 - starting cylinder.			
		2-4	Zone 0 - starting logical sector address.			
		5	Zone 0 - sectors per track.			
		6-7	Zone 0 - sectors per zone.			
		8	Zone 0 - track skew.			
		9	Zone 0 - cylinder skew.			
		10-159	Repeat for zones 1 through 15.			
...						
		160-161	Number of data cylinders.			
		162-164	Maximum logical sector address.			
		165	Track wedge skew.			
		166	Cylinder wedge skew.			
11	3	**Number of user accessable sectors - lsb first.**				
12	1	**Trigger mask**				
		Bit	**Description**			
		0				
		1	Seek timeout.			
		2	Seek fault.			
		3				
		4	ECC error.			
		5	Sequencer read/write error.			
		6	Sequencer underrun/overrun.			
		7	Sequencer timeout.			

Page	Length	Contents		
13	2	**Drive family, model:**		
		Byte 0	**Byte 1**	**Model**
		0Eh	00h	D127
		0Eh	01h	D256
		0Eh	02h	D341
		0Eh	03h	D514
		0Eh	10h	D170
14	6	**Hardware Head Mapping**		
		0	Hardware Head Map - Bit n is set if head n was found during power up.	
		1-2	unused.	
		3	User Head Map - Bit n is set if head n is mapped into use.	
		4-5	unused.	
15	49	**Diskware Overlay**		
		Byte	**Description**	
		0	00h - Overlay 0 - Resident diskware.	
		1-2	Load address.	
		3	Number of sectors.	
		4-5	Cylinder.	
		6	Head.	
		7	Starting sector.	
		8	01h - Overlay 1 - Normal operating diskware.	
		9-15	Same fields as above.	
		16	02h - Overlay 2 - Self scan diskware - main loop - page 0.	
		17-23	Same fields as above.	
		24	03h - Overlay 3 - Self scan diskware - commands - page 1.	
		25-31	Same fields as above.	
		32	04h - Overlay 4 - Self scan diskware - commands - page 2.	
		33-39	Same fields as above.	
		40	05h - Overlay 5 - Self scan diskware - commands - page 3.	
		33-39	Same fields as above.	
		48	0FFh - End marker.	
16	1	**HDA Control Flags**		
		Bit	**Description**	
		0		
		1	allow drive to attempt recal on fatal power-up error.	
		2	enable outputs to dac board by servo.	
		3	disables idle_servo mode for r/w guys to look at signals.	
		4	Enable bump counter.	
		5	Bode.	
		6		
		7		

Page	Length	Contents		
17	272	**Hardware Zone Table - Read / Write Parameters**		
		Byte	**Description**	
		0	Zone 0 - servo threshold.	
		1-2	Zone 0 - synthesizer ratios.	
		3-4	Zone 0 - window centering.	
			0-1	cr addr 0-1
			2	sync power down
			3	synth power down
			4	pd power down
			5	enstho
			6-7	gate delay 1-2
			8	str sign
			9-B	str 0-2
			C	cpratio
		5-6	Zone 0 - servo bandwidth / hysteresis voltage control.	
		7-8	Zone 0 - head 0 equilization / filter cutoff/write current adjustment.	
			0-6	filter 3dB 0-6
			7-9	EQ 0-2
			A	servo
			B-E	wcad 0-4
		...	- repeat for each head -	
		21-22	Zone 0 - head 7 equilization / filter cutoff/write current adjustment.	
		23-367	Repeat for zones 1 through 15.	
18		**- unused -**		
19		**- unused -**		
20	48	**Password Access**		
		Byte	**Description**	
		0-15	User password.	
		16-31	Drive password.	
		32-47	Universal password.	

All pages can be read by the **read configuration** command, and all pages except page 1 can be written via the **write configuration** command. Both the host and the drive are expected to transfer the correct number of bytes for a given page based on the documented page length.

Read Current Cylinder

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode =82 h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return the current cylinder over which the actuator is located. The returned data format is:

Byte	**Contents**
0	Current cylinder - lsb.
1	Current cylinder - msb.

Read ECC Results

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 09h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return the latest values for ECC variables. Data format is as follows:

Byte	**Contents**
0	Number of errors: Bits 0-1 = Number of errors in interleave 0. Bits 2-3 = Number of errors in interleave 1. Bits 4-5 = Number of errors in interleave 2.
1-2	Location of the first error in interleave 0 (lsb first).
3	Value of the first error in interleave 0.
4-5	Location of the first error in interleave 1 (lsb first).
6	Value of the first error in interleave 1.
7-8	Location of the first error in interleave 2 (lsb first).
9	Value of the first error in interleave 2.
10-11	Location of the second error in interleave 0 (lsb first).
12	Value of the second error in interleave 0.
13-14	Location of the second error in interleave 1 (lsb first).
15	Value of the second error in interleave 1.
16-17	Location of the second error in interleave 2 (lsb first).
18	Value of the second error in interleave 2.

Read Id

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				Long
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 12h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to read and return all ID fields on the current cylinder and head, starting at index. If the **long** bit is set to 1, the ID's and the CRC bytes are sent back to the host. The data format is:

Byte	Content
0	Wedge count.
1 - n	ID data.
	8 bytes per wedge ID in the following order if **Long** = 0:
i+0	Flag byte.
i+1	Count byte 2.
i+2	Count byte 1.
i+3	Count byte 0.
i+4	Sector.
i+5	Cylinder/Head. (byte = C4 C3 C2 C1 C0 H2 H1 H0)
i+6	Sequencer status.
i+7	Sequencer error status.
	Note: 1024 bytes is always returned to the host.
	12 bytes per wedge ID in the following order if **Long** = 1:
i+0	Flag byte.
i+1	Count byte 2.
i+2	Count byte 1.
i+3	Count byte 0.
i+4	Sector.
i+5	Cylinder/Head.
i+6	CRC.
i+7	CRC.
i+8	CRC.
i+9	0FFh.
i+10	Sequencer status.
i+11	Sequencer error status.
	Note: 1024 bytes is always returned to the host.
	This data is also used by **format track** long.

Read Index Time

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 11h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to determine the time between index pulses and return the value in microseconds. The returned data format is:

Byte	Contents
0-1	Index time (lsb first).

Read Long Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 0Dh							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command is similar to the **read physical** command except that it returns the ECC field in addition to the data field.

This command causes the drive to read the data and ECC field for the sector specified in **cylinder, head** and **sector.** It also dirties the current segment in the cache.

Lethal returns 526 bytes per sector; the first 512 are the data bytes, followed by 2 crosscheck bytes, the last 12 bytes are ECC bytes.

Read Micro Memory

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Microprocessor Memory Address - msb							
5	Microprocessor Memory Address - lsb							
6	Sub Opcode = 00h							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

This command is used to read the memory in the microprocessor's memory address space. The starting memory address is specified by the **microprocessor memory address** and the **transfer length** specifies the number of bytes to be read.

Read NULLI Table

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 61h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Read the NULLI (slope) table.

Byte	Description
0	Length = 8.
1-3	0.
4-11	Nulli data.

Read Offset Units

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = D4h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Read the microstep and window offset values. The return values are equal to 0.01% of track pitch and window value.

Byte	Description
0-1	Microstep.
2-3	Window Margin.

Read Operating System Information

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = C8h							
7	Reserved = 0							
8	1							
9	Reserved = 0						F	L

Read the operating system sector from the system cylinder. (see **write operating system information**)

Read Perr Variables

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0		SIM	CAV	SSS
2	Cylinder - msb - if SSS = 1							
3	Cylinder - lsb - if SSS = 1							
4	Head - if SSS = 1							
5	Count - if CAV = 1							
6	Sub Opcode = 86h							
7	Address - msb - if CAV = 1							
8	Address - lsb - if CAV = 1							
9	Reserved = 0						F	L

This command causes the drive to return the prediction error for the current track, cylinder and head, for all 64 servo bursts. If the returned word is negative, the head is inside (towards to I.D.) of track center (i.e. step out is required to return to track center).

SSS **1 = Start seek + sample.**
SIM **1 = Sample immediate, do not wait for seek complete.**
CAV **1 = Count and address valid.**
Count **Number of samples.**
Address **Address of the variable to be sampled.**

Byte	**Contents**
0	perr burst 0.
2	perr burst 1.
...	
124	perr burst 63.

Read Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 0Ch							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

This command causes the drive to read the number of sectors specified in **transfer length** from the physical address specified in **cylinder, head** and **sector.**

Note: All cache segments get trashed.

Read Recovery Configuration

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = E6h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Read the current recovery configuration data.

The return data format is as follows:

Byte	Content
0	On the fly ECC.
1	Controller read retry count.
2	Controller seek retry count.
3	Read retry step.
4	Write retry step.

Read Sequencer WCS

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 1Ah							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return the 124 bytes of microprogram in the sequencer's Writable Control Store. The data is organized as 32 words of 4-byte instructions in the following format:

Byte	Content
0-3	Instruction word at WCS address 0 (MSB first)
. . .	
124-127	Instruction word at WCS address 31 (MSB first).

Read Short Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 25h							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

Read sectors using the short sector format.

Transfer Length - contains the number of sectors to write.

Read Variables

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 1Eh							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to return a table of pointers to a group of internal firmware variables. The host may use this information along with the **read/write micro memory**, **peekbuf**, and/or **pokebuf** commands to gain indirect access to the firmware variables. The list of pointers chosen to be returned from **read variables** is product specific - catered to product debugging and testing. The returned data format is:

Byte	**Content**
0	Number of pointers.
1-2	Pointer to first variable - lsb first.
3	Variable location. 0 = microprocessor ram. 1 = buffer ram.
4-6	Next variable pointer and location.
. . .	

See next page for details.

The following is the current list of variables returned for this command:

#	Variable	Location	Description
0	nulli	uP ram.	
1	nulli_slp	uP ram	Nulli steps/track for prediction.
2	nulli_offs	uP ram	Nulli Y intercept for prediction.
3	ss_error_flag	uP ram	Selfscan error flags.
4	diskware_flags	uP ram	Diskware flags.
5	fatal_error_code	uP ram	Fatal error code.
6	revision	uP ram	Revision number.
7	generic_error_base	uP ram	Table of sense key/sense codes internal error codes.
8			- unused -
9	head_offs_tbl	uP ram	Table of head offset values computed during recalibration.
10	bits2	uP ram	Servo status flags. 0 When set, read ch. and clocks in low power mode between servos. 1 Set to disable perr defect filter in servo. 2 Set to allow dithering of VCM on spin up. 3 Set to put servo in read only mode and open bump detect range. 4 Set in seek setup if seek length=ff_int_minsklen, else cleared. 5 A stacked seek is pending. 6 Set when actual track is estimated. 7 Set when actual track is estimated.
11	hda_flags	uP ram	HDA status flags. 1 Allow drive to attempt recal on fatal power-up error. 2 Enable outputs to dac board by servo. 3 Disables idle_servo mode for r/w guys to look at signals. 4 Enable bump counter. 5 Bode.
12	mode	uP ram	Mode number.
13	rcl_error_code	uP ram	RECAL error code.
14	norm_index_tout	uP ram	Normal index timeout value.
15	kloop	uP ram	Loop gain compensation.
16	servo_status	uP ram	Servo Status byte.
17	servo_status	uP ram	Servo Status byte.
18	aeqbl	uP ram	Servo ABurst = BBurst (low) Converson Value.
19	aeqbm	uP ram	Servo ABurst = BBurst (mid) Converson Value.
20	aeqbh	uP ram	Servo ABurst = BBurst (high) Converson Value.
21	slp	uP ram	Servo Edge Mode Transducer Slope.
22	kltg	uP ram	Transducer Gain Compensation.
23	aeqbl_tbl	uP ram	Servo ABurst = BBurst (low) Converson Value vs Head Table.
24	aeqbm_tbl	uP ram	Servo ABurst = BBurst (mid) Converson Value vs Head Table.
25	aeqbh_tbl	uP ram	Servo ABurst = BBurst (high) Converson Value vs Head Table.
26	tna_control_shdw	uP ram	Tna shadow register.
27	kltg_tbl	uP ram	Transducer Gain Compensation vs Head Table.
28	st_ff_acl	uP ram	Single Track ACL Feed Forward.
29	st_ff_dcl	uP ram	Single Track DCL Feed Forward.
30	ff_int_in	uP ram	ID Seek Feed Forward Integrator.
31	ff_int_out	uP ram	OD Seek Feed Forward Integrator.
32	m_nulli	uP ram	NULLI Slope vs ID/OD seek vs NULLI_ZONE matrix.
33	b_nulli	uP ram	NULLI Y Intercept vs ID/OD seek vs NULLI_ZONE matrix.
34	head_map	uP ram	Good Head Map.
35	bits3	uP ram	BITS3, servo bits number 3.
36	bump_counter	uP ram	Bump counter.
37	pmet0	uP ram	Start of servo pmet values.
38	measure_ho	uP ram	Routine for measuring head offsets.
39	fl	uP ram	Flags 1, valid diskware bit 4.
40	aw_lba	uP ram	Base address of ACCUWRITE variables.
41	accuwrite_table	uP ram	Accuwrite table.
42	adjust_kloop	uP ram	Routine for kloop calibration.
43	bits4	uP ram	bits4, latch power fail bit 2.

Reassign Physical

<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td colspan="8">Opcode = 0FFh</td></tr>
<tr><td>1</td><td colspan="3">LUN = 0</td><td colspan="5">Reserved = 0</td></tr>
<tr><td>2</td><td colspan="8">Cylinder - msb</td></tr>
<tr><td>3</td><td colspan="8">Cylinder - lsb</td></tr>
<tr><td>4</td><td colspan="8">Head</td></tr>
<tr><td>5</td><td colspan="8">Sector</td></tr>
<tr><td>6</td><td colspan="8">Sub Opcode = 10h</td></tr>
<tr><td>7</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>8</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>9</td><td colspan="6">Reserved = 0</td><td>F</td><td>L</td></tr>
</table>

This command causes the drive to reassign the physical sector whose address is specified in **cylinder, head** and **sector.**

Recalibrate

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 15h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to recalibrate to cylinder 0.

Sectors From Index

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Logical Sector							
6	Sub Opcode = 24h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Gives the position of the logical sector relative to index.

Data is returned as follows:

Byte	Description
0	Cylinder - lsb.
1	Cylinder - msb.
2	Head.
3	Physical Sector.

Seek and Write - Write Immediate

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 19h							
7	Reserved = 0							
8	Number of Sectors							
9	Reserved = 0						F	L

This command causes the drive to seek to the specified cylinder and head and start writing to the next available sector. The **number of sectors** field specifies the maximum number of sectors the command will write. Data used in writing to the sectors are taken from the sector buffer. Head and Cylinder switching is not supported so the number of sectors may not be written.

This command is normally used to test settling.

Seek Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0			CCD	SD
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 0Ah							
7	Transfer Length - msb							
8	Transfer Lengh - lsb							
9	Reserved = 0						F	L

This command causes the drive to seek to the physical track specified in **cylinder** and **head**.

If **ccd** (cylinder check disable) is set to 1, This allows single step seeks beyond cylinder -13 & 1709.

If **sd** (send data) is set to 1, send the seek time to the host.

Byte	Contents
0	Seek time - lsb.
1	Seek time - msb.

Limitation: the seek timer can only return values up to 65534uS.

Seek Verify

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 0Bh							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command is similar to the **seek physical** command except that the **seek verify** command reads the first header it can find after a seek and verifies that it seeked to the correct target cylinder.

Self Scan Test

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 85h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command starts self scan. See chapter 3 on Self Scan for all of the details.

FYI: Self scan is contained in a diskware overlay.

Servo Verify

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Reserved = 0							
6	Sub Opcode = 81h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to seek to the physical address specified in **cylinder** and **head**. It then performs a servo verify on all 64 of the servo wedges on the track. Two bytes of data are returned. The first byte contains the first bad servo and the second byte contains the error code. There is no provision for returning more than one bad servo per track. The returned data format is as follows.

Byte	Contents
0	Wedge number of the first bad servo (if any).
1	Error code.

The error code bits are as follows:

Bit	Description
7	Sam fault.
6	Sync fault.
5	Data fault.
4	Bump fault.
3	Defect fault.
2	Motor speed fault.
1	Track ID fault.
0	- unused -

Set Mux

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Data 1							
3	Data 2							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 87h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Set the mux output pins.

Data 1 Bits		Description
7-6		**SONCE pin.**
	0 0	Index, once around.
	0 1	Soft index, once around.
	1 X	Phase, spindle phase.
5-3		**TESTIO2 I/O pin.**
	0 0 0	Set signal TESTIO2 as input for chip test (AT write gate)
	0 0 1	SAM output, servo addres mark from TNA.
	0 1 0	ADCTEST output, test signal from ADC block.
	0 1 1	TRGOUT, trigger out from sequencer.
	1 0 0	LSYNDNO output, latched syndrome not zero from sequencer.
	1 0 1	LCRCNO output, latched CRC not zero from sequencer.
	1 1 0	LCMPNO output, latched compare not zero from sequencer.
	1 1 1	BDTEST output, date field during R/W from sequencer.
2-0		**TESTIO1 I/O pin.**
	0 0 0	Set signal TESTIO1 as input for seq trigger in.
	0 0 1	DIV12 output, PHASE divide by 12.
	0 1 0	TESTCLK output, clock signals from UPI block. (see data 2 description, next page)
	0 1 1	Unused.
	1 0 0	BRSTRDY output, Burst ready from TNA.
	1 0 1	SAMSYNFLT output, servo address mark or sync pattern fault from TNA.
	1 1 0	SVOFLT output, servo fault from TNA.
	1 1 1	WFAULT output, write fault from seq.

Data 2 Bits		Description
7-6		Unused.
5-3		TEST CLOCK SELECT
	0 0 0	IFCLK.
	0 0 1	TNACLK.
	0 1 0	ADCCLK.
	0 1 1	SERCLK.
	1 0 0	MTRCLK.
	1 0 1	REFCLK.
	1 1 0	10 MHz clock.
	1 1 1	26.6 MHz clock.
2-0		ADC MUX SELECT
	0 0 0	Synchronous clock enable.
	0 0 1	Asynchronous clock enable.
	0 1 0	ADC conversion at channel 4 is done.
	0 1 1	All MUX signals are or'ed.
	1 0 0	All latched signals are or'ed.
	1 0 1	ADC start.
	1 1 0	ADC clock.
	1 1 1	ADC conversion is done.

Start / Stop

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				IMM
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0						M	S/L
5	Reserved = 0							
6	Sub Opcode = 20h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command requests the drive to spin up or spin down.

If S/L flag is 1, the spin up and recalibration procedure is performed and further drive operations are enabled. If S/L is 0, the drive is spun down or parked depending on the M bit. Further operations requiring read/write and mechanical functionalities are disabled.

If M is 0, Start/Stop will be performed. If M is 1, Load/Unload will be performed.

If IMM is 1, the drive will return with completion status as soon as the operation is initiated. If it is 0, the drive will remain busy until the operation is completed.

M	S/L	Operation	Description
0	0	Stop	Park & spin down.
0	1	Start	Unpark and spin up.
1	0	Load	Unpark and spin up.
1	1	Unload	Park and spin up.

Switch Wedge Set

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 27h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

Attempt to switch to the other servo wedge set on dual wedge drives.

The drive will respond by returning 1 byte back to the host. If it is = 0, the command was successful. If it non-zero, the drive was unable to switch to an alternate wedge set.

Window Margin

<table>
<tr><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td colspan="8">Opcode = 0FFh</td></tr>
<tr><td>1</td><td colspan="3">LUN = 0</td><td>Dmsr</td><td>Rd</td><td colspan="2">Reserved = 0</td><td>Rnd</td></tr>
<tr><td>2</td><td colspan="8">Cylinder - msb</td></tr>
<tr><td>3</td><td colspan="8">Cylinder - lsb</td></tr>
<tr><td>4</td><td colspan="8">Head</td></tr>
<tr><td>5</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>6</td><td colspan="8">Sub Opcode = D2h</td></tr>
<tr><td>7</td><td colspan="8">Data Pattern - msb</td></tr>
<tr><td>8</td><td colspan="8">Data Pattern - lsb</td></tr>
<tr><td>9</td><td colspan="6">Reserved = 0</td><td>F</td><td>L</td></tr>
</table>

Write a data pattern and then perform a series of read tests with different window margins.

Rnd = 0, use the data pattern supplied.
Rnd = 1, a random data pattern is generated.

Rd = 0, write/read.
Rd = 1, read only.

Refer to the Apple Diagnostic Command Spec.

Write Arizona

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Data - msb							
3	Data - lsb							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 80h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to write the 13 bits of data specified in CDB2-3 to the read/write asic.

Write Configuration

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				SP
2	Page Number							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 03h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command is used to write drive configuration information, such as the drive model and serial number, to the drive. The information is organized in pages similar in concept to the Mode Select/Sense pages. The page to be sent is specified in CDB2. The **read configuration** command may be used to read these pages.

The number of bytes implicit for the specified page are transferred from the host. ASCII fields are left justified and filled with spaces on the right. Numeric fields are organized with the least significant byte being sent first.

If the **SP** bit is set, the saveable configuration pages are written to disk. If it is a 0, the pages are only modified in memory.

Refer to the **read configuration** command for a description of the configuration pages.

Write Long Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 0Fh							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command is similar to the **write physical** command except that it writes the ECC field in addition to the data field. This command causes the drive to write the data and ECC fields for the sector specified in **cylinder, head** and **sector.** It also trashes the current segment in the cache.

526 bytes are written. The first 512 are the data bytes, followed by 2 crosscheck bytes, the last 12 bytes are ECC bytes.

Write Micro Memory

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Microprocessor Memory Address - msb							
5	Microprocessor Memory Address - lsb							
6	Sub Opcode = 01h							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

This command is used to write the memory in the microprocessor's memory address space. The starting memory address is specified by the **microprocessor memory address** and the **transfer length** specifies the number of bytes to be written.

Write Operating System Information

<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td colspan="8">Opcode = 0FFh</td></tr>
<tr><td>1</td><td colspan="3">LUN = 0</td><td colspan="5">Reserved = 0</td></tr>
<tr><td>2</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>3</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>4</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>5</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>6</td><td colspan="8">Sub Opcode = CAh</td></tr>
<tr><td>7</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>8</td><td colspan="8">1</td></tr>
<tr><td>9</td><td colspan="6">Reserved = 0</td><td>F</td><td>L</td></tr>
</table>

Writes a sector on the system cylinder with operating system information. This is used by certain customers to load proprietary data on the drive where an end user can't get to it. I guess you could call it a software key.

Write Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 0Eh							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

This command causes the drive to write the number of sectors specified in **transfer length** to the physical address specified in **cylinder, head** and **sector**.

Note: Cache segments get squashed.

Write Sequencer WCS

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = 1Bh							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						F	L

This command causes the drive to accept 128 bytes of microprogram from the host and download it to the sequencer's Writable Control Store. The data is organized as 32 words of 4-byte instructions in the following format:

Byte	Content
0-3	Instruction word at WCS address 0 (MSB first)
. . .	
124-127	Instruction word at WCS address 31 (MSB first).

Write Short Physical

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Cylinder - msb							
3	Cylinder - lsb							
4	Head							
5	Sector							
6	Sub Opcode = 26h							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						F	L

Write data to the drive using the short sector data format. Data sent to the drive should consist of 514 bytes per sector where each block of 514 bytes consists of two bytes of ID followed by 512 data bytes. The two ID bytes should be "3Fh,<sector number>". Sector numbers must be sequential starting with sector 0. A maximum of 32 sectors may be written on any given track.

See the section on short sectors in the self scan chapter for more detail.

Transfer Length - contains the number of sectors to write.

What me worry?

Self Scan

Introduction

Self scan was introduced to reduce production costs by letting the drive test itself without the need for dedicated test equipment, i. e. A drive running self scan only needs a test jumper and power. The first release of self scan replaced digital scan and all of it's associated hardware.

Daytona's self scan is written to replace servo verify, digital scan, and function test plus the associated hardware. This also eliminates handling steps in the manufacturing process. This version will take a raw drive from the servo writer, verify the servo wedges, initialize the drive with SCSI or AT diskware, initialize the system and test cylinders, then run the rest of the production tests.

How It Works

Self scan feeds itself commands stored on the drive's test cylinder. Self scan reads each command and its associated parameter list from the test cylinder, performs the test, and stores the results to the test cylinder. The self scan command and parameter list must be stored on the test cylinder prior to the start of self scan. On Daytona drives, this is either done with the servo writer, or at a diskware station.

Self scan is invoked either by inserting the self scan jumper on the power adapter board or through the self scan superset command. Note: Since the drive checks the self scan pin during power up only, the test jumper must be installed prior to power on to invoke the test.

The following shows the differences in self scan operation when invoked from a super command or a jumper.

Jumper invoked self scan:

1. The LED on the adapter board is turned on for the duration of the test.
2. The test script password is checked for validity.
3. The test script header is copied to the result data.
4. Perform each test, writing the results at the end of each test.
5. Writes the self scan tail, which contains accumulated test data.
6. Sets the LED to a fast blink if self scan passed.
   Sets the LED to a slow blink if self scan failed.

Diagnostic invoked self scan:

1. The drive disconnects for the duration of the test.
2. The LED on the adapter board is turned on for the duration of the test.
3. The test script password is checked for validity.
4. The test script header is copied to the result data.
5. Perform each test, writing the results at the end of each test.
6. Writes the self scan tail, which contains accumulated test data.
7. Turns the LED off and reconnects. Sense information will reflect a failure.

Servo Writer Drive Initialization

The servo writer will perform two functions. One is to write multiple sets of the servo wedges, and the other is to install all of the data used by self scan to test and initialize the drive in it's final configuration. This eliminates the function test station and the drive handling associated with it.

The servo writer will **not** perform the servo verify function. Self scan will do the verify. This will cut down the time the drive spends on the servo writer by approximately 45%. Since the servo writer is an expensive piece of equipment, improving it's throughput will reduce the production costs associated with it. In this case, the throughput almost doubles.

Multiple Wedge Sets

The drive is written with multiple complete sets of servo wedges. This is done to improve the yield from the servo writer. If the servo verify function finds a good set of servo wedges, it will erase the unused or bad wedge set.

Note: This dual wedge set scheme does not affect the servo code. During boot up, the drive will simply sync on one of the wedge sets and ignore the other.

Diskware and System Data

System data will be written to the drive at the servo writer station. The data will consist of diskware for both SCSI and AT drives, serial number, format ids, wedge set ids, and servo writer test data. Multiple copies of the data will be written to each wedge set on the drive for redundancy.

See the table on page 3-78 for the servo written sector layout.

Wedge Set ID Sector

There is a sector located on each wedge set that contains information on how many wedge sets are written to the drive, and what wedge set are we currently on. This data is only used by servo verify.

The ID sector format is as follows:

Byte	Content	
0-511	ID byte. (The ID byte is used as a fill pattern for this sector.)	
	Bits	**Description**
	7-4	Number of wedge sets. (1, 2, ...)
	3-0	Wedge set number. (0, 1, 2, ...)

Short Sectors

The sector format that is used to store data between the servo wedges is non standard. In order to store 256 bytes of data between each wedge, a new format was required due to timing constraints. This sector format does not use an id field. In Daytona code this is referred to as short sectors.

Physical Track Layout

The following shows the physical track layout written by the servo writer. Note there are two identical sets of servo wedges and two identical sets of sector data that are interleaved.

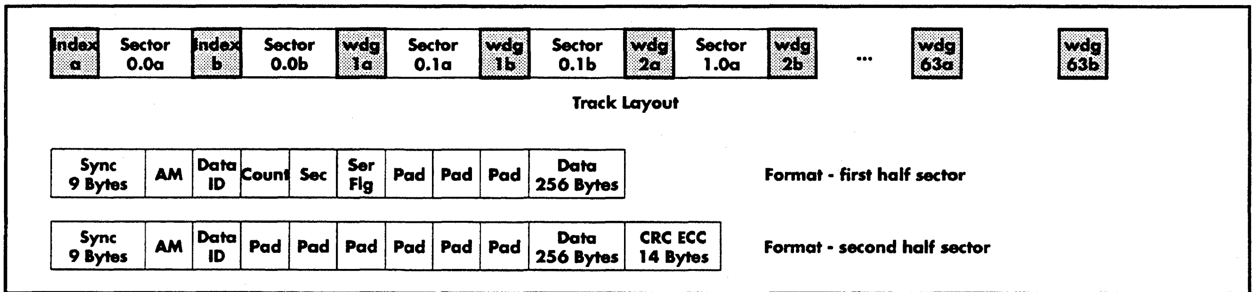


Figure 3-1Physical Track Layout and Short Sector Format.

Sec	Cylinder 749, 789	Cylinder 750, 790	Cylinder 751, 791	Cylinder 752, 792	Cylinder 753, 793
00	Serial Number	Configuration Page 15	AT Diskware Overlay 0	AT Diskware Overlay 2	AT Diskware Overlay 5
01	Self Scan Commands	AT Diskware Overlay 0	Resident - con't	con't	con't
02		Resident	AT Diskware Overlay 1		
03			Normal Operation	AT Diskware Overlay 3	
04				Self Scan	
05				Command Page 1	
06					
07					
08					
09	Servo Writer Test Data				
10	Format IDs - Zone 00				
11	Format IDs - Zone 01				
12					
13					
14					
15				AT Diskware Overlay 4	
16				Self Scan	
17				Command Page 2	
18			AT Diskware Overlay 2		
19			Self Scan		
20			Main Loop		
21			+		
22			Command Page 0		
23					
24					
25	Format IDs - Zone 15				
26	Wedge Set ID Number				
27				AT Diskware Overlay 5	
28				Self Scan	
29				Command Page 3	
30					
31					

Sec		Cylinder 758, 798	Cylinder 759, 799	Cylinder 760, 800	Cylinder 761, 801
00		Configuration Page 15	SCSI Diskware Overlay 0	SCSI Diskware Overlay 2	SCSI Diskware Overlay 5
01		SCSI Diskware Overlay 0	Resident - con't	con't	con't
02		Resident	SCSI Diskware Overlay 1		
03			Normal Operation	SCSI Diskware Overlay 3	
04				Self Scan	
05				Command Page 1	
06					
07					
08					
09					
10					
11					
12					
13					
14					
15				SCSI Diskware Overlay 4	
16				Self Scan	
17				Command Page 2	
18			SCSI Diskware Overlay 2		
19			Self Scan		
20			Main Loop		
21			+		
22			Command Page 0		
23					
24					
25					
26					
27				SCSI Diskware Overlay 5	
28				Self Scan	
29				Command Page 3	
30					
31					

Figure 3-2Servo Written Sector Layout.

Command Data

Command data is stored on the test system cylinder of the drive. Four sectors (2kB) are reserved for the test script. (See chapter5 for more details.)

Test Script Data Structure

The test script data contains a test header, defect scan data patterns, and a command list with associated parameters. The data structure is as follows:

Byte	Description - Test script header.
0-7h	"SELFSCAN" or "LOCKLOCK" (ascii).
8-Fh	Version number.
10-11h	Maximum number of hard errors allowed for this drive.
12-13h	Maximum number of hard errors allowed for each head.
14-15h	Maximum number of hard errors allowed for each cylinder.
16-17h	Maximum number of seek errors allowed for this test.
18-19h	Maximum test time allowed for this test. (seconds)
1A-1Bh	Number of commands - LPT use only.
1C-1Fh	- unused -
20-2Bh	Serial Number
2C-38h	- unused -
39-3Fh	Power On Restart Variables - self scan use only.
40-1FFh	Defect scan data patterns 1-14 - see below for description.
200h	Beginning of the command list - see command descriptions for more detail.
200-n	FFh - End of command list. (n < 800h)

Note: All unused and reserved bytes should be set to 0 or unpredictable results will occur.

Defect Scan Data Patterns

The user can define data patterns to be used in the defect scans. 32 bytes are set aside for each pattern, and the pattern is repeated throughout the data buffer. (The buffer length is set to 1 sector or 1 wedge size in bytes depending on the type of scan) The patterns must follow the following format:

Byte	Description.
0	Pattern length - anything from 1 - 31 bytes.
1-1F	Data pattern

If byte 0 = FFh, then a random pattern will be generated. The following format applies:

Byte	Description.
0	FFh - Generate a random pattern.
1-4	Random number seed. (if = 0, seed is set internally)
5-6	Random pattern length. (if = 0, pattern length is set to the sector or wedge size.)
7-1F	Unused.

Result Data

Result data is stored on the test system cylinder of the drive. Four sectors (2kB) are reserved for the data.

Test Result Data Structure

The test result data contains a copy of the test script header, result data, and a "tail". The result data structure is as follows:

Byte	Description - Result header. (this is a copy of the command header)
0-7h	"SELFSCAN" (ascii).
8-Fh	Version number.
10h	Maximum number of hard errors allowed for this drive.
12h	Maximum number of hard errors allowed for each head.
14h	Maximum number of hard errors allowed for each cylinder.
16h	Maximum number of seek errors allowed for this test.
18h	Maximum test time allowed for this test. (seconds)
1A-1Bh	Number of commands - LPT use only.
1C-1Fh	- unused -
20-2Bh	Serial Number
2C-38h	- unused -
3A-3Fh	Power On Restart variables.
	Result list - see command descriptions for more detail.
40h	Command result data - 16 bytes per command.
. . .	Repeat for each command.
	FFh - End of results.
	Result Data - Tail.
word	Test execution time.
byte	Last command error code.
byte	Last command firmware error code.
word	Total number of seek errors.
byte	Main loop error code.
byte	Main loop firmware error code.
word	Total soft error count for head 0 - zone 0.
. . .	. . .
word	Total soft error count for head 7 - zone 0.
. . .	Repeat the soft error count per head for each zone.
word	Total wiggle error count for head 0.
. . .	. . .
word	Total wiggle error count for head 7.
byte	Total soft errors for head 0 in soft error rate test.
. . .	. . .
byte	Total soft errors for head 7 in soft error rate test.
word	FFFFh - End Marker.

Power On Restart

Power on restart is a feature used by self scan to allow itself to continue testing after a power failure. Self scan will start running at the beginning of the interrupted test, and continue from there.

Data is stored in the command parameter file to allow this to occur.

The data format is as follows:

Byte	Description - Power On Restart Variables - Test Script header.
39h	Power On Restart Status Byte.
3A-3Bh	Count. (incremented each time the power cycles)
3C-3Dh	Command buffer pointer.
3E-3Fh	Result buffer pointer.

Note: There is no way to keep track of the time on the drive when the power is off. Therefore, the test time on the drive will not correspond with the actual chamber time if long or multiple power outages occur.

Self Scan Commands

The commands currently supported are:

Cmd	Description
00h	Invalid Command.
01h	Sequential Defect Scan.
02h	Delete Password.
03h	Format Inline - uses defect information.
04h	Format Media - uses no defect information.
05h	Servo Verify.
06h	Wait for Power Off.
07h	Butterfly Seek.
08h	Head Switch.
09h	Full Stroke Seek.
0Ah	Random Seek.
0Bh	Single Track Seek
0Ch	Start Stop.
0Dh	- unused -
0Eh	- unused -
0Fh	RRO / NRRO.
10h	Contact Start Stop.
11h	Self Optimization.
12h	Scratch Fill.
13h	Physical Random Defect Scan.
14h	Wiggle Test.
15h	Window Centering.
16h	Write Current Optimization.
17h	Window Margin Test.
FFh	End of List.

The command descriptions are contained in the following pages.

Butterfly Seek

Do a butterfly seek measuring the seek times. Note that the average seek time is calculated anew each time through the loop.

Test Algorithm:

```
while (loop count > 0)
    {
    C1 = 0;
    C2 = maximum cylinder;
    average seek time = 0;
    Seek to C1;
    while (C1 <= maxc)
        {
        Do a timed seek to C2;
        total seek time = seek time + total seek time;
        C2—;
        C1++;
        If (C1 <= maxc)
            {
            Do a timed seek to C1.
            total seek time = seek time + total seek time;
            }
        }
    loop count—;
    }
calculate the average seek time;
```

Parameter List Format:

Type	Description
byte	07h - command code.
byte	Loop count.
word	Pass limit - uS - The average seek time must be less than this limit.

Result Data Format:

Byte	Description
0	07h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Minimum seek time - uS.
8-9	Average seek time - uS.
10-11	Maximum seek time - uS.
12-13	Seek error count.
14-15	- unused -

Contact Start Stop

Stop then start the drive, then perform a mini defect scan while the motor is on. This test can run for weeks, and the possibility of the drive crashing exists, so the test data is stored on the drive after every loop. If the test crashes, data can be retrived to find out the loop count of the failure.

The **temporary result flag** signals that the test is not complete.
The **loop count** in the result data contains the number of loops that have been completed.

Parameter List Format: (Note: this input data structure must match the input data structure of defect scan.)

Type	Description	
byte	10h - command code.	
byte	- unused - (defect scan loop count)	
byte	- unused - (defect scan option byte)	The defect scan is forced to be "sequential sector scan"
byte	Read scan count.	i.e. same as if defect scan option byte = 2.
byte	Write scan count.	
byte	Read retry count.	
byte	Write retry count.	
byte	Soft error limit - head.	
byte	- unused -	
word	Soft error limit - drive.	
byte	Margin.	
byte	Margin retries.	
byte*8	Pattern sequence.	
word	Hard read error threshold.	
word	Hard write/verify error threshold.	
word	Start cylinder.	
word	End cylinder.	
word	Offtrack.	
word	Offtrack retries.	
byte*8	Command sequence.	
long	Total sectors transferred.	
long	Random number seed.	
word	Test time limit - S.	
byte	CSS options byte.	
	Bit Description	
	0 0 = Dynamic braking off.	1 = Dynamic braking on.
	1 0 = Auto idle off.	1 = 5 second auto idle.
word	CSS loop count.	
word	Motor on time - S.	
word	Motor off time - S.	

Result Data Format:

Byte	Description
0	10h - command code.
1	CSS option byte.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Completed loop count.
8	Temporary result flag - FFh.
9	- unused -
10-11	Retry count.
12-13	Seek error count.
14-15	Max spinup time.

Defect Scan

Scan the drive to find defects. Any defects found will be entered into a defect map that is eventually used to create a Primary defect list used during normal drive operation.

Parameter List Format:

Type	Description
byte	01h - command code.
byte	Loop count. (for sequential scan)
byte	Defect scan option select. (See the description on the next page.)
byte	Read sequential scan loop count.
byte	Write sequential scan loop count.
byte	Retry read loop count.
byte	Retry write/verify loop count.
byte	Soft errors allowed per head. (soft error rate test)
byte	- unused -
word	Soft errors allowed per drive.
byte	Margin option.
byte	Margin option for retry.
byte*8	Data pattern sequence. (See the description on the next page.)
word	Hard read error threshold.
word	Hard write/read/verify error threshold.
word	Starting cylinder number.
word	Ending cylinder number.
word	Read off-track (bit 15 sign bit, off-track 0fd00H TO 01ffH).
word	Read off-track for retry.
byte*8	Command sequence. (See the description on the next page.)
word*2	Number of sectors/wedges transferred by this command if random scan selected.
word*2	Random number seed if random scan selected.
word	Test end time. If the clock reaches this time, the test will end, no error is generated.

Result Data Format:

Byte	Description Random Scan	Byte	Description Sequential Scan
0	01h - command code.	0	01h - command code.
1	Defect scan option select.	1	Defect scan option select.
2	Command error code.	2	Command error code.
3	Firmware error code.	3	Firmware error code.
4-5	Test time - S.	4-5	Test time - S.
6-9	Num blocks/wedges transferred (lsb first).	6-7	Last error - Cylinder
		8	Last error - Head
		9	Last error - Sector/wedge
10-11	Retry count.	10-11	Retry count.
12-13	Seek error count.	12-13	Seek error count.
14	Scan type.	14	Scan Type
15		15	

Defect Scan Option Select Byte

Bit	If = 0	If = 1	Description
0	Sequential	Random	Scan Type.
1	Wedge	Sector	Scan Format.
2	Track	Wedge/Sector	Scan Length.
3	Normal	No new defects	No new hard defects allowed for this scan.
5	Scan 0	Scan immediate	Scan Start Sector.
6	Off	On	Ignore ID Error
7	Normal	Soft error rate	Soft error rate test. (Defects will not be entered into the defect map, soft errors will be accumulated in dedicated counters.)

Margin Option Byte

Bit	Description		
7-4	Test threshold		
	1111 -	Default.	
	0XXX -	Set threshold.	Threshold = 70% - (XXXB * 5%)
		i.e.	45% = 70% - (101B * 5%)
3-0	1111	Default from zone table.	

Data pattern sequence

8 bytes are set aside for a data pattern sequence. The data patterns are the patterns stored in the test script. The scan routine will look at each byte to see which pattern is run. The legal values for these patterns are as follows:

Value	Description
00h	Skip - no pattern specified.
01h-0Eh	Pattern number 1-14 stored in the command test script data.
0Fh-FEh	illegal
FFh	Random.

Command sequence - random scan

8 bytes are set aside for a command sequence. The sequence will be a combination of the following commands including at least 1 read command.

Cmd #	Command Description
0	NOP(End of command)
1	seek
2	read
3	write

Defect list

Defect scan keeps track of defects with it's own defect list. the list is stored on the test system cylinder of the drive. 8 sectors (4kB) are reserved for the list.

The defect list data structure is as follows:

Cylinder - msb
Cylinder - lsb
Head
Wedge ID
Bytes from wedge - msb
Bytes from wedge - lsb
Count
Sector
Sector end
Sequencer error
Pattern number
Sequencer command
Defect length - msb
Defect length - lsb

Invalid byte = ffh.
Invalid word = ffffh.
Invalid wedge = fdh.

If cylinder - msb = 0FFh, this indicates the end of the defect list.
Error count = 0 if the defect is a scratch fill created defect.

Delete Password

Overwrites self scan password with the lock lock password on the command and result data sectors on the drive. This will prevent self scan from running again and is used to protect the drive after shipment.

Parameter List Format:

Type	**Description**
byte	02h - command code.

Result Data Format:

Byte	**Description**
0	02h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-15	- unused -

End of List

This command number signals the end of the self scan process. There are no more commands in the list.

Parameter List Format:

Type	Description
byte	FFh - command code.

Format Inline

Create the primary defect list (P list) from the self scan defect list, and format the drive's user area with inline spares. The system cylinders are not formatted.

Note: The ID field count byte information must be on the system cylinder.

Parameter List Format:

Type	Description
byte	03h - command code.

Result Data Format:

Byte	Description
0	03h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time.
6-15	- unused -

Format Media

Format the drive's user area ignoring defects. The sector data is initialized to 00h. System cylinders are not formatted.

Note: ID field information must be on the system cylinder.

Parameter List Format:

Type	Description
byte	04h - command code.

Result Data Format:

Byte	Description
0	04h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Last error - cylinder.
8	Last error - head.
9	Last error - sector.
10-15	- unused -

Full Stroke Seek

Measures seek times from cylinder 1 to cylinder 2.

Test Algorithm:

```
target head = head;
while (loop count != 0)
      {
      seek to cylinder 1;
      do a timed seek to cylinder 2;
      total seek time = seek time + total seek time;
      loop count —;
      }
calculate the average seek time;
```

Parameter List Format:

Type	Description
byte	09h - command code.
byte	Loop count to average (byte) if 0 then endless loop.
word	Pass limit - uS - The average seek time must be less than this limit.
word	Cylinder 1.
word	Cylinder 2
byte	Head.
byte	Option Select.

Result Data Format:

Byte	Description
0	09h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Minimum seek time - uS.
8-9	Average seek time - uS.
10-11	Maximum seek time - uS.
12-13	Seek error count.
14-15	- unused -

Option Select Byte

Bit 0 **If = 0**
Seek time is measured from C1 to C2.
If = 1
Seek time is measured from C1 to C2 then from C2 to C1.

Head Switch

Measures head switch times.

Test Algorithm:

```
seek to the specified cylinder and head 0;
while (loop count != 0)
    {
    if (rotate)
        do timed seek, rotating the head after each seek;
    else
        seek to head 1, then do a timed seek to head 2;
    total seek time = seek time + total seek time;
    loop count—;
    }
calculate the average seek time;
```

Parameter List Format:

Type	Description
byte	08h - command code.
byte	Loop count.
word	Pass limit - uS - The average seek time must be less than this limit.
word	Cylinder.
byte	Head 1.
byte	Head 2.
byte	Option Select.

Result Data Format:

Byte	Description
0	08h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Minimum head switch time - uS.
8-9	Average head switch time - uS.
10-11	Maximum head switch time - uS.
12-15	- unused -

Option Select Byte

Bit	If = 0	If = 1	Description
0	Rotate	Step	Rotate through the heads or step from head 1 to head 2.
1	Up	Down	Rotation direction.

Invalid Command

This is an invalid command code. If this value is read from a list as a command, then an error condition exists and all self scan processing is stopped.

Parameter List Format:

Type	Description
byte	00h - command code.

Physical Random Defect Scan

Scan the drive randomly to find defects.

This is actually the **defect scan** command with the appropriate bits set. Take a look at the defect scan command for the parameter formats.

Note: The command, **defect scan**, was designed to cover all types of scans, and it does. Somebody just wanted a unique command number assigned to this command.

Parameter List Format:

Type	**Description**
byte	13h - command code.

All other bytes are the same as in defect scan.

Random Seek

Seek to a random head and cylinder and measure the seek times.

Test Algorithm:

```
while (loop count != 0)
    {
    do a timed seek to a random cylinder and head;
    loop count—;
    }
calculate the average seek time;
```

Parameter List Format:

Type	Description
byte	0Ah - command code.
byte	Loop count to average (byte) if 0 then endless loop.
word	Pass limit - uS - The average seek time must be less than this limit.
word	Number of random seeks to perform per loop.

Result Data Format:

Byte	Description
0	0Ah - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Minimum seek time - uS.
8-9	Average seek time - uS.
10-11	Maximum seek time - uS.
12-13	Seek error count.
14-15	- unused -

RRO / NRRO

Measures the **repeatable** and **non-repeatable runout** on the drive. The self scan result data just contains pass/fail information with no details. The data from the RRO/NRRO measurements will be stored in the test cylinder on two consecutive sectors. The data format is listed below.

The measurements will be taken twice on each platter at the outer and inner cylinders and the results will be stored on the test cylinder. (see chapter 5 for test cylinder layout)

The equations used to calculate these measurements are located in Figure 3-3 on page 3- 98, and the test algorithm can be found on the following page.

Parameter List Format:

Type	**Description**		
byte	0Fh - command code.		
byte	status.		
byte	reserved * **see note below**		
word	"		
word	"		
word	"		
word	RRO limit	% of track width * 10	i. e. 217 (D9h) = 21.7%
word	NRRO limit	% of track width * 10	

Result Data Format:

Byte	**Description**
0	0Fh - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-15	- unused -

RRO/NRRO Data

The following data is stored on the runout result sectors.

RRO / NRRO data format:

Byte	**Description**		
0	Number of wedges.		
1	Number of samples per wedge.		
2-15	Unused.		

Repeat the following data structure for each head on both the inner and outer cylinders.

16	Head.		
17	Unused.		
18-19	Cylinder.		
20-21	RRO	% of track width * 10	i. e. 406 (196h) = 40.6%
22-23	NRRO	% of track width * 10	
. . .			

Note: This routine can be called from the servo verify command therefor the input data structure for this routine must match the data structure for servo verify.

RRO / NRRO test algorithm

```
for (head = 0;head < maxh;head = head+2)
    {
    for (cylinder = 0;cylinder = maxc;cylinder = cylinder + maxc)
        {
        ns = 128;                                  // number of samples = 128
        set sum, sum square, and data sample arrays to 0;
        seek to the first cylinder in the zone;
        wait 1 revolution for settling;
        for (data samples = 0;data samples < ns;data samples++)
            {
            starting at index, aquire the PERR data for this track;
            for (wedge = 0;wedge < num of wedges;wedge++)// get the data for each wedge and keep
                    {                              // a running sum and sum of the squares
                sum[wedge] = sum[wedge] + data sample[wedge];// for all of the samples.
                sum square[wedge] = sum square[wedge] + (data sample[wedge] * data sample[wedge]);
                }
            }
        std dev sum = 0;                           // init the sum of the standard deviations
        ave sum = 0;                               // init the sum of the averages
        ave sum square = 0;                        // init the sum squared of the averages
        for (wedge = 0;wedge < num of wedges;wedge++)
            {
            average[wedge] = sum[wedge] / ns;      // figure the std deviation and ave value for each wedge
            ave sum = ave sum + average[wedge];
            ave sum square = ave sum square + (average[wedge] * average[wedge]);
            std dev[wedge] = sqrt((sum square[wedge] - ((sum[wedge] * sum[wedge]) / ns)) / (ns - 1));
            std dev sum = std dev sum + std dev[wedge];
            }
                                                   // rro = standard deviation of the average wedge values.
        RRO = sqrt((ave sum square[wedge] - ((ave sum[wedge] * ave sum[wedge]) / 64)) / 63;
        RRO = (RRO * 3 * 100) / 2048;
        NRRO = std dev sum / 64;                   // nrro = average of the wedges standard deviations.
        NRRO = (NRRO * 3 * 100) / 2048;
        if (either RRO or NRRO are out of spec)
            {
            set fatal error flag;
            }
        store the data in the system cylinder.
        }
    }
if (fatal error)
    {
    set error code;
    store RRO and NRRO data;
    exit(error);
    }
else
    {
    exit(ok);
    }
```

RRO / NRRO variable descriptions.

n	Number of track samples. (128) Note: Each track sample contains the PERR for each servo wedge.
x	PERR
w	Number of servo wedges. (64)

RAW_X	x	Step 1: For each **track sample**, do the following: 1. Aquire the PERR for each servo wedge. 2. Keep a running sum of the PERRs. 3. Keep a running sum of the squares of the PERRs.
SUM_OF_X	$\sum_1^n x$	
SUM_OF_SQ_OF_X	$\sum_1^n x^2$	
X_BAR	$\bar{x} = \frac{\sum_1^n x}{n}$	Step 2: For each **wedge**, do the following: 1. Calculate the average PERR, X_BAR. 2. Calculate the standard deviation of the PERRs. 3. Keep a running sum of the X_BARs. 4. Keep a running sum of the squares of the X_BARs. 5. Keep a running sum of the standard deviations.
STD_DEV_OF_X	$\sigma = \frac{\sum_1^n x^2 - \frac{\left(\sum_1^n x\right)^2}{n}}{n-1}$	
SUM_OF_X_BAR	$\sum_1^w \bar{x}$	Step 3: 1. Calculate repeatable runout, RRO, as a percentage of track width. This is: (3 sigma X_BAR * 100)/2048 2. Calculate the non-repeatable runout, NRRO, as a percentage of track width. This is the average of the standard devia tions of the wedges. (3 sigma bar * 100)/2048 2048 is the number of steps per track.
SUM_OF_SQ_OF_X_BAR	$\sum_1^w \bar{x}^2$	
SUM_OF_STD_DEV_OF_X	$\sum_1^w \sigma$	

RRO	$\frac{3 * \sigma\bar{x}}{2048} * 100$	sigma X_BAR	$\sigma\bar{x} = \frac{\sum_1^w \bar{x}^2 - \frac{\left(\sum_1^w \bar{x}\right)^2}{w}}{w-1}$
NRRO	$\frac{3 * \bar{\sigma}}{2048} * 100$	sigma bar	$\bar{\sigma} = \frac{\sum_1^w \sigma}{w}$

Figure 3-3RRO/NRRO variable descriptions and equations.

Scratch Fill

This command detects and fills in areas of a scratch in an attempt to eliminate grown defects.

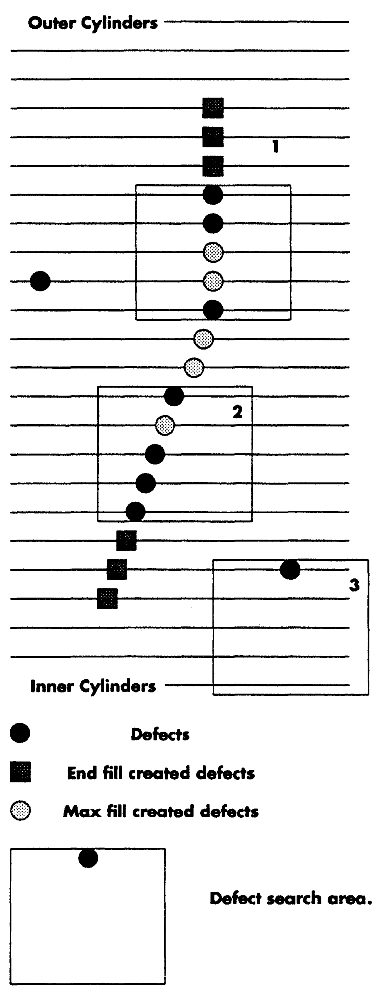
Note: This command converts all of the self scan defects to a bytes from wedge format and stores them back in the defect list with that format.

Parameter List Format:

Type	Description	
byte	12h - command code.	
byte	End fill length.	Number of cylinders added to the ends of a scratch.
byte	Guard band.	Number of bytes on both sides of a defect used to set the defect search area.
byte	Maximum fill length.	Maximum number of cylinders allowed between defects.

Result Data Format:

Byte	Description
0	12h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-15 -	unused -



Self Optimization

This command performs a read channel optimization, correcting the eq and boost values for each head in each zone.

Test Algorithm:

```
Read channel optimization
for (head=0; head  max_head; head ++)
    (
    start_bw=init_bw;
    Init_stress ( win_margin, off_track, bw_transfer_count);
    for ( zone=0; zone  max_zone; zone++)
        (
        if (chk_timeout() == true)
            exit;
        current_cyl=max_cyl(zone);
        while (good_track == false)
            seek(current_cyl);
            offtrack=-200;
            write_guardband(rand_seed (1));
            offtrack=100;
            write_track(off_tarck,rand_seed(2));
            good_track=read_verify();
            if (good_track==false)
                current_cyl--;
        while (good_bw_data== false)
            (
            for (bw=start_bw; lower_slope_found==true I  bw=119 ; bw++)
                set_stress(bw,win_margin,off_track);
                lower_slope_found=get_read_error_count();
                if (lower_slope_found==false)
                    increase_stress(win_margin, off_track)
            if (bw==119)
                increase_stress(win_margin, off_track);
                good_bw_data=false;
            else
                good_bw_data=check_bw_depth();
                    if (good_bw_data==false)
                        decrease_stress(win_margin, off_track);
                    else
                        for (bw=start_bw; upper_slope_found==true I  bw=20 ; bw--)
                            upper_slope_found=get_read_error_count();
                        if (bw==20)
                            increase_stress(win_margin, off_track);
                            good_bw_data=false;
                        else
                            get_opt_bw();
            )
        init_stress(opt_bw_win_margin,opt_bw_off_track, eq_transfer_count);
        while( chk_too_low == false & chk_too_high== false)
            for (eq=1; eq=7; eq++)
                set_stress(eq,eq_transfer_count);
                get_read_error_count();
            if (chk_too_low()== true)
                increase_stress(eq_transfer_count);
            if (chk_too_high()== true)
                decrease_stress(eq_transfer_count);
        get_opt_eq();
        zone==analize_zone_data();
        )
    )
save_zone_table();
write_result();
```

Parameter List Format:

Type	**Description**
byte	11h - command code.
byte	Option select. 00 Test and write results. 01 Test, write results, and update the zone table. 03 Test, write results, update the zone table, and save the zone table to disk.
byte	Maximum time per head in minutes.
byte	Maximum time per zone in minutes.

Result Data Format:

Byte	**Description**
0	11h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6	Result flag. 0 = Pass. 32 = Bypass optimization, use default values.
7-15	- unused -

Servo Verify

This is the first command to run after a servo write. It will verify the servo wedges across the entire surface of the drive. If two or more servo wedge sets are initially written to the drive, and a good set is found during this scan, then the bad or unused set(s) will be erased. If no good set of servo wedges is found, the drive will fail this test and have to be re-servo written. A wedge set will be erased, and data will be stored on the disk.

Since the system cylinder information is written to the drive during servo write, this command must also take care of preserving the data during the erasing of the unused set of servo wedges. The data will be read from the drive and stored in ram, the unused wedges will be erased, and the system cylinders will be formatted. The data will then be written back to the system cylinders.

If RRO/NRRO measurements are to be made, the data from these measurements will be stored in the RRO/NRRO data sector. (see the description for RRO/NRRO) Also, the defective servo wedge information will be stored on the test cylinder as a servo defect list.

Parameter List Format:

Type	Description		
byte	05h - command code.		
byte	Option select. (see below)		
byte	Reserved.		
word	Maximum position error.	% of track width * 10	i. e. 217 (D9h) = 21.7%
byte	Write bump detect level.	% of track width * 10. - double wedge	(max 25.5%)
byte	Write defect detect level.	% of track width * 10. - double wedge	(max 25.5%)
byte	Write bump detect level.	% of track width * 10. - single wedge	(max 25.5%)
byte	Write defect detect level.	% of track width * 10. - single wedge	(max 25.5%)
word	RRO limit	% of track width * 10	
word	NRRO limit	% of track width * 10	

Result Data Format:

Byte	Description
0	05h - command code.
1	Option select.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6	Wedge set used.
7	- unused -
8-9	Error cylinder. (see note below)
10	Error head.
11	Error wedge.
12	Error wedge set.
13	Error type - self scan error code.
14-15	- unused -

Note: Bytes 8-13 of the result data are used to provide more information about the drive when servo verify fails.

The Runout result sectors contain the runout data.

Servo Verify Option Select Byte

Bit	If = 0	If = 1	Description
0	off	on	Verify servo bursts.
1	off	on	RRO/NRRO measurement.
2	off	on	Erase unused servo bursts and initialize diskware.

Drive Initialization

The drive will initialize itself if the appropriate option select bit is set. This includes erasing the unused set of servo wedges, formatting all sectors with a 00h data pattern, and setting up all of the system cylinders. The table on page 3-104 shows what sectors are initialized in the system area.

Note: If drive initialization is selected on a drive that contains normal sectors, only the mode page information will be initialized.

Servo Defect List

Defective servo information will be stored in a defect list on the test cylinder. This data is made available for anyone to use. (see chapter 5 for the test cylinder layout)

Servo defect list format:

Byte	Description
0	Status. **Bit Description** 0 Perr out of range. 1 Track ID error. 2 3 Defect. 4 Bump Error. 5 Data Error. 6 Sync Error. 7 Sam Error.
1	Cylinder - lsb.
2	Cylinder - msb.
3	7-4 Wedge set. 3-0 Head.
4	Wedge number.

Sec	Cylinder -1, -5	Cylinder -2, -6	Cylinder -3, -7	Cylinder -4, -8
00	Servo Writer Test Data	Mode Select 1, 2, ...	Diskware Overlay 0	Diskware Overlay 2
01		Mode Select 3, 4 (hex)	Resident	Self Scan
02		Configuration Pages		Main Loop
03				+
04		Working Defect List		Command Page 0
05				
06	Self Scan Results			
07				
08		Primary Defect List		
09				
10				
11				
12		Temporary Defect List		
13				
14	Self Scan Cmd Parameters			
15				
16		Format ID Bytes - Zone 00		Diskware Overlay 3
17		Format ID Bytes - Zone 01		Self Scan
18		...		Command Page 1
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				Diskware Overlay 4
29				Self Scan
30				Command Page 2
31		Format ID Bytes - Zone 15		
32			Diskware Overlay 1	
33			Normal Operation	
34				
35				
36				
37				
38				
39				
40				Diskware Overlay 5
41				Self Scan
42				Command Page 3
43				
44	RRO/NRRO Results			
45				
46				
47				
48				
49				
50	Servo Verify			
51	Bad Servo List			
52				
53				
...				
...				
64				
65				
66				
67				
68				
69				
70				
71				
72				
73				

Figure 3-4Sectors Initialized by Servo Verify

Servo Verify

```
{
wedge set = 0;
bad wedge set count = 0;
while (wedge set != number of wedge sets)           // check 1 set of servo wedges at a time.
    {
    set all flags = 0;
    syncronize the servo with the current wedge set;
    cylinder = outermost;
    while ((cylinder < maxc) && (fatal error flag = 0))
        {
        head = 0;
        while ((head < maxh) && (fatal error flag = 0))
            {
            seek;
            set the scan data flag;
            while (scan data flag = 1)
                {
                wait for 1 revolution for seek settle;
                starting at index, aquire wedge data for the current track;
                check the wedge data against the specs.;
                if (there were no errors that require a re-scan or we're out of retries);
                    {
                    scan data flag = 0;
                    }
                if (a bad wedge occured in the last data set)
                    {
                    fatal error flag = 1;
                    }
                }
            head++;
            }
        cylinder++;
        }
    if (fatal error flag = 0)                        // if all the wedges were good, no need to check other sets.
        {
        exit wedge set loop;
        }
    else                                             // else this is a bad wedge set.
        {
        wedge set++;
        }
    }
if (wedge set < number of wedge sets)
    {                                                // a good wedge set has been found.
    do RRO and NRRO measurements in three areas of the disk;
    if (RRO and NRRO measurements are ok)
        {
        read in all data from the disk and store it in buffer ram;
        dc erase unused set of servo wedges;
        format the system cylinders;
        configure the drive, writing the data to the system cylinders;
        if (write verify of system cylinders failed)
            {
            set a fatal error condition - minimum number of system copies could not be written.
            }
        }
    else
        {
        set a fatal error condition - RRO/NRRO out of spec;
        }
    }
else
    {                                                // no good servo sets were found. set error then exit.
    set a fatal error condition - no good servo set found;
    }
}
```

Single Track Seek

Measures single track seek times starting at cylinder 1 to cylinder 2.

If the difference between cylinder 1 and cylinder 2 = 1, then the seek test is performed **loop count** number of times.

If the difference between cylinder 1 and cylinder 2 > 1, then the test will single track step from cylinder 1 to cylinder 2 then step back from cylinder 2 to cylinder 1.

Test Algorithm:

```
seek to the test cylinder and head 0;
if (abs(cylinder 1 - cylinder 2) == 1)
    while (loop count != 0)
        seek to cylunder 1 then do a timed seek to cylinder 2;
        loop count—;
else
    step from cylinder 1 to cylinder 2 measuring the seek times for each step;
    step from cylinder 2 to cylinder 1 measuring the seek times for each step;
calculate the average seek time;
```

Parameter List Format:

Type	Description
byte	0Bh - command code.
byte	Loop count.
word	Pass Limit - uS - The average seek time must be less than this limit.
word	Cylinder 1.
word	Cylinder 2.
byte	Head.

Result Data Format:

Byte	Description
0	0Bh - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Minimum seek time - uS.
8-9	Average seek time - uS.
10-11	Maximum seek time - uS.
12-13	Seek error count.
14-15	- unused -

Start Stop

Measures start and stop times.

Test Algorithm:

```
while (loop count != 0)
    {
    do a timed stop of the drive;
    do a timed start of the drive;
    loop count —;
    }
```

Parameter List Format:

Type	Description
byte	0Ch - command code.
word	Loop count.
byte	Waiting time after stop - S.
byte	Test limit time between start to drive ready - S.
byte	Option select. Bit Description 0 1 = Peachfuzz stress test.

Result Data Format:

Byte	Description
0	0Ch - command code.
1	Option select.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	- reserved -
8	Peachfuzz test Vcc before stress.
9	Peachfuzz test Vcc after stress.
10-11	Maximum start time - 1/10'th S.
12-15	- unused -

Third Stroke Seek

Measures third stroke seek times.

Test Algorithm:

```
cylinder = 0;
head = 0;
zone = 0;
seek to the CHS;
while (zone < 3)
    {
    initialize the loop count;
    while (loop count != 0)
        {
        do a timed seek to cylinder = ((maxc/3)*zone) + maxc/3, max head;
        do a timed seek to cylinder = (maxc/3)*zone, head 0;
        }
    zone++;
    }
calculate the average seek time;
```

Parameter List Format:

Type	Description
byte	0Dh - command code.
byte	Loop count.
word	Pass limit - uS - The average seek time must be less than this limit.

Result Data Format:

Byte	Description
0	0Dh - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-7	Minimum seek time - uS.
8-9	Average seek time - uS.
10-11	Maximum seek time - uS.
12-13	Seek error count.
14-15	- unused -

Wait for Power Off

This command suspends selfscan until a power off - power on cycle occurs.

A potential use for this feature is to syncronize the tests during temperature testing. i.e. Run a test at a certain temperature then wait. The temperature is changed then the power is cycled to the drives under test. After power on, a new test is run at the new temperature.

This would get rid of the hassle of trying to run drives in the temperature chamber with computers connected to them.

Parameter List Format:

Type	**Description**
byte	06h - command code.
word	Max time in seconds to wait. Selfscan fails if timeout.
word	Time in seconds between update of results area. This allows selfscan results to show how long we waited in this test.

This command modifies bytes 39 to 3F in the command data as follows:

Byte	**Description**
39	Power on restart status.
3C-3D	Power on restart command buffer pointer.
3E-3F	Power on restart result buffer pointer.

After these bytes are modified, the command data is written back out to the drive. When the power cycle occurs, selfscan will check these bytes. If they are not 0, then selfscan will start running at the command pointed to by the command buffer pointer.

Weighted Average Seek

Measures the weighted average seek time.

Test Algorithm:

```
weighted cylinder = 0;
while (weighted cylinder < maxc)
    {
    seek 1 - do a timed seek to the weighted cylinder;
    seek 2 - do a timed seek to cylinder 0;
    average = (seek 1 + seek 2)/2;
    weighted average = average * (maxc - weighted cylinder);
    weighted cylinder = weighted cylinder + cylinder increment;
    }
```

Parameter List Format:

Type	Description
byte	0Eh - command code.
byte	Loop count, if 0 then endless loop.
word	Pass limit - uS - The average seek time must be less than this limit.
byte	Cylinder increment.

Result Data Format:

Byte	Description
0	0Eh - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-9	Total seek time - uS.
10-11	- unused -
12-13	Seek error count.
14-15	- unused -

Wiggle Test

Detect wiggle phenomenon.

Test Algorithm:

```
SelfScanWiggleTest()
    {
    for (head = 0, wiggle_errs = 0 ; head <= max_head ; head++)
        {
        cyl = FindDefectFreeCylFromID(head);
        for (sectors_xferred = 0 ; sectors_xferred <= max_sectors)
            {
            WriteTrack(erase_data);
            if (ReadTrack() == error)
                wiggle_errs++;
            }
        if (wiggle_errs >= max_errs)
            {
            read_errs = ReadTest(head,cyl);
            if (read_errs * err_ratio < wiggle_errs)
                fail_flag = true;
            }
        }
    return(fail_flag);
    }

ReadTest(head,cyl)
    {
    return(number of read errors in reading 'max_sectors' sectors);
    }
```

Parameter List Format:

Type	Description
byte	14h - command code.
byte	Option select.
word	Maximum number of wiggle errors per head.
byte	Read error ratio. Do not fail for wiggle if: (number of read errors) * (ratio) > (number of wiggle errors)

Result Data Format:

Byte	Description
0	14h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6-15	- unused -

Window Centering

Optimize window strobe for Daytona.

Test Algorithm:

```
save_start_time();
set_DMA_rollover();
init_result_buffer();
for (head = 0 ; head = maxhead ; head++)
    {
    init_off_track(test_offtrack);
    init_transfer_count(transfer_count);
    for (zone = 0 ; zone = maxzone ; zone = zone + 3)
        {
        if (find_good_test_cyl == false);
            exit;
        while (good_result = false);
            {
            for (win_margin = 2 ; win_margin = -2 ; win_margin--)
                {
                set_transfer_count();
                set_margin(win_margin,test_offtrack);
                while (transfer_count > 0)
                    if (chk_test_time_out() = true)
                        exit
                    if (read_track() = fail)
                        {
                        if (servo_fault() = true)
                            {
                            inc_servo_error_count();
                            break;
                            }
                        if (fatal_error = true)
                            break;
                        inc_error_count();
                        }
                    adjust_transfer_count();
                    }
                good_result == chk_result();
                if (error_too_high = true)
                    reduce_stress(transfer_count,test_offtrack);
                if (error_too_low = true)
                    increase_stress(transfer_count,test_offtrack);
                }
            }
        }
    if (good_result = true)
        break;
    }
save_new_window_strobe();


find_good_test_cyl
    target_cylinder = max_cyl(zone);
    while (test = fail)
        seek();
        write_guard_band(off_track = -300);
        write_track(off_track = 200);
        test == verify_track(off_track = 200);
        if (test = fail)
            target_cylinder--;
```

Parameter List Format:

Type	Description
byte	15h - command code.
byte	Option select. 00 Test and write results. 01 Test, write results, and update the zone table. 03 Test, write results, update the zone table, and save the zone table to disk.

Result Data Format:

Byte	Description
0	15h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6	Result Flag 00h - Pass 32h - Bypass - use default values.
7-15	- unused -

Window Margin Test

This test will measure the window margin and threshold values on any set of zones and heads. Error limits may be specified as pass/fail criteria.

Test Algorithm:

```
for (head = 0;head in head map && head <= max head;head++)
    {
    for (zone = 0;zone in zone map && zone <= max zone;zone++)
        {
        seek(max cyl(zone));
        if(good track(zone) == false)
            exit(bad zone);
        for (strobe sigh = plus, cnt = 0;cnt < 2;strobe sign = minus, cnt++)
            {
            for (strobe val = max, flag = fail;flag = fail;strobe val--)
                flag = win mar read test(strobe val);
            save result(win mar val(strobe sign,strobe val));
            }
        seek(current cyl - 1);
        for(thold dir = up, cnt = 0;cnt < 2;thold dir = down, cnt++)
            {
            for(thold val = init val(thold dir), flag = fail;falg = fail;thold val = nextval(thold dir))
                flag = thold read test(thold val);
            save result(thold val, thold dir);
            }
        if (check results(results buffer) == fail)
            save results(error codes);
        }
    }
write results(ss results sectors, results buffer);

thold read test(thold value)
    {
    write trk(9999h = 2t - phase - 1 pattern);
    write trk(6666h = 2t - phase - 2 pattern);
    write trk(2727h = triplet - phase - 1 pattern);
    write trk(7272h = triplet - phase - 2 pattern);
    if (read 10000000 bits(current track, thold val) > max thold errors)
        return(fail);
    else
        return(pass);
    }

win mar read test(strobe val)
    {
    write trk(9999h = 2t - phase - 1 pattern);
    write trk(6666h = 2t - phase - 2 pattern);
    write trk(random - 1 pattern);
    write trk(random - 2 pattern);
    if (read 50000000 bits(current trk, strobe val) > max strobe errors)
        return(fail);
    else
        return(pass);
    }
```

Parameter List Format:

Type	Description
byte	17h - command code.
byte	Bit map of heads to use.
word	Bit map of zones to use.
byte	Lower margin fail limit.
byte	Upper margin fail limit.
byte	Lower threshold fail limit.
byte	Upper threshold fail limit.

Result Data Format:

Byte	Description
0	17h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6	Lower margin fail limit.
7	Upper margin fail limit.
8	Lower threshold fail limit.
9	Upper threshold fail limit.
10-15	- unused -

Result data is stored on cylinder -1 sectors 76-79. The following is a description of the data.

Result Sector Data Format:

Byte	Description
0	Error code. 00h No error. 70h Error seeking to target track. 71h Error finding a good track in the target zone. 72h Unrecovered servo fault. 73h Measurement out of range.
1	Head.
2	Zone.
3-4	Cylinder.
5	- unused -
6	Positive window margin.
7	Negative window margin.
8	Lower limit window margin.
9	Lower limit window margin fail flag. (out of range if > 0)
10	Upper limit window margin.
11	Upper limit window margin fail flag. (out of range if > 0)
12	Lower limit threshold.
13	Lower limit threshold fail flag. (out of range if > 0)
14	Upper limit threshold.
15	Upper limit threshold fail flag. (out of range if > 0)

Write Current Optimization

Optimize write current for Daytona.

Test Algorithm:

```
for (head = 0;head <= max head;head++)
    {
    for (zone = 0;zone <= max zone;zone++)
        {
        seek(max cyl(zone));
        if (good track(zone) == false)
            exit(bad zone);
        for (wr current = max current;wr current >= 0;wr current--)
            {
            for (off track = 100, win strobe = 1;err count < min errs || err count > max errs;adjust(off track, win strobe))
                err count = write read test(wr current, off track, win strobe);
            err cnt(wr current) = err count;
            }
        for (wr current opt = 0, wrc = 0;wrc <= 3;wrc++)
            if (err cnt(wrc) < err cnt(wr current opt))
                wr current opt = wrc;
        wr current(zone) = wr current opt;
        }
    wr current(head) = average(wr current(all zones));
    }
write results(ss result sectors, results buffer);

write read test(wr current, off track, win strobe)
    {
    while(off track needs adjustment && win strobe needs adjustment)
        {
        output port(wr current);
        output arizona(win strobe)
        write trk(1111h = erase pattern);
        step in(100);
        write trk(random - 1 pattern);
        write trk(random - 2 pattern);
        step out(off track);
        for (sec count = 0, err count = 0;sec count < max sectors && err count < max errors;sec count += sectors read)
            if(read track(durrent track) == ecc error)
                err count++;
        adjust stress(off track, win strobe,err count);
        }
    return(err count);
    }
```

Parameter List Format:

Type	Description
byte	16h - command code.
byte	Options.

Result Data Format:

Byte	Description
0	16h - command code.
2	Command error code.
3	Firmware error code.
4-5	Test time - S.
6	Optimization failure code. 1 Error seeking to desired track. 2 Error formatting desired track. 3 Error finding good track in given zone. 4 Error stepping back to center of desired track. 5 Error stepping in on desired track. 6 Error stepping out on desired track. 7 Error in convergence of optimal value:got too far offtrack. 8 Error in convergence of optimal value:too many unstressed errors. 9 Error in stepping back to center after optimum value found. 10 Unrecovered servo fault error reading data. 11 Test timeout.
7	Optimization failure head map. Bit n set if head n could not be optimized.
8-15	- unused -

Miscellaneous

Error Codes

00h	Pass Test.
01h	Average seek time is greater than the limit.
02h	Too many soft sector errors per drive.
03h	Too many soft sector errors per head.
07h	New hard defect found.
08h	Too many hard sector errors per drive.
09h	Too many hard sector errors per head.
0Ah	Too many hard sector errors per cylinder.
10h	Test time is greater than the limit.
11h	Power on restart was not able to recover.
12h	Test results were out of range.
13h	Head/serial number mismatch.
14h	Hardware/user head map mismatch.
19h	Unexpected termination.
20h	Invalid or out-of-range test number requested.
30h	Cannot latch the head.
32h	Cannot optimize read channel.
40h	Too many seek errors.
41h	Initial seek in a seek test failed.
48h	Servo error cannot recovered by retry.
4Ah	Sequencer error.
4Fh	Cannot seek to location after retry.
50h	Start stop test failed start due to time-out spinning up.
54h	Start stop test failed stop (latch stuck open).
58h	Start stop test failed start due to fatal error spinning up.
59h	Spinup error writing test sectors.
5Ah	Peachfuzz stress test failure.
60h	Error writing wedge during AM defect span search.
61h	Error calculating AM defect span.
62h	Invalid sequencer error from Koni.
63h	Selfscan defect list is full.
64h	Too many defects per cylinder.
65h	Too many defects per head.
66h	Too many defects per drive.
67h	Too many wiggle errors.
70h	Seek error in the window margin error test.
71h	No good track found in the window margin test.
72h	Unrecovered servo fault in the window margin test.
73h	Margin out of range in the window margin test.
80h	Failed while format track the drive.
0C0h	Selfscan defect sectors are full.
0C1h	P-list full.

D0h	Internal firmware error.
D1h	Invalid call to wrong selfscan overlay.
D2h	Could no read selfscan overlay.
D3h	Invalid checksum in selfscan overlay.
D4h	" for short sector format.
E0h	Sequencer error.
F0h	Selfscan password not found.
F1h	Error reading selfscan test sectors.
F2h	Error writing selfscan test sectors.
F3h	Error reading servo written system cylinder information.
F4h	Error reading wedge id sector.
F5h	Error switching wedge sets.
F7h	Error RRO/NRRO results were out of spec.
F8h	Error no good servo wedge set was found - servo verify.
F9h	Can't find index in servo verify.
FAh	Fatal servo error in servo verify.
FBh	No good wedge set found in servo verify for 1 set.
FCh	No good wedge set found in servo verify for 2 sets.

Defect Management

The Defect Lists

Three different lists are stored on system cylinder -2:

1. Primary defect list (P list) - this list contains the defects found in selfscan at the factory. Only the factory test software has the capability to define the P list. The P list contains the description for defects only. No information regarding their replacement is included.

2. Working list (W list) - typically, the W list is a union of the P and G lists, plus it contains all information necessary to locate the replacement to all defects.
Grown defect list (G list) - this list contains the defects found in the field during operation of the drive. All user's reassigned defects (i.e. with Reassign Blocks) and auto-reallocated defects are recorded in this list.
Note: In Lethal products, the G list is merged in with the W list, i. e. there is no separate G list.

3. Temporary list - During an update of the W list (a block reallocation for example) the old W list is stored to this area before any modifications are made to it. This allows for the recovery of the old list if an abort happens during the generation of the new list.

The host may access the P and G lists with the Read Defect Data SCSI command. The G list is decoded from information stored in the W list.

The W list is used by defect management whenever a logical-to-physical address conversion is called for. This list is not accessible with standard SCSI commands.

Defect List Storage

Up-to-date versions of the P and W lists are saved on the disk, only the W list needs to be resident in RAM during drive operation. Each defect list may require up to 2048 bytes of storage, therefore, a total of 4 sectors per list are reserved to hold the defect lists on a system track. See Chapter 5 on System Cylinder Layout for the location of the lists. Since the W list is limited to 2048 bytes in size, a maximum of 292 defects may be recorded on a Daytona drive.

Defect List Data Structure

The defect lists maintained and accessed by the defect management system consist of 7 byte defect entries. The P list contains only defect entries while the W list contains both defect and replacement cylinder information. The defect list structure is illustrated below.

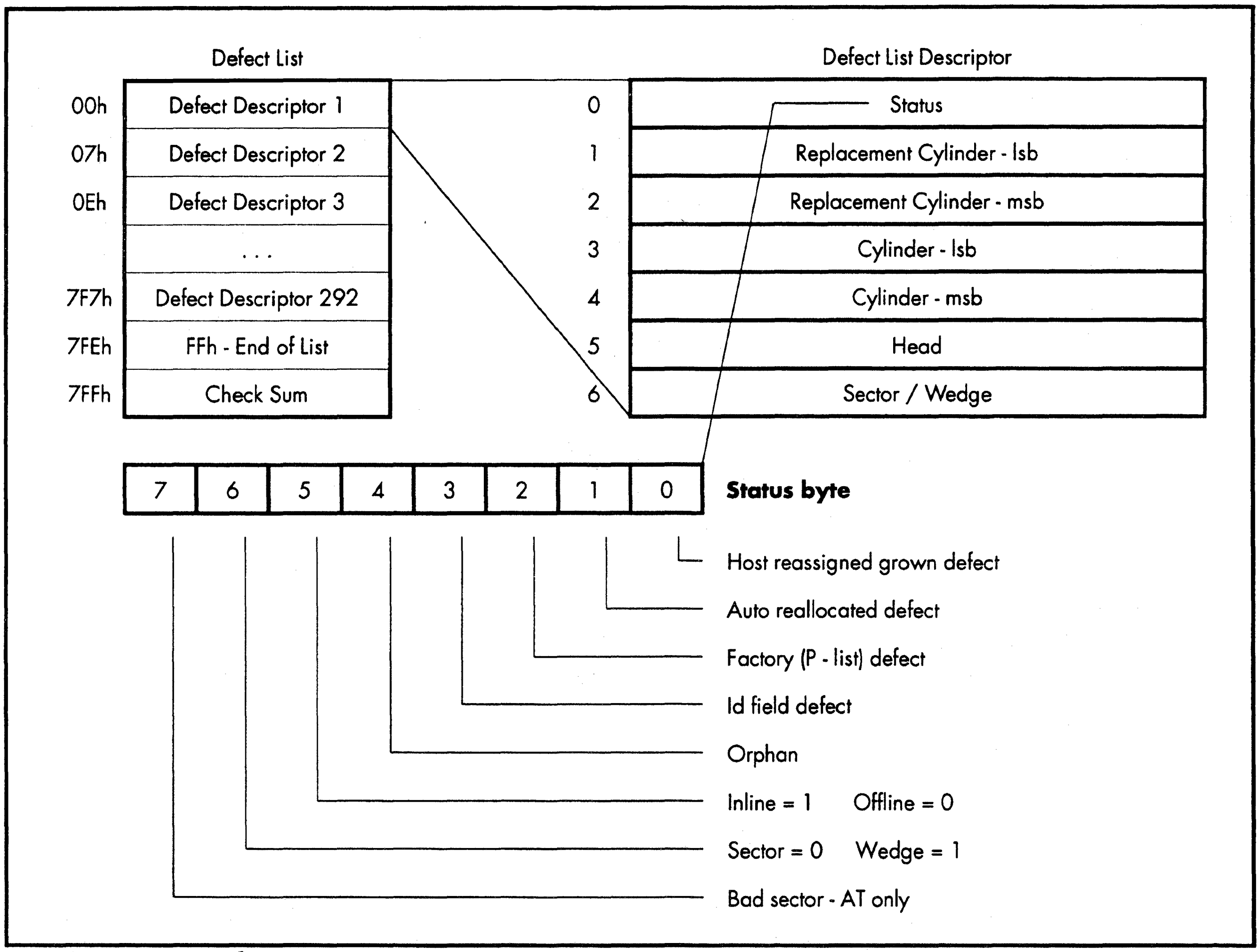


Figure 4.1 Defect List Data Structure

The end of list marker is placed after the last entry in the list.

The checksum is placed at the end of the list, and the empty area in the list is filled with zeros. When this byte is added to the rest of the bytes in the list, the **lsb** of the checksum will equal ascii "L".

Defect type is used to distinguish between P list entries (factory defect) and G list entries (auto reallocated and user reassigned).

Replacement type is used by defect management to find the correct physical sector for a given LBA.

Defective Sector Replacement Strategy

Lethal reserves one alternate sector per cylinder for defect sparing, and it utilizes two methods for sector replacement - inline and offline sparings.

Inline Sparing

Inline sparing is where a defective sector is replaced by the next immediate sector; all sectors thereafter within the same cylinder is shifted, logically, by one. (see figure 4.2) The access penalty is very small for inline replacement which is one sector time. Whenever possible, defects are spared with inline replacement at the factory. In the unlikely event where there are multiple defects on the same cylinder, additional spare sectors must be allocated from adjacent cylinders. This is defined as offline replacement. Accessing the defective sector requires a short seek and latency. All grown defects are offline spared during drive operation. However, the drive will attempt to inline spare all known defects when a Format Unit command is issued.

Offline Sparing

Offline sparing is where a defective sector is replaced by a spare sector located at the end of a cylinder. Defect management will try to replace the defective sector with a spare on the same cylinder. If this is not possible, as in the case of the spare is already in use, defect management will find a spare sector located on an adjacent cylinder. The disadvantage to this is the performance hit caused by the seek. Figure 4.2 containes an example of an offline spare.

Head 0	0	1	2	3	4	5	**Normal physical sector layout.**
Head 1	0	1	2	3	4	5	
Head 0	0	1	2	3	4	5	**Normal logical sector layout. The spare is located on the last sector of the last head.**
Head 1	6	7	8	9	10	Spare	
Head 0	0	1	~~2 bad~~	2	3	4	**Physical sector layout with an inline defect. Note how the physical sectors are now numbered.**
Head 1	0	1	2	3	4	5	
Head 0	0	1		2	3	4	**Logical sector layout with an inline defect. Note how the LBA's moved.**
Head 1	5	6	7	8	9	10	
Head 0	0	1	2	~~3~~	4	5	**Logical sector layout with LBA 3 and LBA 7 offline spared. LBA 3 is now located in the spare location at the end of the cylinder. LBA 7 has been moved to the spare location at the end of the next cylinder.**
Head 1	6	~~7~~	8	9	10	3	
Head 0	11	12	13	14	15	16	
Head 1	17	18	19	20	21	7	

Figure 4.2Inline and Offline Examples.

Orphans

An orphan occurs when a replacement sector goes bad. The replacement is assigned a new sector and the original replacement sector is tagged as an orphan in the defect list. It is no longer used. Defect management skips over defect entries that are tagged orphans.

Track Format - Bad Wedge ID

If a defect occurs in the ID field, there is a potential for losing 3 sectors with the ID per wedge format. This is handled by moving the ID field to a new location in the wedge. The only drawback to this is the ID field moves into a data area. This will sack a sector. The sacked sector will normally become an inline defect.

The following is an explanation on which sector gets sacked by moving the ID field.

The following is a normal track layout

W 0	ID 0	Sector 0	Sector 1a	**W 2**	ID 1	Sector 1b	Sector 2	**W 3**

```
If (ID 1 is bad)
        {
        minimum data size = ID field size in bytes quantized to the next highest multiple of 4;
        If (sector 1b data area > minimum size)
                {
                move ID 1 over 1 ID space;
                take space away from sector 1b and mark sector 1 bad;
                }
        Else
                {
                move ID 1 and sector 1b over 1 ID space;
                take space away from sector 2 and mark it bad;
                }
        set the flag bit in ID 0 to state that ID 1 will be late;
        }
```

Track layout - Sector 1b data area > minimum size (sector 1b sacked)

W 0	ID 0	Sector 0	Sector 1a	**W 2**		ID 1	1b	Sector 2	**W 3**

Track layout - Sector 1b data area < minimum size (sector 2 sacked)

W 0	ID 0	Sector 0	Sector 1a	**W 2**		ID 1	Sector 1b	Sector 2	**W 3**

The sectors that are labled as bad are entered into the Primary defect list and formatted as inline defects.

LBA to CHS Conversion

There are two entry points for performing the LBA to CHS conversion. Given an LBA, the caller invokes INIT_LBA_TO_CHS to initiate the conversion process.

INIT_LBA_TO_CHS determines the destination cylinder for the logical block and scans for known defects from the beginning of that cylinder. The function returns the CHS of the first valid sector plus a value indicating the number of consecutive data sectors starting from the first accessible sector.

It is left to the caller to decide how many sectors are actually required to complete its operation. If sectors are needed in addition to the first series of consecutive sectors, the caller uses the NEXT_LBA_TO_CHS function to locate the next series of sectors. NEXT_LBA_TO_CHS requires no input parameter and returns the same information as INIT_LBA_TO_CHS. Since media defects are sparse, there should be large number of contiguous cylinders with no defects for a typical drive. Basing on this fact, when a location on the disk is accessed, defect management firmware locates a range of "defect-free" cylinders in both directions of the current position. Once the range is defined, subsequent access made within the range will not require any reference to the defect list.

Auto Reallocation

Reallocation during read operation is processed on sector-by-sector basis. When a sector is determined to be defective by the read or write firmware, it is then subjected to write/verify test before it will reallocated. Using the data read from the defective sector, the drive writes to and reads the sector for up to ten times. If any of the ten tests fail, the defect is considered repeatable and the sector is reallocated. If all ten tests pass, then the failure is considered non-repeatable and the sector is left as is.

System Cylinders

General Information

8 cylinders on all drives are reserved for system and test usage. They contain drive configuration information, test data, and diskware. These cylinders are accessable only with physical address commands which are protected diagnostic commands. Customers cannot access these cylinders.

Data is repeated on all heads and is stored on two different cylinder locations for redundancy. Data is read from and written to these areas according to the firmware redundancy algorithm.

The reserved cylinders are assigned as follows:

Cylinder	Description
	Outer System Area.
- 1, -5	Test equipment data.
- 2, -6	System and firmware data.
- 3, -7	Diskware.
- 4, -8	Diskware.

See the table on page 5-131 for the system cylinder sector layouts.

Test Equipment Cylinder

Introduction

This cylinder is reserved for test data. It contains test parameters and data collected during production test.

Description of Sectors

The sector usage is as follows:

Sector	Description
0	Servo writer test data.
1	Analog scan test data.
2	Diskware data.
3	Post op test data.
4	Digital scan test data.
5	QA / DA test data.
6-13	Self scan results.
14-17	Self scan parameters.
18	Error Log.
19-26	Analog scan defect list.
27-34	Self scan defect list.
35-42	Digital scan defect list.
43	
44-45	Self scan RRO/NRRO results.
46-49	Wiggle test results.
50-53	Self scan servo verify bad servo map.
54-61	QA / DA defects.
62-63	
64-71	Self Optimization results.
72-73	Window centering results.

Test Data

The test data sectors (0-5) of all stations are written at the beginning of each test in the following format:

Byte	Format	Description
0-20	ASCII	Test station name. "LSW" - servo writer. "LAS" - analog scan. "LFT" - functional test. "LPO" - post op. "LDS" - digital scan.
30-109	ASCII	Date.
110-111	Binary	Exit code.
112-511	Binary	Reserved (filled with 0).

Self Scan Sectors

See the chapter on Self Scan for all the information.

Defect Lists

Each test that scans for defects maintains its own defect list. Each defect list is allocated a maximum of 8 sectors.

Analog and Digital Scan Defect List Format

The defect lists for analog scan and digital scan are saved in the following format:

Byte	Format	Description
0-13	ASCII	"LAS DEF LST " or "LDS DEF LST".
14-15	Binary	Number of defects (lsb, msb).
16-47	Binary	Head, cylinder skew, zone 0 to 16.

Byte	Description
16	Head skew for zone 0.
17	Cylinder skew for zone 0.
. . .	
46	Head skew for zone 15.
47	Cylinder skew for zone 15.

Byte	Format	Description
48-1023	Binary	Defect list - Analog Scan & Digital Scan (4 bytes per defect).

Byte	Description
48	Cylinder - lsb.
49	Cylinder - msb.
50	Head.
51	Sector.
. . .	

Self Scan Defect List Format

See the chapter on self scan for more details.

System / Firmware Cylinder

Introduction

This cylinder is reserved for system and firmware usage. It contains mode page information, configuration information, defect lists, and format information for the drive.

Description of Sectors

Sector	Description (see detailed descriptions below)
0	Saved mode select pages 1, 2, 20h, 37h, 38h, and 39h.
1	Saved mode select pages 3 and 4.
2-3	Configuration sectors.
4-7	Working defect list.
8-11	Primary defect list.
12-15	Temporary defect list area.
16	Format header bytes zone 0-15.
32	Apple operating system sector.
33	Drive usage sector.
35	Recal record.

Saved Mode Select Pages Sectors

The data stored on these sectors is only the changable part of the mode select pages. See the section on **Mode Pages** for more details.

Configuration Sectors

These sectors containes the drives configuration information such as the revision level, number of heads, etc. See the **Read Configuration** superset command for a detailed explanation of the data contained in this sector.

Defect List Sectors

These sectors contain the defect lists used during the drives normal operation. See the chapter on **Defect Management** for more information.

Format Header Sectors

In order for the firmware to format the drive, it needs to know the count byte information for the split sector data fields. Since there is no simple algorithm to generate this information, the count bytes must be stored in a table. There are 3 count bytes for each wedge ID and there are an average of 50 sectors per track for 16 zones. This add up to approximately 2400 bytes of information. Rather than allocate that much ROM for this purpose, we allocated 16 sectors on the system cylinder to hold this information. Each sector contains the count byte information for a particular zone.

Apple Operating System Sector

This sector is set aside for Apple computer. They will be using this for their own purposes. Since an end user can't access this sector, it is probably being used as a software key.

Sec	Cylinder -1, -5	Cylinder -2, -6	Cylinder -3, -7	Cylinder -4, -8
00	Servo Writer Test Data	Mode Select 1, 2, ...	Diskware Overlay 0	Diskware Overlay 2
01	Analog Scan Test Data	Mode Select 3, 4 (hex)	Resident	Self Scan Main Loop
02	Diskware Data	Configuration Pages		+
03	Post-op Test Data			Command Page 0
04	Digital Scan Test Data	Working Defect List		
05	QA/DA Test Data			
06	Self Scan Results			
07				
08		Primary Defect List		
09				
10				
11				
12		Temporary Defect List		
13				
14	Self Scan Cmd Parameters			
15				
16		Format ID Bytes - Zone 00		Diskware Overlay 3
17		Format ID Bytes - Zone 01		Self Scan
18	Error Log	...		Command Page 1
19	Analog Scan Defect List			
20				
21				
22				
23				
24				
25				
26				
27	Self Scan Defect List			
28				Diskware Overlay 4
29				Self Scan
30				Command Page 2
31		Format ID Bytes - Zone 15		
32		Apple op system sector	Diskware Overlay 1	
33		Drive Usage	Normal Operation	
34				
35	Digital Scan Defect List	Recal Record		
36				
37				
38				
39				
40				
41				Diskware Overlay 5
42				Self Scan
43				Command Page 3
44	RRO/NRRO Results			
45				
46	Wiggle Test Results			
47				
48				
49				
50	Servo Vfy - Bad Servo List			
...				
53				
54	QA/DA Defect List			
...				
61				
64	Self Optimization Results			
...				
71				
72	Window Centering Results			
73				

Figure 5-1System Cylinder Sector Layout.

Diskware

Introduction

The Daytona architecture has been designed to support diskware. Part of the Buffer memory may be used to load firmware from disk and the processor is able to execute the firmware directly from the buffer.

Memory Map

The memory map for Daytona is organized as follows:

Address Range	Length	Description
0000h - 7FFFh	32K	CPU ROM.
8000h - BFFFh	16K	Resident overlay.
C000h - DFFFh	8K	Normal operating overlay or selfscan overlay.
E000h - EFFFh	4K	Buffer access floating block 1.
E000 - E1FF		Sector buffer.
E200 - E5FF		Temporary buffer.
E600 - E7FF		Command history.
E800 - EFFF		W list.
F000h - FAFFh	2.8K	Buffer access floating block 2.
F000 - F1FF		Accuwrite table.
F200 - F3FF		Fixed buffer.
F400 - F6CC		Configuration page data.
F6CD - F7B8		Misc.
F7B9 - F7FF		Free.
F800 - F9FF		Reserved buffer.
FA00 - FA7F		Cache and serial debug port variables.
FA80 - FAFF		Free.
FB00h - FC7Fh	384	Asics.
FC80h - FEFFh	640	Internal ram.
FF00h - FFFFh	256	Internal special function registers.

Floating block 1 memory map

The following is a map of how the floating block CPU addresses map into the absolute address of the DRAM.

Address CPU	Dram	Description Normal Operation	Selfscan Defect Scan	Selfscan All Other
EFFF	1FFFF	w list buffer	W_list	
E800	1F800			
E7FF	1F7FF	command history		
E600	1F600			
E5FF	1F5FF	temporary buffer		
E200	1F200			
E1FF	1F1FF	sector buffer		sector buffer
E000	1F000			
EFFF	1EFFF	buffer ram	result data	result data
E000	1E000			
EFFF	1DFFF	buffer ram	command parameters	command parameters
E000	1D000			
EFFF	1CFFF		defect list	
E000	1C000			
FAFF	6AFF	configuration pages	configuration pages	configuration pages
		mode pages	mode pages	mode pages
F400	6400	variables	variables	variables
F3FF	63FF	fixed buffer		
F200	6200			
F1FF	61FF		selfscan variables	selfscan variables (F050-F1FF)
		accuwrite	result overlay buffer	result overlay buffer (F040-F04F)
F000	6000		command overlay buffer	command overlay buffer (F000-F03F)

The firmware is partitioned between the CPU ROM and the Diskware. The CPU ROM code contains all of the routines necessary to power up the drive and read the diskware into the Buffer. It also contains routines that allow the Diskware to be written to the disk via the host interface. All time critical code is located in the CPU ROM because the processor is able to execute CPU ROM code much faster than Diskware code. The Diskware code contains non time critical code that is not required for powering up the drive. The Diskware code also contains provisions to allow firmware bugs in the CPU ROM code to be corrected by mapping erroneous subroutines from CPU ROM into the Diskware.

There are two areas in the processor address space that may be used for diskware. The Daytona firmware uses only the address range 8000-DFFF for diskware, the other address range 0000-7FFF is always mapped to CPU ROM.

Diskware Code Organization

The diskware code space is partitioned into two parts, a resident part and an overlay part. The Resident diskware is loaded during the drive power up initialization and remains in memory while the drive is powered on. The Overlay diskware is loaded on an as needed basis, at present there are two overlays defined, one for SelfScan and one for normal operation.

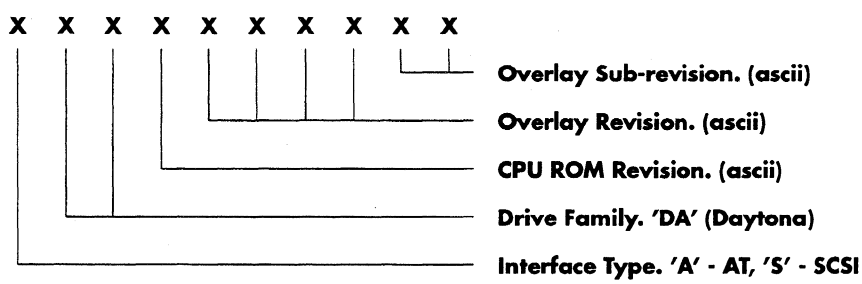
Address Range	Description
8000 - BFFF	Resident Diskware
	Vector Table
	Code
C000 - DFFF	Overlay Diskware
	1 - Normal Operating Code
	Vector table
	Code
	2 - Selfscan Code
	Vector Table
	Code

The Resident Diskware contains a vector table which is used by the CPU ROM code for accessing Diskware subroutines and data, and for mapping erroneous CPU ROM subroutines into Diskware subroutines. During power up initialization a default vector table is copied from CPU ROM, this is replaced by the actual vector table when the Diskware is loaded from disk.

The Daytona firmware has only rudimentary overlay management, in general because of speed considerations it is the responsibility of the calling routine to ensure that the required Overlay is loaded.

Overlay Integrity Verification

The integrity of the Resident and Overlay Diskware is checked by means of a checksum and a revision number. The checksum is the written in the last address of the overlay and is calculated such that the one byte sum of all the bytes in the overlay is zero. The revision number is used check compatibility between the Resident diskware, Overlay diskware and the CPU ROM code, the revision number is stored at the end of the overlay and has the following format.



Compatibility between the CPU ROM code and the Resident Diskware code is determined by comparing the CPU checksum stored in diskware.

Diskware Storage Requirements

The diskware is stored on reserved system cylinders in memory image format. A new configuration page (15) specifies where the overlays are stored on the system cylinders and where the overlays are loaded into the processor memory. Generally system cylinder information is stored in multiple places for redundancy, the overlay configuration page only specifies where the first copy of the diskware is stored. Redundant copies of the diskware are stored according to the firmware redundancy algorithm for system cylinder information. The Daytona firmware stores redundant system cylinder information on all physical heads in both the outer and inner system cylinder areas.

Configuration Page 15 - Overlay Page

Field Offset	Description
0	00h - Overlay 0 - Resident diskware.
1	Load address.
3	Number of sectors.
4	Cylinder.
6	Head.
7	Starting sector.
8	01h - Overlay 1 - Normal operating diskware.
9-15	Same fields as above.
16	02h - Overlay 2 - Self scan diskware - main loop - page 0.
17-23	Same fields as above.
24	02h - Overlay 3 - Self scan diskware - commands - page 1.
25-31	Same fields as above.
32	02h - Overlay 4 - Self scan diskware - commands - page 2.
33-39	Same fields as above.
40	02h - Overlay 5 - Self scan diskware - commands - page 3.
41-47	Same fields as above.
48	FFh - End marker.

Test Process

In the normal test process flow the diskware is written to the disk at the Servo Writer, to allow the manufacture of a generic HDA both AT and SCSI diskware is written to the disk. The Daytona Servo Writer records two sets of servo wedges on the disk and during Selfscan one of the two sets of wedges is selected based on the Selfscan Servo Verify function. To allow the use of the dual servo wedges a special disk format is used by the Servo Writer for writing the diskware. The Servo Writer writes the binary AT and SCSI Diskware Update files (described in the next section) to the data area of the disk. When Selfscan has selected which set of servo wedges to use, the appropriate diskware for the interface is written to the system cylinders using the normal disk format. The cylinder usage is as follows:

Cylinder	Description
761, 801	Servo Write SCSI Diskware Cylinder 4
760, 800	Servo Write SCSI Diskware Cylinder 3
759, 799	Servo Write SCSI Diskware Cylinder 2
758, 798	Servo Write SCSI Diskware Cylinder 1
753, 793	Servo Write AT Diskware Cylinder 4
752, 792	Servo Write AT Diskware Cylinder 3
751, 791	Servo Write AT Diskware Cylinder 2
750, 790	Servo Write AT Diskware Cylinder 1
749	Servo Write System/Test Data
-1, -5	Test Process Cylinder
-2, -6	System Firmware Cylinder
-3, -7	Diskware Cylinder 1
-4, -8	Diskware Cylinder 2

Diskware Update

The Daytona diskware may be updated through the drive interface using UPT or by using the Write Buffer Download Microcode command. UPT uses a disk file which contains the configuration page information and diskware in a binary format as shown below.

Field	Description
Overlay Number	Overlay number, 0 - Resident Memory address.
Load Address	Memory address of overlay.
Number of Sectors	Number of sectors in overlay.
Cylinder	Cylinder at which to store overlay on disk.
Head	Head at which to store overlay on disk.
Starting Sector	Starting sector at which to store overlay on disk.
	Repeat of above for all overlays.
FFh	End marker for overlays.
Fill	Fill 00 to end of sector (512 byte).
Overlay 0 Code	Code for Overlay 0.
	Code for rest of overlays.

UPT uses the information in the file to update the diskware cylinders directly using **write configuration** and **write physical** commands.

Miscellaneous Information

Programmable Trigger

Firmware allows certain conditions to generate a scope trigger. The conditions under which a trigger pulse is generated is controlled by Configuration Page 12 which consists of one byte. The eight bits are used to control whether a pulse is to be generated on an associated condition. If the bit is set and the condition occurs, a 1 microsecond (approximately) pulse is generated. Multiple trigger conditions may be specified at a time. The supported bits and associated conditions are as follows:

Bit	Description
0	—
1	Seek timeout
2	Seek fault
3	—
4	ECC error
5	Sequencer read/write error
6	Sequencer overrun/underrun
7	Sequencer timeout

As as example, to enable a pulse on either a seek timeout or ECC error, enter the following SCSIdiag command line: DEPB 0 18 WRCONF 12

The programmable scope trigger magically appears on microprocessor port P0.7.

Adaptive Cache For Daytona

DAYTONA has an adaptive cache scheme that offers a write segment of fixed size and 1 to 4 read segments. The maximum number of read segments is determined by the value of NUMSEGS for AT and byte 3 of mode page 37H for SCSI. The size of each read segment varies between minimum and maximum limits depending on the size of the request handled through that segment. The maximum limit is the number of sectors in the buffer that can be dedicated for read caching and the minimum limit equals the maximum limit divided by the number of read segments. So at any given moment different read segments can have different sizes.

DAYTONA cache scheme also allows:

i) partial hits : if only a part of the current request can be satisfied from the cached sectors in a given segment then a new prefetch is started on that segment to complete the request and

ii) anticipatory prefetch : if there is a cache hit and the sectors left in the segment are not enough to satisfy another contiguous request of the same size, a new prefetch is started on that segment.

Read algorithm

```
If cache enabled
    do cache scan
Switch
    case (cache hit and prefetch not needed and prefetch not active)
        do cache setup
        break;
    case (cache hit and prefetch not needed and prefetch active)
        extend current prefetch
        break;
    case (cache hit and prefetch needed)
        do cache setup
        start a seek to the first prefetch LBA
        calculate prefetch
        break;
    case (cache miss)
        allocate a new segment
        do cache setup
        start a seek to the first LBA requested
        calculate prefetch
        break;
Do main read loop ;
```

Cache allocation algorithm

```
If (prefetch is enabled)
    segment size = 2 x number of blocks
else
    segment size = number of blocks
segment size = max (segment size, minimum limit)
segment size = min (segment size, maximum limit)
Set reload register = rollover register of the previous segment + 1
Set rollover register = reload register + segment size
If rollover register > top of buffer dedicated to read segments
    reload register = bottom of buffer
    rollover register = reload register + segment size
Dirty all old segments overlapped by the new segment
```

Mode Pages

The following is information on the mode pages. Some of the pages contain information that can be configured by the customer, and this information is denoted by a value in the **Mask** column of the lists. If a bit is set to 1 in the mask, then that bit can be configured by the customer.

Page 0h **Apple disable unit attention page. (Apple default)**

Byte	Mask	Default	Description
0		80h	Page code.
1		02h	Page length.
2	10h	00h	Disable unit attention bit. Apple default = 10h.
3		00h	Not used.

Page 1h **Error recovery parameters.**

Byte	Mask	Default	Description
0		81h	Page code.
1		06h	Page length.
2	FFh	C0h	AWRE,ARRE. Apple default = 24h (Transfer block, Post error)
3	FFh	08h	Retry count.
4	FFh	10h	Maximum ecc error burst on which to perform corrections.
5		00h	2's complement value of microsteps offset from track center.
6		00h	Data strobe offset count.
7		00h	Recovery time limit in units of 10 ms.

Page 2h **Disconnect/reconnect control parameters.**

Byte	Mask	Default	Description
0		82h	Page code.
1		0Ah	Page length.
2	FFh	00h	On reads, how full buffer should be before reconnecting.
3	FFh	FFh	On writes, how empty buffer should be before reconnecting.
4		00h	Bus inactivity limit - msb.
5		00h	Bus inactivity limit - lsb.
6		00h	Disconnect time limit - msb.
7		00h	Disconnect time limit - lsb.
8		00h	Connect time limit - msb.
9		00h	Connect time limit - lsb.
10		00h	Reserved.
11		00h	Reserved.

Page 3h — **Direct access device format parameters.**

Byte	Mask	Default	Description
0		03h	Page code.
1		16h	Page length.
2		00h	Tracks per zone as defined in CCS - msb.
3	00h	02h	Tracks per zone as defined in CCS - lsb.
4		00h	Alternate sectors per zone - msb.
5		01h	Alternate sectors per zone - lsb.
6		00h	Alternate tracks per zone - msb.
7		00h	Alternate tracks per zone - lsb.
8		00h	Alternate tracks per volume - msb.
9		00h	Alternate tracks per volume - lsb.
10	00h	00h	Sectors per track - msb.
11		5Ch	Sectors per track - lsb.
12	00h	02h	Bytes per sector - msb.
13		00h	Bytes per sector - lsb.
14		00h	Interleave - msb.
15		01h	Interleave - lsb.
16		00h	Track skew factor - msb.
17	00h	28h	Track skew factor - lsb.
18		00h	Cylinder skew factor - msb.
19	00h	34h	Cylinder skew factor - lsb.
20		40h	Drive type definition bits. (40H = hard sector format).
21		00h	Reserved.
22		00h	Reserved.
23		00h	Reserved.

Page 4h — **Rigid disk drive geometry parameters.**

Byte	Mask	Default	Description
0		04h	Page code - 04h.
1		12h	Page length - 18.
2	00h	00h	Number of cylinders - msb.
3		06h	Number of cylinders - middle.
4		A8h	Number of cylinders - lsb.
5	00h	02h	Number of heads. (depends on model)
6		00h	Starting cylinder for write precompensation.
7		00h	Starting cylinder for write precompensation.
8		00h	Starting cylinder for write precompensation.
9		00h	Starting cylinder for reduced write current.
10		00h	Starting cylinder for reduced write current.
11		00h	Starting cylinder for reduced write current.
12		00h	Drive step rate.
13		00h	Drive step rate.
14		00h	Landing zone cylinder.
15		00h	Landing zone cylinder.
16		00h	Landing zone cylinder.
17		00h	Reserved.
18		00h	Reserved.
19		00h	Reserved.

Page 8h — **Cache page.**

Byte	Mask	Default	Description
0		88h	Page code.
1		0Ah	Page length.
2	05H	04h	Write cache enable (bit 4 = 1); Read Cache Disable (bit 0) = 0. Apple default = 0.
3		00h	None of the features in the Priority byte is supported.
4		00h	Disable Prefetch Transfer Length - msb is not supported.
5		00h	Disable Prefetch Transfer Length - lsb is not supported.
6		00h	Minimum number of blocks to prefetch - msb.
7		00h	Minimum number of blocks to prefetch - lsb.
8		00h	Maximum number of blocks to prefetch - msb.
9		00h	Maximum number of blocks to prefetch - lsb.
10		00h	Maximum Prefetch Ceiling - msb, not applicable.
11		00h	Maximum Prefetch Ceiling - lsb, not applicable.

Page Ch — **Notch page.**

Byte	Mask	Default	Description
0		0Ch	Page code.
1		16h	Page length.
2		80h	Device is notched, with physical boundaries.
3		00h	Reserved.
4		00h	Maximum number of notches - middle.
5		10h	Maximum number of notches - lsb.
6	FFh	00h	Active notch high not used.
7	FFh	00h	Active notch low by default is 0 (entire device).
8		00h	Starting cylinder - msb.
9	00h	00h	Starting cylinder - middle.
10		00h	Starting cylinder - lsb.
11		00h	Starting head 0.
12		00h	Ending at cylinder max minus 1.
13	00h	06h	Ending cylinder middle.
14		A7h	Ending cylinder lsb.
15	00h	01h	Max physical head. (depends on model)
16		00h	Indicate pages 3, 4 and C are notched.
17		00h	Indicate pages 3, 4 and C are notched.
18		00h	Indicate pages 3, 4 and C are notched.
19		00h	Indicate pages 3, 4 and C are notched.
20		00h	Indicate pages 3, 4 and C are notched.
21		00h	Indicate pages 3, 4 and C are notched.
22		10h	Indicate pages 3, 4 and C are notched.
23		18h	Indicate pages 3, 4 and C are notched.

Page 20h — **Apple serial number page.**

Byte	Mask	Default	Description
0		20h	Page code - 20h.
1		0Ch	Page length - 12.
2-13	00h		Serial number for Apple "############". (drive serial number - ascii)

Page 30h — **APPLE COMPUTER, INC page.**

Byte	Mask	Default	Description
0		30h	Page code.
1		14h	Page length.
2-21			ASCII string for Apple "APPLE COMPUTER, INC ".

Page 32h — **Auto power down page.**

Byte	Mask	Default	Description
0		B2h	Page code.
1		02h	Page length.
2	FFh	00h	Standby mode disabled with 0. Apple default = 5 (seconds)
3	FFh	00h	Shutdown mode disabled with 0.

Page 36h — **Apple diagnostic page.**

Byte	Mask	Default	Description
0		36h	Page code.
1		02h	Page length.
2	00h	C0h	Diagnostic mode field.
3	00h	00h	Burn in test/medium corrupted field.

Page 37h — **Tako (Cache) page.**

Byte	Mask	Default	Description
0		B7h	Page code.
1		0Eh	Page length.
2	3Fh	03h	Read cache, prefetch. Apple default = 7 (read cache,prefetch,prefetch only)
3	FFh	02h	Number of segments in cache. Must be 1 to 4. Apple default = 1.
4		00h	Minimum number of blocks to prefetch.
5		00h	Maximum number of blocks to prefetch.
6-15		00h	Unused.

Page 38h — **Auto power down page (Alternate).**

Byte	Mask	Default	Description
0		B8h	Page code.
1		04h	Page length.
2		00h	Reserved.
3	FFh	00h	Auto shutdown time (0 = disabled).
4		00h	Reserved.
5		00h	Reserved.

Page 39h — **Quantum page.**

Byte	Mask	Default	Description
0		B9h	Page code.
1		06h	Page length.
2	FBh	10h	Quantum configuration bits, RUEE set. Apple default = 8 (fill data pattern enabled)
3	DFh	00h	Quantum configuration bits.
4-7		00h	Reserved.

AT Defaults

Read cache	on
Prefetch	on
Prefetch only	off
Write cache	on
Accuwrite	off
Auto read	off
Auto transfer	off
Number of segments	2
Auto read allocation	on
Auto write allocation	on
Reallocate uncorrectable read	on
Retry count	8
ECC correction span	16
Auto-idle	on

Firmware Error Codes

Numerical Listing

Code		SCSI Key	SCSI Code	Description
00	00h			No error.
01	01h	4	01h	No index signal.
02	02h	4	98h	Timeout in settling.
03	03h	3	03h	Servo write fault.
04	04h	1	96h	Recovered seek timeout.
05	05h	4	06h	Recalibrate failure.
06	06h			
07	07h	1	06h	Recovered prediction update error.
08	08h	3	11h	Unrecoverable data field ecc error.
09	09h	1	17h	Data error recovered via retries.
10	0Ah	4	99h	Bump retry counter expired.
11	0Bh	6	9Ah	A target attempted to re-select.
12	0Ch	3	13h	Data field sync timeout.
13	0Dh	1	13h	Recovered data sync timeout.
14	0Eh	3	14h	No record found.
15	0Fh	1	14h	Recovered no record found.
16	10h	4	15h	Seek error.
17	11h	1	15h	Recovered seek error.
18	12h	3	16h	Data sync or marker not found.
19	13h	1	16h	Recovered data sync or marker not found.
20	14h	1	18h	Data error recovered with ECC, 2 consecutive syndromes equal.
21	15h	1	18h	Data error recovered with ECC, 2 consecutive syndromes equal (PERR).
22	16h	3	19h	Bad defect list.
23	17h	5	1Ah	Parameter overrun.
24	18h	4	1Bh	Synchronous transfer error.
25	19h	3	1Ch	No primary defect list found.
26	1Ah	5	9Bh	Invalid period or offset in synchronous message.
27	1Bh	5	9Ch	Active initiator attempted a select while disconnected.
28	1Ch	5	20h	Invalid command.
29	1Dh	5	21h	Invalid lba.
30	1Eh	5	22h	Invalid function requested.
31	1Fh	5	24h	Invalid bits set in CDB.
32	20h	5	25h	Invalid lun specified.
33	21h	5	26h	Invalid fields in parameters.
34	22h	4	9Dh	Motor never gets up to speed.
35	23h	6	29h	Reset occurred.
36	24h	6	2Ah	Mode select parameters were changed.
37	25h	1	18h	Data error recovered via ECC on last retry.
38	26h	1	18h	Data error recovered via ECC on last retry (PERR).
39	27h	3	32h	Defect list is full.
40	28h	4	40h	Ram error (most likely found in a diagnostic).
41	29h	B	40h	Ram parity error.
42	2Ah			
43	2Bh			
44	2Ch	4	43h	Message reject error.
45	2Dh	B	87h	Logical assertion (firmware consistency check) error.
46	2Eh	B	45h	Initiator did not reselect.
47	2Fh	4	9Fh	Internal ROM checksum error.
48	30h	B	47h	SCSI bus parity error.
49	31h	B	48h	Initiator detected error.

SCSI Key #	Description
1	Recovered errors.
2	Not Ready Errors.
3	Medium Errors.
4	Hardware Errors.
5	Illegal Requests
6	Unit Attention Errors.
B	Abort Errors.
E	Miscompare Errors.

Code		SCSI Key	SCSI Code	Description
50	32h	4	A0h	External PROM checksum error.
51	33h	3	80h	Error in writing to a system sector.
52	34h	3	81h	Error in reading from a system sector.
53	35h	4	9Eh	Motor drops out of legal speed range.
54	36h	1	9Eh	Recovered from motor out of speed.
55	37h	4	84h	Failure in writing to sequencer format table.
56	38h	4	85h	Reject of message that should never have been sent.
57	39h	4	86h	Unexpected error from sequencer.
58	3Ah	1	86h	Recovered unexpected sequencer error.
59	3Bh	B	0Bh	For Abort message.
60	3Ch	4	89h	Bad head amplifier.
61	3Dh	4	8Ah	Head read from ID not equal to selected head.
62	3Eh	5	8Ah	Invalid head specified.
63	3Fh	5	8Bh	Invalid cylinder specified.
64	40h	5	8Ch	Attempt by intruding initiator to select drive a second time
65	41h	5	8Dh	Bytes per block / bytes per sector gives a remainder.
66	42h	6	8Eh	Unexpected SIC interrupt occurred.
67	43h	2	B0h	Drive is up to speed and recalibrating.
68	44h	2	B1h	Drive is spinning up.
69	45h	2	B2h	Drive has not been told to spin up.
70	46h	3	32h	No more alternate sectors available.
71	47h	5	8Fh	Invalid sector specified.
72	48h	4	90h	Synchronous acknowledge error.
73	49h	4	91h	FIFO unload error.
74	4Ah	4	92h	FIFO load error.
75	4Bh	4	93h	FIFO predicted full error.
76	4Ch	4	94h	Undocumented SCSI asic error.
77	4Dh	4	95h	Sequencer timeout.
78	4Eh	1	95h	Recovered solo timeout.
79	4Fh	4	96h	Bump timeout.
80	50h	1	96h	Recovered bump error.
81	51h	E	1Dh	Read Buffer miscompare.
82	52h	4	A1h	Sequencer rollover register failure.
83	53h	4	02h	Seek timeout.
84	54h	1	02h	Recovered seek timeout.
85	55h	3	A3h	Failure in reading a sector in Reassign Blocks command.
86	56h	5	AEh	Some parameter(s) in the mode pages found to be bad during init.
87	57h	3	31h	Failed to write fill data pattern in Format Unit command.
88	58h	3	10h	ID ecc error.
89	59h	1	10h	Recovered ID ECC error.
90	5Ah	4	A7h	Offtrack timeout.
91	5Bh	3	12h	AM mark not found for ID field.
92	5Ch	1	12h	Recovered am mark not found for id field.
93	5Dh	3	AAh	Data read was written after reallocation of uncorrectable data.
94	5Eh	1	13h	Recovered reallocated uncorrectable data read.
95	5Fh	1	ABh	Requested format in Read Defect Data not available.
96	60h	1	98h	Recovered settle timeout.
97	61h	4	97h	Underrun error.
98	62h	1	97h	Recovered underrun error.
99	63h	4	06h	Recal fault - timed out getting burst data
100	64h	4	06h	Recal fault - AEQBM <= offset value
101	65h	4	06h	Recal fault - AEQBH > 0ffh
102	66h	4	06h	Recal fault - timed out getting burst data
103	67h	4	06h	Recal fault - AEQB burst data not in range
104	68h	4	06h	Recal fault - AEQBH burst data <= mid value
105	69h	4	06h	Recal fault - AEQBL burst data >= mid value
106	6Ah	4	06h	Recal fault - AEQBH and low are too close
107	6Bh	4	06h	Recal fault - if slope >= 256
108	6Ch	4	06h	Recal fault - no sam unparking
109	6Dh	4	06h	Recal fault - seek timeout in recal

Code	SCSI Key	SCSI Code	Description
110 6Eh	4	06h	Recal fault - timed out getting head offsets
111 6Fh	4	FFh	Physical heads found by recal does not match page 8.
112 70h	4	FFh	
113 71h	4	FFh	
114 72h	4	FFh	
115 73h	5	FFh	Bad sector number in format track descriptor list.
116 74h	5	FFh	Bad descriptor in format track descriptor list.
117 75h	3	14h	Bad Block Mark set for ID (AT).
118 76h	4	ACh	Latch stuck open.
119 77h	4	9Ch	Selfscan spin-up timeout failure.
120 78h	4	03h	WUS write fault.
121 79h	1	03h	Recovered WUS Write fault.
122 7Ah	3	82h	ID CRC error, internal continuing processing.
123 7Bh	1	82h	Recovered ID CRC error, internal continuing processing..
124 7Ch	3	81h	Diskware read error.
125 7Dh	3	81h	Resident checksum error.
126 7Eh	3	81h	Overlay checksum error.
127 7Fh	3	81h	Diskware short sector read error.
128 80h	4	81h	Rom/overlay diskware incompatability.
129 81h	4	81h	Rom/resident diskware incompatability.
130 82h	4	81h	Resident/overlay diskware incompatability.
131 83h	3	03h	Write fault. (write gate & wedge)
132 84h	1	03h	Recovered write fault. (write gate & wedge)
133 85h	5	C2h	Not applicable Apple command for Pioneer.
134 86h	2	C0h	Apple burn in test in progress.
135 87h	1	31h	Apple medium format corrupted.
136 88h	5	49h	Invalid message.
137 89h	6	3Fh	Microcode (diskware) has changed.
138 8Ah	3	FFh	Microcode download error.

Listing by Error Type

Recovered Errors

Code		Key	Description
04	04h	1	Recovered seek timeout.
07	07h	1	Recovered prediction update error.
09	09h	1	Data error recovered via retries.
13	0Dh	1	Recovered data sync timeout.
15	0Fh	1	Recovered no record found.
17	11h	1	Recovered seek error.
19	13h	1	Recovered data sync or marker not found.
20	14h	1	Data error recovered with ECC, 2 consecutive syndromes equal.
21	15h	1	Data error recovered with ECC, 2 consecutive syndromes equal (PERR).
37	25h	1	Data error recovered via ECC on last retry.
38	26h	1	Data error recovered via ECC on last retry (PERR).
54	36h	1	Recovered from motor out of speed.
58	3Ah	1	Recovered unexpected sequencer error.
78	4Eh	1	Recovered solo timeout.
80	50h	1	Recovered bump error.
84	54h	1	Recovered seek timeout.
89	59h	1	Recovered ID ECC error.
92	5Ch	1	Recovered am mark not found for id field.
94	5Eh	1	Recovered reallocated uncorrectable data read.
95	5Fh	1	Requested format in Read Defect Data not available.
96	60h	1	Recovered settle timeout.
98	62h	1	Recovered underrun error.
121	79h	1	Recovered WUS Write fault.
123	7Bh	1	Recovered ID CRC error, internal continuing processing..
132	84h	1	Recovered write fault. (write gate & wedge)
135	87h	1	Apple medium format corrupted.

Not Ready Errors

Code		Key	Description
67	43h	2	Drive is up to speed and recalibrating.
68	44h	2	Drive is spinning up.
69	45h	2	Drive has not been told to spin up.
134	86h	2	Apple burn in test in progress.

Medium Errors

Code		Key	Description
03	03h	3	Servo write fault.
08	08h	3	Unrecoverable data field ecc error.
12	0Ch	3	Data field sync timeout.
14	0Eh	3	No record found.
18	12h	3	Data sync or marker not found.
22	16h	3	Bad defect list.
25	19h	3	No primary defect list found.
39	27h	3	Defect list is full.
51	33h	3	Error in writing to a system sector.
52	34h	3	Error in reading from a system sector.
70	46h	3	No more alternate sectors available.
85	55h	3	Failure in reading a sector in Reassign Blocks command.
87	57h	3	Failed to write fill data pattern in Format Unit command.
88	58h	3	ID ecc error.
91	5Bh	3	AM mark not found for ID field.
93	5Dh	3	Data read was written after reallocation of uncorrectable
117	75h	3	Bad Block Mark set for ID (AT).
122	7Ah	3	ID CRC error, internal continuing processing.
124	7Ch	3	Diskware read error.
125	7Dh	3	Resident checksum error.
126	7Eh	3	Overlay checksum error.
127	7Fh	3	Diskware short sector read error.
131	83h	3	Write fault. (write gate & wedge)
138	8Ah	3	Microcode download error.

Hardware Errors

Code		Key	Description
01	01h	4	No index signal.
02	02h	4	Timeout in settling.
05	05h	4	Recalibrate failure.
10	0Ah	4	Bump retry counter expired.
16	10h	4	Seek error.
24	18h	4	Synchronous transfer error.
34	22h	4	Motor never gets up to speed.
40	28h	4	Ram error (most likely found in a diagnostic).
44	2Ch	4	Message reject error.
47	2Fh	4	Internal ROM checksum error.
50	32h	4	- unused -
53	35h	4	Motor drops out of legal speed (3664 rpm) range.
55	37h	4	Failure in writing to sequencer format table.
56	38h	4	Reject of message that should never have been sent.
57	39h	4	Unexpected error from sequencer.
60	3Ch	4	Bad head amplifier.
61	3Dh	4	Head read from ID not equal to selected head.
72	48h	4	Synchronous acknowledge error.
73	49h	4	FIFO unload error.
74	4Ah	4	FIFO load error.
75	4Bh	4	FIFO predicted full error.
76	4Ch	4	Undocumented SPICY error.
77	4Dh	4	SOLO timeout.
79	4Eh	4	Bump timeout.
82	52h	4	Sequencer rollover register failure.
83	53h	4	Seek timeout.
90	5Ah	4	Offtrack timeout.
97	61h	4	Underrun error.
99	63h	4	Recal fault - timed out getting burst data
100	64h	4	Recal fault - AEQBM <= offset value
101	65h	4	Recal fault - AEQBH > 0ffh
102	66h	4	Recal fault - timed out getting burst data
103	67h	4	Recal fault - AEQB burst data not in range
104	68h	4	Recal fault - AEQBH burst data <= mid value
105	69h	4	Recal fault - AEQBL burst data >= mid value
106	6Ah	4	Recal fault - AEQBH and low are too close
107	6Bh	4	Recal fault - if slope >= 256
108	6Ch	4	Recal fault - no sam unparking
109	6Dh	4	Recal fault - seek timeout in recal
110	6Eh	4	Recal fault - timed out getting head offsets
111	6Fh	4	Physical heads found by recal does not match page 8.
118	76h	4	Latch stuck open.
119	77h	4	Selfscan spin-up timeout failure.
120	78h	4	WUS Write fault.
128	80h	4	Rom/overlay diskware incompatability.
129	81h	4	Rom/resident diskware incompatability.
130	82h	4	resident/overlay diskware incompatability.

Illegal Requests

Code		Key	Description
23	17h	5	Parameter overrun.
26	2Ah	5	Invalid period or offset in synchronous message.
27	2Bh	5	Active initiator attempted a select while disconnected.
28	2Ch	5	Invalid command.
29	2Dh	5	Invalid lba.
30	2Eh	5	Invalid function requested.
31	2Fh	5	Invalid bits set in CDB.
32	30h	5	Invalid lun specified.
33	31h	5	Invalid fields in parameters.
62	3Eh	5	Invalid head specified.
63	3Fh	5	Invalid cylinder specified.
64	40h	5	Attempt by intruding initiator to select drive a second time
65	41h	5	Bytes per block / bytes per sector gives a remainder.
71	47h	5	Invalid sector specified.
86	56h	5	Some parameter(s) in the mode pages found to be bad during init.
115	73h	5	Bad sector number in format track descriptor list.
116	74h	5	Bad descriptor in format track descriptor list.
133	85h	5	Not applicable Apple command for Pioneer.
136	88h	5	Invalid message.

Unit Attention Errors

Code		Key	Description
11	0Bh	6	A target attempted to re-select.
35	23h	6	Reset occurred.
36	24h	6	Mode select parameters were changed.
66	42h	6	Unexpected SIC interrupt occurred.
137	89h	6	Microcode (diskware) has changed.

Abort Errors

Code		Key	Description
41	29h	B	Ram parity error.
45	2Dh	B	Logical assertion (firmware consistency check) error.
46	2EH	B	Initiator did not reselect.
48	30h	B	SCSI bus parity error.
49	31h	B	Initiator detected error.
59	3Bh	B	For Abort message.

Miscompare Errors

Code		Key	Description
81	51h	E	Read Buffer miscompare.

Debug Port

Introduction

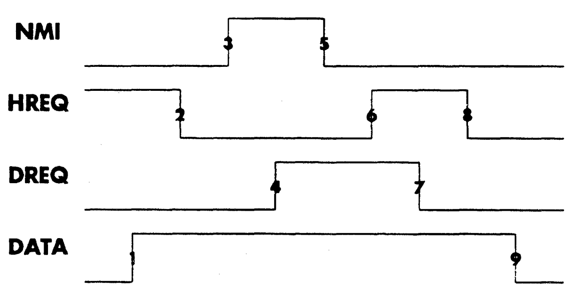
A debug port has been implemented on Daytona products by utilizing the four vendor specific test pins.

The following pages contain information about the commands and data structures used to implement this function.

See the serial debug port specification for information about the hardware aspects of implementing this function.

Attention Protocol

The following diagram shows the signal sequence necessary to initiate an attention phase with the Daytona drives.



1. DATA is asserted by the host.
2. HREQ is de-asserted by the host.
3. NMI is asserted by the host.
4. DREQ is asserted by the drive.
5. NMI is de-asserted by the host.
6. HREQ is asserted by the host.
7. DREQ is de-asserted by the drive.
8. HREQ is de-asserted by the host.
9. DATA is de-asserted by the host.

After a command has been completed, the line signals should be reverted back to what they were before the command was issued.

Command Descriptions

Command Summary

Opcode	Command
00	Read Micro Memory
01	Write Micro Memory
04	Call Subroutine
0C	Debug Mode Control
1C	Peek Buffer
1D	Poke Buffer
C1	Run Mode Control
C2	Configuration Control
C3	Read Registers
C4	Write Registers

Call Subroutine

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Subroutine Address - msb							
3	Subroutine Address - lsb							
4	Unused							
5	Unused							
6	Sub Opcode = 04h							
7	R1 - A							
8	R0 - X							
9	Reserved = 0						0	0

This command causes the drive to call the subroutine whose address is specified in **subroutine address.**

This command also allows for the implementation of quick and simple commands without the need to modify the command decode tables and the firmware documentation. This will also helps to reduce the proliferation of commands.

R1, **R0** contain values that are loaded into the cpu's **AX** register before the routine is called.

Configuration Control

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = C2h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						0	0

This is the command that is used to define the structure used for determining the configuration of all breakpoints, watchpoints, enables, disable, watch statements and some miscellaneous control bytes. The structure is sent to the drive following the command block and is defined as follows.

byte	**debug flags**	
	bit 0	-
	bit 1	breakpoint active
	bit 2	watchpoint active
	bit 3	-
	bit 4	-
	bit 5	register display active
byte	**break flag**	
	bit 0	breakpoint 1 is active
	bit 1	breakpoint 2 is active
	bit 2	breakpoint 3 is active
	bit 3	breakpoint 4 is active
	bit 4	watchpoint 1 is active
	bit 5	watchpoint 2 is active
	bit 6	watchpoint 3 is active
	bit 7	watchpoint 4 is active
struct * 4	**break point**	
struct * 4	**watch point**	

Note: - see below for structure definitions.

The individual structure elements are:

break point

word	Address to break on

watch point

word	Address of data to break on
word	Data value used for comparison
byte	flag
	bit 0 = any change
	bit 1 = equal to compare data
	bit 2 = not equal to compare data
	bit 3 = greater than compare data
	bit 4 = less than compare data
	bit 7 = word compare instead of byte

Debug Mode Control

<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td colspan="8">Opcode = 0FFh</td></tr>
<tr><td>1</td><td colspan="3">LUN = 0</td><td colspan="4">Reserved = 0</td><td>E</td></tr>
<tr><td>2</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>3</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>4</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>5</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>6</td><td colspan="8">Sub Opcode = 0Ch</td></tr>
<tr><td>7</td><td colspan="8">Reserved = 0</td></tr>
<tr><td>8</td><td colspan="8">1</td></tr>
<tr><td>9</td><td colspan="6">Reserved = 0</td><td>0</td><td>0</td></tr>
</table>

This command is used to enter or to exit from the debug mode. This command, with E set to 1, is to be issued before any of the other Sub Op code 0Cxh commands can be used. To return to normal execution, this command with E set to 0, must be sent. On entry, all break/watchpoints will be cleared. The exit option will cause all breakpoints to be cleared and any replaced instructions will be restored to their original values.

E = 1 - Enable the debug mode.
E = 0 - Disable the debug mode

This command will return two bytes indicating the debug code revision in the drive. The first byte is the major revision number indicating any changes in the data structures that are sent from the host (refer to the Configuration Control command). The second revision number indicates firmware revisions that do not affect the data structure for the specific major revision number.

Peek Buffer

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Page Number							
4	Buffer Memory Address - msb							
5	Buffer Memory Address - lsb							
6	Sub Opcode = 1Ch							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						0	0

This command is used to read the contents of the cache buffer RAM.

CDB byte	**Description**
Buffer Memory Address	The starting buffer location.
Transfer length	Specifies the number of bytes to be written.

The address range is from 00000h thru 1FFFFh.

Poke Buffer

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Page Number							
4	Buffer Memory Address - msb							
5	Buffer Memory Address - lsb							
6	Sub Opcode = 1Dh							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						0	0

This command is used to write to the cache buffer RAM.

CDB byte	**Description**
Buffer Memory Address	The starting buffer location.
Transfer length	Specifies the number of bytes to be written.

The address range is from 00000h thru 1FFFFh.

Read Micro Memory

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Microprocessor Memory Address - msb							
5	Microprocessor Memory Address - lsb							
6	Sub Opcode = 00h							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						0	0

This command is used to read the memory in the microprocessor's memory address space.

CDB byte	Description
Microprocessor memory address	Address to start reading at.
Transfer length	Specifies the number of bytes to be read.

Read Registers

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = C3h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						0	0

This command will return 22 bytes of register data to the host.

The data format is as follows:

Byte	Register Description
0-1	PC
2-3	PSW
4-5	RP7
6-7	RP6
8-9	RP5
10-11	RP4
12-13	RP3
14-15	RP2
16-17	RP1
18-19	RP0
20-21	SP

Run Mode Control

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0			Mode	
2	Reserved = 0							
3	Reserved = 0							
4	Start Address - msb							
5	Start Address - lsb							
6	Sub Opcode = C1h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						0	0

This command set the mode of execution that the debugger will use. Controls for information that is returned and where to start execution from is defined in this command.

Mode	**Action**
00	Continue executing from the last breakpoint.
01	Reserved.
10	Single-step one time then break.
11	Run from the starting address in cdb 4 & 5.

Start Address The address to start executing from if the Mode is set to 11b.

After the execution of this command, the completion will be pending until either a breakpoint, watchpoint, or the next instruction (if single-step mode is selected) occurs. The following describes the data that is returned upon the occurrence of the event:

byte	**break_flag**	
	bit 0	breakpoint 1 occurred
	bit 1	breakpoint 2 occurred
	bit 2	breakpoint 3 occurred
	bit 3	breakpoint 4 occurred
	bit 4	watchpoint 1 occurred
	bit 5	watchpoint 2 occurred
	bit 6	watchpoint 3 occurred
	bit 7	watchpoint 4 occurred
word	**program counter value when the event occurred.**	
word	**psw**	**Processor status word**
word	**rp7**	**Register pair 7 (HL)**
word	**rp6**	**Register pair 6 (DE)**
word	**rp5**	**Register pair 5 (UP)**
word	**rp4**	**Register pair 4 (VP)**
word	**rp3**	**Register pair 3**
word	**rp2**	**Register pair 2**
word	**rrp1**	**Register pair 1 (BC)**
word	**rp0**	**Register pair 0 (AX)**

Write Micro Memory

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Microprocessor Memory Address - msb							
5	Microprocessor Memory Address - lsb							
6	Sub Opcode = 01h							
7	Transfer Length - msb							
8	Transfer Length - lsb							
9	Reserved = 0						0	0

This command is used to write the memory in the microprocessor's memory address space.

CDB byte	Description
Microprocessor memory address	Address to start writing at.
Transfer length	Specifies the number of bytes words to be written.

Write Registers

	7	6	5	4	3	2	1	0
0	Opcode = 0FFh							
1	LUN = 0			Reserved = 0				
2	Reserved = 0							
3	Reserved = 0							
4	Reserved = 0							
5	Reserved = 0							
6	Sub Opcode = C4h							
7	Reserved = 0							
8	Reserved = 0							
9	Reserved = 0						0	0

This command will send 22 bytes of register data to the cpu.

The data format is as follows:

Byte	**Register Description**
0-1	PC
2-3	PSW
4-5	RP7
6-7	RP6
8-9	RP5
10-11	RP4
12-13	RP3
14-15	RP2
16-17	RP1
18-19	RP0
20-21	SP

Index

A

B

C

Z