WD 1000 Winchester Disk Controller OEM Manual

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

TABLE OF CONTENTS

SECTION 1 INTRODUCTION

1.1 General Description

The WD1000 is a stand-alone, general purpose Winchester Controller board designed to interface up to four Winchester disk drives to a host processor. The drive signals are based upon the floppy look-alike interface available on the Shugart Associates' SA1000, the Seagate Technology ST506, the Quantum Q2000, and other compatible drives. All necessary buffers and receivers/drivers are included on the board to allow direct connection to the drive. Either a 34 pin (5-1/4" drive) or a 50 pin (8" drive) connector is provided, as well as four 20 pin data connectors.

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 WD1000 is based upon a proprietary chip set called the WD1100, specifically designed for Winchester Control.

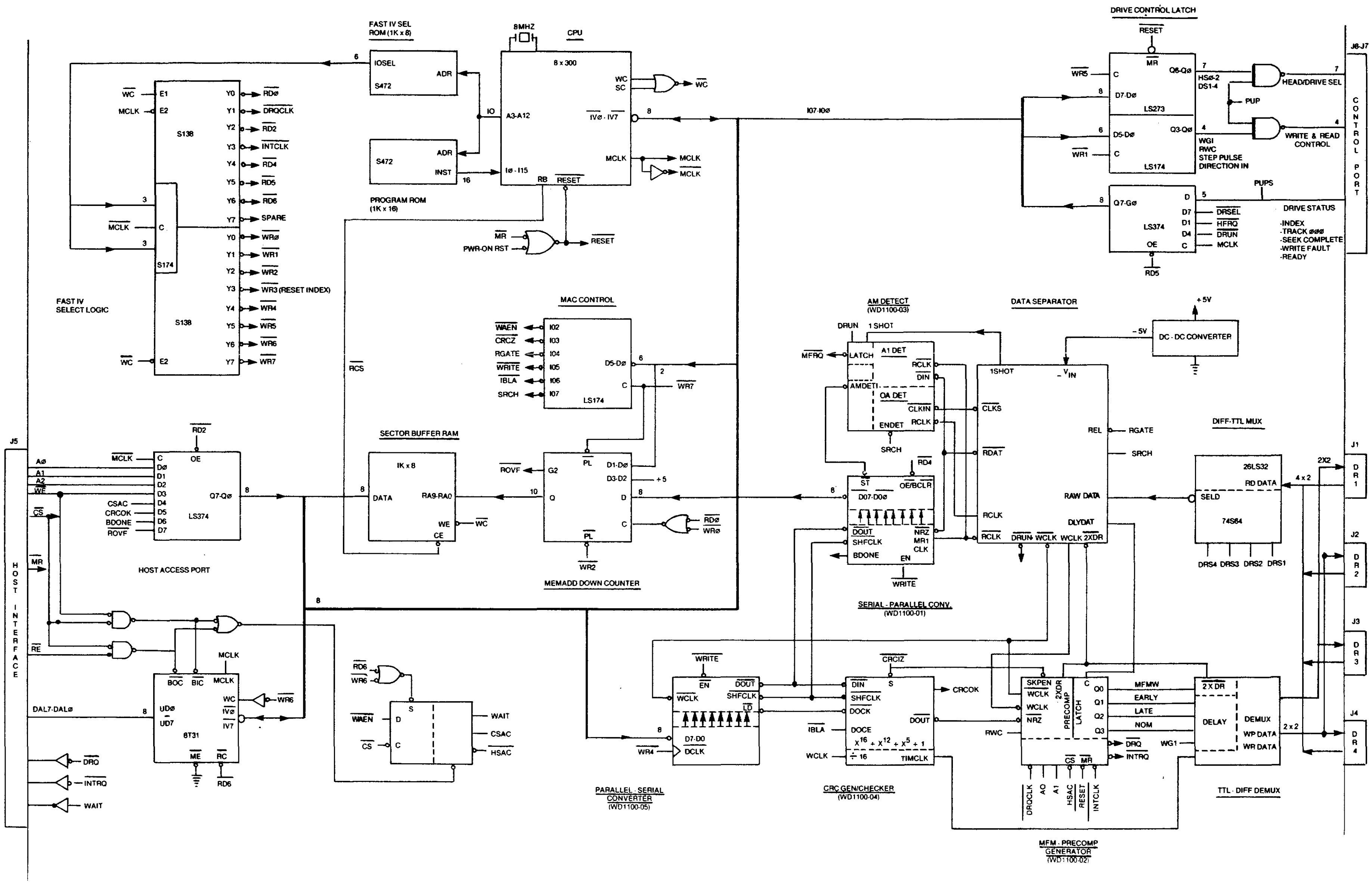
1.2 Features

* 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 (ROM selectable)
* Unlimited Sector Interleave Capability
* Overlap Seek capability
* Implied Seek on all commands
* Automatic Retries on all errors
* Automatic Restore and Re-seek on seek error
* 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:	10 uS to 7.5 mS (0.5 mS increments)
Data Transfer Rate:	4.34 Mbits/sec (SA1000, Q2000) 5.000 Mbits/sec (ST506)
Write Precomp Time:	10 nanoseconds
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.) -8 to -18V, 50 mA
Ambient Temperature Operating:	0°C. to 50°C. (32°F. to 122°F.)
Relative Humidity:	20% to 80%
MTBF:	10,000 POH
MTTR:	30 minutes
Length:	9.9 in. (24.9 cm)
Width:	6.8 in. (17.1 cm)
Height:	0.75 in. (1.9 cm)
Mounting Centers:	6.375 X 9.375 in. (16 X 23.6 cm)

1.4 System Block Diagram



SECTION 2 INTERFACE CONNECTORS

2.1 Organization

The WD1000 has seven on board connectors. These connectors consist of a power connector, a host interface connector, a drive control connnector, 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 WD1000.

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 Connector

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 WD1000.
2.2.4 RE-	When Read Enable- is active along with CS-, the host may read data from a selected register of the WD1000.
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.
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 WD1000. Sector Number, Cylinder Number and SDH 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 -V	-V input from the host supplies -8 to -15V to the on-board -5 volt regulator (VR1). *This input is necessary unless the optional on board DC to DC converter is used.* This power input is also available on J6, pin 2.
2.2.11 GND	All even numbered pins on this connector are to be used as signal grounds. Power grounds are available on J6, pin 1.
2.2.12 +5V	8 power pins for regulated +5 volts. This power input is also available on J6, pin 3.

2.2.13 50 Pin Host Interface Connector

The host interface connector (J5) is a 50 pin card edge connector on tenth-inch centers that mates with Burndy #FRE 50 B-3. 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	-V
36	35	INTRQ
38	37	DRQ
40	39	MR-
	41	Not Connected
	42	Not Connected
	43-50	+5V

2.3 Drive Control Connectors

The drive control connector is a (relatively) low speed bus that is daisy chain connected to each of the drives (up to four) in the system. To properly terminate each TTL level output signal from the WD1000, 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. This output is valid only during Write and Format Commands.

2.3.2
Write Gate- This output signal allows data to be written on the disk.

2.3.3
Seek Complete- Informs the WD1000 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 WD1000 that some fault has occurred on the selected drive. The WD1000 will not execute commands when this signal is true.

2.3.6
HS0-HS2- Head Select lines are used by the WD1000 to select a specific R/W head on the selected drive.

2.3.7
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.

2.3.8
Ready- Informs the WD1000 that the desired drive is selected and that its motor is up to speed. The WD1000 will not execute commands unless this line is true.

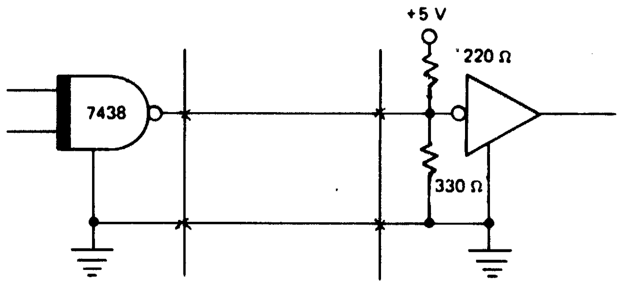
2.3.9
Step- This line is pulsed once for each cylinder to be stepped. The direction of the step will be determined by the DIRECTION line. The step pulse period is determined by the internal stepping rate register during implied seek operations or explicitly during Seek and Restore commands. During auto restore, the step pulse period is determined by the SEEK COMPLETE time from the drive.

2.3.10
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.

2.3.11
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)



2.3.13 50 Pin Drive Control Connector

This drive control connector (J8) 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		NC
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

2.3.14 34 Pin Drive Control Connector

This drive control connector (J7) 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:

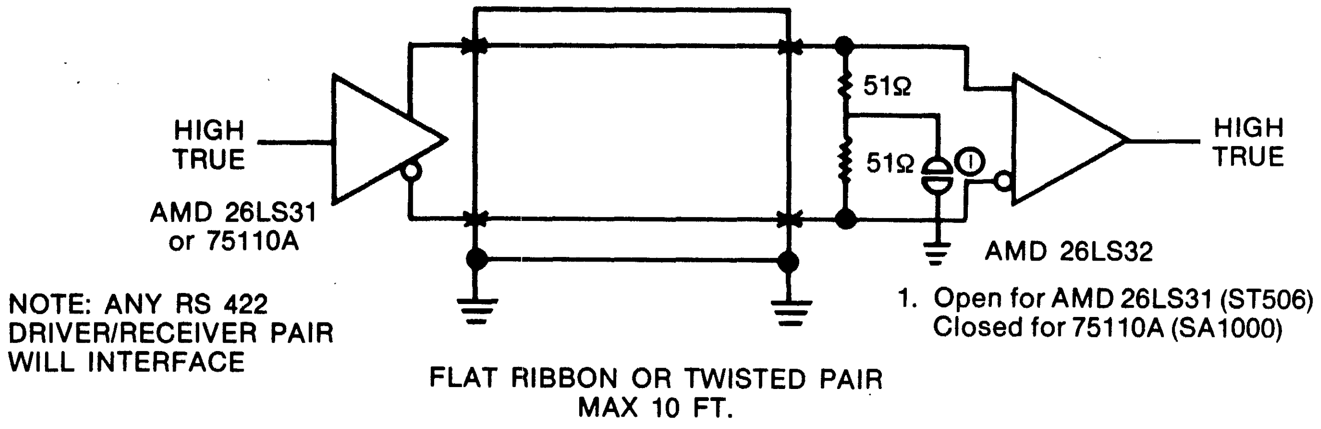
Signal Ground	Signal Pin	I/O	Signal Name
1	2	O	-RWC
3	4	O	-Head Select 2
5	6	O	-Write Gate
7	8	I	-Seek Complete
9	10	I	-TR000
11	12	I	-Write Fault
13	14	O	-Head Select 0
15	16		NC
17	18	O	-Head Select 1
19	20	I	-Index
21	22	I	-Ready
23	24	O	-Step
25	26	O	-Drive Select 1
27	28	O	-Drive Select 2
29	30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

2.4 Drive Data Connector

Four data connectors (J1-4) are provided for clock signals and data between the WD1000 and each drive. All lines associated with the transfer of data between the drive and the WD1000 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		NC
8	7		NC
	9	O	+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

2.4.1 Differential Data Driver/Receiver


HIGH TRUE
AMD 26LS31 or 75110A
NOTE: ANY RS 422 DRIVER/RECEIVER PAIR WILL INTERFACE
51Ω
51Ω
1
HIGH TRUE
AMD 26LS32
1. Open for AMD 26LS31 (ST506) Closed for 75110A (SA1000)
FLAT RIBBON OR TWISTED PAIR MAX 10 FT.

2.5 Power Connector

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

Pin	Signal Name
1	GROUND
2	-8 to -15 V unregulated *
3	+5 V regulated

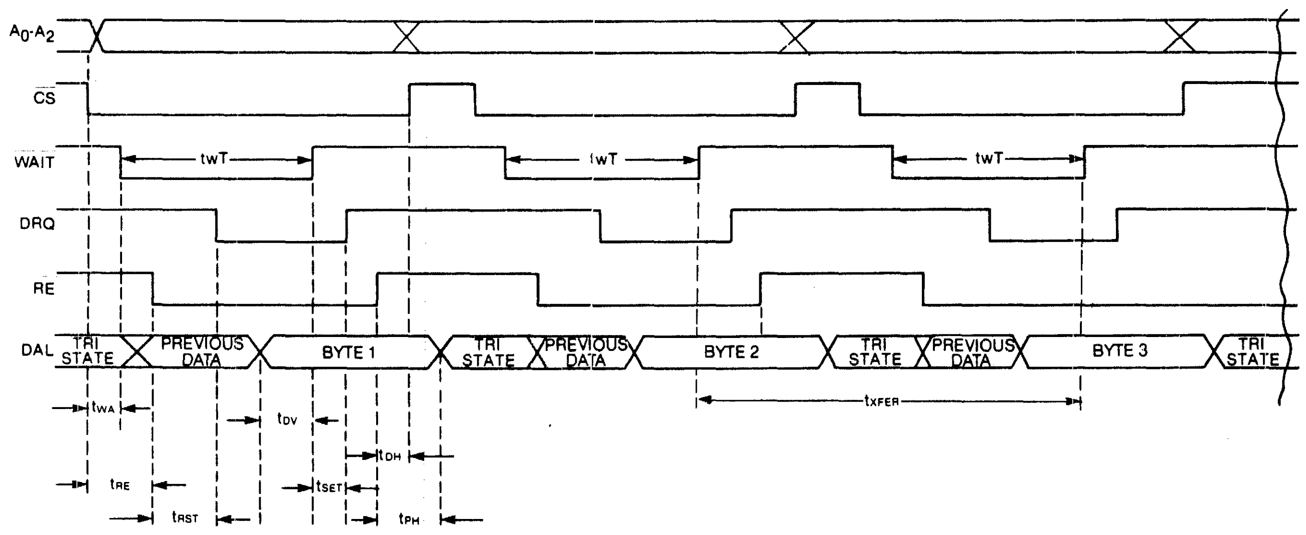
* This pin is ground on REV B and below.

SECTION 3 INTERFACE TIMING

3.1 Host Interface Timing

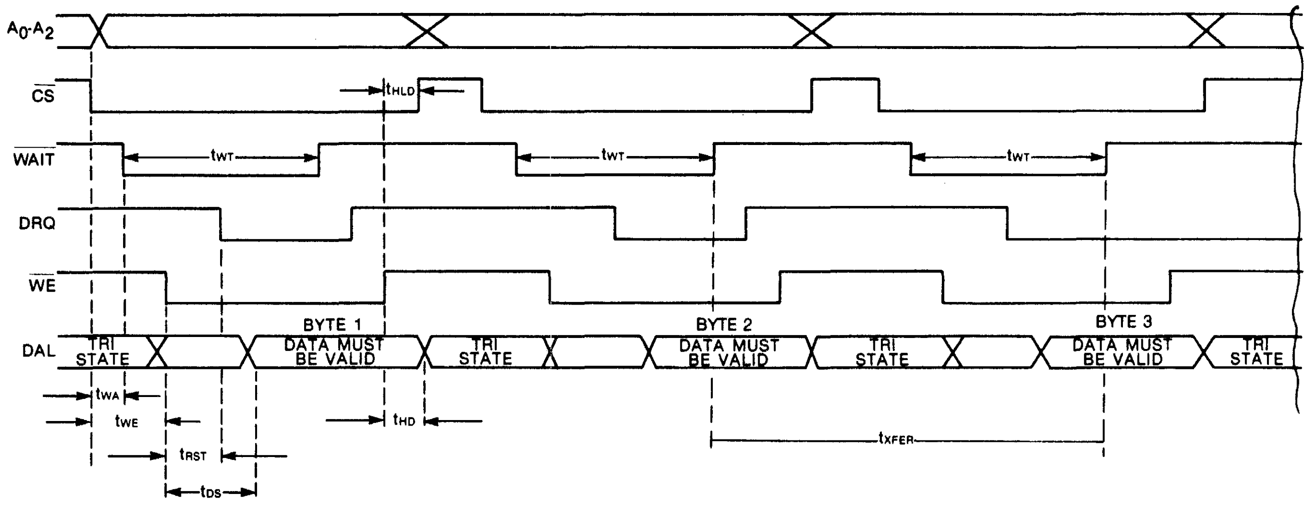
3.1.1 Host Read Timing

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		40	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-		100	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

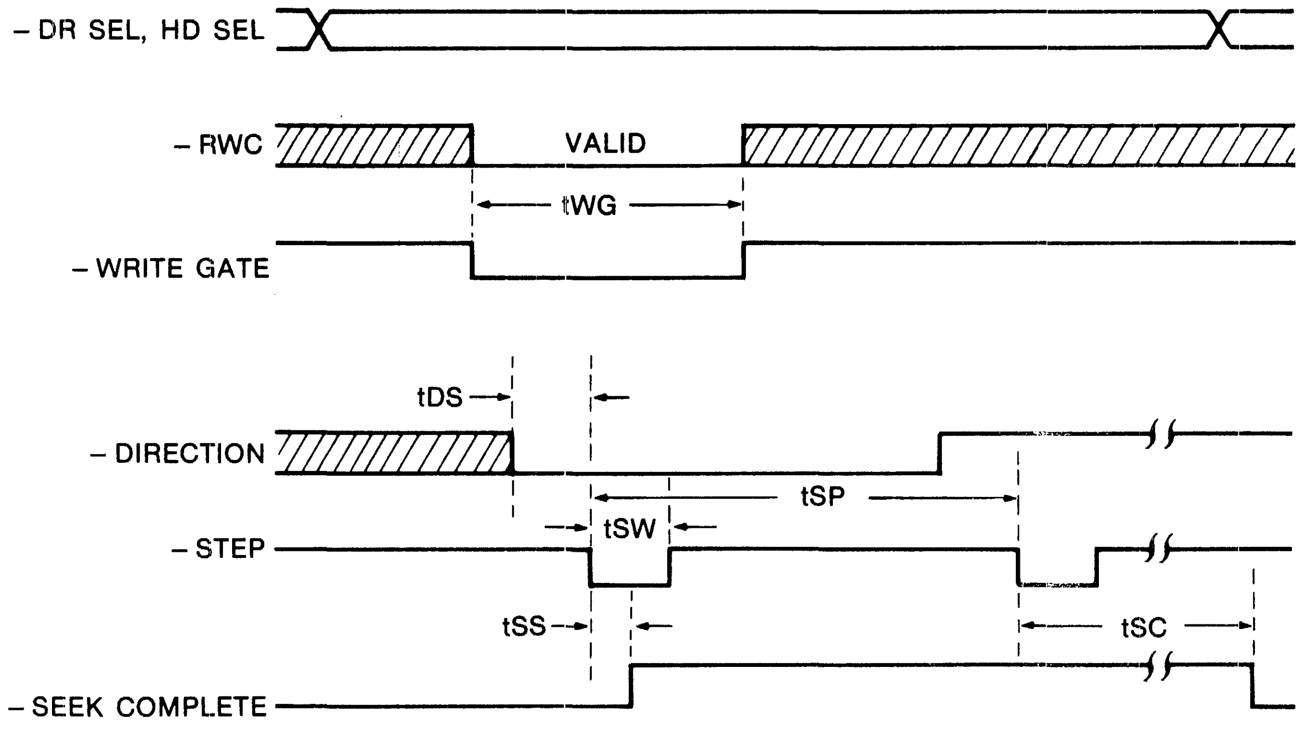


3.2 Drive Control Timing

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	0.01	7.5	mS
t_{SS}	Step to Seek Complete false		9	uS
t_{SC}	Last Step to Seek Complete		128	Index times

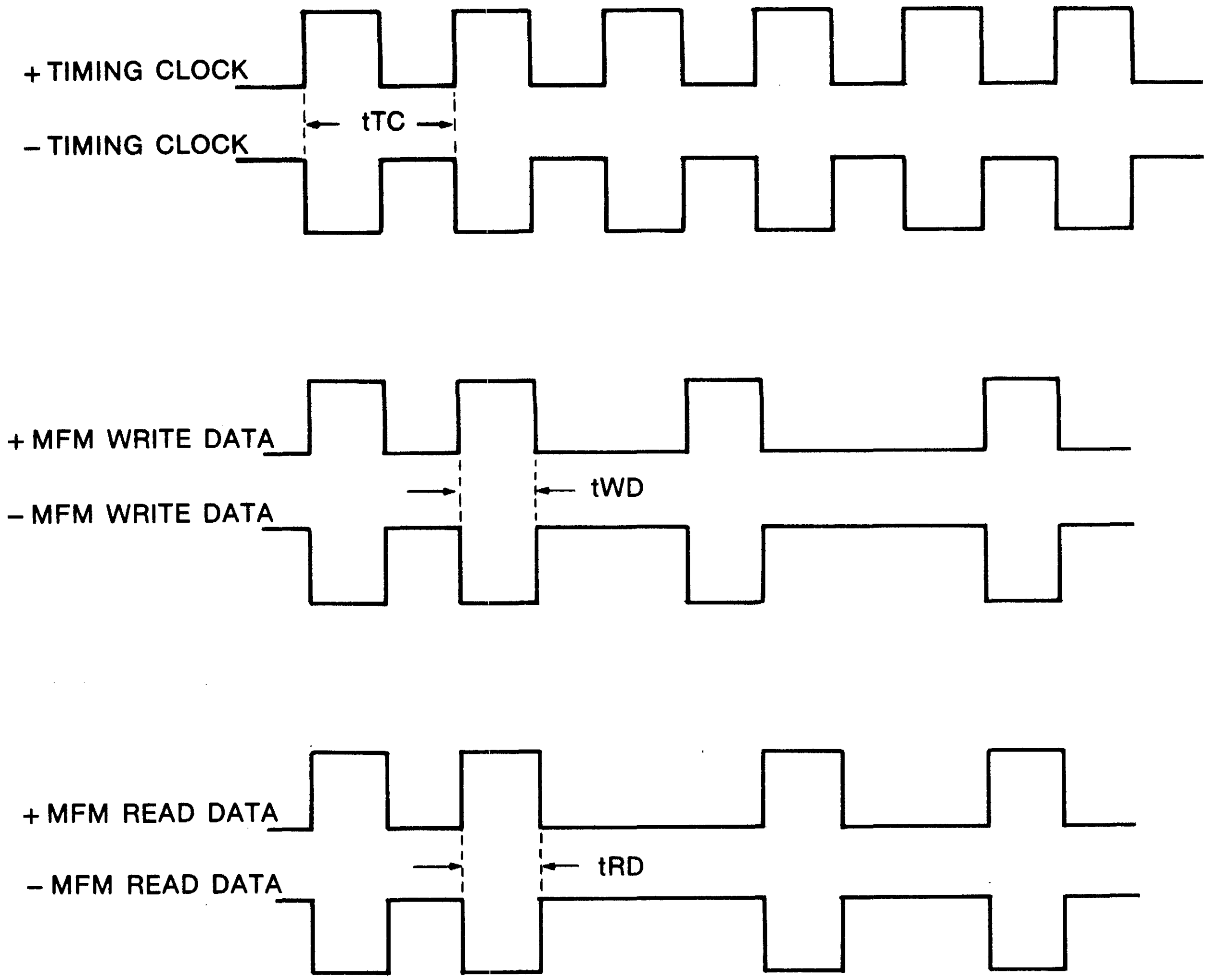
Notes:

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 WD1000 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:

- The WD1000 is accessing data internally to send to the host during a read operation
- The WD1000 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 WD1000. When the WD1000 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 WD1000 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 WD1000 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.

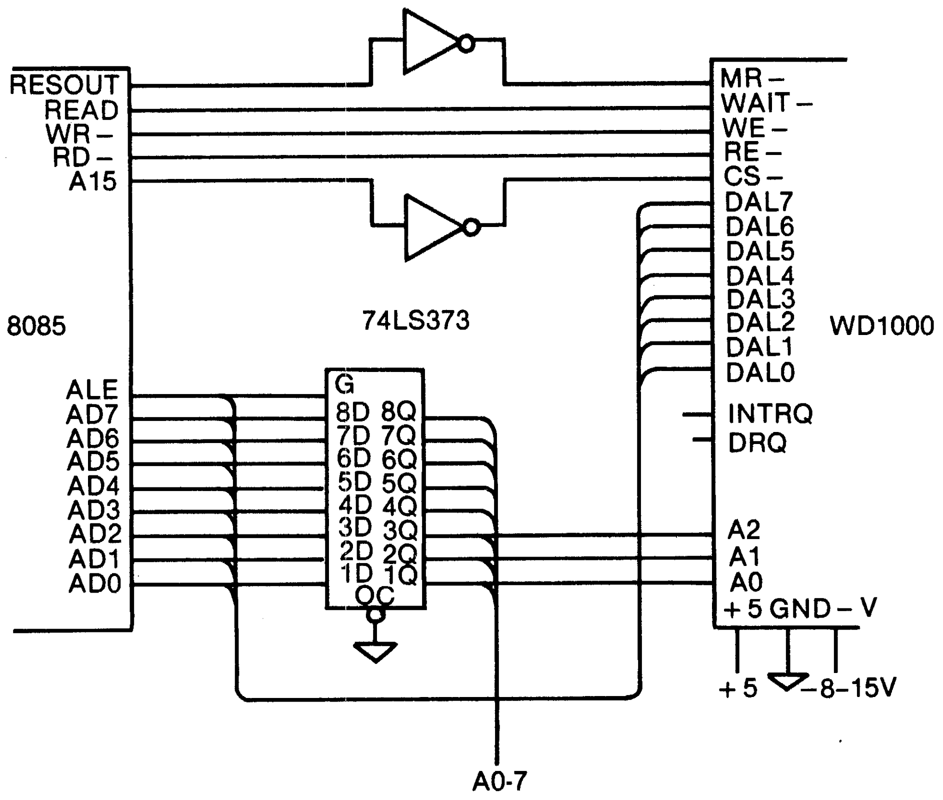
Do not confuse the function of WAIT- with the BUSY bit of the Status Register. The BUSY bit is a status indicating that the WD1000 is communicating with the disk and WAIT- is simply a bus synchronization signal.

4.1.1 WAIT timing

The user can modify the timing of the wait signal by selecting a jumper. The WD1000 is shipped with a jumper (or trace) between E4 and E5. This enables waits as soon as the CS - signal is asserted. This timing is a requirement for some processors and compatible with most. If the host system requires the WAIT- signal to be asserted only when RE - or WE - are asserted in conjunction with CS -, the trace at E4 and E5 should be cut and a jumper should be installed between E4 and E3.

4.2 Host interfacing example

The example below illustrates the absolute minimum of hardware required to interface to a small 8085 micro-computer 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 WD1000. 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 WD1000 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.

5.3.2 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 sector Size, Drive select, and Head select bits. The SDH register is a R/W register organized as follows:

5.3.4 Cylinder Number

These two registers form the cylinder number where the head is to be positioned on a Seek, Read, or Write 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:							A	9	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 R/W register and may be read or written to by the host.

5.3.6 Sector Count

This register is loaded with the number of sectors to be formatted during a Format command. During the Format command, this register is decremented to zero and must be re-loaded for each format operation.

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 10 nanoseconds from nominal.

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

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

5.3.9 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 WD1000 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 WD1000 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	CRC Error - Data Field
5	Write Fault	CRC Error - ID Field
4	Seek Complete	ID Not Found
3	Data Request	—
2	—	Aborted Command
1	—	TR000 Error
0	Error	DAM not found

5.5 Status Register Bits

5.5.1 Error

When set, indicates that one or more bits are 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 Data Request

Functions identically to the 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.

5.5.3 Seek Complete

Indicates the condition of the seek complete line on the selected drive.

5.5.4 Write Fault

Indicates the condition of the Write Fault Line on a selected drive. The WD1000 will not execute any command if this bit is set.

5.5.5 Ready

Indicates the condition of the READY line of the selected drive. The WD1000 will not execute any commands unless the ready bit is set.

5.5.6 Busy

After issuing a command, this bit will be set indicating that the WD1000 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 the ID field.

5.6.2 TROOO Error

Will be set during a Restore command if, after issuing 1023 stepping pulses, 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 must 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 CRC Error Data

Indicates that a CRC error was encountered in a data field during a Read Sector Command.

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.

SECTION 6 COMMANDS

The WD1000 executes five easy to use macro commands. Most commands feature automatic 'implied' seek, which means the host system need not tell the WD1000 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 CRC errors. If the R/W head mis-positions, the WD1000 will automatically perform a restore and a re-seek. If the error is completely unrecoverable, the WD1000 will simulate a normal completion to simplify the host system's software.

Commands are executed by loading the 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 WD1000 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:

		BITS							
Type	Command	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	0	0	0
III	Write Sector	0	0	1	1	0	0	0	0
III	Format Track	0	1	0	1	0	0	0	0

6.1.1 Stepping Rates

r_3 - r_0 - Stepping Rate					
0000	=	10uS	1000	=	4.0mS
0001	=	0.5mS	1001	=	4.5mS
0010	=	1.0mS	1010	=	5.0mS
0011	=	1.5mS	1011	=	5.5mS
0100	=	2.0mS	1100	=	6.0mS
0101	=	2.5mS	1101	=	6.5mS
0110	=	3.0mS	1110	=	7.0mS
0111	=	3.5mS	1111	=	7 5ms

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.

6.2 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.

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 1023 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 WD1000 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 WD1000 will attempt to read the sector. The WD1000 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 incurred a data CRC error.

6.3.1.3 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 read the sector once again. An auto-restore will be performed only once per read or write sector command.

6.3.1.4 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. If the Data CRC Error 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.

6.3.1.5 Error Severity Levels

Although the WD1000 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. Data CRC Error
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.

6.3.1.6 Normal Completion

If the WD1000 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.

6.4 Type III Commands

This type of command is characterized by a transfer of a block of data from the host to the WD1000 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 as 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 WD1000 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

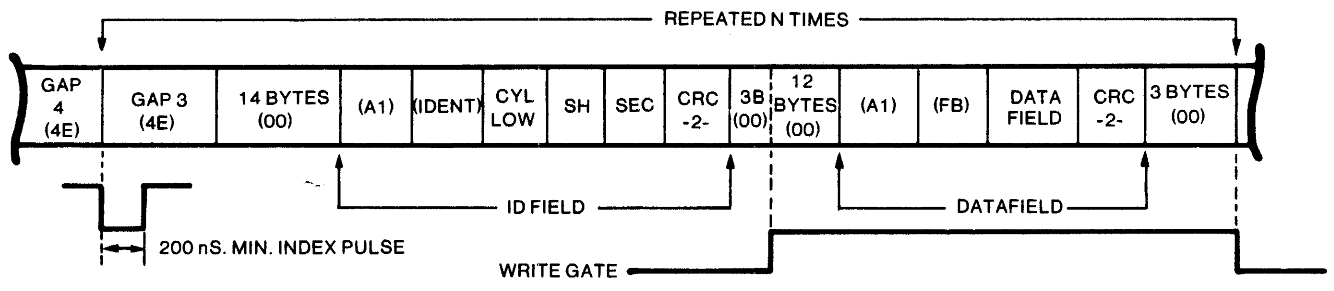
If no errors are encountered so far, a Seek command is executed. No verification of track positioning accuracy is performed because the track may not have any ID fields present. After the Seek operation has been performed, the Seek Complete line is sampled. If the Seek Complete line is not asserted within 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 starts writing a pattern of 4E's until the index is encountered. 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 is 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 CRC 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

6) GAP 4 values are:

Sector length	Gap 3	Gap 4	Sector Count
128	15	356	54
256	15	352	32
512	30	800	17

SECTION 7 PROGRAMMING

Users familiar with floppy disk systems will find programming the WD1000 a pleasant surprise. A substantial amount of intelligence that was required by the host computer has been incorporated into the WD1000. The WD1000 performs all needed retries, even on data CRC and head positioning errors. Most commands feature automatic 'implied' seek which means that seek commands need not be issued to perform basic read/write functions. The WD1000 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 WD1000 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 WD1000 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 writable. 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 WD1000 can recall all the Task File parameters sent to it, it is recommended that Task File parameters be stored in the WD1000 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
   (WD1000 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 WD1000's 10 uS stepping rate require a slower rate (usually 3 mS or more) for Restore operations. This is why the WD1000 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.

The WD1000 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 WD1000 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 WD1000 has completed a command is through interrupts. The INTRQ line on the WD1000 makes a low to high transition whenever the disk controller requires CPU intervention. This allows the host CPU to run other tasks while the WD1000 is reading or writing data to the disk.

7.2.4 Use of the Error bit

Since the WD1000 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 WD1000 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
```

7.3 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 WD1000 buffer to the host. This command features implied seek with an implicit stepping rate. To execute a Type II 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
   (WD1000 will attempt to read sector)
2. Wait for interrupt or for Busy bit in Status Register to be reset
3. Do block move from WD1000 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 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
   (WD1000 will attempt to read sector)
   (DMA controller will move data from WD1000 to memory)
3. Wait for interrupt from WD1000
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 WD1000'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 WD1000 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 WD1000 to system memory. The transfer address is in HL and the byte count is in B:

```
READIT: IN   DATA    ;Get data from WD1000 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 WD1000 data register address in C:

```
READIT: INIR      ;Transfer buffer from WD1000 to memory
```

7.3.3 Using DMA

There are several features in the WD1000 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 WD1000 interrupts for DMA operation.

7.3.3.1 Partial Sector Transfers

The WD1000 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 WD1000 will interrupt the host after the sector has been transferred during a DMA read operation, but if a partial sector has been read, the WD1000 will not know that the operation has been completed. For this reason, the 'transfer complete' interrupt must come from the DMA controller. There still is a problem. During write sector operations, the DMA controller will interrupt the system after the buffer has been transferred to the WD1000 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 WD1000 has a provision built into its command structure to detect read operations.

7.3.3.2 Interrupt Source Selection

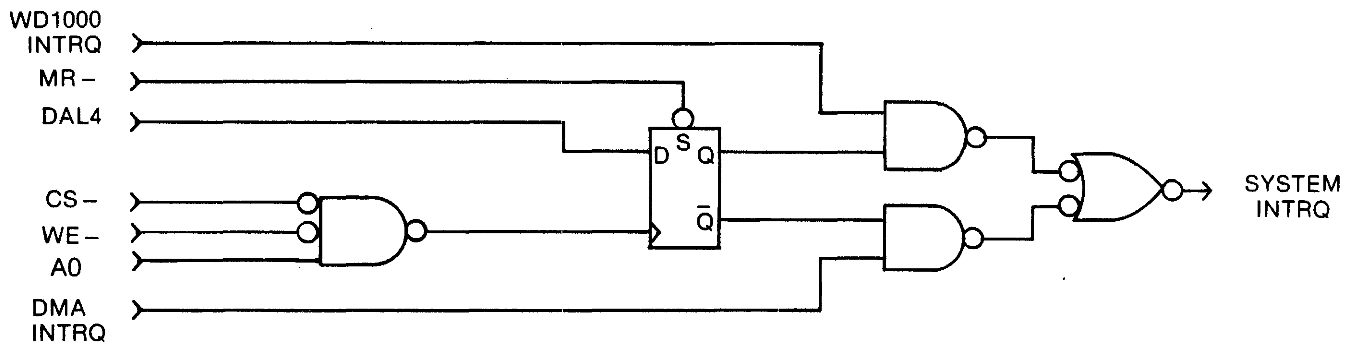
Bit 4 of all commands determine whether the operation will be a read sector operation or something else. Those commands that require the interrupt from the WD1000 will have this bit set to a 1. The read sector command (the only one that might need the DMA controller's interrupt) has this bit set to a 0.

7.3.3.3 Clearing Hardware DRQ

During partial sector reads, the DMA controller will stop the DMA transfer before the WD1000 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 is passed its parameters only after the Task File is updated.

7.3.3.4 Interrupt Selection Circuit

If the user is reading partial sectors with the WD1000 and wants to have the system automatically configure its interrupts, a circuit similar to the following will have to be implemented:



7.3.4 Simulated Completions

All WD1000 commands 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:

- Simplifies masking and generation of interrupts
- Simplifies non-error handling portions of the system software
- Eliminates the software overhead of handling different types of errors
- Simplifies system software error handling validation (any error is handled the same as any other error)
- Prevents system failure in the event of some obscure error condition that the systems 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 WD1000 buffer. Like Type II commands, these commands feature implied seek with an implicit stepping rate. To execute a 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 WD1000 buffer
   (WD1000 will attempt to write 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 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 WD1000)
   (WD1000 will attempt to write sector or format)
3. Wait for interrupt from WD1000
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 byte will be either a 00 or an 80 in hex. If the first byte is a 00, the sector is marked as good. If the first 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 byte is the logical sector number of the next sector to be formatted. This number will be recorded on the disk.

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 of the disk 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.

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 R/W 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 us 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) 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 in a certain place on the disk, it sometimes makes sense to have *two* interleaves. One could be used for regular disk operations and the other for directory functions.

To simplify driver software, the WD1000 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 format operations. 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 byte pair in brackets represents the second logical sector.

7.5 Bad Block Mapping

The Winchester and thin film technology drives that the WD1000 interfaces to 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.

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 WD1000 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 WD1000 allows the user to set a marker that is recorded into the ID field. When the WD1000 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 precludes the possibility of using the bad block marker to map out bad areas in an ID field. In the following interleave table, the user has marked the fifth sector (logical sector 1 as a bad block):

Interleave table with 32 sectors and 4:1 interleave with logical sector one marked as a bad block:

```
00 00 00 08 00 10 00 18 [80  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
```

Because the bad block bit cannot be read when there's a flaw in the ID field, it cannot be used as a primary method of mapping out bad blocks.

SECTION 8 THEORY OF OPERATION

8.1 General

The WD1000 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 Micro-controller, the 8X300, 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" 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.

The disk controller is built around 5 basic sections:

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

8.2 Processor Functions

All functions of the WD1000 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 8X300, a bipolar micro-controller particularly well suited for handling data efficiently at high rates.

The 8X300 operates 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 micro-processsors 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 (IA0-IA13) is capable of directly accessing 8K words of program storage, however, the WD1000 uses only the first 10 address lines, limiting onboard program storage to 1K words. Program data is input to the 8X300 (U50) on the Instruction Data Bus (ID0-ID15) as 16 bit words which are decoded to perform the desired operation. All bus designations utilized by the 8X300 are reversed from the traditional LSB to MSB weighting, so that on the WD1000 these lines have all been renamed on the schematic to provide a more conventional designation system for the board.

8.2.1 Fast IO Select

An extension byte has been added onto the instruction data memory to provide port access decoding on an instruction-by-instruction basis. This "Fast IO Select" byte, is not processed by the 8X300, rather it is decoded by auxiliary hardware to provide 8 read strobes and 8 write strobes which route data to the various devices distributed along the interface vector bus.

The Fast IO byte is latched into a 6-bit latch (U27) on the trailing edge of MCLK to ensure the data remains stable during the entire instruction. This data selects a read strobe and a write strobe through two 1-of-8 decoders (U20 and U26) which are alternately enabled by the -WC control strobe produced by the 8X300. To provide edges on read strobes during sequential read operations from various ports, the read strobe decoder (U20) is always disqualified at the end of instructions by +MCLK' which is a delayed copy of +MCLK. This delay compensates for timing races through the Fast IO latch (U27) and the control signals.

Because each decoder has a unique input, it is possible to select any read port with any write port during each instruction. 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 can be thought of as simply an 8-bit bidirectional IO bus of the 8X300. In fact, it has been renamed as IO0-IO7 to reflect this distinction.

8.2.2 Internal Bus Control

Several bus control signals are produced by the 8X300 to identify and strobe the data on the IO bus. WC (write control) is a signal which determines the direction of the data to and from peripherals. When WC is false (during the first half cycle) data is being input to the 8X300 from the IO bus. When WC is true (during the second half cycle), data is being output from the 8X300 onto the IO bus. SC (select control) is a signal which becomes active during the second half cycle instead of WC if the IO bus contains an 8-bit IO address. In the WD1000 both WC and SC are combined by a NOR gate (U24) to indicate all accesses to any port. This arrangement allows 8-bit immediate data moves from the 8X300 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. 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 8X300 are logically divided into two address spaces. These two address spaces are qualified with RIGHT BANK SELECT (-RB) and LEFT BANK SELECT (-LB). All WD1000 I/O ports except RAM data appear in the right bank address space. The -LB signal is run directly to the -CS inputs of the RAMs to avoid the propagation delays associated with the Fast I/O Select logic, allowing slower RAMs to be used.

8.2.3 Reset Circuit

The 8X300 is held reset for approximately 40 mS after initial power on. This is accomplished by an RC network (R42, C68 and CR3) which drives a Schmitt trigger (U31) to provide a proper rise/fall time on the -RESET line of the 8X300. Alternate reset of the processor can be accomplished by dropping -MR (J5 pin 39) whenever the host wishes to reset the controller. A Schmitt trigger (U31) is provided with a 4.7K pull-up (R43) to buffer the -MR input from the host. -RESET also propagates to the drive control latches (U52 and U48) and their associated line drivers (U54-56) and host interface WAIT (U32), DRQ and INTRQ latches (U30), ensuring proper initialization of these functions during power up and subsequent resets from the host.

8.2.4 Processor Power Supply

Power is supplied to the 8X300 from the +5 Volt (Vcc) power bus. Due to the internal operation of the 8X300 an on-chip voltage reference is provided to produce bias to an external pass transistor (Q4) which drops Vcc to the 8X300 to approximately +3.0 Volts. All signals into and out of the 8X300 are internally level shifted to be TTL compatible.

8.2.5 Read and Write Ports

Throughout the circuit, output ports are formed by D type latches using write strobes (WR0-7) to latch data into the ports. Reading of ports is universally accomplished by using read strobes (RD0, RD2, RD4-6) that enable selected tri-state output devices on the IO bus. Additionally, two read strobes are used to clock the host DRQ and INTRQ latches (U30) and one read strobe is left unused as a "dummy" port 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 8X300 does not permit data to be saved or retrieved from dedicated program storage, RAM must be installed on the IO bus. RAM must, therefore, be accessed via the IO bus by I/O instructions like all other port accesses. To provide for addressing the RAM, three latch/counters (U40, 45 and 46) are connected to the IO bus to receive and store addresses required to access the RAM (U33 and U39).

8.2.6.1 RAM Addressing

The RAM address bus (RA0-RA9) uniquely addresses 1 of 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 (U40) will be reset providing an overflow status which can be read by the processor on U43. 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 WD1000 this function is used for setting sector buffer lengths of 128, 256, or 512 bytes.

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 which means that effective addresses are stable to the RAM by at least the 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 -LB which is output by the 8X300. Data to be read from RAM will be placed on the IO bus whenever -LB is low and -WC is high. Data is written into a selected RAM cell on the trailing edge of -WC if -LB is low. During writes, both -WC and -LB 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. This function is used for scratchpad storage during program execution. This mode of RAM access requires 2 or 3 instruction cycles for each random access to the RAM as opposed to 1 for sequential access using the post-decrement feature.

8.2.7 MAC Control Port

Basic control of the various functional sections of the WD1000 is accomplished by a dedicated 6-bit control port called MAC CNTRL (U34). MAC CNTRL enables the functions of the WAIT control circuitry (-WAEN), CRC generation (-CRCIZ), gating of read data into data separation circuitry (RGATE), selection of read or write functions (-WRITE), control of CRC check word output (-1BLA) and AM detection (SRCH). MAC CNTRL output states are latched into the port by a write strobe (WR7). Additionally, any time MAC CNTRL is loaded with a new byte, the lower 2 data bits (IO0-1) are strobed into the upper 2 address counter/latch bits (RA8,9).

All remaining ports are distributed among the basic functional sections of the WD1000 and will be described in detail within the discussion of those functions.

8.3 Serial Data Separation

The WD1000 controller contains onboard circuitry 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 therefore maximum 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".

The fact that clock bits are not recorded with every data bit cell requires circuitry 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 parallelization into bytes.

8.3.1 Incoming Data Selection

In the WD1000, serial data is input from up to 4 radially connected drives via a quad RS-422 differential receiver (U15). 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 (U23). 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 input of another AND/OR/INVERT gate (U18) which will 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 (Q3, U17 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 U17 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) 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 therefore 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. Furthermore, great 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 switching function is initiated immediately after the RGATE goes true and will only switch read data into the VCO after 16 consecutive ones or zeros (high frequency) are detected by a one-shot (U10) and counter (U12) connected directly to the raw MFM data. The one-shot is adjusted for a pulse width of 1.25 times the RCLK period. This is 250 +/-10 nS for ST506 compatible drives and 287 +/-10 nS for SA1000 compatible drives. These adjustments of the DRUN one-shot (U10) provide 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 the up/down counter (U12), whose count mode 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 the -DRUN latch output (U13 pin 8) low which switches read data in and reference clock out. An AND/OR/INVERT gate (U18) performs the switching. -DRUN is read through U42 by the 8X300 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 (U2 and associated circuitry) and the error amplifier and filter (U1). As previously stated, the VCO runs at a frequency twice that of the RCLK rate.

8.3.5 VCO

RCLK is produced by the VCO thru a divide-by-two counter (U8). The VCO is a discrete LC oscillator with a shunt capacitor formed by a hyper-abrupt tuning diode (CR1). The diode varies its capacitance as a function of the amount of reverse DC bias applied to its PN junction. As bias decreases the capacitance increases pulling the oscillator down in frequency and, conversely, as the bias increases, the oscillator frequency rises.

The VCO performs conventionally with one interesting exception. To help the VCO lock onto the incoming signals more quickly, a provision has been made to allow an external timing signal to freeze the output of the VCO in the high state. This is accomplished by disqualifying U2 in the VCO feedback circuit and by removing bias from the transistor (Q2) which provides loop gain in the oscillator. -PHASEUP performs this function and is present just after the switch over from reference clock to MFM data is made.

The width of -PHASEUP is directly related to the difference in timing between the positive going transition of the VCO output and the positive transition of the second data/clock bit of the MFM data stream. -PHASEUP causes the output of the VCO to rise in phase with the MFM data from the drive. This eliminates the need for the VCO to perform a frequency acquisition to lock onto the data stream, but rather only adjust its phase slightly to center data/clocks within the RCLK. The phase acquisition is much faster and easier to achieve and results in vastly improved performance.

8.3.6 Error Amplifier

Control of the VCO is accomplished by the error amplifier and filter. The error amplifier is a balanced differential amplifier whose output sources or sinks current to the filter stage. The output of the error amplifier is pulse width modulated by the phase detector (U6, 7). 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 cathode of the hyper-abrupt tuning diode (CR1). This effectively increases the reverse bias on the diode which reduces its capacitance and therefore increases the VCO frequency slightly to match the phase of the incoming data. Whenever the VCO is running too fast, the error amplifier produces pump-down pulses to the filter. The diode then receives decreased reverse bias and, therefore, more capacitance and lower VCO frequency.

The operating point of the tuning diode (CR1) is initially set for an open loop VCO frequency of two times RCLK by setting -OSC ADJUST and monitoring the VCO output. This adjustment places the initial bias through the bias divider (R18-20, R22 and C8-9) at approximately -2.8 V to -3.2 V. At this setting the most responsive region of the diode is being used giving higher gain in the VCO. To keep the initial bias voltage close to three volts with varying disk data rates, the VCO inductor (L1) is 3.3 uH for 5 Mhz drives and 3.9 uH for 4.34 Mhz drives.

The VCO is forced to match the phase of the incoming data. Once the VCO is close to the phase of the incoming data, the pump pulses will become very small or missing completely. 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.

Previously, we said that great care is taken to insure that the VCO starts on the same phase as the incoming data. If this were not the case, the error amplifier would produce very large pumps in an attempt to pull the VCO onto frequency and phase. Due to the gain of the error amplifier and the required characteristics of the filter, the integrated pump pulses would overcompensate, causing the VCO to overshoot in its attempt to lock-on. This action would continue in a diminishing fashion until lock on-occurred. Unfortunately, the data sync fields it was trying to acquire would be over by the time the VCO finally acquired lock-on. -PHASEUP is, therefore, extremely important to the overall ability of the data separator to function reliably.

8.3.7 Sample on Phase Detector

The circuitry which feeds the error amplifier is called the Sample on Phase Detector. This circuit consists of several D latches (U7) and a delay line (DL1). The function of the circuit is to provide time windows during which the leading edges of the incoming MFM data can be compared to the leading edges of the VCO clock. These windows are approximately 50 nS in length. The windows are initiated by a leading edge of any data bit as it enters the detector. They are terminated by that same data bit, edge-delayed by a net 50 nS (60 nS in the delay line minus approximately 10 nS in propagation delays).

When both the delayed data bit 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, either both pumps are on or both pumps are off, producing no net pump-up or down.

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. The manner in which this is accomplished is by 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 shift is accomplished by tapping the data off the sample on phase detector delay line at the 60 nS tap and inverting the VCO clock to the RCLK divider (U8). The delayed data clocks a pair of latches (U9). The 'data' latch has its D input and CLEAR connected to +RCLK and the 'clock' latch has its D input and CLEAR connected to -RCLK.

If a 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 1/4 the RCLK period without being lost. The output of each latch is then further extended by feeding directly into another stage of latches (U3) 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 WD1000.

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 create circuitry 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 (U8) 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 (U14) by raising the SEARCH signal. This signal releases a latch (U13) which will remove -DHOLD from the RCLK divider (U8) 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 zeros which are detected by the one-shot (U10) in the DRUN circuit. Once released, the phase of RCLK vs. data and clocks will remain stable throughout the read of an ID field 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 is processed through specialized circuitry to prepare it for parallel processing by the 8X300. This processing consists of 3 functional circuits.

1. AM detection (U14)
2. Serial to parallel conversion (U29)
3. CRC checking circuit (U19)

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. Furthermore, it is equally impossible to determine byte boundaries as well. This problem is solved by a uniquely recorded data/clock pattern called an Address Mark (AM). The AM consists of a data pattern of HEX 'A1' with a missing clock pattern of HEX '0A'. Normally a data byte of HEX 'A1' requires a clocking pattern of HEX '0E'. In fact, due to the rules of MFM data encoding, an alternating clock pattern such as HEX 'A' or HEX '5' cannot exist legally.

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 WD1000 first acquires phase lock over a field of zeros. After this acquisition is acheived, the processor releases the AM detector (U14) by raising the SEARCH control line (SRCH) on the MAC CNTRL port (U34). Due to the circuitry associated with the VCO to RCLK divider, the -RDAT output of the data separator (U3 pin 8) will be high and the -CLKS output (U3 pin 6) 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, thereby 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; their 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 (U4) to precisely time the leading edge of -AMDET to the rising edge if RCLK. This synchronization relaxes timing requirements on the data and clock inputs of the serial to parallel converter (U29).

8.4.2 Serial to Parallel Conversion

After an AM has been detected, the Serial to Parallel Converter (U29) 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 (U36). The act of reading the latches also clears off the pending BDONE flag. As successive bytes are processed, the BDONE is serviced by the processor as data becomes available.

Outputs from the serial to parallel device also include -SHFTCLK and -DOUT. -SHFTCLK is actually RCLK propagated through the device. -DOUT is the Q output of the last stage of the shift register string. -DOUT and -SHFTCLK are tri-stated along with BDONE, and are active only when -WRITE is high, indicating a read mode of operation. -DOUT and -SHFTCLK are routed to the CRC generator checker device.

8.4.3 CRC Checking Circuit

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 WD1000, a Cyclic Redundancy Check (CRC) is performed on all data transfers from the disk. The CRC is an error detection code consisting of 16 additional bits which are appended to every ID field and data field on the disk. These bits are produced by dividing the data stream serially with a large polynomial. This division produces a unique 16 bit value for any information passed through the CRC generator.

As data is being read from the disk, the CRC generator re-computes the original CRC bits. After the last two bytes (containing the original recorded CRC) are read, the value in the CRC generator must always be zero. When this happens, the data was correctly read and the controller will not flag an error. If, however, the CRC generator is not zeroed after it has checked all bytes of the recorded data, then the controller will flag the data as erroneous and enter into a retry condition. If after attempting to correctly read the data 16 times, the controller still cannot get correct data, the read is aborted and the host is informed that the data in the buffer is questionable.

The WD1000 uses the same device to generate and check CRC's for data being written and read on the disk. The polynomial used is:

$X^{16} + X^{12} + X^{5} + 1$ (commonly called the CRC-CCITT polynomial).

During read operations, the processor polls the condition of the DRUN circuitry. When DRUN is true, it begins to search for an address mark. Once the AM is located, the processor will start to read parallel data which has been converted from NRZ data by the serial to parallel device. The processor will terminate this activity when it has received the information it is looking for or if an error is detected.

While the processor is reading the parallel data, the CRC generator is reconstructing the CRC check value. The CRC generator is initialized by the processor setting -CRCIZ low for at least 250 nS during the search for the AM. -CRCIZ is originated on the MAC CNTRL port (U34). Upon receiving the -CRCIZ signal, the CRC generator/checker will preset all 16 of its internal polynomial division shift registers to logic ones and arm an internal latch which will enable the checking function on the leading edge of the first non-zero data to enter the device.

It should be remembered that prior to an AM there is always a field of zeros (all data bits low) so the first non-zero data bit into the device will always be the most significant bit of the AM (HEX A1).

Once enabled by the first non-zero data bit, the CRC device will shift succeeding data bits into a feedback shift register string with exclusive OR gates tied to the feedback nodes on the first, fifth, twelfth and sixteenth 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.

When the last bit of an ID or DATA field is processed, the pattern in the registers should be equivalent to the 16 bits appended to the fields during original recording. The appended bits are also entered into the CRC device. If all of the bits in the appended field are identical to the bits in the registers, then the exclusive OR gates in the register string will have flipped all of the ones to zeros and the CRC will have been satisfied.

The output of each register stage is tied to a 16 bit wide comparator which goes true when all of its inputs are zeros. The output of the comparator is retimed to remove any decoding slivers and is output as CRCOK. The processor can read CRCOK through U43 to see if a CRC error occurred.

After the CRC bits are processed, the data stream will contain at least one more byte of zeros. It is the nature of the CRC polynomial that if no bits are set to ones in the registers none will be flipped if a constant input of zeros is shifted into the registers. This provides a convenient latching function for the CRCOK flag which will remain true for at least 1 byte after the last CRC check byte, giving the processor time to read the flag.

The data, clock and BDONE are supplied to the CRC device on a 3-bit mini bus. During read operations, the Serial to Parallel device (U29) will be sourcing these lines because the WRITE control line from MAC CNTRL (U34) is low which enables tri-state drivers on these lines. The Parallel to Serial device (U37) will have its tri-state drivers disabled.

8.5 Serial Data Generation

The WD1000 records data on the disk in MFM format. In order to produce the proper data format, the WD1000 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 (U37)
2. CRC generation (U19)
3. MFM and precompensation (U30)

8.5.1 Parallel to Serial Conversion

Parallel data is converted into a serial NRZ data stream by the Parallel to Serial device (U37). The processor enables this conversion by lowering the -WRITE signal on MAC CNTRL (U34). -WRITE causes the tri-state buffers present on the parallel to serial device to become active, supplying the CRC 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 U43 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 a NRZ serial data stream. The shifting clock is also brought out as SHFCLK to be used as the clock for the CRC device.

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 Generation

The CRC generator/checker (U19) is used to generate the CRC bits and to append them to the end of the data being written to the disk. This is the complementary function to that performed during reads. 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 after the data instead of reading the CRC and checking it.

The initial state of the shift registers within the device is forced to all ones by the processor pulsing -CRCIZ for approximately 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 (HEX 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 device. This will start the CRC polynomial generator and the CRC will be computed.

To write the CRC check bits, the processor will drop the -1BLA (1 Byte Look-Ahead) signal on MAC CNTRL port (U34) just after the last data byte was sent to the parallel to serial converter. The internal switchover from data to check bits is synchronized by the -LD signal being active during the falling edge of -SHFCLK.

This will cause the CRC generator to begin dumping the computed CRC onto the NRZ data stream at the conclusion of the last data byte. In this fashion, the device is able to append the proper CRC information to the end of a field of data. -1BLA is maintained at a low state for the duration of the unloading process which lasts for 16 bit times.

During the unloading process, the CRC 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 which is a requirement for the proper operation of the CRC device during read operations. The NRZ data with CRC appended is then 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 complimentary 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. This line is subject to decoding slivers, so it is run through a re-timing latch (U21) to clean it up.

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. In any event, there is a method which can be applied to reduce the effect of this shift on the data called "write precompensation".

Precompensation 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/clock bit to be written early, late or on time. As with MFMW, these signals are subject to decoding slivers and must be retimed by U21.

The processor can enable or disable the generation of these signals by controlling the RWC (Reduce Write Current) line from the MAC CNTRL port (U34). When RWC is high, precomp is in effect. When RWC is low, no precomp is generated and the NOM output of the device is held true.

The delay line actually performs the precomp with the help of an AND/OR/INVERT gate (U22). 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 (U16) where it is converted to a differential form and then is sent to the drive. The AND/OR/INVERT gate has one other function. If the controller is not writing, the WGI (Write Gate Internal) signal will be low. This is inverted by U38 and applied to the AND/OR/INVERT gate's fourth section. This resulting high input effectively inhibits the gate from accepting MFMW data.

8.6 Host Interface

All data transfers between the host and the WD1000 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 WD1000 is selected by the three address lines (A0-2). All accesses to the WD1000 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 WD1000 provides a not ready signal (WAIT-). For systems using interrupts and/or DMA, the WD1000 provides INTerrupt ReQuest (INTRQ) and Data ReQuest (DRQ).

Accessing the WD1000 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 WD1000 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 WD1000 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 WD1000 are not implemented in hardware, it takes the 8X300 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 8X300 must do to access the desired register. After the data has been written or read, the WD1000 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 (U34) called WAit ENable (WAEN-). If the WD1000 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 control latch (U32), transferring the WAEN- state through the latch, qualifying the wait drivers (U44,54). 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- [BIC or BOC in some installations] 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 (U49). This release is caused by the logical OR of RD6- and WR6- on U25 which presets the wait latch (U32) to a non-wait request condition. The WAIT- signal is stretched to the trailing edge of the RD6- or WR6- by U2.

If WAEN- is de-asserted, the WD1000 generates no waits at all. In this case, the host will read the dummy status written to the communications latch by the 8X300. 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 WD1000 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 (U49) 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 WREQ- signal being asserted, causes a signal, Card Select ACcess (CSAC), to be generated. The 8X300 samples this signal at U43 every 250 nS, and if asserted, reads the status of A0-2 and WE- (U43). 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 WD1000 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 (U30) as an auxiliary function of the chip. The WD1000 sets INTRQ using INTCLK- and sets DRQ Using DRQCLK-, both of which are produced by U20. Interrupts are cleared by HSAC- (Host Select ACcess) and A0,1 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. During Power On Reset or Master Reset (MR-), INTRQ and DRQ is reset.

SECTION 9 MAINTENANCE

The WD1000 requires no scheduled preventative maintenance. There are are a few adjustments associated with the data separation circuitry that may need to be adjusted if a drive with a different data rate is installed. Remember, the inductor L1 must be the proper value for the data rate being used and Y1 must be selected for a fundamental frequency of four times the data rate.

9.1 VCO Adjustments

Data separation circuitry on the WD1000 uses a voltage controlled oscillator (VCO) which phaselocks 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:

Ground the cathode of the tuning diode (CR1) to the closest accessible ground using a low inductance shorting cable. This cable should consist of the shortest piece of wire possible to make the connection.

Connect a frequency counter to the VCO buffered output on TP9.

Apply power to the board. Ensure a logic "1" exists at TP17. A logic "0" on TP17 will inhibit the VCO and make adjustments impossible. If a logic "1" is not present, verify that the DRUN circuitry (U10-U13) is functioning and adjusted properly. Refer to section 9.3 for adjustments to DRUN.

Vary the OSC ADJ pot (R22) to verify the range of adjustment values listed in the following table are obtainable. After the range has been verified, adjust R22 to the final setting listed in the table.

Y1 Frequency	Tank Coil	VCO Range	Final Setting
20.000Mhz	L1 = 3.3 uh	9.0-11.0 MHz	10.0 Mhz +/-1 KHz
17.360Mhz	L1 = 3.9 uh	7.5-9.5 MHz	8.68 MHz +/-1 KHz

Turn off power to the board. Disconnect all jumpers and test equipment.

9.2 Error Amplifier Adjustments (Static)

The phase detection technique used to correct the frequency and phase of the VCO employs a balanced sample and hold error amplifier. To ensure reliable operation of the data separator, the error amplifier must be properly balanced. The balance adjustment must be made using the following procedures:

Apply power to the board. Ground TP20 or U8 pin 6 to turn off the right half of the error amplifier (U1). In this state current will only flow in the left half of the amplifier.

Connect a 100 ohm resistor between U1 pin 8 and ground.

Adjust the BAL pot (R1) until a reading of 0 V +/- 20 mV is observed on TP5.

Remove the ground from TP20. Install a ground to the PUP1 signal line, accessible on U7 pin 1.

Verify a reading of 0 V +/- 50 mV on TP6. This indicates that the side to side balance of the error amplifier is within tolerance.

Re-adjust R1 until the reading at TP6 is 0 V +/- 20 mV.

Turn off power to the board. Disconnect all jumpers and test equipment.

9.3 Error Amplifier Adjustments (Dynamic)

After static balance adjustments are performed, the error amplifier should be adjusted for balanced dynamic operation.

This adjustment requires that the controller be constantly reading data on the innermost cylinder of a formatted drive. While reading data, monitor TP9 with a frequency counter and adjust the error amplifier balance pot (R1) until the most stable display reading is obtained. This indicates that the VCO is being locked on every attempt to read data.

Turn off power to the board. Disconnect all test equipment.

9.4 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 (U10) which is adjustable by the DRUN pot (R26). R26 must be adjusted according to the following procedures:

The DRUN adjustment is made with the WD1000 in an operating test configuration with a host, drive and power source. Once the proper setup is made, apply power to the WD1000 and all test equipment.

Monitor TP14 (-DRUN) 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 TP14, adjust R26. The period of the DRUN single shot should be adjusted to 1.25 times the period of RCLK.

Turn off power to the WD1000. Disconnect 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 WD1000's operating requirements. As you might notice, there is no retry software included in these examples. That is because the WD1000 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

```
;*************************************************************;
;                                                             ;
;            WD1000 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
;driver for the WD1000. It assumes that the WD1000 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 WD1000 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 WD1000 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 WD1000
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 WD1000
RSWAIT:     IN       STATUS                 ;Wait 'till restore done
            ANA      A
            JM       RSWAIT
            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 WD1000 task file
            MVI        A,READ               ;Get READ command
            OUT        COMND                ;Output command to WD1000

;Wait for WD1000 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 WD1000 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 WD1000
            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 WD1000 task file
            MVI         A,WRITE         ;Get WRITE command
            OUT         COMND           ;Output command to WD1000

;Transfer sector from system memory to WD1000
;(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 WD1000
            INX         H               ;Increment memory pointer
            DCR         B               ;Decrement byte counter and continue
            JNZ         WRITLP          ;if we haven't transferred 256 yet

;Wait for WD1000 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 WD1000 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
            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

```
;************************************************************;
;                                                            ;
;            WD1000 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 is 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 WD1000
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.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 WD1000 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 WD1000
            SLEEP       IOWAIT                     ;Put job in an I/O wait state
RSWAIT:     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 WD1000 task file
            LOCK                        ;Disable interrupts 'til we go to sleep
            MOVB        #READ,@#COMND   ;Issue READ command to WD1000

;Wait for WD1000 to read in a sector
            SLEEP       IOWAIT          ;While waiting, go into I/O wait state

;When we wake up, transfer sector from WD1000 to system memory
;(Transfer address in R3, WD1000 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 (WD1000 data
;register) and puts them in a block of memory pointed to by R3. R0 bytes
;are moved.
            MABB        R4,R3           ;Move data from WD1000 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 WD1000 task file
           MOVB       #WRITE,@#COMND ;Issue WRITE command to WD1000

;Transfer sector from system memory to WD1000
;(Transfer address in R3, WD1000 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 WD1000 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 WD1000

;Wait for WD1000 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 WD1000
           RORB       R0                  ;Rotate Error bit to carry
           BCC        DONEOK              ;and return to OS if no error
           MOVB       @#ERROR,R0          ;Get WD1000 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 WD1000 sector number register
            MOVB        R0,(R4)+        ;and send to sector number register
                                        ;and increment R4 to point to WD1000
                                        ;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
            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 WD1000 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 WD1000
            MOV         #DATA,R4        ;Index WD1000 Data Reg for R/W ops
            RTN
```

A.2.5 Interrupt Service routine

```
;**************************************************************;
;                                                              ;
;                 Interrupt Service Routine                    ;
;                                                              ;
;**************************************************************;

;This routine gets called whenever the WD1000 interrupts. All it does is
;read the status register of the WD1000 to clear INTRQ and revives the
;original calling job.

INTSER:     PUSH        R0              ;Save this register
            MOVB        @#STATUS,R0     ;Acknowledge the interrupt to WD1000
            WAKEUP      IOWAIT          ;Wake up job (from I/O wait state)
            POP         R0
            RTT                         ;Return from trap
            END
```

APPENDIX B INTERLEAVE CALCULATING PROGRAM

This simple 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.

```
10      PRINT"WD1000 Interleave calculating program"
20      PRINT
30      INPUT"Number of sectors? ";COUNT
40      INPUT"Interleave Factor? ";INTER
50      DIM HEX$(16),SEC(COUNT)
60      FOR INDEX=1 TO 16
70      READ HEX$(INDEX)
80      NEXT
90      FOR INDEX=1 TO COUNT
100     SEC(INDEX)=-1
110     NEXT
120     FOR INDEX=0 TO COUNT-1
130     IF RES>=COUNT THEN RES=RES-COUNT
140     IF SEC(RES+1)=-1 THEN SEC(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(SEC(INDEX)/16)
210     PRINT HEX$(X+1);HEX$(SEC(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 PROGRAMMER'S QUICK REFERENCE

C.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

C.2 Valid Commands

		BITS							
TYPE	COMMAND	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	0	0	0
III	Write Sector	0	0	1	1	0	0	0	0
III	Format Track	0	1	0	1	0	0	0	0

C.3 SDH Register Format

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

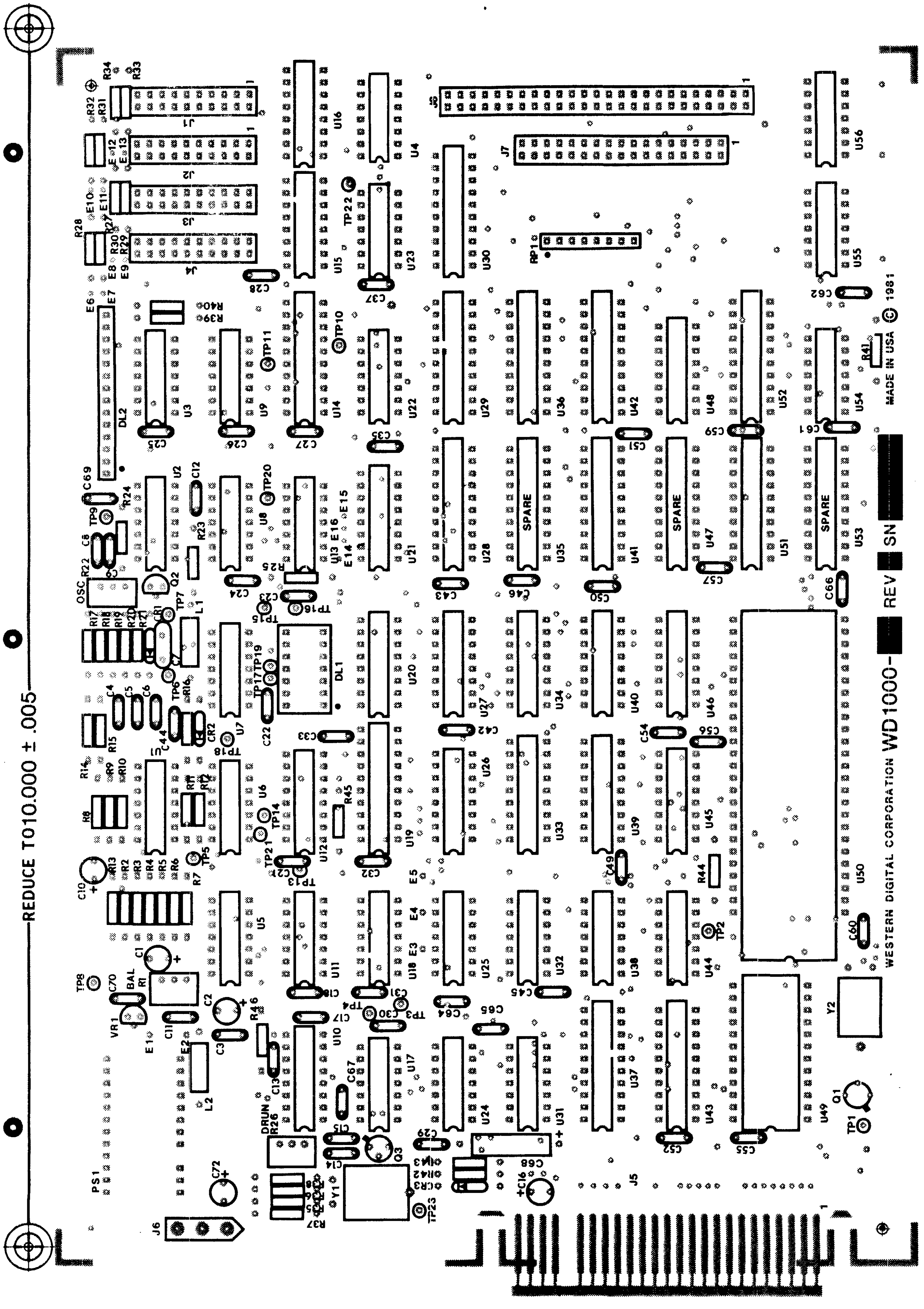
Bit 6	Bit 5	Sector Size
0	0	256 Bytes
0	1	512 Bytes
1	1	128 Bytes

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

C.4 Status and Error Register Bits

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



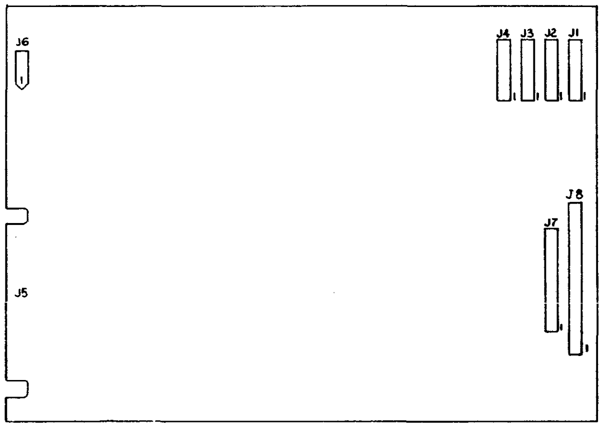
WESTERN DIGITAL CORPORATION WD-1000 F

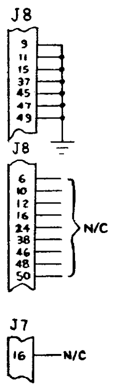
CHG	E.O. NO.	DATE	BY	CHKD
FO	257	8/[illegible]	[illegible]	[illegible]

D.2 SCHEMATIC

D-2

MNEMONIC	MEANING	LOCATION
$\overline{\text{AMDET}}$	ADDRESS MARK DETECT	(C31)
A2-AØ	TASK FILE ADDRESS SELECT BITS 2-Ø	(D16)
$\overline{\text{BIC}}$	BUS INPUT CONTROL	(C15)
$\overline{\text{BOC}}$	BUS OUTPUT CONTROL	(C15)
$\overline{\text{CLKS}}$	CLOCK DATA	(C18)
$\overline{\text{CRCIZ}}$	CYCLIC REDUNDANCY CHECK WORD INITIALIZE	(B3)
CRCOK	CRC OKAY	(C27)
$\overline{\text{CS}}$	CARD SELECT	(C16)
CSAC	CARD SELECT ACCESS CONTROL	(A11)
DAL7-DALØ	DATA ACCESS LINE	(C16)
-DIRECTION IN	DIRECTION CONTROL	(B9)
DHOLD	DATA HOLD	(A19)
DLYDAT	DELAYED DATA	(B19)
DRQ	DATA REQUEST	(B9)
$\overline{\text{DRQCLK}}$	DATA REQUEST CLOCK	(B2)
$\overline{\text{DRSEL}}$	DRIVE SELECT	(B27)
DRS4-DRS1	DRIVE SELECT BITS 4-1	(C12)
$\overline{\text{DRUN}}$	DATA RUN	(B19)
$\overline{\text{HFRQ}}$	HIGH FREQUENCY	(C31)
$\overline{\text{HSAC}}$	HOST SELECT ACCESS CONTROL	(A10)
HS2-HSØ	HEAD SELECT BITS 2-Ø	(D9)
IA9-IAØ	INSTRUCTION ADDRESS LINES BITS 9-Ø	(D5)
ID15-IDØ	INSTRUCTION DATA BITS 15-Ø	(D6, B6)
-INDEX	INDEX PULSE FROM DRIVE	(B16)
$\overline{\text{INTCLK}}$	INTERRUPT CLOCK	(B2)
INTRQ	INTERRUPT REQUEST	(B9)
IO7-IOØ	I/O LINES 7-Ø	(C5)
$\overline{\text{IVBØ}}$-$\overline{\text{IVB7}}$	INPUT VECTOR BUS BITS Ø-7	(C6)
MCLK	MASTER CLOCK	(B5)
$\overline{\text{MFMW}}$	MODIFIED FREQUENCY MODULATION WRITE STREAM	(C26)
$\overline{\text{MR}}$	MASTER RESET	(B15)
$\overline{\text{OSC}}$	OSCILLATOR OUTPUT	(C20)
RA9-RAØ	RAM ADDRESS BITS 9-Ø	(C3)
RCLK	READ CLOCK	(C19)
$\overline{\text{RCS}}$	RAM CHIP SELECT	(D4)
$\overline{\text{RDAT}}$	READ DATA	(D18)
RD6-RD4 RD2 RDØ	READ CONTROL LINES	(B2)
$\overline{\text{RE}}$	READ ENABLE	(C16)
-READY	READY STATUS FROM DRIVE	(A16)
$\overline{\text{RESET}}$	RESET SIGNAL	(A5)
RGATE	READ GATE	(B3)
$\overline{\text{ROVF}}$	RAM OVERFLOW	(C3)
RWC	REDUCE WRITE CURRENT	(C12)
-SEEK COMPLETE	SEEK COMPLETE STATUS FROM DRIVE	(A16)
SRCH	SEARCH	(B3)
-STEP PULSE	STEP PULSE TO DRIVE	(C9)
TIMCLK	TIMING CLOCK FOR SA1000	(C27)
TRACK ØØØ	TRACK ØØØ STATUS FROM DRIVE	(B16)
$\overline{\text{WAEN}}$	WAIT ENABLE	(B3)
$\overline{\text{WAIT}}$	MEMORY NOT READY SIGNAL	(B9)
WCLK	WRITE CLOCK	(A20)
$\overline{\text{WE}}$	WRITE ENABLE	(B16)
WGI	WRITE GATE INTERNAL	(C12)
-WRITE FAULT	WRITE FAULT STATUS FROM DRIVE	(A16)
WR7-WRØ	WRITE CONTROL LINES	(A2)
$\overline{\text{1BLA}}$	1 BYTE LOOK AHEAD	(B3)
$\overline{\text{2XDR}}$	2 X DATA REFFERENCE CLOCK	(A20)





NOTES: UNLESS OTHERWISE SPECIFIED.

(1) OPTIONAL JUMPER COMPONENT SIDE ONLY
SOLDER FOR: SA-1000 DRIVE
OPEN FOR: ST-506

2 RESISTOR VALUES ARE IN OHMS
±5%, 1/4W

(3) L1 = 3.3μH FOR ST-506 DRIVE
L1 = 3.9μH FOR SA-1000 DRIVE

(4) Y1 = 20.000 MHZ FOR ST-506 DRIVE
Y1 = 17 360 MHZ FOR SA-1000 DRIVE

(5) JUMPER IF -VIN (J5-33 OR J6-2) IS USED. OPEN
IF PS-1 (-5V DC/DC CONVERTOR) IS USED

(6) JUMPER E5 TO E4 TO QUALIFY $\overline{\text{WAIT}}$ BY $\overline{\text{CS}}$
JUMPER E5 TO E4 IF $\overline{\text{BIC}}$·$\overline{\text{BOC}}$ TO QUALIFY $\overline{\text{WAIT}}$

(7) OPTIONAL COMPONENTS: USE EITHER ONE (PS1 OR VR1)

(8) NOT USED, USED ONLY WHEN FOR EXPANDING
MEMORY TO 1K.

(9) C53 HAS BEEN DELETED ON REV., "F" ARTWORK.

(10) U4 IS AN ADDITION ON REV., "F" AND THEREFORE
PHYSICALLY OUT OF SEQUENCE.

(11) JUMPER E14 TO E16 AND CUT TRACE TO E15 TO
ACCOMODATE -4002 PART IN U30.

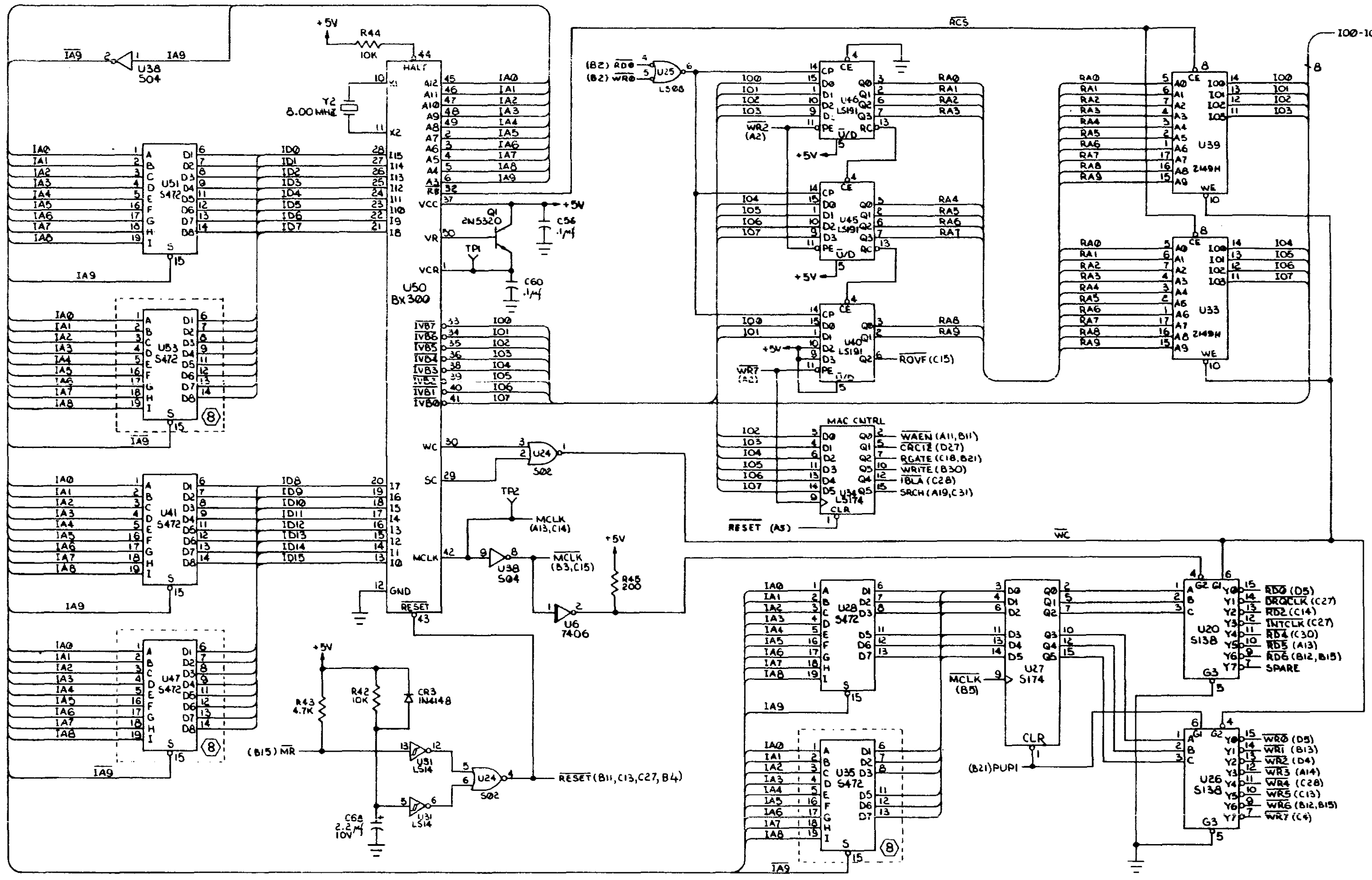
DRAWN	[illegible] 8/14/81	WESTERN DIGITAL CORPORATION	TITLE WD-1000 DISK INTERFACE
CHECK	[illegible]		
DESIGN			DWG NO. 60-031001-20 REV FO
ENGR	[illegible]	ISSUE DATE 4-1-81 SIZE D	
OPER		SCALE	SHEET 1 OF 5

NOTES: UNLESS OTHERWISE SPECIFIED.

D.2.1 MICRO CONTROLLER

D-3

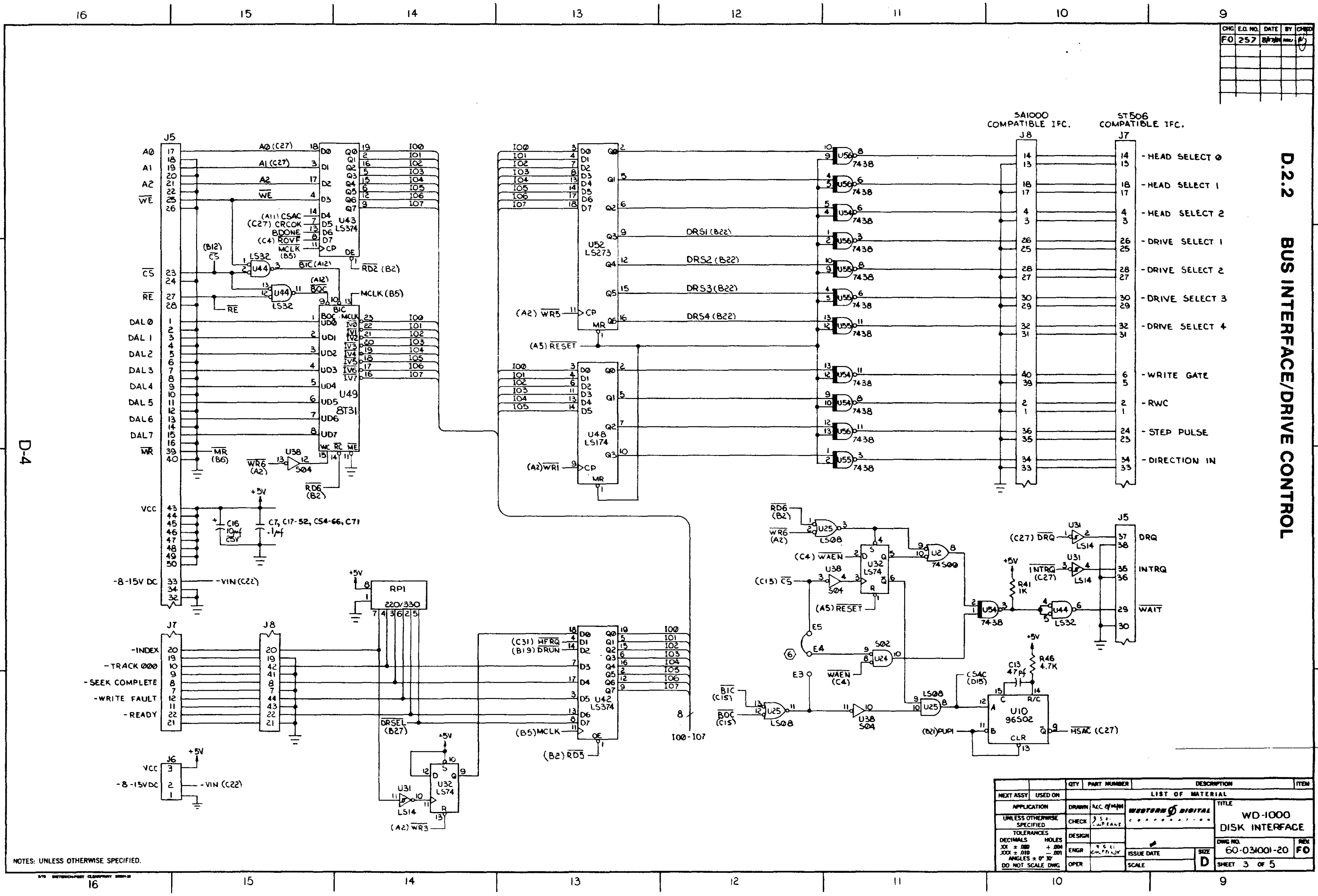
CHG.	E.O. NO.	DATE	BY	CHKD
FO	257	8/17/81	[illegible]	[illegible]



NOTES UNLESS OTHERWISE SPECIFIED

DRAWN		WESTERN DIGITAL CORPORATION	TITLE WD-1000 DISK INTERFACE
CHECK	8-5-81 [illegible]		
DESIGN			DWG NO. 60-031001-20 REV FO
ENGR	8-5-81 [illegible]	ISSUE DATE	SIZE D
OPER		SCALE	SHEET 2 OF 5

D.2.2 BUS INTERFACE/DRIVE CONTROL



CHG	E.O. NO.	DATE	BY	CHKD
F0	257	8/7/81	[illegible]	[illegible]

LIST OF MATERIAL

WESTERN DIGITAL CORPORATION

TITLE: WD-1000 DISK INTERFACE

DRAWN: R.C. 6/4/81

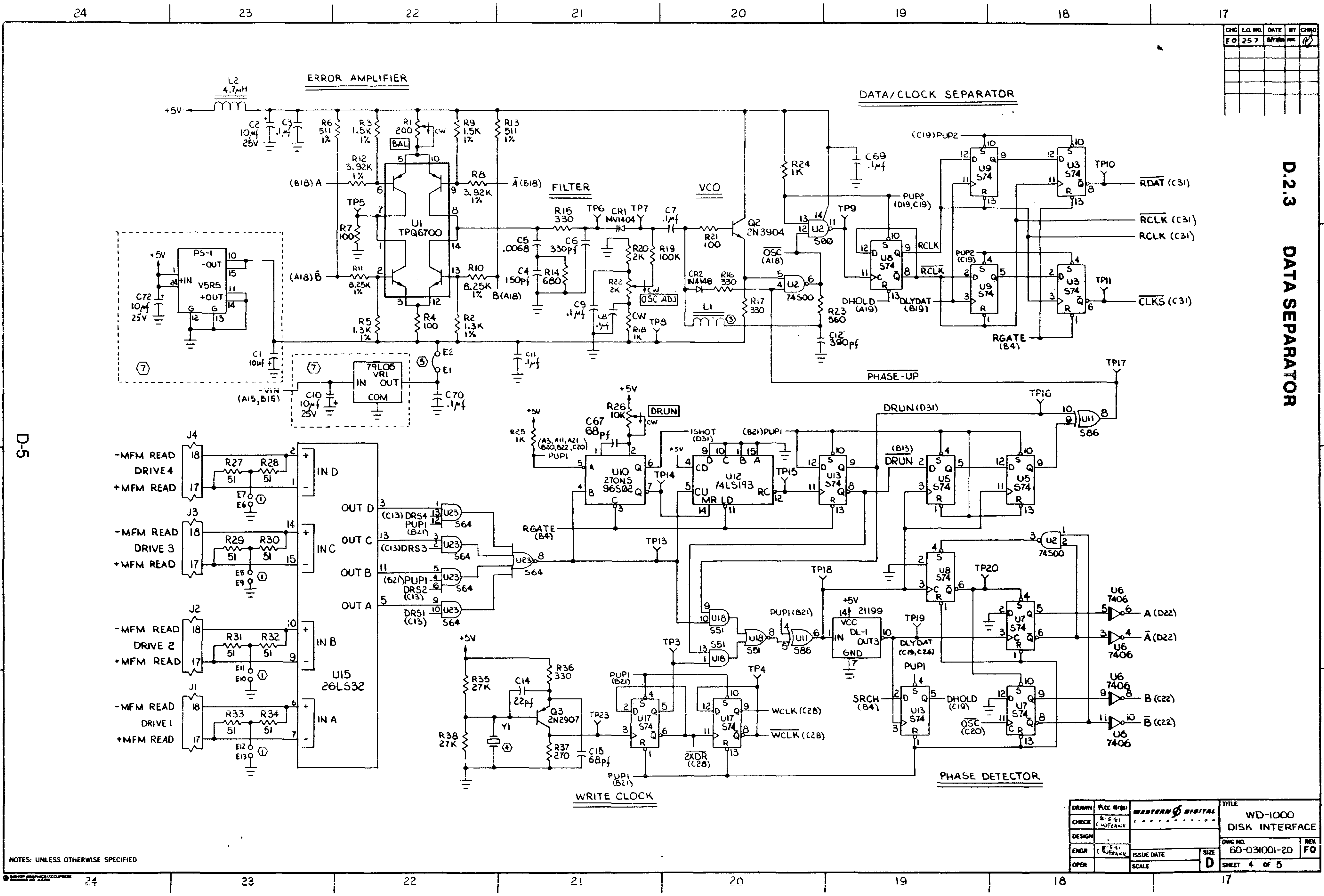
SIZE: D

DWG NO. 60-031001-20 REV F0

SHEET 3 OF 5

UNLESS OTHERWISE SPECIFIED TOLERANCES DECIMALS .XX ± .030 .XXX ± .010 HOLES +.004 -.001 ANGLES ± 0° 30' DO NOT SCALE DWG

D.2.3 DATA SEPARATOR

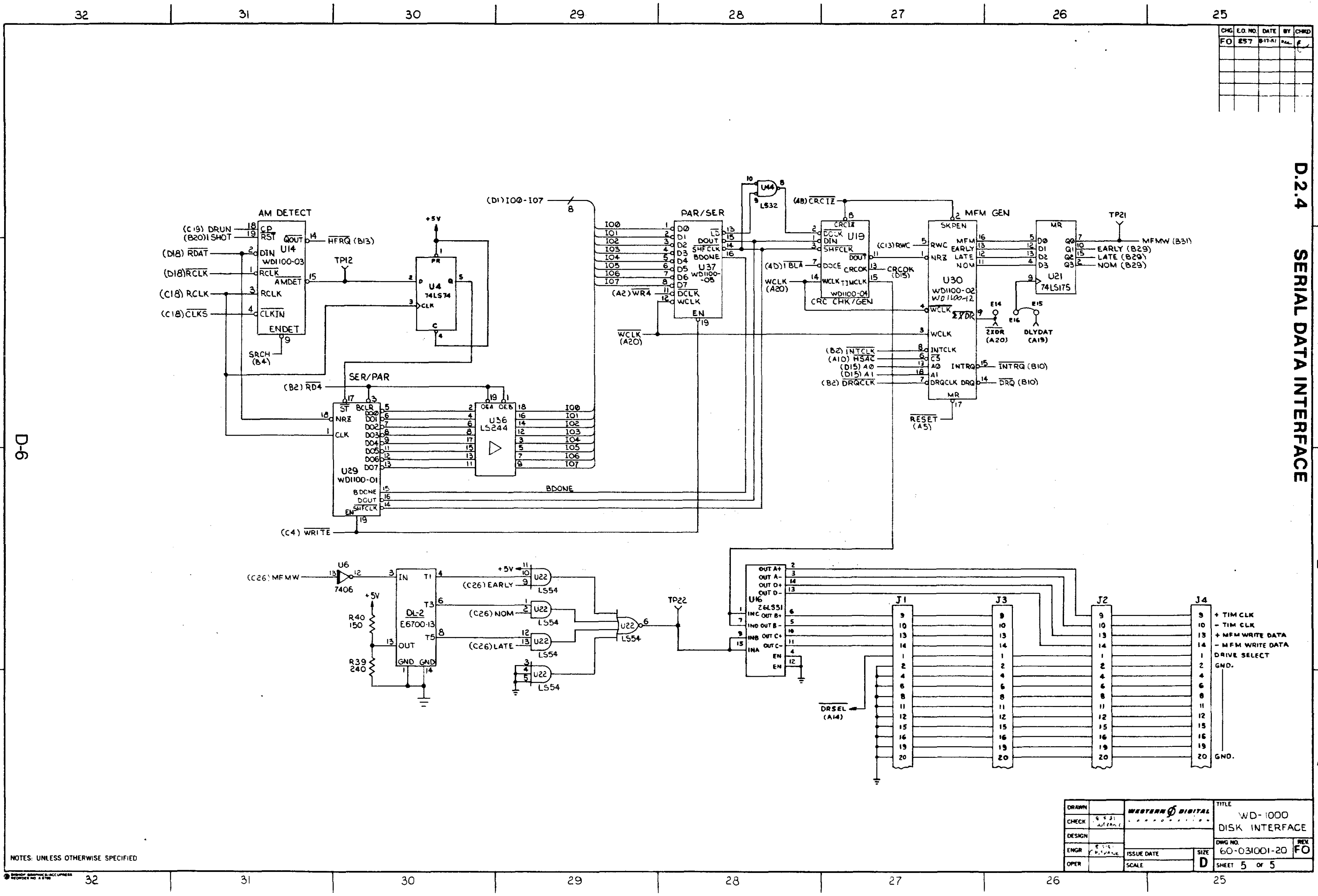


CHG	E.O. NO.	DATE	BY	CHKD
F0	257	8/17/81	[illegible]	[illegible]

NOTES: UNLESS OTHERWISE SPECIFIED.

DRAWN	R.CC 8/1/81	WESTERN DIGITAL CORPORATION	TITLE WD-1000 DISK INTERFACE
CHECK	8-5-81 C.W.FRANK		
DESIGN			DWG NO. 60-031001-20 / REV F0
ENGR	8-5-81 C.W.FRANK	ISSUE DATE	SIZE D
OPER		SCALE	SHEET 4 OF 5

D.2.4 SERIAL DATA INTERFACE



NOTES: UNLESS OTHERWISE SPECIFIED

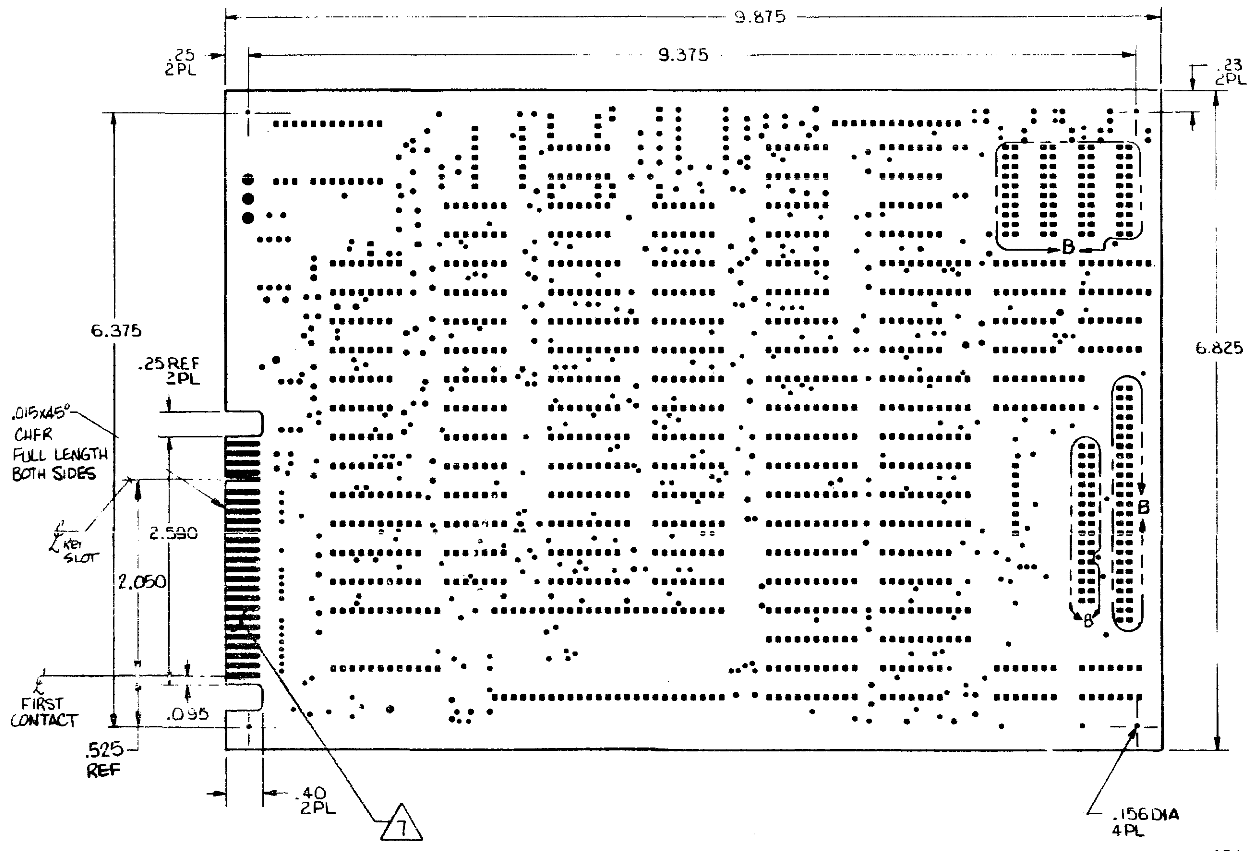
CHG	E.O. NO.	DATE	BY	CHKD
FO	857	8-17-81	[illegible]	[illegible]

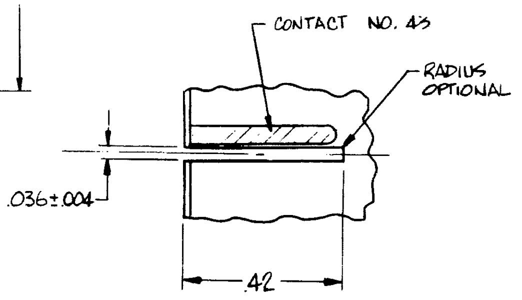
WESTERN DIGITAL

TITLE WD-1000 DISK INTERFACE

DWG NO. 60-031001-20 REV FO

SIZE D SHEET 5 OF 5





DETAIL - C

KEY SLOT

4X SIZE

8. SILKSCREEN CHARACTERS IN WHITE INK ON COMPONENT SIDE
7. PLATE CONNECTOR AREA 50 MICROINCHES GOLD OVER 500 MICROINCHES NICKEL
6. PLATE HOLES AND EXPOSED TERMINAL AREA WITH SN60 SOLDER
5. SOLDER MASK OVER BARE COPPER BOTH SIDES
4. HOLES LOCATED AT TERMINAL AREA CENTERS WITHIN .010DIA OF TRUE POSITION
3. FRONT-TO-BACK REGISTRATION WITHIN .005
2. PLATE HOLES THROUGH .001 MIN WALL
1. MAT'L: FL-GFN 062C-1/1-A1A

FINISHED HOLE SIZES		
LETTER	DIA ±.002	QTY
UNMARKED	.030	1612
A	.062	3
B	.040	164