WD1001

Winchester Disk Controller

OEM Manual

Document 80-031003-00 A1

Michael J. Friese

WESTERN DIGITAL
C O R P O R A T I O N

SECTION 1 INTRODUCTION

1.1 General Description

The WD1001 is a stand-alone, general purpose Winchester Controller board with error correction (ECC) capabilities. It is designed to interface up to four Winchester disk drives to a host processor. There are two versions of the board, the WD1001-55/85 and the WD1001-05. The WD1001-55/85 is a full featured version that is the same size as the original WD1000. This version can operate 8"drives (-85) or 5-1/4" drives (-55). The WD1001-05 is a smaller version of the board that can be mounted directly to most 5-1/4"drives.

The drive signals are based upon the floppy look-alike interface available on the Shugart Associates' SA1000, the Seagate Technology ST506, and other compatible drives. All necessary buffers and receivers/drivers are included on the board to allow direct connection to the drive. Four 20 pin radial connectors are provided for data. Either a 34 pin (5-1/4"drive) or a 50 pin (8" drive) connector is provided for drive control.

Communications to and from the host computer are made via a separate computer access port. This port consists mainly of an 8 bit bi-directional bus and appropriate control signals. All data to be written to or read from the disk, status information, and macro commands are transferred via this 8 bit bus. An on board sector buffer allows data transfers to the host computer independent of the actual data transfer rate of the drive.

The WD1001 is based upon a proprietary chip set called the WD1100, specifically designed for Winchester Control.

1.2 Features

* Single 5V Supply
* Built-in Data Separator
* Built-in Write Precompensation logic
* Data rates up to 5 Mbits/sec
* Control for up to 4 drives
* Control for up to 8 R/W heads
* 1024 cylinder addressing range
* 256 sector addressing range
* CRC generation/verification
* Automatic formatting
* 128,256, or 512 bytes per sector (User selectable)
* Unlimited sector interleave capability
* Multiple sector reads and writes
* Overlap seek capability
* Implied seek on all commands
* Automatic retries on all errors
* Automatic restore and re-seek on seek error
* Error correction on data field errors
* Diagnostic reads and writes for checking error correction
* 8 Bit Host Interface

1.3 Specifications

Encoding method:	MFM
Cylinders per Head:	Up to 1024
Sectors per Track:	Up to 256 (512 byte sec)
Heads:	8
Drive Selects:	4
Step rate:	35 uS to 7.5 mS (0.5 mS increments)
Data Transfer Rate:	4.34 Mbits/sec (SA1000) 5.000 Mbits/sec (ST506)
Write Precomp Time:	12 nanoseconds
CRC Polynomial:	$X^{16}+X^{12}+X^5+1$
ECC Polynomial:	$X^{32}+X^{28}+X^{26}+X^{19}+X^{17}+X^{10}+X^6+X^2+1$
Reciprocal ECC Polynomial:	$X^{32}+X^{30}+X^{26}+X^{22}+X^{15}+X^{13}+X^6+X^4+1$
Miscorrection Probability:	256 byte sector- <8.0 E-6 512 byte sector- <1.5 E-5
Non-detection Probability	- 2.3 E-10
Correction Span:	5 bits
Sectoring:	Soft
Host Interface:	8 Bit bi-directional Bus
Drive Capability:	10 LS Loads
Drive Cable Length:	10 ft. (3 M) max.
Host Cable Length:	3 ft. (1 M) max.
Power Requirements:	+5V +-5%, 3.0A Max (2.5A typ.)
Ambient Operating Temperature:	0 C to 50 C (32 F to 122 F)
Relative Humidity:	20% to 80%
MTBF:	10,000 POH
MTTR:	30 minutes
Length (-55/85):	9.9 in. (24.9 cm)
Length (-05):	8.5 in. (21.6 cm)
Width (-55/85):	6.8 in. (17.1 cm)
Width (-05):	5.75 in. (14.6 cm)
Height (All):	0.75 in. (1.9 cm)
Mounting Centers (-55/85):	6.375 x 9.375 in. (16 x 23.6 cm)
Drive Mount Centers (-05):	3.12 in. (7.9 cm)

1.4 Simplified System Block Diagram
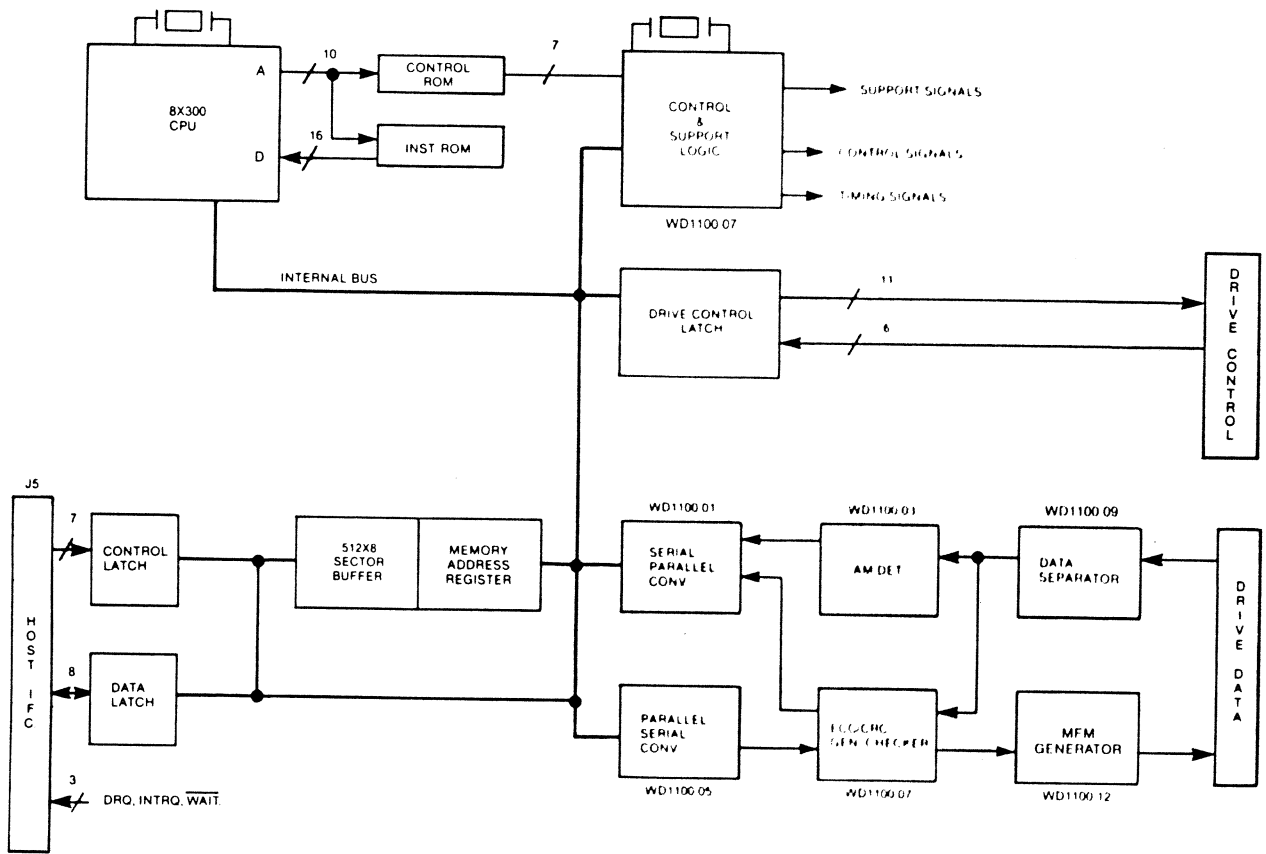
SECTION 2 INTERFACE CONNECTORS

2.1 Organization

The WD1001 has seven on board connectors. These connectors consist of a power connector, a host interface connector, a drive control connector, and four high speed data connectors.

The drive control cable is daisy-chained to each of the four drives. Although there are places on the board for two drive control connectors, only one would normally be installed for any particular configuration.

The drive data connectors carry differential signals and are radially connected. Up to four drives can be accommodated by the WD1001.

The host interface connector provides interface signals that are compatible with most microprocessors and many mini-computers with very little interfacing effort.

2.2 Host Interface Signals

The Host Interface connector (J5) consists of an eight bit bi-directional bus, three bit address bus, and seven control lines. All commands, status, and data are transferred over this bus. The control signals are as follows:

2.2.1 DAL0-DAL7

8 bit bi-directional Data Access Lines. These lines remain in a high-impedance state whenever the CS-line is inactive.

2.2.2 CS-

When Card Select- is active along with RE- or WE-, Data is read or written via the DAL bus. CS- **must** make a transition for each byte read from or written to the task file.

2.2.3 WE-

When Write Enable- is active along with CS-, the host may write data to a selected register of the WD1001.

2.2.4 RE-

When Read Enable- is active along with CS-, the host may read data from a selected register of the WD1001.

2.2.5 A2-A0

These three Address lines are used to select one of eight registers in the Task File. They must remain stable during all read and write operations.

2.2.6 WAIT-

Upon receipt of a CS-, the WAIT- line may go active. It returns to the inactive state when the DAL lines are valid on a read, or data has been accepted on a write. The function of this line should not be confused with the BUSY bit in the status register. The WAIT- signal is intended only as a bus synchronization mechanism. This signal is **not** an optional signal.

2.2.7 INTRQ

The INTerrupt ReQuest Line is activated whenever a command has been completed. It is reset to the inactive state when the Status Register is read, or a new command is loaded via the DAL lines. INTRQ can also reset by accessing the Sector Number register.

2.2.8 DRQ

The Data ReQuest line is activated whenever the sector buffer contains data to be read by the host, or is awaiting data to be loaded by the host. This line is reset whenever the Data register is read from or written to. The DRQ line will continue to toggle until the buffer is exhausted or until a write or read is performed on the Cylinder Low register.

2.2.9 MR-

The Master Reset line initializes all internal logic on the WD1001. In order to insure a complete reset, this line must be asserted for at least 1uS. The Sector Number, Cylinder Number and SDH registers are cleared, stepping rate is set to 7.5 mS, Write Precomp is set to cylinder 128, and Sector Count is set to 1. The DRQ and INTRQ lines are reset.

2.2.10 GND

All even numbered pins on this connector are to be used as signal grounds. Power grounds are available on J6, pin 1 of the -55/85 and on pins 2 and 3 of the -05.

2.2.11 +5V

Eight power pins for regulated +5 volts on the -55/85 only. This power input is also available on J6, pin 3 of the -55/85 and pin 4 of the -05.

2.2.12 Host Interface Connector

The Host Interface connector (J5) for the -55/85 is a 50 pin card edge connector on tenth-inch centers that mates with Burndy #FRE50B-3. The Host Interface connector for the -05 is a 40 pin vertical header that mates with a Burndy #FRS40BS. The cable used should be flat ribbon cable or twisted pair with a length of less than three feet. The connector pin-outs are as follows:

Signal Ground	Signal Pin	Signal Name
2	1	DAL0
4	3	DAL1
6	5	DAL2
8	7	DAL3
10	9	DAL4
12	11	DAL5
14	13	DAL6
16	15	DAL7
18	17	A0
20	19	A1
22	21	A2
24	23	CS-
26	25	WE-
28	27	RE-
30	29	WAIT-
32	31	Not Connected
34	33	Not Connected
36	35	INTRQ
38	37	DRQ
40	39	MR-
	41	Not Connected *
	42	Not Connected *
	43-50	+5V *

* WD1001-55 and 85 only

2.3 Drive Control Signals

The Drive Control connector (J7 and J8 on -55/85, J7 on -05) is a (relatively) low speed bus that is daisy chain connected to each of up to four drives in the system. To properly terminate each TTL level output signal from the WD1001, the last drive in the daisy chain should have a 220/330 ohm line termination resistor pack installed. All other drives should have no termination. Drive Control Signals are as follows:

2.3.1 RWC-

When the Reduce Write Current line is activated with Write Gate, a lower write current is used to compensate for greater bit packing density on the inner cylinders. The RWC- line is activated when the cylinder number is greater than or equal to four times the contents of the Write Precomp Register.

2.3.2 Write Gate-

This output signal allows data to be written to the disk.

2.3.3 Seek Complete-

Informs the WD1001 that the head of the selected drive has reached the desired cylinder and has stabilized. Seek Complete is not checked after a SEEK command, thus allowing overlapped seeks.

2.3.4 Track 000-

Indicates that the R/W heads are positioned on the outermost cylinder. This line is sampled immediately before each step is issued.

2.3.5 Write Fault-

Informs the WD1001 that some fault has occurred on the selected drive. The WD1001 will not execute commands when this signal is true.

2.3.6 HS0-HS2-

Head Select lines are used by the WD1001 to select a specific R/W head on the selected drive.

2.3.7 Sector-

For hard sectored drives, this line is used to indicate the sector boundaries during formatting. Note that this line is not used unless special PROMs are installed to handle hard sectored drives.

2.3.8 Index-

Is used to indicate the index point for synchronization during formatting and as a time out mechanism for retries. This signal should pulse once each rotation of the disk.

2.3.9 Ready-

Informs the WD1001 that the desired drive is selected and that its motor is up to speed. The WD1001 will not execute commands unless this line is true.

2.3.10 Step-

This line is pulsed once for each cylinder to be stepped. The direction of the step will be determined by the Direction In- line. The step pulse period is determined by the internal stepping rate register during implied seek operations or explicitly during Seek and Restore commands. During auto restore, the step pulse period is determined by the Seek Complete- time from the drive.

2.3.11 Direction In-

Determines the direction of motion of the R/W head when the step line is pulsed. A high on this line defines the direction as out and a low defines direction as in.
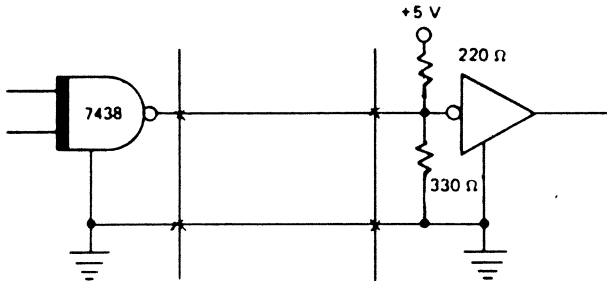
2.3.12 DS1-DS4-

These four Drive Select lines are used to select one of four possible drives.

2.3.12 Control Driver/Receiver

The control lines have the following electrical specifications:

True= 0.0 V to 0.4 V at I_{in} = 40 ma.(max)
False= 2.5 V to 5.25 V at I_{in} = - 0 ma.(open)

2.3.14 50 Pin Drive Control Connector

This Drive Control connector (J8 on -85 only) is a 50 pin vertical header on tenth-inch centers that mates with Burndy #FRS50BS. The cable used should be flat ribbon cable or twisted pair with a length of less than 10 feet. The cable pin-outs are as follows:

Signal Ground	Signal Pin	I/O	Signal Name
1	2	O	RWC-
3	4	O	Head Select 2-
5	6		NC
7	8	I	Seek Complete-
9	10		NC
11	12		NC
13	14	O	Head Select 0-
15	16	I	Sector-
17	18	O	Head Select 1-
19	20	I	Index-
21	22	I	Ready-
23	24		NC
25	26	O	Drive Select 1-
27	28	O	Drive Select 2-
29	30	O	Drive Select 3-
31	32	O	Drive Select 4-
33	34	O	Direction In-
35	36	O	Step-
37	38		NC
39	40	O	Write Gate-
41	42	I	TR000-
43	44	I	Write Fault-
45	46		NC
47	48		NC
49	50		NC

2.3.15 34 Pin Drive Control Connector

This Drive Control connector (J7 on -55 and -05) is a 34 pin vertical header on tenth-inch centers that mates with Burndy #FRS34BS. The cable used should be flat ribbon cable or twisted pair with a length of less than 10 feet. The cable pin-outs are as follows:

Signal Ground	Signal Pin	I/O	Signal Name
1	2	O	RWC-
3	4	O	Head Select 2-
5	6	O	Write Gate-
7	8	I	Seek Complete-
9	10	I	TR000-
11	12	I	Write Fault-
13	14	O	Head Select 0-
15	16	I	Sector-
17	18	O	Head Select 1-
19	20	I	Index-
21	22	I	Ready-
23	24	O	Step-
25	26	O	Drive Select 1-
27	28	O	Drive Select 2-
29	30	O	Drive Select 3-
31	32	O	Drive Select 4-
33	34	O	Direction In-

2.4 Drive Data Signals

The Drive Data connectors carry the high speed differential MFM data between the drive and the WD1001. Due to the loading characteristics of these differential lines, each of the drives have their own data connector. Drive Data Signals are as follows:

2.4.1 Drive Selected-

This signal is not used on the WD1001.

2.4.2 Timing Clock+

One half of the differential Timing Clock signal. This line contains a square wave signal equal to 1/64 the frequency of the write clock crystal. (-55 and -85 only)

2.4.3 Timing Clock-

This is the complementry version of Timing Clock+. (-55 and -85 only)

2.4.5 MFM Write Data+-

Differential MFM data from the controller to the disk.

2.4.5 MFM Read Data+-

Differential MFM data from the disk to the controller.

2.4.6 Drive Data Connectors

Four Data connectors (J1-4) are provided for clock signals and data between the WD1001 and each drive. All lines associated with the transfer of data between the drive and the WD1001 system are differential in nature and may not be multiplexed. The Data connectors are 20 pin vertical headers on tenth-inch centers that mate with Burndy #FRS20BS. The cable used should be flat ribbon cable or twisted pair with a length of less than 10 feet. The cable pin-outs are as follows:

Signal Ground	Signal Pin	I/O	Signal Name
2	1	I	Drive Selected
4	3		NC
6	5		Write Protect-
8	7		NC
	9	OI	Timing Clock+ *
	10	O	Timing Clock- *
11			GND
12			GND
	13	O	MFM Write Data+
	14	O	MFM Write Data-
15			GND
16			GND
	17	I	MFM Read Data+
	18	I	MFM Read Data-
19			GND
20			GND

* -55 and -85 only

2.4.7 Differential Data Driver/Receiver

HIGH TRUE

AMD 26LS31 or 75110A

51Ω

51Ω

HIGH TRUE

AMD 26LS32
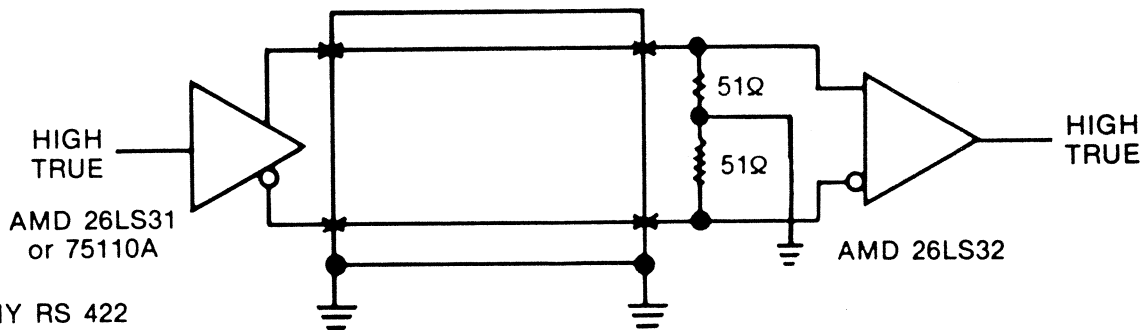
NOTE: ANY RS 422 DRIVER/RECEIVER PAIR WILL INTERFACE

FLAT RIBBON OR TWISTED PAIR MAX 10 FT.

2.5 Power Connector

A three pin connector (J6) is provided for power input to the -85 and -55 board. The customer supplied mating connector housing is Molex 03-09-1032. The pin-outs are as follows:

| Pin	Signal Name
1	Ground
2	Open *
3	+5 V regulated

* This pin was ground on the REV B and below WD1000 (Non-ECC predecessor of the WD1001) and -V (-7 to -15) on Rev. C through G.

A four pin connector (J6) is provided for power input to the -05 board. The customer supplied mating connector housing is Amp 1-480424. The pin-outs are as follows:

Pin	Signal Name
1	No Connection
2	Ground
3	Ground
4	+5 V regulated

SECTION 3 INTERFACE TIMING

3.1 Host Interface Timing

3.1.1 Host Read Timing

Symbol	Characteristic	Min	Max	Units
t_{RE}	RE active from A0-2, CS-	0		nS
t_{WA}	WAIT- active from CS-		100	nS
t_{RST}	INTRQ, DRQ reset from RE-		100	nS
t_{DV}	Data valid before WAIT- inactive	10		nS
t_{DH}	Hold Time Data from RE- inactive		20	nS
t_{WT}	WAIT- active period	0	6	uS
t_{HLD}	Hold time A0-2, CS- from RE-	0		nS
t_{SET}	WAIT- inactive to DRQ		750	nS
t_{XFER}	Transfer time per byte	1.75		uS
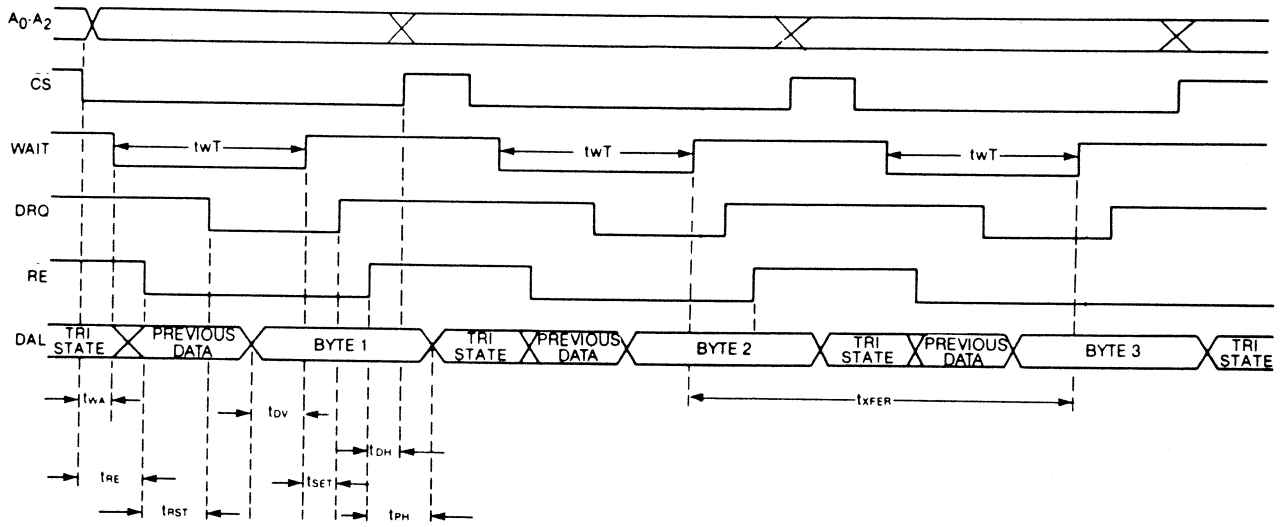
3.1.2 Host Write Timing

Symbol	Characteristic	Min	Max	Units
t_{WE}	WE- active from A0-2, CS-	0		nS
t_{WA}	WAIT- active from CS-		l00	nS
t_{RST}	INTRQ, DRQ reset from WE-		100	nS
t_{DS}	Data valid delay from WE-		100	nS
t_{HD}	Data Hold from WE-	20		nS
t_{WT}	Wait active period	0	6	uS
t_{HLD}	Hold time A0-2, CS- from WE-	0		
t_{XFER}	Transfer time per byte	1.75		uS

3-2

3.2 Drive Control Timing

Symbol	Characteristic	Min	Max	Units
t_{WG}	Write gate pulse width	1 sector	2 rotation	
t_{DS}	Direction to step delay	250		nS
t_{SW}	Step pulse width	5 (typical)		uS
t_{SP}	Programmed Step pulse period	.035	7.5	mS
t_{SS}	Step to Seek Complete false		30	uS
t_{SC}	Last Step to Seek Complete		128	Index times

Notes:

1. Write gate pulse width will vary depending on the sector size and the rotation rate of the disk.

2. Step pulse period will be equal to seek complete time during auto restore.

3.3 Drive data timing

Symbol	Characteristic	Min	Max	Units
t_{TC}	Timing clock period	WCLK/16	(typical)	
t_{WD}	Write data pulse width	60	120	nS
t_{RD}	Read data pulse width	25		nS
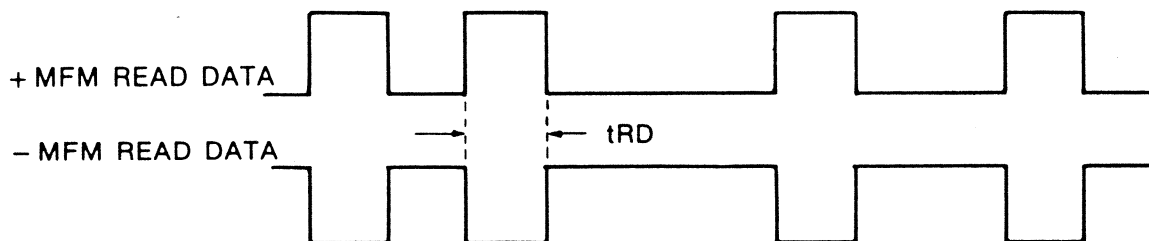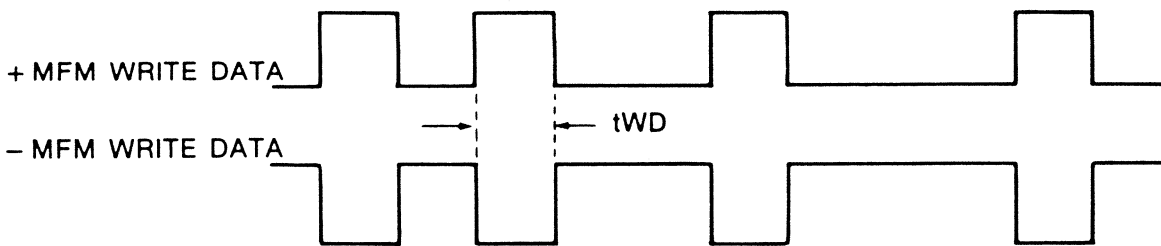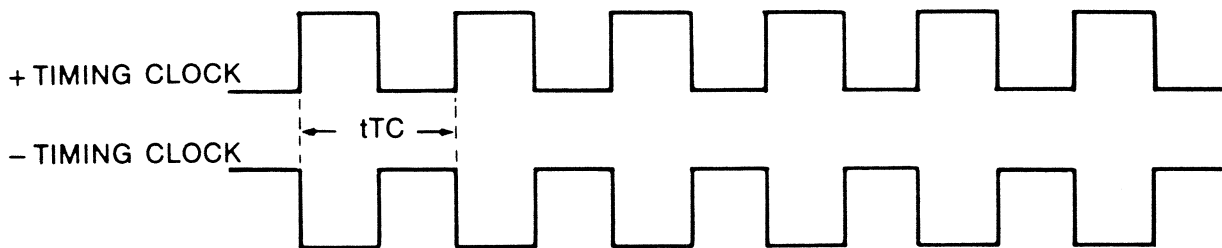
SECTION 4 HOST INTERFACING

The WD1001 is designed to easily interface to most micro-computers and many mini-computers. All interfacing is done through the Host Interface Connector (J5). The interface is very similar to Western Digital's family of LSI peripheral chips. There's only one exception: the inclusion of a WAIT line.

4.1 Waits

The WAIT- control line goes true whenever either of the following are true:

O The WD1001 is accessing data internally to send to the host during a read operation.
O The WD1001 has not accepted the data from the host during a write operation.

The definition of the WAIT- line is very similar to the WAIT signal found on many Intel and Zilog products. WAIT- is also similar to the REPLY signal on Western Digital and DEC processors.

Wait will not necessarily make a transition for each access to the WD1001. When the WD1001 can return the requested data within 100 nS, there will not be any transition of the WAIT- line. This should be interpreted as an instant REPLY on Western Digital Processors.

If the WD1001 cannot return the requested data within 100 nS, it will assert its WAIT- line. The period of the WAIT- signal will vary from 750 nS to 6 uS, with 1.25 uS being about average. The period of WAIT- only approaches 6 uS during a read or write which happens immediately **after** a command is written to the command register. This means that longer waits may be encountered during the first read or write to any WD1001 register if that first read or write happens within approximately 6 uS of a command being issued.

During the time that WAIT- is asserted, the host system **must** hold all of its strobe and address lines stable. On write operations, the DAL lines must also be held stable.

Do not confuse the function of WAIT- with the Busy bit of the Status register. The Busy bit is a status indicating that the WD1001 is communicating with the disk and WAIT- is a bus synchronization signal.

4.2 Host interfacing example

The example below illustrates the absolute minimum of hardware required to interface to a small 8085 microcomputer system. In this example, we are not using buffers or completely decoding the I/O. In a real system, the user would probably want to completely decode the I/O to minimize the amount of I/O or memory space required to interface the WD1001. If the interface cable length is kept to a few inches, it is often permissible to interface it directly to a microcomputer's **buffered** bus.
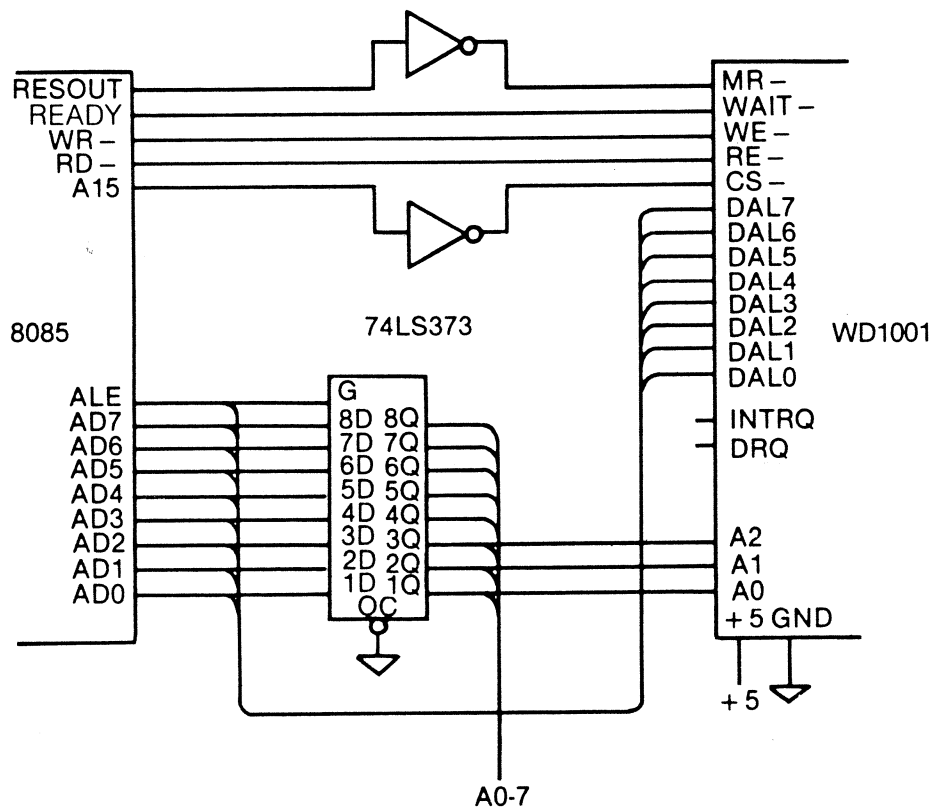
SECTION 5 TASK FILE

5.1 Task File Basics

The WD1001 performs all disk functions through a set of registers called the Task File. These registers are loaded with parameters such as Sector Number, Cylinder Number, etc., prior to issuing a command. Individual registers are selected via A0-2. The following registers are available:

5.2 Register Array

CS-	A2	A1	A0	RE-	WE-
1	X	X	X	Deselected	Deselected
0	0	0	0	Data Register	Data Register
0	0	0	1	Error Register	Write Precomp
0	0	1	0	Sector Count	Sector Count
0	0	1	1	Sector Number	Sector Number
0	1	0	0	Cylinder Low	Cylinder Low
0	1	0	1	Cylinder High	Cylinder High
0	1	1	0	Size/Drive/Head	Size/Drive/Head
0	1	1	1	Status Register	Command Register

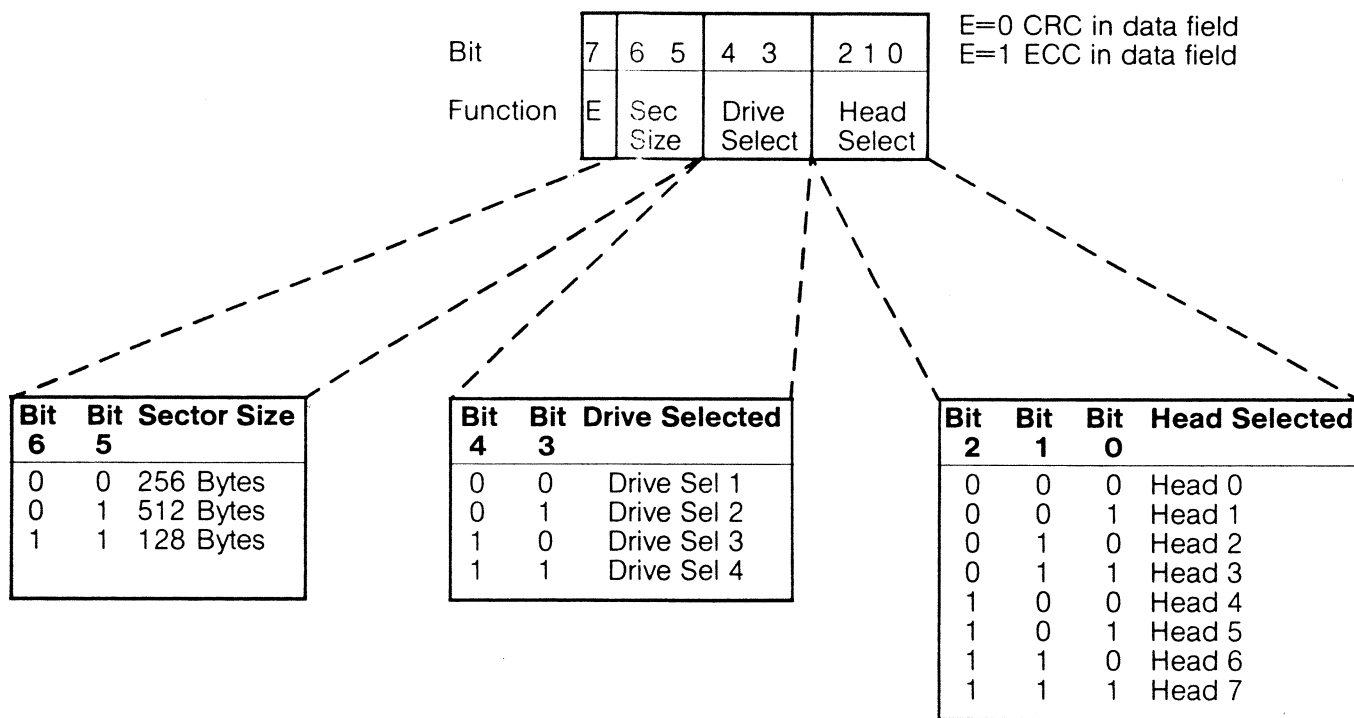## 5.3 Register Definitions
5.3.1 Command Register

All commands are loaded into this register after the task registers have been set. Writing to this register will cause the INTRQ Line to be reset. The Command register is a write-only register.

5.32 Status Register

After execution of a command, the Status register is internally loaded with status information pertaining to the command executed. The host must read this register to determine successful execution of the command. The Status register is a read-only register; it cannot be written to by the host. If the busy bit is set, no other bits in this register are valid. Accessing this register will cause the INTRQ line to be reset.

5.3.3 SDH Register

This register contains the ECC mode, sector Size, Drive select, and Head select bits. The SDH register is a read/write register organized as follows:

Bit	7	6 5	4 3	2 1 0
Function	E	Sec Size	Drive Select	Head Select

E=0 CRC in data field
E=1 ECC in data field

Bit 6	Bit 5	Sector Size
0	0	256 Bytes
0	1	512 Bytes
1	1	128 Bytes

Bit 4	Bit 3	Drive Selected
0	0	Drive Sel 1
0	1	Drive Sel 2
1	0	Drive Sel 3
1	1	Drive Sel 4

Bit 2	Bit 1	Bit 0	Head Selected
0	0	0	Head 0
0	0	1	Head 1
0	1	0	Head 2
0	1	1	Head 3
1	0	0	Head 4
1	0	1	Head 5
1	1	0	Head 6
1	1	1	Head 7

5.3.4 Cylinder Number

These two read/write registers form the cylinder number where the head is to be positioned on a Seek, Read, Write, or Format command. Internally, a separate set of cylinder register values are maintained for each drive. The two least significant bits of the Cylinder High register form the most significant bits of the cylinder number as illustrated below:

	Cylinder High								Cylinder Low							
Register bits:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Cylinder bits:							9	8	7	6	5	4	3	2	1	0

5.3.5 Sector number

This register is loaded with the desired sector number prior to a Read or Write command. The Sector Number register is a read/write register and may be read or written to by the host.

5.3.6 Sector Count

This read/write register is loaded with the number of sectors to be processed. On Read or Write multiple commands, the number of sectors to be transferred is loaded into this register. During a Format command, this register is loaded with the number of sectors to be formatted. During the course of a command, the Sector Count register is decremented towards zero and should be re-loaded for each command.

5.3.7 Error Register

This register contains specific fault information pertaining to the last command executed. This register is valid only if the Error bit in the Status register is set. The Error register is read only.

5.3.8 Write Precomp

The Write Precompensation register holds the cylinder number where the RWC line will be asserted and Write Precompensation logic is to be turned on. This write-only register is loaded with the cylinder number divided-by-4 to achieve a range of 1024 cylinders. For example, if write precompensation is desired for cylinder 128 (80 Hex) and higher, this register must be loaded with 32 (20 Hex). The Write Precompensation delay is fixed at 12 nanoseconds from nominal. On drives that require separate write precompensation and reduce write current cylinders, set the Write Precomp register to the cylinder where write current reduction is desired.

5.3.9 Data Register

This register is the user's window to the on-board full sector buffer. It contains the next byte of data to be written to or read from the internal sector buffer. The Data register is accessed once for each byte in the sector. When the DRQ (Data ReQuest) line is asserted, the sector buffer contains data in a read command, or is awaiting data to be written during a write command into the Data register. If the WD1001 is interfaced using programmed I/O, data transfers to this register can be implemented using block moves. This register may not be read from or written to except in the context of a valid command.

5.4 Status Registers

There are two registers in the WD1001 that are used to monitor the execution of commands. They are the Status register and the Error register. Each bit of these registers is used to define a particular type of status or error condition:

Bit	Status Register	Error Register
7	Busy	Bad Block Detect
6	Ready	Uncorrectable
5	Write Fault	CRC Error - ID Field
4	Seek Complete	ID Not Found
3	Data Request	–
2	Corrected	Aborted Command
1	–	TR000 Error
0	Error	DAM Not Found

5.5 Status Register Bits

5.5.1 Error

When set, indicates that a bit is set in the Error register. It provides an efficient means of checking for an error condition by the host. This bit is reset on receipt of a new command.

5.5.2 Corrected

Indicates that there was a read error condition either in the data field or the ECC check bits themselves, and that the controller was able to correct the condition.

5.5.3 Data Request

Functions almost identically to the hardware DRQ line. When set, it indicates that the sector buffer is ready to accept data or contains data to be read out by the host. The Data Request bit is reset when the sector buffer has been fully read from or written to. Normally, the host need not consult this bit to determine if a byte should be transferred.

5.5.4 Seek Complete

Indicates the condition of the Seek Complete line on the selected drive.

5.5.5 Write Fault

Indicates the condition of the Write Fault line on a selected drive. The WD1001 will not execute any command if this bit is set.

5.5.6 Ready

Indicates the condition of the Ready line of the selected drive. The WD1001 will not execute any commands unless this bit is set.

5.5.7 Busy

After issuing a command, this bit will be set, indicating that the WD1001 is busy executing a command. No other bits or registers are valid when this bit is set.

5.6 Error Register Bits

5.6.1 DAM not found

Will be set during a Read Sector command if, after successfully identifying the ID field, the Data Address mark was not detected within 16 bytes of ID field.

5.6.2 TR000 Error

Will be set during a Restore command if, after issuing 1024 stepping pulses, the Track 000 line was not asserted by the drive.

5.6.3 Aborted Command

Indicates that a valid command has been received that cannot be executed, based on status information from the drive. For example, if a write sector command has been issued while the Write Fault line is set, the Aborted Command bit will be set. Interrogation of the Status and/or Error registers by the host can be performed to determine the cause of failure.

5.6.4 ID Not Found

When set, this bit indicates that an ID field containing a specified cylinder, head, sector number or sector size was not found.

5.6.5 CRC Error ID

Indicates that a CRC error was encountered in an ID field.

5.6.6 Uncorrectable

Indicates that an error was detected while reading the data field or ECC check bits and the error was so severe that the controller was not able to correct the condition.

5.6.7 Bad Block Detect

Indicates that a Bad Block Mark has been detected in the specified ID field. If the command issued was a write sector command, no writing will be performed. If generated from a read sector command, the data field will not be read. Note that Bad Block will not be detected if the flaw is in the ID field unless multiple ID fields were written.

SECTION 6 COMMANDS

The WD1001 executes five easy to use macro commands. Most commands feature automatic 'implied' seek, which means the host system need not tell the WD1001 where the R/W heads of each drive are or when to move them. The controller automatically performs all needed retries on all errors encountered including data field errors. If the data field contains an error, the controller will perform a correction, if possible. If the R/W head mis-positions, the WD1001 will automatically perform a restore and a re-seek. If the error is completely unrecoverable, the WD1001 will simulate a normal completion to simplify the host system's software.

Commands are executed by loading a command byte into the Command register while the controller is not busy. (Controller will not be busy if it has completed the previous command.) The task file must be loaded prior to issuing a command. No command will execute if the Seek Complete or Ready lines are false or if the Write Fault line is true. Normally it is not necessary to poll these signals before issuing a command. If the WD1001 receives a command that is not defined in the following table, undefined results will occur.

6.1 Command Summary

For ease of discussion, commands are divided into three types which are summarized in the following table:

TYPE	COMMAND	BITS							
		7	6	5	4	3	2	1	0
I	Restore	0	0	0	1	r_3	r_2	r_1	r_0
I	Seek	0	1	1	1	r_3	r_2	r_1	r_0
II	Read Sector	0	0	1	0	D	M	L	0
III	Write Sector	0	0	1	1	0	M	L	0
III	Format Track	0	1	0	1	0	0	0	0

L=Long Read/Write   D=DMA Read Interrupt
M=Multiple Sector r_x=Stepping Rate

6.1.1 Stepping Rates

r_3-r_0 – Stepping Rate						
0000	=	35	uS	1000	=	4.0 mS
0001	=	0.5	mS	1001	=	4.5 mS
0010	=	1.0	mS	1010	=	5.0 mS
0011	=	1.5	mS	1011	=	5.5 mS
0100	=	2.0	mS	1100	=	6.0 mS
0101	=	2.5	mS	1101	=	6.5 mS
0110	=	3.0	mS	1110	=	7.0 mS
0111	=	3.5	mS	1111	=	7.5 ms

6.1.2 DMA Read

D -DMA Read Mode

0 = Programmed I/O Mode
1 = DMA Mode

The DMA bit is used to position INTRQ in relation to DRQs during the read sector command. If the DMA bit is reset (D=0), the interrupt will occur before the first DRQ. This allows the programmed I/O host to intervene and transfer the data from the sector buffer. If the DMA bit is set (D=1), then the interrupt will occur only after the system DMA controller has transferred the entire buffer of data.

6.1.3 Long Read and Write

If the Long bit is set, a special diagnostic read or write will be performed. During normal reads or writes, the ECC check bytes are not visible to the user. The Long bit allows the user to read and write these normally invisible bytes.

During a Read Long, the WD1001 will return a sector that is four bytes longer than the selected sector size. These four bytes will be the ECC check bits as recorded on the disk. During a Write Long, the host gives the WD1001 a sector that is four bytes longer than normal. These four extra bytes are recorded in place of the ECC bytes that are normally written after each sector.

The Read and Write Long option may only be used when the WD1000 is in ECC mode.

6.2 Type I Commands

These commands simply position the R/W heads of the selected drive. Both commands have explicit stepping rate fields. The lower four bits of these commands form the stepping rate which is stored for later Read, Write or Format operations.

6.2.1 Restore

The Restore command is used to calibrate the position of the R/W head on each drive by stepping the head outward until the TR000 line goes true. Upon receipt of the Restore command, the Busy bit in the Status Register is set. Cylinder High and Cylinder Low registers are cleared. The lower four bits of the command byte are stored in the stepping rate register for subsequent implied seeks. The state of Seek Complete, Ready and Write Fault are sampled, and if an error condition exists, the Aborted command bit in the Error register is set, the Error bit in the Status register is set, an interrupt is generated, and the Busy bit is reset.

if no errors are encountered thus far, the internal head position register for the selected drive is cleared. The TR000 line is sampled. If TR000 is true, an interrupt is generated and the Busy bit is reset. If TR000 is not true, stepping pulses at a rate determined by the stepping rate field are issued until the TR000 line is activated. When TR000 is activated, the Busy bit is reset and and interrupt is issued. If the TR000 line is not activated within 1024 stepping pulses, the TR000 Error bit in the Error Register and the Error bit in the Status Register are set, the Busy bit is reset, and an interrupt is issued.

6.2.2 Seek

The Seek command positions the R/W head to a certain cylinder. It is primarily used to start two or more concurrent seeks on drives that support buffered stepping. Upon receipt of the Seek command, the Busy bit in the Status Register is set. The lower four bits of the command byte are stored in the stepping rate register for subsequent implied seeks. The state of Seek Complete, Ready and Write Fault are sampled, and if an error condition exists, the Aborted command bit in the Error register is set, the Error bit in the Status register is set, an interrupt is generated, and the Busy bit is reset.

If no errors are encountered thus far, the internal head position register for the selected drive is updated, the direction line is set to the proper direction and a step pulse is issued for each cylinder to be stepped. When all stepping pulses have been issued, the Busy bit is reset and an interrupt is issued. Note that the Seek Complete line is not sampled after the Seek command, allowing multiple seek operations to be started using drives with buffered seek capability.

6.3 Type II Commands

This type of command is characterized by a transfer of a block of data from the WD1001 buffer to the host. This command has an implicit stepping rate as set by the last Restore or Seek command.

6.3.1 Read Sector

The Read Sector command is used to read a sector of data from the disk to the host computer. Upon receipt of the Read command, the Busy bit in the Status register is set. The state of Seek Complete, Ready and Write Fault are sampled, and if an error condition exists, the Aborted Command bit in the Error register is set, the Error bit in the Status register is set, and a normal completion is simulated.

6.3.1.1 Implied Seek

If no errors are encountered so far, a Seek command is executed. The Seek Complete line is sampled. If the Seek Complete line does not go true within 128 Index pulses, then the Aborted command bit in the Error register is set, the Error bit in the Status register is set, and a normal completion is simulated.

6.3.1.2 Retries

Once the head has settled over the desired cylinder, the WD1001 will attempt to read the sector. The WD1001 performs all retries necessary to recover the data during the read command. The controller attempts to read the desired sector up to 16 times. It will attempt a retry if it does not find an ID, if the ID of that sector has a bad CRC, if the Data Address Mark (DAM) couldn't be found, or even if the data was actually read from the disk but was in error.

6.3.1.3 Error Correction

If an error was detected while reading the data field, the controller will attempt to correct the error. If the error was correctable, the Corrected bit in the status register will be set and the command resumed. If it was uncorrectable, the Uncorrectable Error bit will be set, the Error bit in the Status register is set, and a normal completion is simulated.

6.3.1.4 Auto Restore

Every time the controller encounters an error, it records the occurrence of that error in an internal register. If, after 16 retries, the controller was not able to get a match on the ID field, it assumes that the head was possibly mis-positioned and executes an auto-restore. During the auto-restore, the stepping rate is implied to be equal to the Seek Complete period. If the TRK000 does not go true within 1024 steps, the TRK000 Error bit in the Error register is set, the Error bit in the Status register is set and a normal completion is simulated.

After the auto-restore has been successfully completed, the controller re-seeks and attempts to read the sector once again. An auto-restore will be performed only once per read or write sector command.

6.3.15 Hard Errors

If the controller encounters a non-recoverable error, the controller examines its internal error history register. It then sets the bit in the Error Register of the highest severity error incurred. If the Uncorrectable bit is set, the data that last produced that error will be available in the sector buffer. The Error bit in the Status Register is set and a normal completion is simulated.

6.3.1.6 Error Severity Levels

Although the WD1001 might encounter any number of errors in the course of executing a command, it only reports the most severe error. Errors are ranked from most severe to least severe as follows:

1. Aborted Command
2. TR000 Error
3. Bad Block *
4. Uncorrectable
5. Data Address Mark Not Found
6. ID CRC Error
7. ID Not Found

* – Bad block will only be detected if there is no ID CRC Error or ID Not Found Error in the sector with the Bad Block bit set.

6.3.1.7 Normal Completion

If the WD1001 encountered no errors, it is considered a normal completion. The busy bit is reset. The status of the DMA bit in the command byte is examined. If this bit is reset (D=0; programmed I/O mode) then an interrupt is issued at this time. DRQs are then generated for each byte to be read from the buffer. (Note: It is recommended that programmed I/O transfers should take place as a block move without consulting the DRQ bit in the Status Register.) After all the data has been moved from the buffer, the DMA bit in the command byte is consulted again. If this bit is set (D=1; DMA mode) then an interrupt will be issued.

6.3.2 Multiple Sector Reads

If the M bit in the command byte is set, then the WD1001 will attempt to read multiple sectors. After all the data has been transferred from the sector buffer to the host on a read, the Sector Number register is incremented, the Sector Count register is decremented, and if the Sector Count is non zero, the read command will be re-executed. When Sector Count reaches zero or if a fatal error is encountered, the WD1001 will stop and interrupt the host.

When a Correctable error is encountered during a multiple sector read, the occurrence of the error is logged, but no interrupts are generated. After the whole multiple transfer is complete, the host can read the Corrected bit of the Status register to determine if any automatic corrections have taken place.

6.4 Type III Commands

This type of command is characterized by a transfer of a block of data from the host to the WD1001 buffer. These commands have implicit stepping rates as set by the last Restore or Seek command.

6.4.1 Write Sector

The Write Sector command is used to write a sector of data from the host computer to the disk. Upon receipt of the Write command, the controller generates DRQs for each byte to be written to the buffer. (Note: It is recommended that programmed I/O transfers should take place ᴄ a block move without consulting the DRQ bit in the Status register.)

After all data has been sent to the sector buffer, the Busy bit in the Status register is set. The state of Seek Complete, Ready and Write Fault are sampled, and if an error condition exists, the Aborted Command bit in the Error register is set, the Error bit in the Status register is set, an interrupt is generated, and the Busy bit is reset.

6.4.1.1 Implied Seek

If no errors are encountered so far, a Seek command is executed. The Seek Complete line is sampled. If the Seek Complete line doesn't go true within 128 Index pulses, then the Aborted Command bit in the Error register is set, the Error bit in the Status register is set, an interrupt is generated and the Busy bit is reset.

6.4.1.2 Retries

Once the head has settled over the desired cylinder, it will attempt to read the ID of the sector. The WD1001 performs all retries necessary to recover the ID during the write command. The controller attempts to read the ID of the desired sector up to 16 times. It will attempt a retry if it doesn't find an ID or if the ID of that sector has a bad CRC.

6.4.1.3 Auto Restore

Every time the controller encounters an error, it records the occurrence of that error in an internal register. If, after 16 retries, the controller was not able to get a match on the ID field, it assumes that the head was possibly mis-positioned and executes an auto-restore. During the auto-restore, the stepping rate is implied to be equal to the Seek Complete period. After the auto-restore has been successfully completed, the controller re-seeks and attempts to write the sector once again.

6.4.1.4 Hard Errors

If the controller encounters a non-recoverable error, the controller examines its internal error history register. It then sets the bit in the Error register of the highest severity error incurred. The Error bit in the Status register is set, an interrupt is generated and the Busy bit is reset.

If the proper sector is located, the sector buffer is written to the disk, an interrupt is generated and the Busy bit is reset.

6.4.2 Format Track

The Format command is used for initializing the ID and data fields on a particular disk. Upon receipt of the Format command, the controller generates DRQs for each byte of the interleave table to be written to the buffer. Information on setting up an interleave table can be found in Section 7. In all cases, the number of bytes transferred to the buffer must correspond to the current sector size.

After all data has been sent to the buffer, the Busy bit in the Status register is set. The state of Seek Complete, Ready and Write Fault lines are sampled. If an error condition exists, the Aborted Command bit in the Error register is set, the Error bit in the Status register is set, an interrupt is generated and the Busy bit is reset.

6.4.2.1 Implied Seek

If no errors are encountered so far, a Seek command is executed. No verification of track positioning accuracy is performed because the track may not have any ID fields present. After the Seek operation has been performed, the Seek Complete line is sampled. If the Seek Complete line is not asserted within 128 Index pulses, the Aborted Command bit in the Error register is set, the Error bit in the Status register is set, an interrupt is generated and the Busy bit is reset.

Once the head has settled over the desired cylinder, the controller waits until the Index line is asserted. Once the index is found, a number of ID fields and nulled data fields are written to the disk. The number of sectors written is equal to the contents of the Sector Count Register. As each sector is written, the Sector Count Register is decremented, and consequently, must be updated before each format operation.

After the last sector is written, the controller back-fills the track with 4E's. When the next index pulse after the last sector written is encountered, the format operation is terminated, an interrupt is generated and the Busy bit is reset.

6.4.2.2 Track Format

The Format command formats the track using the following format:
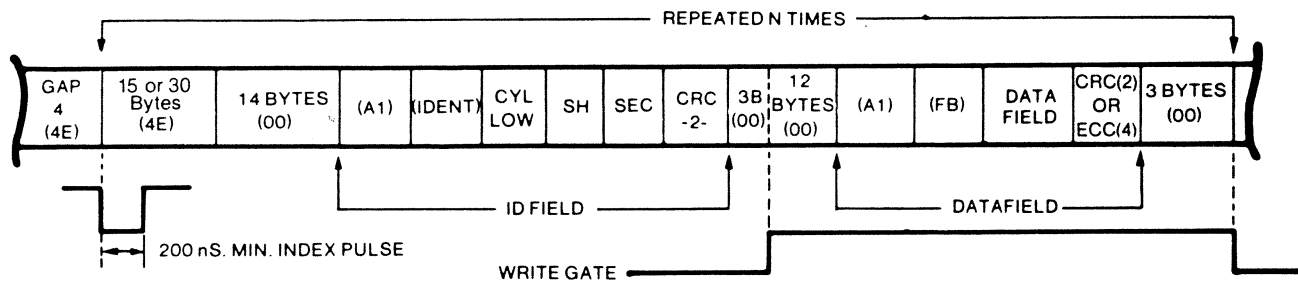
NOTE:

1) When MSB of SH byte = 1, bad block is detected.
2) Write Gate turn-on is 3 bytes after the ID field's CRC byte.
3) Write Gate turn-off is 3 bytes after the Data Field's check bytes.
4) 12 bytes of zeroes are re-written on a Data Field update.

5) The 2 LSB's of the IDENT byte are used for Cylinder high.
   These values are:
   FE - 0 to 255   cylinders
   FF - 256 to 511   cylinders
   FC - 512 to 767   cylinders
   FD - 768 to 1023 cylinders

SECTION 7 PROGRAMMING

Users familiar with floppy disk systems will find programming the WD1001 a pleasant surprise. A substantial amount of intelligence that was required by the host computer has been incorporated into the WD1001. The WD1001 performs all needed retries, even on head positioning errors. If there is an error in the data field, the WD1001 will attempt to correct it. Most commands feature automatic 'implied' seek which means that seek commands need not be issued to perform basic read/write functions. The WD1001 keeps track of the position of up to four read/write head assemblies, so the host system does not have to maintain track tables. All transfers to and from the disk are through an on-board full sector buffer. This means that data transfers are fully interruptable and can take place at any speed that is convenient to the system designer. In the event of an unrecoverable error, the WD1001 simulates a normal completion so that special error recovery software is not needed.

This section assumes that the user has read sections five (Task File) and six (Commands).

7.1 Setting up Task Files

Before any of the five commands may be executed, a set of parameter registers called the Task File must be set up. For most commands, this informs the WD1001 of the exact location on the disk that the transfer should take place. For a normal read or write sector operation, the Sector Number, the Size/Drive/Head, Cylinder Number and Command registers (usually in that order) will be written.

Note that most of these registers are readable as well as writeable. These registers normally are not read from, but this feature is provided so that error reporting routines can determine physically where an error occurred without recalculating the sector, head and cylinder parameters.

Since the WD1001 can recall all the Task File parameters sent to it, it is recommended that Task File parameters be stored in the WD1001 as they are calculated. This will save the programmer a few instructions and microseconds by not maintaining two copies of the same information.

7.1.1 Cylinders and Tracks

Since most hard disk drives contain more than one head per positioner, it is more efficient to step the R/W head assemblies of most disk drives by cylinders, not tracks. In other words, the disk driver software should be designed to read or write all data that is directly accessible by all the heads on a positioner before stepping to a new cylinder. The following example illustrates a cylinder-by-cylinder sequential file read on a four head, two platter disk drive:

Physical Cylinder	Logical Head Number	Physical Head Side	Physical Platter
25	3	Top	B
26	0	Bottom	A
26	1	Top	A
26	2	Bottom	B
26	3	Top	B
27	0	Bottom	A

7.2 Type I Command Programming

Restore and Seek are Type I commands. These commands position the R/W heads of the selected drive and set the implied stepping rate register. No data is transferred to or from the Data Register. To execute a Type I command, the system software must do the following functions in this order:

1. Set up Task File and issue command with stepping rate (WD1001 will attempt to execute Type I command)
2. Wait for interrupt or for Busy bit in Status Register to be reset
3. Check Error bit in Status Register for proper completion

7.2.1 Stepping Rates

Most drives that use the WD1001's 35 uS stepping rate require a slower rate (usually 3 mS or more) for Restore operations. This is why the WD1001 allows you to have explicit stepping rates on both Restore and Seek. Upon power up, it is a good practice to issue a Restore command with a slower stepping rate to recalibrate the head assembly. After waiting for that operation to complete, issue a Seek command with the faster stepping rate to set the stepping rate for subsequent implied seeks.

7.2.2 Use of Busy bit

There are two different ways to sense the completion of a command. The first way, for smaller single user systems, is to poll the Busy bit of the Status Register. The Busy bit (bit 7) is set whenever the controller starts a disk operation and is reset whenever the controller is ready to communicate with the host computer.

SECTION 7 PROGRAMMING

The WD1001 busy bit is located in the same place as the sign bit of many computers to simplify the polling process.

This is one way to poll this bit using 8080 code:

```
WAIT:   IN      STATUS      ;Input WD1001
                            ;Status register
        ANA     A           ;Update 8080
                            ;sign flag
        JM      WAIT        ;Wait if busy (sign)
                            ;bit set
```

Here's one way to poll the busy bit using PDP-11 code:

```
WAIT:   MOVB    @#STATUS,R0 ;Input status,
                            ;update sign flag
        BMI     WAIT        ;Wait if busy (N)
                            ;bit set
```

7.2.3 Use of Interrupts

Another more efficient way of notifying the CPU that the WD1001 has completed a command is through interrupts. The INTRQ line on the WD1001 makes a low to high transition whenever the disk controller requires CPU intervention. This allows the host CPU to run other tasks while the WD1001 is reading or writing data to the disk.

7.2.4 Use of the Error bit

Since the WD1001 simulates normal completions, it acts the same whether or not errors are encountered. The only way to check error status is to check the Error bit in the Status register. The WD1001 Error bit is located so that it can be easily tested by rotating it into the carry bit of many processors. The contents of the Error register are not valid unless the Error bit is set.

This is one way to check the Error bit using 8080 code:

```
        IN      STATUS      ;Get status (if not
                            ;already in A)
        RAR                 ;Rotate error bit
                            ;into C
        JC      ERROR       ;Jump if error
                            ;found
```

In certain hardware configurations, this can check the error bit using PDP-11 code:

```
        BIT     @#STATUS,#1 ;Bit test the
                            ;error bit
        BNE     ERROR       ;Branch if
                            ;error found
```

7.2.5 Use of the Corrected bit

Correctable errors are usually quite benign and can almost always be ignored. However, some systems designers may wish to log their occurrence. The Corrected bit is positioned in the Status register to facilitate error logging. Correctable and fatal errors can be detected with the following 8080 code:

```
        IN      STATUS      ;Get WD1001 status
        ANI     5           ;Mask off Error and
                            ;Correct bits
        JNZ     SOMERR      ;Jump if we have
                            ;either a correct-
                            ;able or fatal error
```

7.3 Type II Command Programming

The Read Sector command is the only Type II command. This command is characterized by the transfer of a block of data from the WD1001 buffer to the host. This command features implied seek with an implicit stepping rate. To execute a Type II single sector command in **programmed I/O mode,** the system software must do the following functions in this order:

1. Set up Task File and issue command with DMA bit reset (WD1001 will attempt to read sector)
2. Wait for interrupt or for Busy bit in Status Register to be reset
3. Do block move from WD1001 buffer to system memory
4. Check Error bit in Status Register for proper completion

Note: Steps 3 and 4 above can be reversed.

To execute a Type II single or multiple sector command in **DMA mode with interrupts,** the system software does the following:

1. Set up Task File and issue command with DMA bit set
2. Set up DMA controller (WD1001 will attempt to read single or multiple sectors) (DMA controller will move data from WD1001 to memory)
3. Wait for interrupt from WD1001
4. Check Error bit in Status register for proper completion

Note: The above sequence is preferred but steps 1 and 2 above can be reversed.

7.3.1 DMA Mode

The DMA mode bit (D) in the above read sector examples is a special bit in the command byte that is used to optimize the WD1001's interrupts during programmed I/O and DMA operations. If the DMA bit is reset (D=0) the interrupt will come before the buffer is transferred. This allows a programmed I/O host to intervene and transfer the buffer of data. If the DMA bit is set (D=1) then the interrupt will happen only after the data has been transferred. This allows the host to go uninterrupted until the entire buffer has been transferred.

7.3.2 Block Moves

The WD1001 performs all transfers between it and the disk drive through an on-board full sector buffer. Once the disk has been read, the data is available to the host at any rate from DC to as high as a byte every 1.75 uS. In programmed I/O applications there is no need to consult the DRQ bit in the status register to determine if another byte is ready to be processed. Once an interrupt occurs or the busy bit is reset on a read, the host computer should do a block move of all the bytes in the sector.

The following 8080 code demonstrates a transfer from the WD1001 to system memory. The transfer address is in HL and the byte count is in B:

```
READIT:  IN    DATA    ;Get data from WD1001
                       ;sector buffer
         MOV   M,A     ;Store it in memory
         INX   H       ;Increment memory
                       ;pointer
         DCR   B       ;Decrement byte counter
         JNZ   READIT  ;Do it again if whole
                       ;sector not xfered
```

The following Z-80 instruction does it all. The transfer address is in HL, byte count is in B and WD1001 data register address in C:

```
READIT:  INIR          ;Transfer buffer from
                       ;WD1001 to memory
```

7.3.3 Using DMA

There are several features in the WD1001 which simplify the use of DMA. Of course, there's the DRQ line that makes a low to high transition for each byte to be transferred. As mentioned earlier, there is a special bit in the Read Sector command which optimizes the WD1001 interrupts for DMA operation.

7.3.4 Multiple Sector Transfers

The WD1001 can transfer more than one sector per command if interfaced using DMA **and** interrupts. Transfers as large as an entire track can be executed. The Sector Count register holds the number of records to be transferred. (If Sector Count is zero then 256 records will be transferred.) The Sector Number register holds the starting sector of the transfer. When a multiple sector transfer is successfully completed, the Sector Count register will be equal to zero and the Sector Number register will be equal to the last sector transferred plus one.

If a fatal error is encountered during a multiple sector transfer, the Sector Number register will be left pointing the sector that contained the fatal error and the Sector Count register will hold the number of sectors that were not transferred.

If a correctable error is encountered during a multiple sector read, the Corrected bit in the Status register will be set but the operation will not be terminated because correctable errors are not considered fatal.

7.3.4.1 Partial Sector Transfers

The WD1001 allows partial sector transfers on read operations. This allows the user to read the first part of a sector and then discard the rest. During programmed I/O, the byte counter in the block move routine is set to the number of bytes to be read. During DMA operations, the DMA controller is set with the number of bytes to be transferred.

Normally the WD1001 will interrupt the host after the sector has been transferred during a DMA read operation, but if a partial sector has been read, the WD1001 will not know that the operation has been completed. For this reason, the 'transfer complete' interrupt must come from the DMA controller. There is, still, a problem. During write sector operations, the DMA controller will interrupt the system after the buffer has been transferred to the WD1001 but before the data has been written. Some systems with advanced interrupt handling capabilities can easily mask off the spurious DMA interrupt. For those that can't, the WD1001 has a provision built into its command structure to detect read operations.

7.3.4.2 Interrupt Source Selection

Bit 4 of all commands determines whether the operation will be a read sector operation or something else. Those commands that require the interrupt from the WD1001 have this bit set to a 1. The read sector command (the only one that might need the DMA controller's interrupt) has this bit set to a 0.

7.3.4.3 Clearing Hardware DRQ

During partial sector reads, the DMA controller will stop the DMA transfer before the WD1001 has a chance to issue its last data request. Because of this, the DRQ line may be set the next time transfer parameters are sent to the DMA controller. To avoid spurious (and often fatal) DRQ's, the user must do a hardware clear of the DRQ line. This is accomplished by reading or writing the Cylinder Low register. (This will only clear the DRQ line. The DRQ bit in the Status Register will be indeterminate.) This action is typically done before a subsequent read or write sector command in the normal course of updating the Task File. Care should be exercised to insure that the DMA controller has passed its parameters only after the Task File is updated.

7.3.4.4 Interrupt Selection Circuit

If the user is reading partial sectors with the WD1001 and wants to have the system automatically configure its interrupts, a circuit similar to the following will have to be implemented:
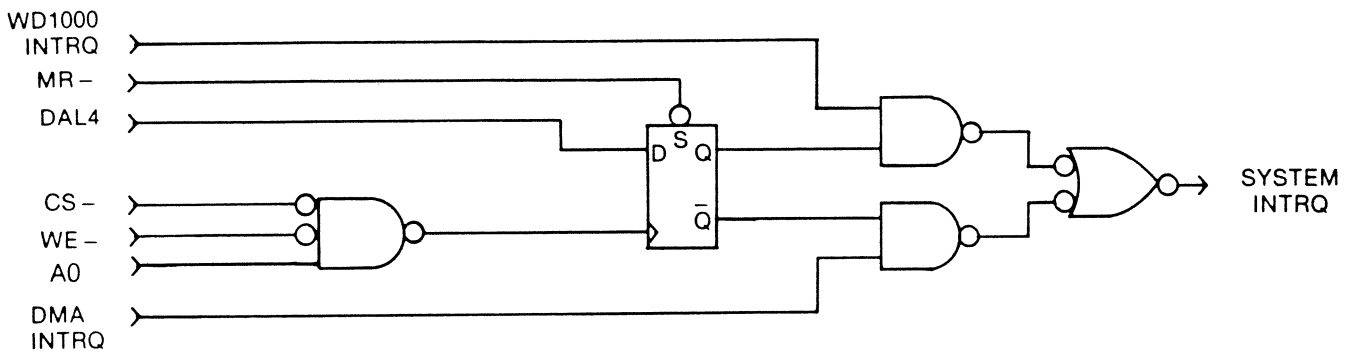
7.3.5 Simulated Completions

All WD1001 commands (except multiple sector transfers) act in precisely the same manner, whether or not an error was encountered. The only way to detect that an error has occurred is to sample the Error bit in the Status Register. Simulated Completions offer the system designer several tangible benefits:

o Simplifies masking and generation of interrupts

o Simplifies non-error handling portions of the system software

o Eliminates the software overhead of handling different types of errors

o Simplifies system software error handling validation (any error is handled the same as any other error)

o Prevents system failure in the event of some obscure error condition that the system programmer did not anticipate

7.4 Type III Command Programming

Write Sector and Format are Type III commands. These commands are characterized by the transfer of a block of data from the host to the WD1001 buffer. Like Type II commands, these commands feature implied seek with an implicit stepping rate. To execute a single sector Type III command in **programmed I/O mode,** the system software must do the following functions in this order:

1. Set up Task File and issue command
2. Do block move from system memory to WD1001 buffer (WD1001 will attempt to write a sector or format)
3. Wait for interrupt or for Busy bit in Status Register to be reset
4. Check Error bit in Status Register for proper completion

To execute a single or multiple sector Type III command in **DMA mode with interrupts,** the system software does the following:

1. Set up Task File and issue command
2. Set up DMA controller (DMA controller will move data from memory to WD1001) (WD1001 will attempt to write sector or format)
3. Wait for interrupt from WD1001
4. Check Error bit in Status Register for proper completion

Note: Steps 1 and 2 above can be reversed.

7.4.1 Formatting

The format command is very similar to the write sector command, except instead of filling the sector buffer with user data, it is filled with interleave and bad block information. Two bytes will be written to the buffer for each sector to be formatted.

The first (lower) byte will be either a 00 or an 80 in hex. If the lower byte is a 00, the sector is marked as good. If the lower byte is an 80, the sector will set the Bad Block bit in the Status Register if there is any attempt to read or write to it. Please see cautions in section on bad block mapping.

The second (upper) byte is the logical sector number of the next sector to be formatted. This number will be recorded on the disk. The Sector Number register is not used during Format.

On a 32 sector per track disk, 32 pairs of formatting information must be supplied to the drive during each format operation. To start the format operation the buffer must be completely filled, even if the sector table is not as long as the buffer. On a 32 sector per track disk, 64 bytes of formatting information are supplied. If the sector size is 256 bytes then 192 bytes of garbage must be passed to the controller to start the format operation.

Since the contents of the sector buffer do not imply how many sectors are to be formatted, a dedicated register is provided. This **Sector Count** register must be loaded with the number of sectors to be formatted before each and every format operation. To calculate the maximum number of sectors per track, see Appendix C.

7.4.2 Interleaving

If we try to read physically sequential sectors on the disk, there is not enough time for us to set up to read or write the next sector before it has passed by the read/write head. This means that the disk will have to make a complete rotation to pick up the next sector. If we were to read all 32 sectors on a particular track it would take 32 rotations, or about a half a second per 8K bytes. This performance can be tremendously improved by allowing the system to read or write more than one sector per rotation. This can be accomplished with interleaving.

Suppose our system takes less than three sector times (3/32 rotational period with 256 byte sectors) to digest the data that it has read and to set up the next read operation. That means that if we can arrange to

have the second logical sector placed physically only four sectors away from the first one, the controller will be able to read it without much delay. This four to one interleave factor will allow us to potentially read the entire track in only four rotations. In our particular example, this will increase the throughput by a factor of eight.

The simplest way to determine the optimum interleave for any particular system is through experimentation. If the system maintains its directories or virtual memory swapping areas in a certain place on the disk, it sometimes makes sense to have **more than one** interleave.

To simplify driver software, the WD1001 will automatically map logical to physical sectors to achieve interleave. This logical to physical map is recorded on each track of the the disk in the ID fields of the sectors. This map is recorded on the disk during the format operation.

Here is an example of an interleave table for a 32 sector track with 4:1 interleave and no bad blocks:

Interleave table with 32 sectors and 4:1 interleave

```
00 00 00 08 00 10 00 18 00 01 00 09 00 11 00 19
00 02 00 0A 00 12 00 1A 00 03 00 0B 00 13 00 1B
00 04 00 0C 00 14 00 1C 00 05 00 0D 00 15 00 1D
00 06 00 0E 00 16 00 1E 00 07 00 0F 00 17 00 1F
```

Remember: The balance of the buffer must be filled with something to start the format operation.

The first byte in each byte pair in the preceding example is set to 00. This marks each block as a 'good' block. The second byte of each byte pair is the logical sector number. The first byte pair above represents the first logical sector of the track. The underlined byte pair represents the second logical sector.

7.5 Bad Block Mapping

The Winchester and thin film technology drives that interface to the WD1001 often do not have perfect media. Imperfections in the media allow much more latitude in what the media manufacturers can ship, significantly bringing down the cost of the media and, consequently, the drives.

The user is required to map out these imperfections. There are many ways it can be done, some of which are highly operating system dependent. Here are a few ideas:

7.5.1 Sector Pre-allocation

If the operating system supports random sector or group allocation, the bad blocks can sometimes be mapped out by recording an un-deletable file using all the bad sectors on the disk. When the operating system tries to write to the bad block, it will see that the sector or group that contains the error has already been allocated. The operating system will automatically map over the bad sector.

There are a couple of minor restrictions associated with this form of bad block mapping. The file that contains the bad sector must never be moved to another section of the disk. The bad sector file may not be read (for obvious reasons) and reads or writes to the disk, that do not consult the disk allocation map (physical reads/writes), are not allowed.

7.5.2 Alternate Tracks

This method works on most operating systems but, it requires more software overhead. Whenever a read or write is attempted, the track number (cylinder and head select) is checked against a table maintained by the operating system or driver. If the track number matches the table, the driver knows that there is a flaw somewhere on that track. The driver will look up the alternate track for that flawed track and the read or write will be performed elsewhere.

The primary disadvantages of this type of bad block mapping is its rather high software overhead. When the system is brought up, the alternate track table has to be read from some flawless area of the disk. After it has been read, every read or write operation must check the alternate track table before performing its respective operation.

The WD1001 bad block marking and detection facility is useful for eliminating the software overhead of looking up each track number before each write is performed. During format, all the sectors of the bad track should be written redundantly with the bad block bit set (See section on Bad Block bit below). When any read or write is attempted on the bad track, the WD1001 will interrupt with the Error bit in the Status register and the Bad Block bit in the Error register set. The driver can then then look up the alternate track in its internal table and resume the operation.

7.5.3 Spare Sectors

This method is probably the simplest to implement in most systems. Its primary disadvantage is that at least one sector must be set aside as a spare for each track. During format, the physical sector that contains the flaw is written with some illegal sector number. The physical sector following it contains the real logical sector and its data. In the following interleave table, the user mapped out the fifth physical sector by telling the WD1001 to write a logical sector number of FF to it.

Interleave table with 32 sectors and 4:1 interleave with physical sector five mapped out:

```
00 00 00 08 00 10 00 18 00 FF 00 01 00 09 00 11
00 19 00 02 00 0A 00 12 00 1A 00 03 00 0B 00 13
00 1B 00 04 00 0C 00 14 00 1C 00 05 00 0D 00 15
00 1D 00 06 00 0E 00 16 00 1E 00 07 00 0F 00 17
```

Please note that when formatting the disk in this manner, at least one sector must have an illegal sector number. Also, since we have allocated one sector to bad block mapping, we no longer have a sector 1F.

7.5.4 Bad Block Bit

The WD1001 allows the user to set a marker that is recorded into the ID field. When the WD1001 attempts to read or write a sector with a bad block mark set, the operation will be aborted and the Error bit in the Status register and the Bad Block bit in the Error register will be set. The Size, Head, Cylinder, Sector and ID CRC fields of the selected sector must be correct in order to detect a bad block mark. This means the ID field must be error free in order to detect the bad block mark.

To insure that the Bad Block bit can be read even though some ID fields may be defective, the ID fields must be recorded redundantly. In order to make this possible, the WD1001 includes a special variation of the Format command. When a Bad Block bit in the interleave table is set during format, the WD1001 records only the ID field of that sector. No data field is recorded. This helps to make room for redundant ID fields.

In the following interleave table, the user has marked all the sectors with a bad block mark and recorded all sectors redundantly. The interleave is not very important here because the driver (hopefully) will not attempt to read bad sectors sequentially.

Interleave table with redundant sectors, no interleave, and all sectors marked as bad blocks:

```
80 00 80 01 80 02 80 03 80 04 80 05 80 06 80 07
80 08 80 09 80 0A 80 0B 80 0C 80 0D 80 0E 80 0F
80 10 80 11 80 12 80 13 80 14 80 15 80 16 80 17
80 18 80 19 80 1A 80 1B 80 1C 80 1D 80 1E 80 1F
80 00 80 01 80 02 80 03 80 04 80 05 80 06 80 07
80 08 80 09 80 0A 80 0B 80 0C 80 0D 80 0E 80 0F
80 10 80 11 80 12 80 13 80 14 80 15 80 16 80 17
80 18 80 19 80 1A 80 1B 80 1C 80 1D 80 1E 80 1F
```

SECTION 8 THEORY OF OPERATION

8.1 General

The WD1001 hard disk controller is a discrete implementation of all functions required to control SA1000/ST506 compatible Winchester hard disk drives via a standard data and control bus. The controller is fabricated using a mix of high-speed bipolar and NMOS devices contained on a single, 2-sided PC board. The design of the circuitry makes use of a high-speed Microcontroller, the 8X305, newly developed NMOS support devices, Schottky, and low power Schottky devices to achieve low component count and low cost while maintaining high performance and reliability. All I/O connections are made using standard ribbon cable connectors. Standard pin-out configurations for disk interface connectors are provided to permit direct pin-for-pin connection to SA1000 compatible 8″ drives and ST506 compatible 5-1/4″ drives. All power for the board can be supplied from a single +5 Volt power supply on a separate connector. All host to disk data transfers are buffered by an onboard RAM to achieve totally asynchronous transfers to and from the disk by the host. In addition, the WD1001 has the ability to perform error correction (ECC) using firmware and a custom designed NMOS device.

The WD1001 is available in two different PC board layouts. The larger size board is designated the WD1001-55 or -85 depending on the size of drives that is supports. The smaller board is designated the WD1001-05 and it controls 5-1/4″ drives only. The component designations differ for the two products. **The component designations will be printed together with the WD1001-55 and -85 first and the WD1001-05 second.** A colon (:) will be used to separate the two designations.

The disk controller is built around 5 basic sections:

1. Processor functions
2. Serial data separation
3. Data conversion, checking and correcting
4. Serial data generation
5. Host interface functions

8.2 Processor Functions

All functions of the WD1001 controller are ultimately controlled by the onboard processor. Due to the high data rates associated with hard disk drives, processing of data and control of machine functions within the circuitry requires a processor capable of extremely fast execution speed. The processor used is the 8X305 (or 8X300), a bipolar microcontroller, particularly well suited for handling data efficiently at high rates.

The 8X305 is operated at a basic clock rate of 8 MHz and performs all operations within 2 clock cycles, giving it a speed of 4 MIPS (Million Instructions Per Second) or one instruction executed every 250 nS. The architecture of the processor is different from most popular microprocesssors in that no common data or address bus is provided to be shared by RAM, ROM or peripheral devices.

Instructions are fetched from ROM via a dedicated instruction address and data bus. The instruction address bus (IA13-IA0) is capable of directly accessing 8K words of program storage. The WD1001, however, uses only the first 10 address lines, limiting onboard program storage to 1K words. Program data is input to the 8X305 (U46:U9) on the Instruction Data Bus (ID15-ID0) as 16 bit words which are decoded to perform the desired operation. All bus designations utilized by the 8X305 are reversed from the traditional LSB to MSB weighting. On the WD1001 these lines have been renamed on the schematic, using traditional weighting to provide a more conventional designation system for the board.

Data is transferred between the processor and its ports on a separate 8-bit bus called the "IO" bus. This bus is active low. It must be noted that this bus is in no way related to the instruction data bus and should be thought of as simply an 8-bit bidirectional IO bus for the 8X305. In fact, it has been renamed as IO0-IO7 to reflect this definition.

8.2.1 Fast IO Select

An extension byte has been added onto the instruction data memory (U40:U14) to provide port access decoding on an instruction by instruction basis. This "Fast IO Select" byte is not processed by the 8X305; rather it is decoded by auxiliary hardware to provide 8 read strobes, 8 write strobes, and 8 single bit output ports which route data to the various devices distributed along the IO bus.

The write part of the Fast IO byte is latched into a 4-bit latch (U26:U25) and the input of a decoder/latch (U33:U30) on the trailing edge of MCLK. This ensures that data remains stable during the entire instruction. The read strobe and write strobe are selected through a pair of 1-of-8 decoders (U33:U30 and U34:U24) which are alternately enabled by the SC- control strobe produced by the 8X305. It also provides latch-address data to the addressable latch (U27:U27) for drive and ECC control signals. To provide edges on read strobes during sequential read operations from various ports, the read strobe decoder (U33:U30) is always disqualified at the end of instructions by MCLK-. Because each decoder has a unique input and the latch is directly addressable, it is possible to select any read port with any write port, during each instruction.

8.2.2 Internal Bus Control

Several bus control signals are produced by the 8X305 to identify and strobe the data on the IO bus. SC- is a signal which determines the direction of the data to and from peripherals. When SC- is false, (during the first half cycle) the 8X305 inputs data from the IO bus. When SC- is true (only during the second half cycle), the 8X305 outputs data to the IO bus. The WD1001 uses SC- to indicate all writes to any port. This arrangement allows 8-bit immediate data moves from the 8X305 to any output port within one instruction, instead of the normal 5-bit immediate moves provided for by the instruction set.

All instruction fetches occur late in the second cycle of the preceding instruction. This time is marked by the generation of a 65 nS (nominal) active high pulse called MCLK, which occurs every instruction. On some ports, MCLK is also used to latch data prior to being input on the IO bus to insure stability during reads, and to disqualify read strobes which would otherwise remain true into the second clock cycle of any instruction which does not write to a port.

All I/O ports on the 8X305 are logically divided into two address spaces. This address space is qualified with RIGHT BANK SELECT (RB-). All WD1001 I/O ports, except RAM, appear in the left bank address space. The RAM is placed in the right bank and the right bank signal is run directly to the CS- inputs of the RAMs to avoid the propagation delays associated with the Fast I/O Select logic. This allows slower RAMs to be used and provides better access margins on read operations.

8.2.3 Reset Circuit

The 8X305 is held reset for approximately 40 mS after initial power-on. This is accomplished by an RC network (R24:R5, C24:C16 and CR1:CR1) which drives a Schmitt trigger (U39:U17) to provide a proper rise/fall time on the RESET- line of the 8X305 and various port latches. Alternate reset of the WD1001 can be accomplished by asserting MR- (J5 pin 39) whenever the host wishes to reset the controller. A Schmitt trigger (U39:U17) is provided with a 4.7K pull-up (R23:R2) to buffer the MR- input from the host. RESET- also propagates to the drive control latches (U43:U22 and U27:U27), the host interface Support Logic Chip (U19:U28), and the INTRQ and DRQ latches (U16:U23).

8.2.4 Processor Power Supply

Power is supplied to the 8X305 from the +5 Volt (V_{cc}) power bus. Due to the internal operation of the 8X305, an on-chip voltage reference is provided to produce bias to an external pass transistor (Q6:Q1)

which drops V_{cc} to the 8X305 to approximately +3.0 Volts. This supply is used internally by the 8X305 logic and all signals of the 8X305 are internally level shifted to be TTL compatible.

8.2.5 Read and Write Ports

Throughout the circuit, output ports consist of D type latches using write strobes (WR0-7) to latch data into the ports. Reading of ports is accomplished by using read strobes RD0, RD2, and RD4-6. The read strobes individually enable selected tri-state output devices on the IO bus. Additionally, two read strobes are used to clock the host DRQ and INTRQ latches (U16:U23) for instructions not requiring data from a port. This ensures glitchless operation of the Fast IO port decoders.

8.2.6 Read/Write Memory

Since the 8X305 does not permit data to be saved or retrieved from dedicated program storage, RAM must be installed on the IO bus and it must be accessed via the IO bus by I/O instructions like all other port accesses. To provide for addressing the RAM, three latch/counters (U30:U7, U37:U3 and U42:U6) are connected to the IO bus to receive and store addresses required to access the RAM (U29:U8 and U36:U4).

8.2.6.1 RAM Addressing

The RAM address bus (RA0-RA9) uniquely addresses 1024 memory locations. As each counter chip reaches a count of 0, it will set a borrow condition to the next higher counter which will be decremented at the end of the next access to RAM. When all bits of the address have been reset, the ROVF- bit on the last counter (U42:U6) will be reset, providing an overflow status which can be read by the processor on U41:U4. By setting various beginning address values, ROVF- can be used to mark the end of any RAM access loop from 1 to 1024 bytes in length. In the WD1001 this function is used for setting sector buffer lengths of 128, 256 and 512 bytes in the normal mode and 132, 260, and 516 bytes in the long mode.

8.2.6.2 Sector Buffering

All data read from the disk or written to the disk is passed through the RAM to provide buffering required for asynchronous data transfer between the host and disk. The counters are post-decremented so that addresses are stable to the RAM by at least one instruction prior to the actual access. This preselection feature effectively reduces RAM access time to the output enable and propagation time of the RAM for read operations and the width of the minimum WR- strobe pulse for write operations.

8.2.6.3 RAM Accessing

RAM access is initiated by RB- which is output by the 8X305. Data to be read from RAM will be placed on the IO bus whenever RB- is low and SC- is high. Data is written into a selected RAM cell on the trailing edge of SC- if RB- is low. During writes, RB- will be low for at least 120 nS so that data setup time requirements are met.

8.2.6.4 Scratchpad Operations

Because the RAM address counters are presettable, direct reads and writes to a specific address are possible. However, resolution of the address is limited to the nearest fourth address location. This limitation is imposed so that a full ten bit address can be specified with a single eight bit output. The least significant two bits are implied to be 10 (binary).

8.2.7 Miscellaneous Control Ports

Control of the various functional sections of the WD1001 is accomplished by a dedicated 6-bit control port called MAC CNTRL (U31:U20) and an addressable latch (U27:U27). MAC CNTRL enables the functions of the WAIT control circuitry (WAEN-), ECC generation (ECCIZ-), gating of read data into data separation circuitry (RGATE), selection of read or write functions (WRITE-), control of ECC check word output (1BLA-) and AM detection (SRCH). MAC CNTRL output states are latched into the port by a write strobe (WR7).

The addressable latch controls drive functions and CRC/ECC selection. This latch enables direction of the drive (DIRIN-), stepping pulses to the drive (STEP-), reduced write current (RWC-), write precomp (WPC-), write enable (WRITE GATE-), and the switching enable between CRC and ECC (CRC-/ECC).

All remaining ports are distributed among the basic functional sections of the WD1001 and will be described in detail within the discussion of those functions.

8.3 Serial Data Separation

The WD1001 controller utilizes an NMOS device (WD1100-09) especially designed to process incoming MFM data from the drive by a process called data separation. Here, some background information may be helpful:

> In order to provide maximum data recording density and storage efficiency, data is recorded on the disk using a Modified Frequency Modulation (MFM) technique. This technique requires• clock bits to be recorded only when two successive data bits are missing in the serial data stream. This reduces the total number of bits required to record a given amount of information

on the disk. This results in an effective doubling of the amount of data capacity, hence the term "double density".

Because clock bits are not recorded with every data bit cell, circuitry is required that can remain in sync with data during the absence of clock bits. Synchronous decoding of MFM data streams requires the decoder circuitry to synthesize clock bit timing when clocks are missing and synchronize to clock bits when they are present. This is accomplished by using a phase locked oscillator employing an error amplifier/filter to sync onto and hold a specific phase relationship to the data and clock bits in the data stream. The synthesized clock called RCLK can then be used to separate data bits from clock bits and to shift the resultant serial data into registers for byte parallelization.

8.3.1 Incoming Data Selection

In the WD1001, serial data is input from up to 4 radially connected drives via a quad RS-422 differential receiver (U3:U38). The receiver converts differential input data to TTL levels for use by the controller. The data from the selected drive is then routed to the data separation circuitry by a 4-section AND/OR/INVERT gate (U9:U33). Due to the fact that different drives produce varying data pulse widths, the data is first routed through a one-shot (U7:U39) to provide a consistent data pulse width. At this point data and clocks are still combined and appear as 50 nS nominal active high pulses spaced at intervals of 1, 1.5 or 2 times the RCLK period. This data is presented to the Data Separater chip (U6:U34) which will then gate either MFM data or a reference clock into the first stage of the VCO error amplifier circuitry.

8.3.2 Reference Clock

The reference clock is derived from the write clock crystal oscillator (U4:U14 and associated circuitry). This oscillator uses a fundamental crystal cut to oscillate at 4 times the RCLK frequency. The 4X output is then divided by U4 to produce both a 2X clock (2XDR) which is used as a reference and a 1X clock (WCLK) which is used to produce MFM write data for the disk. The crystal (Y1:Y2) frequency is 20.000 MHz for ST506 compatible drives or 17.360 MHz for SA1000 compatible drives.

8.3.3 Clock Gating

The gating of the reference and MFM data into the data separator is dependent on the condition of the read gate (RGATE) signal and the spacing of the data on the serial stream after RGATE is brought true. Due to the techniques which are employed to separate data from clocks, it is necessary to run the VCO at a rate twice the data clock (RCLK) rate. The VCO is set to an open loop frequency of 2X RCLK. Any variations

in this rate due to variations in disk rotational speed must be compensated for by the VCO, but instantaneous shifts in data due to the effects of adjacent bit cells on the disk and minor noise must be ignored. Also, the response of the VCO must be adjusted to effectively ride over missing clock bits which occur as a result of the MFM recording technique. The resultant compromise between response and reject requirements of the VCO cause the VCO to have a tendency to become locked onto harmonics of the data rate rather easily. This is likely to occur if the VCO is connected to a data stream over a field of data which has data bits spaced at 1.5 or 2 times the actual RCLK time intervals.

To provide protection against this undesirable condition, the VCO is always held locked onto a stable clock running at 2X RCLK frequency whenever the controller is not actually reading data. Care is taken to switch in read data to the VCO error detector only when it is known that the data stream frequency is equal to the RCLK frequency. This can occur only when the data is a solid stream of all ones or all zeros.

8.3.4 High Frequency Detector

The switch from reference clock to live data is initiated immediately after the RGATE goes true and will only occur after 16 consecutive ones or zeros (high frequency) are detected on the raw MFM. This detection is accomplished by a one-shot (U7:U39) and the data separator. The one-shot is adjusted for a pulse width of 1.25 times the RCLK period (250 +/- 10 nS for ST506 compatible drives and 287 +/-10 nS for SA1000 compatible drives). The adjustment of the one-shot provides tolerance of up to 1/4 RCLK period in jitter on the MFM data bits while still being able to distinguish MFM zeros or ones from other data patterns.

Each clock or data bit on the serial stream triggers the one-shot. If the time between successive triggers is less than the one-shot time constant, the one-shot remains retriggered. As the one-shot is triggered by data stream bits, so is a counter in the Data Separator Device (U6:U34), whose reset is controlled by the state of the one-shot outputs. While the one-shot is being retriggered the counter counts up. When any data bit fails to reach the one-shot before its time constant is over, the one-shot resets and in turn clears the counter. Only when 16 successive retriggers occur can the counter reach its terminal count. At this time, the counter overflow goes true and sets a latched DRUN- output low, which switches read data in and reference clock out. DRUN- is read through U38:U18 by the 8X305 to determine the condition of the MFM data stream.

At this point data and clocks have finally been connected to the first stage of the data separator. The heart of the data separator is the VCO (U1:U42), the error amplifier, filter and the Data Separator Support Logic Chip (WD1100-09).

8.3.5 Sample on Phase Detection

When an input signal is applied to the system, its phase relationship is detected within the Data Separator Device. The function of this phase detector is to provide windows, during which the leading edge of the incoming MFM data can be compared to the leading edges of the VCO clock. The windows are approximately 50 nS in length. The windows are initiated by the leading edge of any data bit as it enters the detecter. They are terminated by that same data bit, edge delayed by 60 nS or the VCO output (OSC-). When both the delayed data bit (delayed by DL1:DL1) and the nearest VCO edge arrive at the detector, the detector is reset until the next data bit arrives on the MFM data stream. The delayed data bit sets its half of the detector latches to produce a pump up condition at the error amplifier. The VCO clock edge sets its half of the detector to produce a pump-down condition. When the circuit is balanced, both pumps are on or off, producing no net pump-up or pump-down.

8.3.6 Error Amplifier

Control of the VCO is accomplished by the error amplifier, filter, and Data Separator Chip. The error amplifier is a balanced current mirror, whose output sources or sinks current to the filter stage. Whenever the VCO is running too slow, the error amplifier receives pulses from data bits before pulses from the VCO clock. This causes the error amplifier to produce pump-up pulses to the filter. The filter integrates these pulses, producing an average increase in the voltage to the VCO. Whenever the VCO is running too fast, the error amplifier produces pump-down pulses to the filter. It must be noted, however, that some slight error will always be present because, without pumps, the filter will float and the VCO will drift. The overall gain of the error amplifier and the VCO will maintain this error very small, resulting in very close tracking between the VCO output phase and the incoming data phase.

8.3.7 VCO

The WD1001 uses a single chip VCO (U1:U42), which simplifies circuitry and adjustments. The operating point of the VCO is initially set by adjusting the variable capacitor (C8:C28) for a 10 Mhz (when running on 5 Mhz drives) output center frequency and the frequency control voltage input to 2.5V +/- 0.5V. It should be noted here that the frequency range voltage and the frequency output are adjusted to the 'locked' center frequency with the same variable capacitor adjuster.

The output of the error amplifier and filter is fed to the VCO and represents how far the VCO frequency is from that of the incoming signal. The error signal, which is proportional to the difference, allows the VCO frequency to shift from center frequency and become the same frequency as the input signal. When the loop is in lock, the difference frequency component (error voltage) is DC and will be passed by the low pass filter. Thus, the lock range is limited by the range of the error voltage that can be generated. The lock range is essentially a DC parameter and is not effected by the band edge of the low pass filter. It can be defined as the frequency range, usually centered about the VCO initial free running frequency, over which the loop can track the input signal once lock has been acquired.

Frequency control is actually a matter of frequency range. The difference component may fall outside the band edge of the low pass filter and be removed along with the the sum frequency component. If this is the case, no information is transmitted around the loop and the VCO remains at its initial free running frequency. As the input frequency approaches that of the VCO, the frequency of the difference component decreases and approaches the band edge of the low pass filter. Now, some of the difference component is passed, which tends to drive the VCO towards the frequency of the input signal. This, in turn, decreases the frequency of the difference component and allows more information to be transmitted through the low pass filter to the VCO. This is essentially a positive feed-back, which causes the VCO to snap into 'lock' with the input signal. With this in mind, The term 'capture range' can be defined as the frequency range centered about the VCO initial free running frequency over which the loop can acquire lock with the input signal.

As previously stated, the VCO runs at a frequency twice that of the RCLK rate. By setting the center frequency equal to twice the data rate, the VCO will lock to the data and give an exact synchronized clock.

8.3.8 Window Extension

Once the VCO has been locked onto the phase of the incoming data, the actual separation of data and clocks can occur using a technique called window extension. This technique causes data bits to first have their leading edges shifted into the center of the RCLK half cycles and then to be latched or extended until the next rising edge of the RCLK. The delayed data clocks a pair of latches (U13:U37). The 'data' latch has its D input and CLEAR connected to +RCLK and the 'clock' latch has its D input connected to RCLK-.

If an MFM data bit enters the latches while RCLK+ is high, it will be extended as a data bit. If RCLK- is high, it will be extended as a clock bit. Due to the this extension technique, bits can jitter approximately ¼ the RCLK period without being lost. The output of each latch is then further extended by feeding directly into another stage of latches (U14:U32) and clocked on alternate edges of RCLK. The final outputs of the data extension/separation stage are two separate signals, one consisting solely of NRZ data, and the other NRZ clocks. NRZ data and clocks are finally in a form suitable for processing by subsequent circuitry within the WD1001.

8.3.9 Clock Detection

Due to the nature of MFM data encoding, it is impossible to know exactly if MFM bits are data or clocks. This ambiguity results in having to assume that bits on RCLK- are actually data bits until the VCO is locked on and a unique data/clock pattern is detected. This is accomplished by holding the VCO to RCLK divider (U12:U36) reset until it is fairly certain that bits on the data stream are actually clocks belonging to a field of zero data.

Once this assessment has been made, the processor releases the AM detector (U20:U31) by raising the SEARCH signal. This signal releases a latch (U12:U36) which will remove DHOLD- from the RCLK divider on the next rising edge of a MFM data bit so that CLOCKS will be on the RCLK- phase and DATA will be on the RCLK+ phase. The processor makes its assessment of the state of the data stream solely on the occurrence of a significant run of high frequency as detected by the one-shot (U7:U39) in the DRUN circuit. Once released, the phase of RCLK versus data and clocks will remain stable throughout the read of an ID or data field. Whenever SEARCH is dropped, the VCO to RCLK divider is once again reset and no RCLKS are produced.

8.4 Data Conversion and Checking

MFM data which has been separated to form NRZ data and clocks are processed through specialized circuitry to prepare it for parallel processing by the 8X305. This processing consists of 3 functional circuits.

1) AM detection (U20:U31)
2) ECC/CRC checking circuit (U18:U35)
3) Serial to Parallel Conversion (U25:U13)

Each function will be discussed separately but bear in mind that many interdependencies exist.

8.4.1 AM Detection

As previously stated, it is impossible to know whether serial data bits are actually data or clock bits by just looking at the data stream. Also, it is equally impossible to determine byte boundaries of the data stream. This problem is solved by a uniquely recorded data/clock pattern called an Address Mark (AM). The AM consists of a data pattern of hexadecimal 'A1' with a missing clock pattern of hex '0A'. Normally a data byte of hex 'A1' requires a clocking pattern of hex '0E'.

The AM is used to uniquely identify the start of a field of information (Data or ID field) within each sector. Preceding each AM on the disk there is always a long run of 'zero' data. Zeros have a clock bit for every RCLK. When attempting to read information from the disk, the WD1001 first acquires phase lock over a field of zeros. After this acquisition is achieved, the processor releases the AM detector (U20:U31) by raising the SEARCH control line (SRCH) on the MAC CNTRL port (U31:U20). Due to the circuitry associated with the VCO to RCLK divider, the RDAT- output of the data separator will be high and the CLKS- output will be low. RCLK- will be the shifting clock for RDAT- and RCLK+ will be the shifting clock for CLKS-. These 4 signals are routed into the AM detector.

Inside the AM detector, the RDAT- is shifted into an 8-bit synchronous serial shift register and clocked on the falling edge of RCLK-. CLKS- are shifted into a similar shift register on the falling edge of RCLK+. The output stage of the RDAT- register is dumped into an 'A1' comparator and the output stage of the CLKS- register is dumped into a '0A' comparator. AM detection occurs when both detectors are true, setting the AMDET latch. At the instant AM occurs, the exact relationship between data and clocks is known. It is also known that data is being clocked by RCLK- so CLKS- can actually be discarded; its only purpose was in detecting AM. The AMDET- signal is used as a synchronization signal to start subsequent conversion circuitry. The AMDET- signal remains true until the processor again de-asserts the SEARCH control line.

The AMDET- signal is processed by a D latch, located in the Support Logic device (WD1100-07), to precisely time the leading edge of AMDET- to the rising edge of RCLK. This synchronization relaxes timing requirements on the data and clock inputs of the serial to parallel converter (U25:U13). Also, a one-bit delay is placed in the AMDET path within the Support Logic Device to compensate for correct timing with the ECC architecture.

8.4.2 Error Detection and Correction

Data recorded on magnetic media is prone to several types of errors which could render data unusable if some form of error detection were not employed.

On the WD1001, error detection is performed on all data transfers from the disk. The ID fields use a 16 bit Cyclic Redundancy Check (CRC) and the data fields use either the same CRC or a special 32 bit Error Correction Code (ECC). The CRC and ECC fields are appended to the data field that they are to protect.

The WD1001 uses the same device to generate and check CRCs and ECCs. Normally, the WD1001 uses CRCs in ID fields and ECCs in data fields. Also, as a software selectable option, the WD1001 may be used in a CRC only mode by appending CRC check bytes to both ID and Data fields. Although either polynomial could be used for both fields, the use of the ECC polynomial is limited to data only. On-the-fly correction of ID fields cannot be done and CRC provides adequate checking for the ID fields. The ECC and CRC polynomials used are as follows:

$$ECC = X^{31}+X^{28}+X^{26}+X^{19}+X^{17}+X^{10}+X^6+X^2+1$$

$$CRC = X^{16}+X^{12}+X^5+1 \quad \text{(Also known as CRC-CCITT)}$$

As data is being read from the disk, the CRC/ECC generator re-computes the original check bits. After all the data are read, the value in the CRC/ECC generator is exclusive ORed with the check bits recorded on the disk. The result is called the syndrome. If the syndrome was zero, the data was correctly read. Otherwise, an error occurred. If the field was protected by a CRC, the data can often be recovered by a retry which the WD1001 performs automatically. If the field was protected by an ECC, the non-zero syndrome is used by the on board processor to compute the displacement and the error vector within the sector. This information is then used to correct the data if a single burst of no more than five bits in error occurred.

The CRC/ECC generator is initialized by setting ECCIZ- low for at least 250 nS during the search for the AM. ECCIZ- is originated on the MAC CNTRL port (U31:U20). Upon receiving the ECCIZ- signal, the ECC generator will preset all 32 of its internal polynomial division shift registers to logic ones and arm an internal latch which will start the CRC/ECC generator on the leading edge of the first non-zero bit (hopefully an AM) to enter the device.

Once enabled by the first non-zero data bit, the ECC device (U18:U35) will shift succeeding data bits into a feedback shift register string with exclusive OR gates tied to the feedback nodes of the registers. As each

RCLK occurs, the registers will divide the incoming data and a unique pattern of ones and zeros will appear across the registers. The ECCEN input line is set low, indicating that the internal circuitry is ready to begin the computation of the ECC/CRC check bytes.

Sometime before the last byte of data and after the next to the last byte of data is transferred through the device, the DCSS line is set low. When the last bit of an ID or data field is processed, the pattern in the registers should be equivalent to the 16 or 32 check bits appended to the fields during original recording. The check bits on the disk are exclusive ORed with the check bits in the CRC/ECC device to produce the syndrome.

Data is deserialized after being processed by the ECC/CRC device and Byte-Sync boundaries are marked by byte-sync pulses obtained from the Data Support logic device on the RBS input. The byte-sync pulses are internally ANDed with the RWCP line to insure the smooth transition of check/syndrome bytes on the DOUT output line, after the last bit of data has been entered into the device. A one-bit time delay occurs on the DOUT line because an internal latch is used to deglitch the output line.

8.4.3 Serial to Parallel Conversion

After data has been processed by the ECC device, the Serial to Parallel Converter (U25:U13) takes over. NRZ data and RCLK are used to shift data bits into an 8-bit serial to parallel shift register. As each bit is shifted, a divide-by-8 counter circuit is incremented. After every eighth bit of data is shifted, the counter produces an overflow pulse, marking byte boundaries in the serial data stream. The overflow bit from the counter resets the counter, clocks the data from the shift register into an 8-bit parallel latch, and sets a tri-state flag register called BDONE. The flag can be read by the processor to see if any converted data is ready to be read from the latches.

When the processor sees BDONE in the true state, it services the device by gating data onto the IO bus using read strobe 4 (RD4-) in conjunction with a tri-state buffer (U24:U12). The act of reading the latches also clears the BDONE flag. As successive bytes are processed, BDONE is serviced by the processor as data becomes available.

8.5 Serial Data Generation

The WD1001 records data on the disk in MFM format. In order to produce the proper data format, the WD1001 uses several specialized devices to process the parallel data supplied by the host into a serial MFM data stream. The data supplied by the host is temporarily stored in the buffer RAM until the correct sector is located for the data to be written.

The process of writing is essentially the opposite of reading except that the data separator circuitry is not required and the generation of the MFM data stream is produced by synchronous clocking techniques.

The functional sections of the serial data generation section are listed below:

1) Parallel to Serial conversion (U17:U19)
2) ECC/CRC generation (U18:U35)
3) MFM and precompensation (U16:U23)

8.5.1 Parallel to Serial Conversion

Parallel data is converted into a serial NRZ data stream by the Parallel to Serial device (U17:U19). The processor enables this conversion by lowering the WRITE- signal on MAC CNTRL (U31:U20). WRITE- causes the tri-state buffers present on the parallel to serial device to become active, supplying the ECC device with data, clocks, and BDONE strobes.

The processor presents parallel data on the IO bus along with the WR4- write strobe which latches the data into the parallel port on the trailing edge of the strobe. The write strobe also resets any pending BDONE. Inside the parallel to serial device, the parallel latches are loaded into a serial shift register on every eighth WCLK transition. As the data is transferred to the shift registers, the BDONE status flag is set. The processor reads this flag through U41:U5 to determine when to write the next parallel byte to the device. The timing of the parallel accesses is at a rate 1/8 that of the bit rate of the NRZ data stream. For ST506 compatible drives the byte timing is 1.6 uS and for SA1000 drives it is 1.84 uS.

The output of the last register in the shift string is brought out of the device as NRZ serial data.

Whenever it is desired to write a repetitive string of identical data bytes, the processor can simply ignore the BDONE flag and permit the device to reload the data from its latches over and over again for as long as required to generate the field. This feature of the device is used in writing certain fields used in formatting.

8.5.2 CRC/ECC Generation

The CRC/ECC generator/checker (U18:U35) is used to generate the CRC/ECC bits and to append them to the end of the data being written to the disk. The operation of the polynomial generator is identical to read operations except that at the end of the data field the processor sets a signal which causes the device to output the computed CRC/ECC after the data instead of producing the syndrome.

The initial states of the shift registers within the device are forced to all ones by the processor pulsing ECCIZ- for at least 250 nS while the parallel to serial device is outputting all zeros on the NRZ data line. At that time, a latch is set which holds the registers at ones until the first non-zero data bit enters the device.

The first non-zero bit will be the MSB of the AM (hexadecimal A1) of the data field to be written. When the processor decides that enough zeros have been written to satisfy the sync field requirements, it will store a hex A1 in the parallel to serial device. At the proper time (in sync with BDONE) the parallel to serial device will begin to send the MSB of the AM to the CRC/ECC device. This will start the CRC or ECC polynomial generator and the ECC will be computed.

To write the ECC/CRC check bits, the processor will assert the One Byte Look-Ahead (1BLA-) signal on MAC CNTRL port (U31:U20) just after the last data byte was sent to the parallel to serial converter. The internal switchover from data to check bits is synchronized to the next byte time by the WCP- signal. Once the switch takes place, the CRC/ECC generator will begin dumping the computed CRC or ECC onto the NRZ data stream. The net effect of this is to append the proper CRC or ECC information to the end of a field of data. 1BLA- is maintained true for the duration of the unloading process which lasts for up to four byte times.

During the unloading process, the ECC registers back-fill with zeros. This feature is handy because by leaving 1BLA- low for additional time, zeros will always be written after the CRC or ECC which is a requirement of the format of the disk. The NRZ data from the CRC/ECC device is sent to the MFM generator device.

8.5.3 MFM Generation

The conversion from NRZ write data to MFM write data takes place in the MFM/Precomp device. This device accepts NRZ data and a complementary WCLK and produces MFM data and clocks by sending the data through circuitry which decides when and where to write clocks on the data stream under the MFM encoding rules. The proper encoding of the data into MFM requires the device to apply three rules to the data.

1) If the current data cell contains a data bit then no clock bit will be generated.
2) If the previous data cell contained a data bit then no clock bit will be generated.
3) If the previous data cell and the present data cell are vacant then produce a clock bit in the current clock cell.

The terms 'data cell' and 'clock cell' are defined by the state of the WCLK. While WCLK is low it is a data cell and while high it is a clock cell. It can be seen then, that both clock and data cells are 1/2 the period of WCLK or 100 nS for ST506 compatible drives and 115 nS for SA1000 drives. Also, note that by the rules stated above, a clock and data bit can never occur within the same WCLK period and legal spacings for bits can be 1, 1.5, or 2 times the WCLK period only. The rules are implemented within the device by shift registers that hold the next two, last and present data bits and combinatorial logic. The state of WCLK is considered and the appropriate bit cells are filled and combined on the MFMW output line of the device.

8.5.3.1 Write Precompensation

The MFM data stream is now totally compatible with the recording rules and may be sent to suitable line drivers for transmission to the drive except for one modification. Due to the decreasing radius on the physical surface of the disk, the inside tracks have less circumference and therefore exhibit a increase in recording flux density over the outside tracks. This increase in flux density aggravates a problem in magnetic recording known as 'dynamic bit shift'.

Dynamic bit shift comes about as the result of one bit on the disk (a flux reversal) influencing an adjacent bit. The effect is to shift the leading edge of both bits closer together or further apart than recorded. The net result is that enough jitter is added to the data recorded on the inside tracks to make them harder to recover without error.

Write precompensation is used to reduce the effect of dynamic bit shift. It is a way of predicting which direction a particular bit will be shifted and intentionally writing that bit out of position in the opposite direction to the expected shift. This is done by examining the next two data bits, the last and the present bits to be written and producing three signals depending on what these bits are. The three signals are EARLY, LATE and NOMINAL. They are used in conjunction with a delay line to cause the leading edge of a data or clock bit to be written earlier, later, or on time.

The processor can enable or disable the generation of these signals by controlling the Write Pre-Comp (WPC) line from the addressable latch (U27:U27). When WPC is high, precomp is in effect. When WPC is low, no precomp is generated and the nominal output of the device is held true.

The delay line actually performs the precomp with the help of an AND/OR/INVERT gate (U23:U40). MFMW pulses are applied to the input of the delay line and depending on which of the three precomp signals is present the AND/OR/INVERT gate selects

a different tap on the delay line. Nominal data is actually tapped from the second tap, early data from the first and late data from the third. From the AND/OR/INVERT gate the MFMW data is sent to the input of an RS-422 driver (U15:U24) where it is converted to a differential form and then is sent to the drive.

8.6 Host Interface

All data transfers between the host and the WD1001 take place over an eight bit bi-directional bus (J5) consisting of eight Data Access Lines (DAL0-7). The source or destination register inside the WD1001 is selected by the three address lines (A0-2). All accesses to the WD1001 are controlled by Card Select (CS-), Read Enable (RE-) or Write Enable (WE-). Since the access time for any particular read or write operation will vary, the WD1001 provides a not ready signal (WAIT-). For systems using interrupts and/or DMA, the WD1001 provides INTerrupt ReQuest (INTRQ) and Data ReQuest (DRQ).

Accessing the WD1001 is essentially like accessing variable speed RAM. The host must provide a valid address in A0-2 along with a CS-. Immediately or after a short set up time, the host may assert RE- or WE-. If access time on the WD1001 will be over 100 nS, then WAIT- will be asserted. The host **must** keep all address lines and strobes stable while WAIT- is true. When the WD1001 de-asserts WAIT-, the data has been accepted on a write or the data is on the DAL bus on a read.

8.6.1 Wait Enable

Since most of the registers in the WD1001 are not implemented in hardware, it takes the 8X305 a finite amount of time to actually fetch the requested data on a read or store data on a write. This time varies depending on the amount of processing the 8X305 must do to access the desired register. After the data has been written or read, the WD1001 de-asserts the WAIT- line, allowing the host to terminate the current bus cycle.

The generation of the WAIT- signal is controlled by a bit in the MAC latch (U31:U20) called WAit ENable (WAEN-). If the WD1001 is ready to accept random accesses to its task file, WAEN- will be asserted. On each bus access, the host **must** drop the Card Select (CS-) line on J5. The leading edge of CS- clocks the wait line (Support Logic Chip, U19:U28) transferring the WAEN- state through the chip. This clocking action is required to insure that WAIT- will not be asserted in the middle of any bus access already in progress. After the wait latch has been clocked, CS- causes WAIT- to be asserted to the host.

The WAIT- line is released on the trailing edge of any read or write strobe to the communications latch (U45:U1). This release is caused by the logical OR of RD6- and WR6- in U19:U28 which presets the wait latch to a non-wait request condition. The WAIT- signal is stretched to the trailing edge of the RD6- or WR6-.

If WAEN- is de-asserted, the WD1001 generates no waits at all. In this case, the host will read the dummy status written to the communications latch by the 8X305. This feature is used by the microcode to simulate a busy condition when the host reads the status register in non-interrupt driven systems. When the WD1001 becomes un-busy, the WAEN- line will be asserted and operations on the host interface bus will be monitored once again.

8.6.2 Bus Gating

During all accesses by the host, one of two signals will be produced to gate the bus. During read operations, CS- and RE- are ANDed, producing Bus Output Control (BOC-). This signal gates the contents of the communication latch (U45:U1) onto the DAL bus. During write operations, CS- and WE- produce Bus Input Control (BIC). This signal latches the state of the DAL lines into an internal R/S latch.

8.6.3 Register Selection

The combination of a host read or write operation along with the WAEN- signal being asserted, causes a signal, Card Select ACcess (CSAC), to be generated. The 8X305 samples this signal at U41 every 250 nS, and if asserted, reads the status of A0-2 and WE- (U41:U5). The state of A0-2 and WE- determine which register is to be accessed (A0-2) and in what direction that access will take place.

8.6.4 Interrupts and DRQs

The WD1001 produces INTerrupt ReQuests (INTRQ) to signal the end of all disk operations and Data ReQuests (DRQ) to signal data ready to DMA controllers. INTRQ and DRQ originate on the MFM generator (U16:U23) as an auxiliary function of the chip. The WD1001 sets INTRQ using INTCLK- and sets DRQ using DRQCLK-, both of which are produced by U33. Interrupts are cleared by HSAC- (Host Select Access) and A0, A1 when the host reads the Status register, issues a command, or accesses the Sector Number register. DRQs are cleared when the host accesses the Data or Cylinder Low registers. DRQs will be re-issued for each byte to be transferred. HSAC- is a 200 nS version of the CSAC- signal, which is produced by the Data Separator Support Device (WD1100-08). During Power-On Reset or Master Reset (MR-), INTRQ and DRQ are reset.

SECTION 9 MAINTENANCE

The WD1001 requires no scheduled preventative maintenance. There are three adjustments associated with the data separation circuitry that may need to be adjusted if a drive with a different data rate is installed.

The WD1001 is available in two different PC board layouts. The larger size board is designated the WD1001-55 or -85 depending on the size of drives that it supports. The smaller board is designated the WD1001-05 and it controls 5-1/4″ drives only. The component designations differ for the two products. **The component designations will be printed together with the WD1001-55 and -85 first and the WD1001-05 second.** A colon (:) will be used to separate the two designations.

9.1 DRUN Adjustments

To facilitate the process of acquiring phase lock on data being read from a disk, a hardware detector is utilized to indicate when the read/write head of the drive is over a recorded field of all ones or all zeros. The detector depends on the timing of a one-shot (U7:U39) which is adjustable by the DRUN pot (R32:R8). DRUN must be adjusted according to the following procedures:

The DRUN adjustment is made with the WD1001 in an operating test configuration with a host, drive and power source.

Monitor DRUN- (U7 pin 7:U39 pin 6) with a 10X oscilloscope probe while attempting to read a sector of data from the drive. The scope should be set to trigger on a high to low transition. While observing DRUN-, Adjust R32.R8. The period of the DRUN single shot should be adjusted to 1.25 times the period of RCLK.

Disconnect all test equipment.

Y1:Y2 Frequency	DRUN Period	Freq. Range	Final Setting
20.000 Mhz	250 nS	9.0-11.0 MHz	10.0 Mhz +/-1 KHz
17.360 Mhz	280 nS	7.5-9.5 MHz	8.68 MHz +/-1 KHz

9.2 Oscillator Frequency

Data separation circuitry on the WD1001 uses a voltage controlled oscillator (VCO) which phase locks onto incoming data and provides a clock suitable for separating data and clock bits on an MFM encoded data stream. The VCO must be adjusted using the following procedures:

Set Balance potentiometer R3:R23 to 100 Ohms +/- 5 Ohms. This is a rough balance adjustment.

Connect a frequency counter to the VCO buffered output on U1:U42 pin 10.

Connect a Volt meter to the Center Frequency input of U1:U42 pin 1.

Make all connections to the board, including the host and drive. Adjust the variable capacitor C8:C30 until the frequency output locks onto the desired center frequency for the drive being used which is 10.000 MHz for ST506 or 8.68 MHz for SA1000. Once this 'lock-on' frequency is achieved, continue the same adjustment for an input voltage of 2.5 +/-0.5V.

9.3 Balance Adjustment

The balance adjustment fine tunes the relationship of delayed data (DLYDAT) to read clock (RCLK).

Attach channel 1 of a dual trace oscilloscope to U18:U37, pin 2 (RCLK). Set scope for 20 nS/div, with a positive slope. Attach the other channel to U18:U37, pin 3 (DLYDAT).

While the controller is doing constant reads to a single sector, adjust the potentimeter R3:R23 so that the edge of DLYDAT is moving to the right with respect to RCLK. When the WD1001 starts encountering errors, move the RCLK signal to the left until the WD1001 starts to function again. Note the exact position of DLYDAT on the screen in nanoseconds. We will call this the "right capture value".

RCLK

|←—— RIGHT CAPTURE VALUE
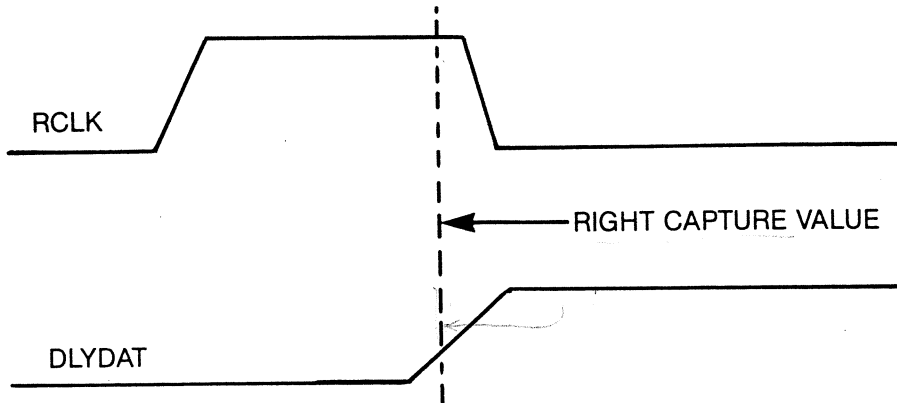
DLYDAT

Continue moving DLYDAT to the left until the WD1001 fails again. Move DLYDAT to the right until the controller functions again. This is the left capture value.

|←— RIGHT CAPTURE VALUE

RCLK

LEFT CAPTURE VALUE —→

DLYDAT
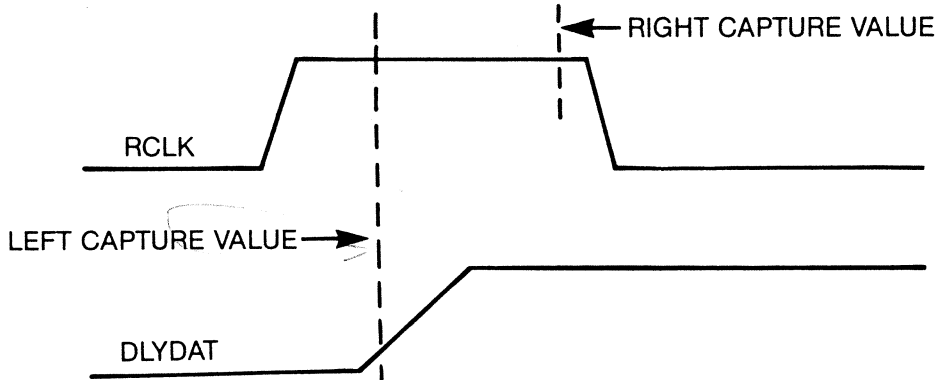
The final adjustment for the WD1001 is determined using the following formula:

(left capture value-right capture value) X 0.7 = Final adjustment value

Adjust pot to final adjustment value.

LEFT CAPTURE VALUE ——→ |←— RIGHT CAPTURE VALUE

70%   30%

RCLK

FINAL ADJUSTMENT VALUE

DLYDAT
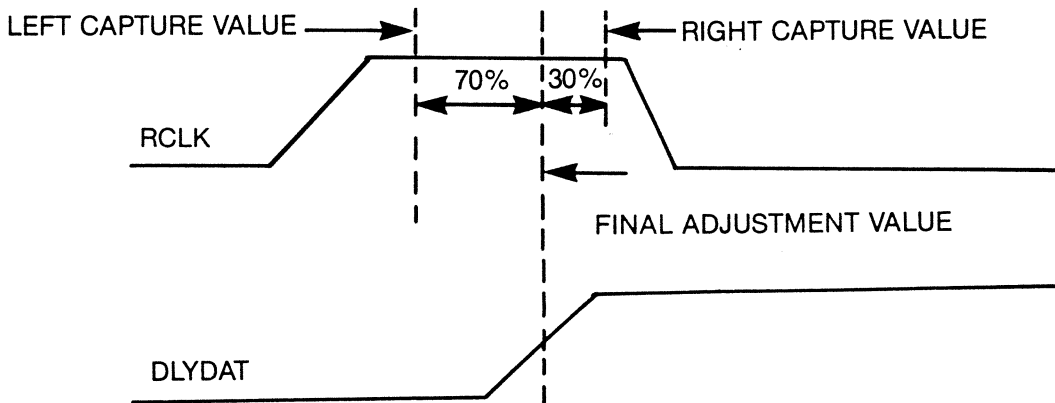
Re-adjust variable capacitor C8:C30 until a value of 2.5 +/- 0.5 volts appears on the voltmeter.

Remove all test equipment.

APPENDIX A DISK DRIVER EXAMPLES

Sometimes an example is worth a thousand words. Hopefully, these sample disk drivers will be the catalyst to get your first driver running. Note that even though these drivers are very simplistic, they represent everything needed to satisfy the WD1001's operating requirements. As you might notice, there is no retry software included in these examples. That is because the WD1001 does all needed retries.

Two examples are presented. The first is a programmed I/O, programmed status driver using the eight bit Intel 8085 microprocessor. The second example is programmed I/O and interrupt driven and is written for the sixteen bit Western Digital WD16 microprocessor.

A.1 Polled Status Driver

```
;****************************************************
;                                                  ;
;                                                  ;
;  WD1001 Hard Disk Controller Driver              ;
;     Example for 8085 Microprocessor              ;
;  with programmed I/O and polled status           ;
;                                                  ;
;                                                  ;
;****************************************************
```

;This driver is intended to demonstrate one simple approach to writing a driver for the
;WD1001. It assumes that the WD1001 is interfaced using programmed I/O without
;interrupts.

;The specifications of the imaginary demonstration drive are:

```
;Sector size:          256 bytes
;Sectors per Track:     33
;Surfaces per drive:    4 (two platters)
;Cylinders per drive:   512
;Stepping rate:         2 milliseconds
STRATE  =    2          ;Define stepping rate for assembler
```

;Since we're allowing the WD1001 to map around the bad blocks for us, we have to
;sacrifice one sector per track. This brings down the logical sector per track count to 32.

;Experienced systems programmers will note that we are not making our driver as flexible
;as it should be. Since WD1001 compatible drives will be introduced in the future and
;present manufacturers will be increasing the density of their current drives, the driver that
;you write should be built with plenty of equates and conditional assemblies.

;Our imaginary operating system can access up to 65536 logical records of 256 bytes
;each. It has three types of calls: Initialize, Read, and Write. Three numerical parameters are
;passed in the following registers:

```
;     Drive number           C
;     Logical record number   DE
;     Transfer address        HL
```

;Upon completion of all commands, the carry bit of the 8085 will be reset if the operation
;terminated properly, and set if there was an error. If there was an error during read or write,
;the error handling routine will decode it and print it out on the user console.

```
;************************
;     EQUATES          ;
;************************

;***   Port Definition   ***
BASADD =    0C8                     ;Base address of WD1001
DATA    =   BASADD                  ;Data register
ERROR   =   BASADD+1                ;Error Register
WPC     =   BASADD+1                ;Write Precomp
SECNT   =   BASADD+2                ;Sector Count
SECNO   =   BASADD+3                ;Sector Number
CYLLO   =   BASADD+4                ;Cylinder Number
CYLHI   =   BASADD+5                ;Cylinder High
SDH     =   BASADD+6                ;Size/Head/Drive
STATUS  =   BASADD+7                ;Status register
COMND   =   BASADD+7                ;Command register

;***   Command Definition   ***


REST    =   10                      ;Restore command
READ    =   20                      ;Read command (programmed I/O mode)
WRITE   =   30                      ;Write command
```

A.1.1 Initialization

```
;************************
;     INITIALIZE       ;
;************************
```

;This routine is called once whenever the system is powered up or reset. It sets the
;stepping rate and restores the head on the selected drive.

```
RESTOR:  CALL   UPTASK             ;Select drive, don't care about record
         MVI    A,REST+<STRATE*2>  ;Get stepping rate and restore
         OUT    COMND              ;Output command to WD1001
RSWAIT:  IN     STATUS             ;Wait 'till restore done
         ANA    A                  ;by updating sign flag in 8085
         JM     RSWAIT             ;and wait 'till bit 7 (Busy) goes low
         RAR                       ;Put error bit in carry
         RET                       ;Return to operating system
```

A.1.2 Read Sector

```
;******************
;     READ        ;
;******************
```

;This is the read routine for our imaginary operating system.
```
READIT:  CALL   UPTASK             ;Update WD1001 task file
         MVI    A,READ             ;Get READ command
         OUT    COMND              ;Output command to WD1001
```

;Wait for WD1001 to read in a sector
```
RWAIT:   IN     STATUS             ;Check Busy bit
         ANA    A                  ;by updating sign flag in 8085
         JM     RWAIT              ;and wait 'till bit 7 goes low
```

```
;Transfer sector from WD1001 to system memory
;(Transfer address in HL)

            MVI     B,0             ;Init byte counter to 256 bytes
READLP:     IN      DATA            ;Get a byte of data from WD1001
            MOV     M,A             ;Move it to memory
            INX     H               ;Increment memory pointer
            DCR     B               ;Decrement byte counter and continue
            JNZ     READLP          ;if we haven't transferred 256 yet
            IN      STATUS          ;Re-read status for errors
            JMP     DONE            ;Now check the completion status
```

A.1.3 Write Sector

```
;******************;
;                  ;
;      WRITE       ;
;******************;

;This is the write routine for the driver
WRITIT:     CALL    UPTASK          ;Update WD1001 task file
            MVI     A,WRITE         ;Get WRITE command
            OUT     COMND           ;Output command to WD1001

;Transfer sector from system memory to WD1001
;(Transfer address in HL)
            MVI     B,0             ;Init byte counter to 256 bytes
WRITLP:     MOV     A,M             ;Get a byte of data from memory
            OUT     DATA            ;Move it to WD1001
            INX     H               ;Increment memory pointer
            DCR     B               ;Decrement byte counter and continue
            JNZ     WRITLP          ;if we haven't transferred 256 yet

;Wait for WD1001 to write the sector
WWAIT:      IN      STATUS          ;Check Busy bit
            ANA     A               ;by updating sign flag in 8085
            JM      WWAIT           ;and wait 'till bit 7 goes low


;******************;
;                  ;
;      DONE        ;
;******************;

;Both READ and WRITE commands finish here to check for errors
DONE:       RAR                     ;Rotate Error bit to carry
            RNC                     ;and return to OS if
                                    ;no error
            IN      ERROR           ;Get WD1001 error code

;<<< Place error reporting routine here >>>

            STC                     ;Set carry to flag an error
            RET                     ;and return to OS with error
```

A.1.4 Task File Updating

```
;*******************************
;                              ;
;  UPTASK SUBROUTINE  ;
;*******************************
;
```

;This subroutine sets up the task file registers

```
;Sector number
UPTASK:   MOV    A,E           ;Get lower 8 bits of record number
          ANI    31.           ;Mask off lower 5 bits (bits 0-4)
          OUT    SECNO         ;and send to sector number register


;Size/Drive/Head
          MOV    A,E           ;Get lower 8 bits again
          RLC                  ;Rotate remaining 3 bits
          RLC                  ;to get an effective right shift of 5
          RLC                  ;Mask off next two bits (5-6)
          ANI    3.            ;to make head number
          MOV    B,A           ;and store it away momentarily
          MOV    A,C           ;Get drive number
          ADD    A             ;and left shift it by 3
          ADD    A
          ADD    A
          ORA    B             ;OR in head number and
          ORI    80            ;OR in ECC flag and size field
          OUT    SDH           ;send it to Size/Drive/Head register


;Cylinder low
          MOV    A,E           ;Get last bit of lower record number
          RAL                  ;and put it in carry
          MOV    A,D           ;Get upper half of record number
          RAL                  ;Left shift it and merge in carry
          OUT    CYLLO         ;Send it to lower cylinder register


;Cylinder high
          MVI    A,0           ;Clear all bits except for the
          RAL                  ;the least significant and send
          OUT    CYLHI         ;to the upper cylinder register
          RET
          END
```

A.2 Interrupt Driven Driver

```
;**************************************************.
;                                                  ;
;    WD1001 Hard Disk Controller Driver            ;
;         Example for the Western Digital          ;
;              WD16 Microprocessor                 ;
;        using interrupts and programmed I/O       ;
;                                                  ;
;**************************************************.
```

;This example driver demonstrates the type of driver that one would use in a multitasking
;environment. Like the 8085 driver, it uses programmed I/O for data transfers. Unlike the last
;driver, this one supports interrupts.

;This new driver uses the same drive used in the 8085 example.

;Like the last driver, this one can access up to 65536 logical records of 256 bytes each. It
;still has three types of calls: Initialize, Read, and Write. Three numerical parameters are
;passed in the following registers:

```
;            Drive number              R1
;            Logical record number     R2
;            Transfer address          R3
```

;Upon completion of all commands, the carry bit of the WD16 will be reset if the operation
;terminated properly and set if there was an error. If there was an error during read or write,
;the error handling routine will decode and print it out on the user console.

```
;************************.
;      EQUATES          ;
;************************.
```

```
;***   Port Definition   ***
BASADD =    0FFD8                    ;Base address of WD1001
DATA   =    BASADD                   ;Data register
ERROR  =    BASADD+1                 ;Error Register
WPC    =    BASADD+1                 ;Write Precomp
SECNT  =    BASADD+2                 ;Sector Count
SECNO  =    BASADD+3                 ;Sector Number
CYLLO  =    BASADD+4                 ;Cylinder Number
CYLHI  =    BASADD+5                 ;Cylinder High
SDH    =    BASADD+6                 ;Size/Head/Drive
STATUS =    BASADD+7                 ;Status register
COMND  =    BASADD+7                 ;Command register

;***   Command Definition   ***
REST   =    10                      ;Restore command
READ   =    20                      ;Read command (programmed I/O mode)
WRITE  =    30                      ;Write command

;***   Drive stepping rate   ***
STRATE =    2                       ;Drive stepping rate
```

A-5

A.2.1 Initialization

```
;**********************;
;     INITIALIZE       ;
;**********************;
```

;This routine is called once whenever the system is powered up or reset. It sets the
;stepping rate and restores the head on the selected drive.


```
RESTOR:  CALL    UPTASK                 ;Select drive, don't care about record
```

;Since we're interfacing to a multitasking computer, we want to put the calling job to sleep
;while we do the restore. The SLEEP instruction in this listing is actually a monitor call to the
;operating system which does the dirty work. Once the WD1001 is done with the restore, it
;will interrupt the CPU and the interrupt service routine will wake the job. Putting the job to
;sleep will allow other tasks to use the CPU while the restore is in progress.


```
        LOCK                                    ;Disable interrupts until sleep
        MOV     #REST+<STRATE*2>,@#COMND ;Get stepping rate and restore
                                                ;and output command to WD1001
        SLEEP   IOWAIT                          ;Put job in an I/O wait state
        MOVB    @#STATUS,R0                     ;Get status register
        RORB    R0                              ;Put error bit in carry
        RTN                                     ;Return to operating system
```

A.2.2 Read Sector

```
;*****************;
;     READ        ;
;*****************;
```

;This is the read routine for our imaginary operating system.
```
READIT:  CALL    UPTASK                 ;Update WD1001 task file
         LOCK                           ;Disable interrupts 'til we go to sleep
         MOVB    #READ,@#COMND          ;Issue READ command to WD1001
```

;Wait for WD1001 to read in a sector
```
         SLEEP   IOWAIT                 ;While waiting, go into I/O wait state
```

;When we wake up, transfer sector from WD1001 to system memory
;(Transfer address in R3, WD1001 data register address in R4)
```
         MOV     #256.,R0               ;Init byte counter to 256 bytes
```

;The following instruction, MABB (Move Address to Block of Bytes), does a block move by
;reading the data pointed to by R4 (WD1001 data register) and puts them in a block of
;memory pointed to by R3. R0 bytes are moved.
```
         MABB    R4,R3                  ;Move data from WD1001 to memory
         BR      DONE                   ;Now check the completion status
```

A.2.3 Write Sector

```
;*******************
;     WRITE        ;
;*******************
```

```
;This is the write routine for our imaginary driver
WRITIT:    CALL    UPTASK              ;Update WD1001 task file
           MOVB    #WRITE,@#COMND      ;Issue WRITE command to WD1001

;Transfer sector from system memory to WD1001
;(Transfer address in R3, WD1001 data register address in R4)
           MOV     #256.-1.,R0         ;Init byte counter to 256-1 bytes
```

;The following instruction, MBBA (Move Block of Bytes to Address) is the converse of the
;MABB instruction, above. This time the block of memory pointed to by R3 will be moved
;to the WD1001 data register (pointed to by R4). R0 bytes will be moved.

```
           MBBA    R3,R4               ;Move most of the data
           LOCK                        ;Disable ints. 'till last byte xferred
           MOVB    @R3,@R4             ;Write last byte of data to WD1001

;Wait for WD1001 to write the sector
           SLEEP   IOWAIT              ;Wait in I/O wait state
```

```
;*******************
;     DONE         ;
;*******************
```

```
;Both READ and WRITE commands finish here to check for errors
DONE:      MOVB    @#STATUS,R0         ;Get status byte from WD1001
           RORB    R0                  ;Rotate Error bit to carry
           BCC     DONEOK              ;and return to OS if no error
           MOVB    @#ERROR,R0          ;Get WD1001 error code
```

;<<< Place error reporting routine here >>>

```
           LCC     CARRY               ;Set carry to flag an error
DONEOK:    RTN                         ;and return to OS with error
```

A.2.4 Task File Updating

```
;********************************;
;  UPTASK SUBROUTINE  ;
;********************************;

;This subroutine sets up the task file registers

;Sector number
UPTASK:    MOV     R2,R0              ;Get record number
           AND     #31.,R0            ;Mask off lower 5 bits (bits 0-4)
           MOV     #SECNO,R4          ;Index WD1001 sector number register
           MOVB    R0,(R4)+           ;and send to sector number register
                                      ;and increment R4 to point to WD1001
                                      ;Cylinder Low register


;Size/Drive/Head
;(The following instruction does a right arithmetic shift 5 times)
           SSRA    R2,5               ;Discard SECNO bits from record number
           MOV     R2,R0              ;Get remaining record number bits
           AND     #3.,R0             ;Mask off bits 5-6 to make head number
;(The following instruction does a left arithmetic shift 3 times)
           SSLA    R1,3               ;Shift drive number into position
           OR      R1,R0              ;OR drive number and head together
           OR      #80,R0             ;OR ECC flag and sector size field
           MOVB    R0,@#SDH           ;send it to Size/Drive/Head register


;Cylinder Low
           SSRA    R2,2               ;Discard HEAD bits from record number
           MOVB    R2,(R4)+           ;Send Cylinder Low to WD1001 and
                                      ;increment R4 to point to Cylinder High


;Cylinder High
;(The following instruction SWAps the upper and lower Byte of a word)
           SWAB    R2                 ;Get upper byte of cylinder
           AND     #1,R2              ;Mask to least significant bit
           MOVB    R2,(R4)            ;and send it to WD1001
           MOV     #DATA,R4           ;Index WD1001 Data Reg for R/W ops
           RTN
```

A.2.5 Interrupt Service routine

```
;******************************************;
;INTERRUPT SERVICE ROUTINE;
;******************************************;

;This routine gets called whenever the WD1001 interrupts. All it does is read the status
;register of the WD1001 to clear INTRQ and revives the original calling job.


INTSER:    PUSH    R0                 ;Save this register
           MOVB    @#STATUS,R0        ;Acknowledge the interrupt to WD1001
           WAKEUP  IOWAIT             ;Wake up job (from I/O wait state)
           POP     R0
           RTT                        ;Return from trap
           END
```

APPENDIX B INTERLEAVE CALCULATING

This BASIC program simplifies the process of generating interleave tables. It is written in a fairly standard subset of the BASIC language and should run on many BASIC interpreters and compilers. Some implementations of BASIC may require the variable names to be converted to single letter names and the IF THEN ELSE constructs may have to be re-written.

The two questions at the beginning of the program should be answered in decimal. The interleave table is printed in hexadecimal.

B.1 BASIC Interleave Calculating Program

```
10   PRINT "WD1001 Interleave calculating program"
20   PRINT
30   INPUT "Number of sectors? ";COUNT
40   INPUT "Interleave Factor? ";INTER
50   DIM HEX$(16),SECTOR(COUNT)
60   FOR INDEX=1 TO 16
70   READ HEX$(INDEX)
80   NEXT
90   FOR INDEX=1 TO COUNT
100  SECTOR(INDEX)=-1
110  NEXT
115  RES=0
120  FOR INDEX=0 TO COUNT-1
130  IF RES>=COUNT THEN RES=RES-COUNT
140  IF SECTOR(RES+1)=-1 THEN SECTOR(RES+1)=INDEX ELSE RES=RES+1:
     GOTO 130
150  RES=INTER+RES
160  NEXT
170  PRINT
180  PRINT "Interleave table with";COUNT;"sectors and";
     INTER; ":1 interleave"
190  FOR INDEX=1 TO COUNT
200  X=INT(SECTOR(INDEX)/16)
210  PRINT HEX$(X+1);HEX$(SECTOR(INDEX)-X*16+1),
220  NEXT
230  PRINT
240  DATA 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
250  END
```

APPENDIX C CALCULATING SECTORS PER TRACK

WD1001 compatible disk drives are constantly changing. Higher bit packing densities and higher accuracy spindle motors help to increase the amount of data that can be put on a track. This appendix will help you to determine the maximum number of sectors that can be recorded on your disk drive.

The unformatted byte capacity can be figured from this formula:

Capacity=Bits per second / Revolutions per second x (1-Error) / 8

We'll take a hypothetical drive (the same one as in the disk driver examples) with a data rate of 5M bits per second and revolution rate of 3600 RPM. Our drive has a spindle speed accuracy of 3%. These numbers applied to our formula yield:

10,104=5,000,000/60x(1-0.03)/8

To be on the safe side, we will always round down when we come up with a fractional value. The unformatted capacity of this drive is 10,104 bytes. To figure the number of sectors per some number of bytes apply this formula:

Sectors=Capacity / (Data field size+Gap3+Check bytes+Other overhead)

Using 512 byte sectors with a Gap3 size of 30 bytes, running ECC we end up with:

17=10,104/(512+30+4+41)

The BASIC program on the next page can be used to automate the sector per track calculations presented here.

C.1 BASIC Sectors per Track Utility

```
 10 PRINT "WD100X Sectors per Track Calaulating utility"
 20 PRINT
 30 INPUT"Data rate of drive in bits per second: ";DATARATE
 40 INPUT"Revolutions per minute: ";RPM
 50 INPUT"Rotational speed error on percent: ";RERROR
 60 CAPACITY=INT(DATARATE/RPM*60*(1-RERROR/100)/8)
 70 PRINT"Unformatted capacity" is CAPACITY"bytes."
 80 PRINT
 90 INPUT"Data field size in bytes (128, 256, etc.): ";SIZE
100 INPUT"Formatted with CRC or ECC: ";ECCMODE$
110 ECCMODE$=LEFT$(UCS(ECCMODE$),1)
120 IF ECCMODE$<>"E" AND ECCMODE$<>"C" THEN 100
130 IF ECCMODE$="E" THEN CHECKBYTES=4 ELSE CHECKBYTES=2
140 IF SIZE>256 THEN GAP3=30 ELSE GAP3=15
150 SECTORS=INT(CAPACITY/(SIZE+GAP3+CHECKBYTES+41))
160 PRINT "Formatted capacity is";SECTORS*SIZE;"bytes per track using";
170 PRINT SECTORS;"sectors per track."
180 END
```

APPENDIX D PROGRAMMER'S QUICK REFERENCE

D.1 Task File

CS-	A2	A1	A0	RE-	WE-
1	X	X	X	Deselected	Deselected
0	0	0	0	Data Register	Data Register
0	0	0	1	Error Register	Write Precomp
0	0	1	0	Sector Count	Sector Count
0	0	1	1	Sector Number	Sector Number
0	1	0	0	Cylinder Low	Cylinder Low
0	1	0	1	Cylinder High	Cylinder High
0	1	1	0	Size/Drive/Head	Size/Drive/Head
0	1	1	1	Status Register	Command Register

D.2 Valid Commands

TYPE	COMMAND	Bits 7 6 5 4 3 2 1 0
I	Restore	0 0 0 1 r_3 r_2 r_1 r_0
I	Seek	0 1 1 1 r_3 r_2 r_1 r_0
II	Read Sector	0 0 1 0 D M L 0
III	Write Sector	0 0 1 1 0 M L 0
III	Format Track	0 1 0 1 0 0 0 0

L=Long Read/Write   D=DMA Read Interrupt
M=Multiple Sector r_x=Stepping Rate

D.3 SDH Register Format

Bit	7	6 5	4 3	2 1 0
Function	E	Sec Size	Drive Select	Head Select

E=ECC Mode

Bit 6	Bit 5	Sector Size
0	0	256 Bytes
0	1	512 Bytes
1	1	128 Bytes

Bit 4	Bit 3	Drive Selected
0	0	Drive Sel 1
0	1	Drive Sel 2
1	0	Drive Sel 3
1	1	Drive Sel 4

Bit 2	Bit 1	Bit 0	Head Selected
0	0	0	Head 0
0	0	1	Head 1
0	1	0	Head 2
0	1	1	Head 3
1	0	0	Head 4
1	0	1	Head 5
1	1	0	Head 6
1	1	1	Head 7

D.4 Status and Error Register Bits

Bit	Status Register	Error Register
7	Busy	Bad Block Detect
6	Ready	Uncorrectable
5	Write Fault	CRC Error - ID Field
4	Seek Complete	ID Not Found
3	Data Request	-
2	Corrected	Aborted Command
1	-	TR000 Error
0	Error	DAM not found