3100 Class Schedule

Monday

8:30 To 4:30 Tolland

3100 Overview and CPU Theory

Tuesday

Noon To 5:00 Tolland

Interpolator Theory

6:00 To 9:00 Main Plant

Operations

Wednesday

Noon To 4:00 Tolland

OEM Interface, Parallel, Power Supply, MTA

5:00 To 9:00 Main Plant

Diagnostics

Thursday

Noon To 4:00 Tolland

Instruction Set / GSIBug

5:00 To 9:00 Main Plant

GSIBug

Friday

8:00 To End Tolland

SASI, XEBEC - Test

SWITCHES AND JUMPERS

DEFINITIONS AND SETTINGS

CPU BOARD

Momentary: Abort switch (red) - program defined
Reset switch (black) - hardwired

Dip Switches: Used to run on board diagnostics.
(bank of 8) All 8 switches are program defined.

	ON	OFF
Switch 1	Boot from floppy at program load time OR To exit a running diagnostic	Start selected diagnostic

Switches 2 thru 8 Diagnostic Program Defined

INTERPOLATOR

IC's 73 and 75 (DIP switches - 2 banks of 8)	Software configuration switches
IC 76 (DIP switches - 1 bank of 2)	Motor Pulse Width Switches OFF=narrow pulse ON=wider pulse
IC 8 (DIP switches - 1 bank of 4)	Interpolator selection All off=Board Ø

SASI BOARD

Jumpers 1 thru 8 are located near IC11.

Jumper 2, 4 > Base Address
6 Interrupt request level 3

XEBEC FLOPPY CONTROLLER

Jumper S to 1

MAG. TAPE CONTROLLER

Jumper 7 to 8 on J2 Interrupt Request Level 1
Jumper 1 to 2 on J3
3 to 4
5 to 6
7 to 8
> Base Address

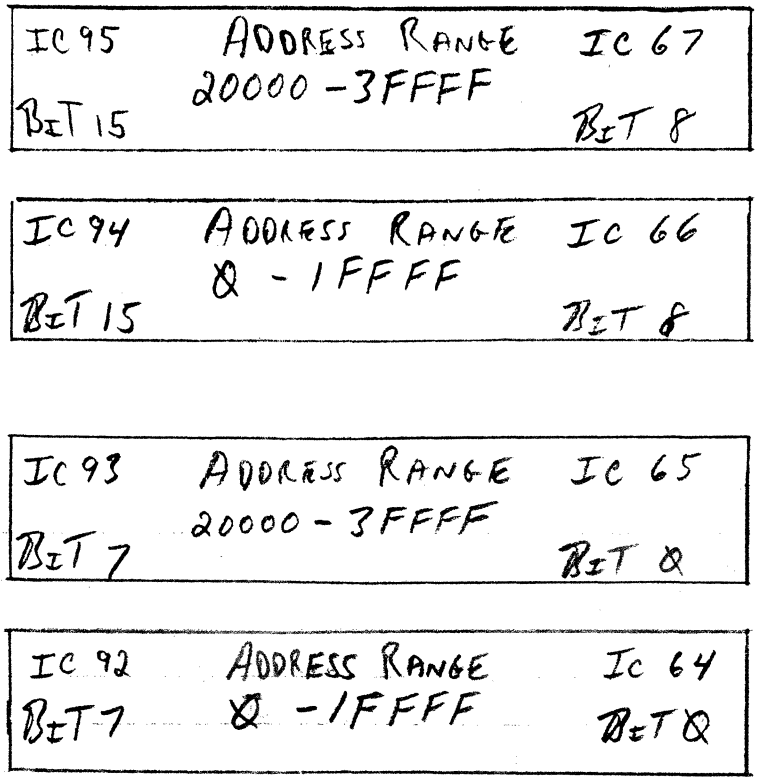
VIEWPOINT TERMINAL

Switch S1: 3,5,7,8 are up

PARALLEL INTERFACE

Jumper:
3 on JP1
2,3,4 on JP2

CPU RAM MEMORY LAYOUT

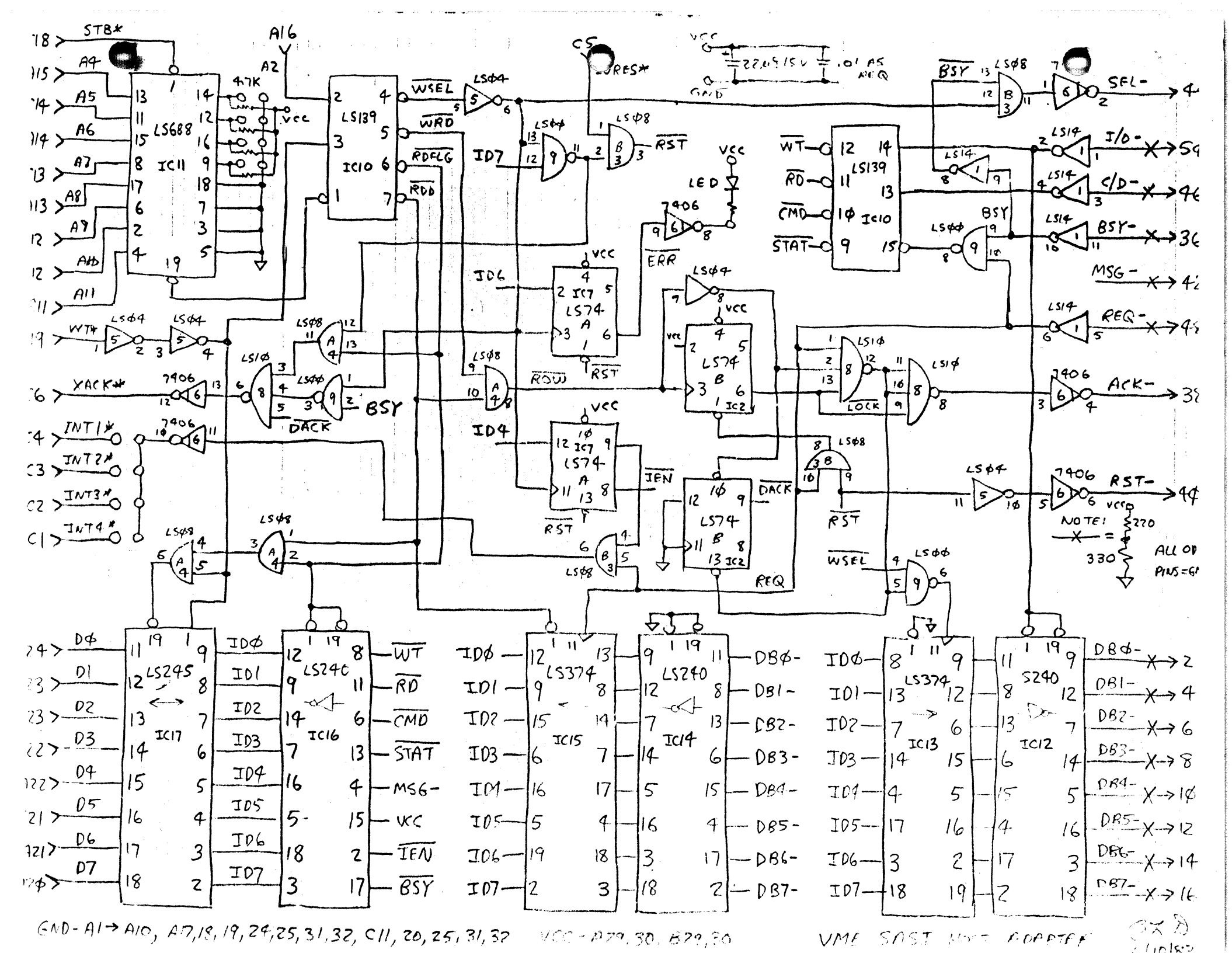


FRONT

3100 Control PC Boards

August 30, 1983

Standard	- GNC-1	CPU	Released (3)	
	- GEC-1	INTRP	Rel. 9/9	
	- GTP-206	Backplane	Released	
	- GTP-215	CTL Panel	Released (5)	
	- XEBEC S-1401	Floppy	Purch.	
	- GVM-1 (OWL)	SASI (Floppy)	Rel. 9/2	
Variable (One required)	- GTP-222	OEHI & OEHC	Rel. 9/2	RELEASED 9/
	To Be Done - GTP-2xx	OEHF	Rel. -	
	Options - GTP-2xx	35	Rel. -	
	- GTP-2xx	DEC Emitter Film Cutter	Rel. -	
Options	- Motorola MVME435	Mag Tape	Stock	
	- OWL GVM-2	Parallel	Rel. -	NOT RELEASED Protos 6/24
	- GTP-2xx	NEXT Quick Look	Rel. -	
	- Motorola MVME410	Dual 16 bit	Purch.	Used w/PPT and Series 1
	- GTP-2xx	IBM Series 1	Rel. -	Mounted on connector panel


STB*
A16
A2
A4
A5
A6
A7
A8
A9
A10
A11
LS688
IC11
4.7K
Vcc
CS
DRES*
LS139
IC10
WSEL
WRD
RDFLG
RDD
LS04
LS08
RST
ID7
ID6
ID4
LED
7406
ERR
WT
RD
CMD
STAT
BSY
SEL-
I/O-
C/D-
MSG-
REQ-
ACK-
RST-
WT*
XACK*
INT1*
INT2*
INT3*
INT4*
DACK
ROD
IEN
LS74
IC7
IC2
LS10
LOCK
REQ
NOTE: 220
330
ALL OD
PINS=6
D0
D1
D2
D3
D4
D5
D6
D7
LS245
IC17
LS240
IC16
LS374
IC15
IC14
IC13
IC12
ID0
ID1
ID2
ID3
ID4
ID5
ID6
ID7
DB0-
DB1-
DB2-
DB3-
DB4-
DB5-
DB6-
DB7-
MSG-
VCC
GND - A1→A10, A17,18,19,24,25,31,32, C11, 20, 25, 31, 32
VME SAST HOST ADAPTER

IC IDENTIFICATION

CPU

74LS138 - 1 of 8 Binary Decoder
74LS148 - Priority Encoder (Binary Out)
74LS155 - Dual 1 to 4 Line Demultiplexers
74LS161 - Synchronous 4 Bit Counter
74LS164 - 8 Bit Parallel-out Shift Register
74LS175 - Quad D-Type Flip-Flops w/Common Clock and Clear
74LS191 - Synchronous Up/Down Counters
74LS244 - Octal Tri-State Line Drivers
74LS245 - Octal Tri-State Bus Transceiver
74LS367 - Six Tri-State Buffers
74LS368 - Six Tri-State Inverting Buffers
74LS373 - Octal Tri-State D-Type Transparent Latches (Register)
555 - Timer

INTERPOLATOR

7442 - BCD to Decimal Decoder
7497 - 6 Bit Binary Rate Multipliers (BRM)
74LS123 - One Shot
74LS136 - Quad 2 Input Exclusive-OR w/open Collector Outputs
74LS138 - 1 of 8 Binary Decoder
74LS139 - Dual 2 to 4 Line Decoder
74LS157 - Quad 2 to 1 Line Data Selectors/Multiplexers
74LS161 - Synchronous 4 Bit Counter
74LS175 - Quad D-Type Flip-Flops
74LS191 - Up/Down Counter
74LS240 - Octal Tri-State Buffers
74LS273 - Octal D-Type Latches
74LS283 - 4 Bit Binary Adder w/Look Ahead Carry
74LS393 - Dual 4 Bit Binary Counter w/Individual Clocks

14490 - Hex Switch Debounce Circuits
26LS31 - Quad Differential Drivers

INTERRUPT AUTO VECTOR

Level

7	Abort Switch
6	DMA
5	Plotter Port
4	Duart
3	I/O Channel 1 (TAPE)
2	I/O Channel 2 (PARALLEL, PAPER TAPE)
1	I/O Channel 3 (FLOPPY)

MEMORY MAP

Address Range	Assigned to
00000 thru 3FFFF	RAM Memory (256KB)
40000 thru 7FFFF	Peripherals
80000 thru BFFFF	EPROM (16 Bit, Only 16KB is used)

PERIPHERAL BASE ADDRESSES

40000 thru 41FFF	I/O Channel (D0-D7) 4KB on Byte Boundary
40000	Mag Tape
40060	Floppy
40100	Parallel (16 Bit)
42000 thru 43FFF	I/O Channel (Redundant)
44000 thru 47FFF	DMA Port (D0-D15)
48000 thru 4BFFF	Duart Port (D0-D7)
4C000 thru 4EFFF	Plotter Port (D0-D15)
50000 thru 53FFF	Diagnostic Port (D0-D7)

DMA PORT

R/W to Address	Control Signal	Data Bits	Function
W to $44000	$\overline{\text{LPS}}$-Load Port Select (Interpolator Board selection)	D0,D1,D2	Selects 1 of 8 ports for DMA transfer.
		D3	Controls A17 during DMA transfer. DMA buffer is lower (0) or upper (1) 128KB of RAM.
		D4-D15	N/A
W to $44002	$\overline{\text{LBP}}$ Load Buffer Pointer	D0-D15	Correspond to A1→A16 Pointer can be set to any even address in RAM above $4000. Area below $4000 is hardware inhibited.
W to $44004	$\overline{\text{LBC}}$ Load Buffer Counter	D0-D15	Correspond to the number of single word transfers. (64KW maximum)
W to $44006	$\overline{\text{STARTD}}$ Enables DMA Transfer	D0-D15	N/A
R to $44004	$\overline{\text{RBC}}$ Read Buffer Counter	D0-D15	Can be read any time. Reading clears DMAIRQ.

During DMA interrupt service, the buffer counter must be read to determine transfer completion (zero count) or a fault condition (non-zero).

DMA will be disabled on each $\overline{\text{DMAIRQ}}$. $\overline{\text{DMAIRQ}}$ is generated on the completion or fault. A fault condition occurs during an access of RAM below $4000 or trying to access non existing RAM. In both cases DMA will time out.

WRITING TO PLOTTER PORT

A10-A13 Contain Port Select Code

R/W to Address	Control Signal	Data Bits	Function
W to 4CXX0	$\overline{\text{LDM}}$ Load M Register	D0-D15	See Next Page
W to 4CXX2	$\overline{\text{LDR}}$ Load R Register	D0-D15	Preset to 1/2 Ratio
W to 4CXX4	$\overline{\text{LDD}}$ Load D Register	D0-D15	Distance Word
W to 4CXX6	$\overline{\text{LDC1}}$ Load Control 1	D0=1	Clears Start/Stop Status Bit 0, non latched
		D4=1	Turns error LED on, latched
		D5=1	Turns start LED on, latched
		D6	Not used
		D7=1	Enables DMA transfer to selected device
		Other Bits	N/A
W to 4CXX8	$\overline{\text{LDC2}}$ Load Control 2	D0-D7	Motor Pulse Clock Value
		D8-D15	Halt Clock Value
W to 4CXXA	Generates Clear	N/A	Write to this address clears device
W to 4CXXC W to 4CXXE	Reserved to control special photohead functions.		

M WORD FOR 3100

15	14	13	12	11	10	9	8	7 - 0
NMV (Delay)	Tool Select	2WD. Xfer	MME (45°)	PUD	MJA	B Sign	A Sign	Velocity

Ratio is sent on a 3 word transfer; this word is used until the next 3 word transfer is given.

If Bit 13 is set, this is a 2 word transfer (M,D) otherwise, transfer 3 words (M,R,D).

If Bit 14 is set, load tool registers with MØ→M7; decode M8,M9 for tool select.

M9	M8	
0	0	- Pen Select
0	1	- Aperture Select
1	0	- Reserved
1	1	- Reserved

Velocity Bits M2,M1,M0	Pen Selection
	0-
	1-Pen 1
	2-Pen 2
	3-Pen 3
	4-Pen 4
	5-Pen 5
	6-Pen 6
	7-SHTOP

Velocity Bits M3,M2 / M1,M0 > 0 thru 12 - CP0 thru CP11

M4 = Side Select

Bit 15 (NMV) means no move. Setting this Bit will disable motor pulses. Executes delay based on velocity and distance bits value.

Bit 12 (MME) means min/max equal. Setting this Bit will generate a 45° line.

READING FROM PLOTTER PORT

R/W to Address	Control Signal	Data Bits	Function
R to 4CXX0	$\overline{\text{RDW1}}$ Read Word 1	D0-D15	Read Status Word

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
+X Slew	-X Slew	+Y Slew	-Y Slew	Plotter on X And Y Limit Switches	-	-	-	Buffer Is Empty	-	No Photo Head Mode	-	-	ESP Fault	Lamp Fault	Start/ Stop Switch

Condition is true when specified Bit=1.

Generates IRQ

R to 4CXX2	$\overline{\text{RDW2}}$ Read Word 2	D0-D15	Read Configuration Switches (see next page)
R to 4CXX4	$\overline{\text{RDW3}}$ Read Word 3	D0-D15	Read Status Word and clear IRQ

R to 4CXX6
R to 4CXX8
R to 4CXXA
> Reserved for OEH status.

Note: A10 thru A13 contain port select code for systems containing multiple interpolators (ports).

OEH I MODULE

Module has redundant 6 Bit M Register.

D0-D4 For aperture select
D11 - To detect pen down move

Bits 14 and 9 = 1, and Bit 8 = 0 will load module M register when XFB=1.

After aperture select has been initiated and the attempt to open the shutter follows it (M11=1), the circuit generates wait signal to interpolator. This prohibits the transfer and execution of the next command set (M,R,D) until aperture select is done. However, if the shutter is not going to be opened, the plotter can be moved while aperture select is in progress.

3100 POWER UP SEQUENCE

(Applies to proms marked GSI bug Ø5 only)

After the reset or program load switch is depressed, the bank of 8 DIP switches on the CPU board are read in.

Then:

A) If switches 2,3,6,7 are on and 1,4,5,8 are off, we will enter "GSIBUG".

B) If switch 1 is off and all the rest are on, the "Extended Memory Test" will begin to run.

Note: See "Memtest" description for operating instructions.

C) Any other switch setting will cause:

1st step A "Power up memory test" to run for 7 seconds (approx.) When the test starts, the start LED on the control box will turn on. If the test completes successfully, the start LED will be turned off. If an error is detected, the error LED on the control box will turn on and an error message will be displayed in the LEDs on the CPU board. (See "Memtest" description for details.) Then the program will stop (loop).

2nd step If the memory test runs successfully, then the "boot" program will issue the internal and RAM diagnostic commands to the floppy controller before attempting to read from the disc. If these two commands or any other commands do not complete successfully, the controller error code will be displayed in LEDs on the CPU board, the error LEDs on the "SASI" board and the control box will turn on. Then the "boot" program will start from the beginning again (keep trying).

FATAL ERROR

A fatal error in the "boot" program occurs when we get an error during the "request sense status" command. We issue this command after an error is detected during another command to get the error code. A fatal error will turn on the "SASI" board and control box error LEDs and then the program will stop. At this point, the only option is to depress reset or program load to start the power up sequence from the beginning.

3rd step The "boot" program reads in our software starting at track 1 and sector 1. If an error occurs during this process it will be handled as described above in step 2.

Xebec Controller (S-1401C) Error Codes

The following are the error codes and descriptions that apply when this controller drives the Tandon TM-100 5¼ inch floppy drive on the 3100.

If an error occurs while booting, the error code will be displayed in the LEDs on the CPU board, provided a fatal error does not occur (see first page).

EXAMPLE: Refer to last page.

HEX CODE	
14	The most significant digit, 1, will be displayed on LED bank 3 in binary. LED #5 is the LSB and LED #8 is the MSB.
	The least significant digit, 4, will be displayed on LED bank 2 in binary. LED #1 is the LSB and LED #4 is the MSB.

HEX CODE	DEFINITION
00	The controller detected no error during the execution of the previous operation.
06	After stepping maximum number of cylinders, controller did not received track 00 signal from drive.
10	An error was detected in the CRC of the target ID zone.
11	Write fault.
12	A write was attempted on a write protected floppy diskette.
14	Sector Not Found: The controller found the correct cylinder and head, but not the target sector.
15	Seek Error: The controller detected an incorrect cylinder or track, or both.
16	Format track timeout error.
17	Format track not complete.
19	Two sided error.
1A	Wrong data mark found.
1D	Lost data in FDC.
1E	Data CRC error detected.

HEX CODE	DEFINITION
1F	FDC busy error. The LSI FDC chip was busy executing a command. This is a timeout or chip reset failure.
20	Invalid Command: The controller has received an invalid command from the host.
21	Illegal Disk Address. The controller detected an address that is not within the limits of the drive characteristics.
22	Something in the 8 bytes of data which specify the drive is not permissable.
23	Invalid interleave.
30	RAM Error: The controller detected a data error during the RAM sector-buffer diagnostic.
31	Program Memory Checksum Error: During its internal diagnostic, the controller detected a program-memory checksum error.

MEMTEST

Overview

This is the memory test for the RAM (currently 256KB) on the 3100 CPU board. This program resides in prom on the CPU board and uses RAM ONLY to output error messages to the CRT. ERROR MESSAGES OUTPUT TO THE CRT USE RAM FOR A PRINT BUFFER. THE AREA OF RAM USED IS BELOW ADDRESS $800. THEREFORE, IF LOWER MEMORY ($0-$20000) IS "DEAD", OUTPUTTING AN ERROR MESSAGE TO THE CRT WON'T WORK. IN THIS INSTANCE, THE ERROR CODE DISPLAYED IN THE LEDS ON THE CPU BOARD MUST BE USED.

During one complete pass, this program writes and verifies 4 different data patterns, and verifies RAM addressing twice. One pass takes about 25 seconds.

A portion of "Ramtest" is also run on initial power up. Locations $0000 thru $3FFFE are tested by writing and verifying test pattern $0000, then one pass of the address test is run. If successful, the start LED on the control box will turn on when the test is started and off when it ends, then the system will boot. Upon detecting an error, the control box error LED will light and if switch 2 is on, an error message will be output to the CRT. REGARDLESS OF SWITCH 2'S POSITION, AN ERROR CODE WILL BE DISPLAYED IN THE LEDS ON THE CPU BOARD.

NOTE: The power up ramtest takes about 6 seconds start to finish.

Datatest

This test is specifically designed to test 64K RAM chips. The program writes the test pattern to all of memory first then goes back and compares what was written to what it reads. This sequence is done to force refreshes between the write and read portions of the test. The four test patterns used are:

$0000
$FFFF
$5555
$AAAA

The last two test patterns will be complimented before being written to a memory location (except for the first location and every 257th location). This results in a true checkerboard pattern within a RAM array of 256 by 256.

Address Test

A "march pattern" test is run immediately after each of the first two test pattern data tests: $0000 and $FFFF.

The "march pattern" test "walks" A 1 or A 0 bit thru a RAM chip array previously filled with bits of the opposite polarity, insuring along the way that every location in the array answers to it's own address only.

CPU Board Switch Assignments

Switch 1 OFF Starts extended memory test.
ON Stops extended memory test.

Switch 2 ON Error messages output to CRT as well as LEDS.
OFF Error messages output to LEDS only and stop testing.

Switch 8 has meaning only when error messages are being output to the CRT (switch 2 "on").

Switch 8 ON Halt on single bit error and wait for start switch before testing next location.
OFF Display errors but do not halt.

CPU Board LEDS

5-8 ON = Error in upper 128KB
5-8 OFF = Error in lower 128KB

1-4 = Failing bit in binary
LED 1 is the lsb
LED 4 is the msb

(see last page)

Control Box LEDS

Start = turns on when testing begins
turns off when testing ends

Error = turns on when error detected

CONTROL BOX SWITCHES

The start switch is used after an error has been detected to restart testing at the failing location plus 2 only when switch 2 is "on" (outputting error messages to CRT).

OPERATING INSTRUCTIONS

1. Set CPU board switch 1 "off" to start the extended test.
2. Set CPU board switch 8 and 2 to desired position.
3. Depress program load switch. If the test is running, the control box start LED should be lit.

When you want to stop the test, turn switch 1 "on", the test will then complete the current pass, turn off the start LED and stop.

Displaying Errors

If switch 2 is off:

Upon detecting an error, the program will turn on the control box error LED and display the failing bit and address range in the CPU LEDS as described above. If more than one bit is found to be failing, the first failing bit will be displayed and the display will then flash on and off. In either case, testing will stop.

If switch 2 is on:

Error messages will be output to the CRT in the following format.

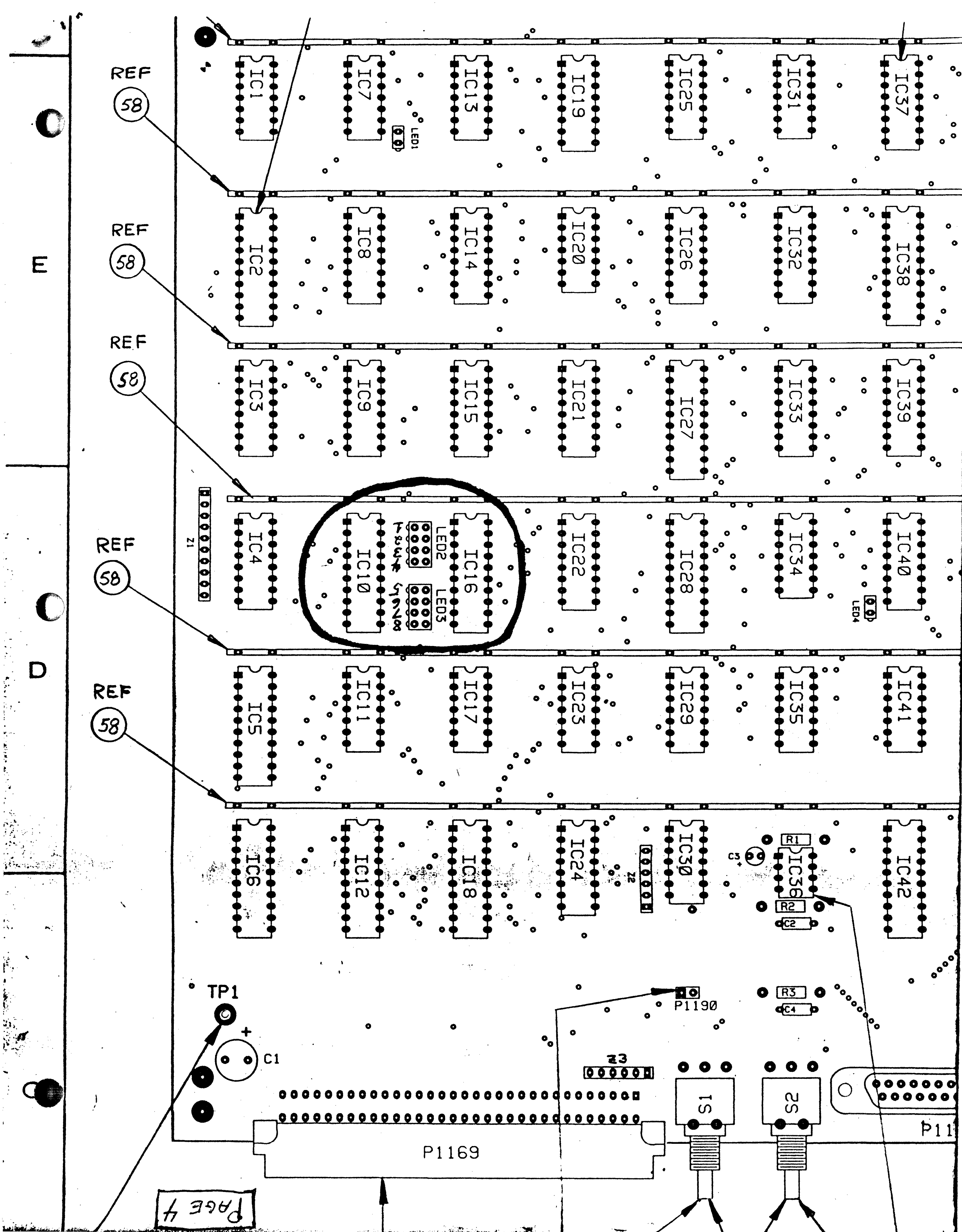
Failed at: (address) Wrote: (test pattern) Read:(data read back)

If switch 8 and 2 are "on":

The start switch will cause testing to resume at the failing address +2.

If switch 8 is "off" and switch 2 is "on":

The error will be displayed and testing will resume at the failing address +2.


E
D
REF 58
REF 58
REF 58
REF 58
REF 58
IC1
IC7
IC13
IC19
IC25
IC31
IC37
LED1
IC2
IC8
IC14
IC20
IC26
IC32
IC38
IC3
IC9
IC15
IC21
IC27
IC33
IC39
Z1
IC4
IC10
LED2
IC16
LED3
IC22
IC28
IC34
IC40
LED4
IC5
IC11
IC17
IC23
IC29
IC35
IC41
IC6
IC12
IC18
IC24
Z2
IC30
R1
C3
IC36
R2
C2
IC42
TP1
P1190
R3
C4
C1
Z3
S1
S2
P1169
PAGE 4

INTERPOLATOR DIAGNOSTICS

Presently Interpolator Diagnostics allow the user to input MRD words directly to the interpolator to test different hardware functions. Future release will include the ability to test DMA transfers to the interpolator.

The program allows the input of a group of MRD words with the ability to repeat up to 2,147,483,647 times.

Input data format is in hex.

	MMMM	RRRR	DDDD
Ex1	00FF	0001	8000

controls velocity

Ex1 sets M word velocity to max
Ratio = 1
Distance =32768

To interpret MRD transfer results decimal display counters are connected to PIN 2(XMP+) and PIN 21 (YMP+) of the plotter connector. Make sure the interpolator is not halted before starting a MRD word transfer (halt led off).

ALL TESTING IS PERFORMED WITH THE TABLE DISCONNECTED*

RATIO EQUATION

For specific calculation of major and minor pluses use the following equation

$$\text{Ratio} = \frac{\text{Minor} \times 65536 - 32768}{\text{major}}$$

Ex. 2 If D word=10 what ratio needed to generate 1 minor pluse?

$$R = \frac{1 \times 65535 - 32768}{10}$$

R=3276.8 or 3277 = CCD hex

MAX VELOCITY

The interpolator diagnostic can be used to calculate maximum velocity. Results of this test are compared to know acceptable data for the table in question.

Ex3 For 41 Table - find velocity on axis
known Δt =154.2us @ stepsize (ss) Δx=.0005

THEN V41
calculated

$$= \frac{.0005 \text{ In/step}}{154.2 \times 10^{-6} \text{ sec/step}}$$

=3.24 in/sec

or

3.24 in/sec x 60 sec/min=194.4 in/min

SO INPUT

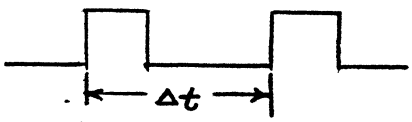
MMMM	RRRR	DDDD
00FF	XXXX	XXXX

max velocity

Repeat as necessary to read pluse width

MEASURE Δt WITH SCOPE

Ex.



150 us measured

measured Δt=7.5 div (20us/div)
=150us

THEN V41 =
measured

$$= \frac{.0005 \text{ in/step}}{150 \times 10^{-6} \text{ sec/step}}$$

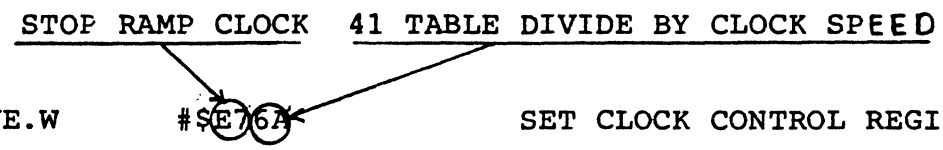
=3.33 in/sec or 3.33 in/sec x 60 sec/min=200 in/min

OR SIMPLY COMPARE Δt known Vs Δt meas. 154.2 us $\cong$ 150us

INTERPOLATOR TEST DATA

HEX M WORD	HEX R WORD	HEX D WORD	DEC. MAJOR PLUSES	DEC. MINOR PLUSES	
00FF	0001	8000	32768	1	
000F	0001	8000	32768	1	HO HUM
00FF	8000	0001	1	1	
00FF	00F9	8000	32768	125	
00FF	00FA	8000	32768	125	
00FF	00FB	8000	32768	126	
00FF	00FC	8000	32768	126	
00FF	8000	00FA	250	125	
00FF	FFFF	8000	32768	32768	
00FF	8000	FFFF	65535	32768	
00FF	OCCD	000A	10	1	
00FF	0002	4000	6384	1	
00FF	01F4	FFFF	65535	500	
00FF	0800	00FF	255	8	

DIRECT MOVE INTERPOLATOR




```
        MOVE.W      #$E76A          SET CLOCK CONTROL REGISTER
        MOVE.W      #Ø,$4CØØA       CLEAR DEVICE (CAUSES HALT)

        BSR         WAITBUFF        WAIT FOR BUFFER EMPTY
        MOVE.W      D1 $4CØØØ       M-WORD
        MOVE.W      D2 $4CØØ2       R-WORD
        MOVE.W      D3,$4CØØ4       D-WORD
SELF    BRAS        SELF
```

Routine to wait for plotter buffer to be empty. When outputing moves directly to the plotter(such as slew or initialization) we must make sure the previous command has been accepted by the plotter.

```
WAIT BUFF   MOVE.W   $4CØØØ,D7
            TST.B    D7            is buffer empty bit set.
            BPL.S    WAIT BUFF
```

FLOPPY TEST

COMMANDS:

RD	Read sector from floppy
WR	Write sector to floppy
FM	Format diskette
DA	Specify data to be written (see "DA" below)
UN	Specify unit number (Ø)
HD	Specify head (Ø or 1)
TR	Specify track number (Ø thru 79)
SC	Specify sector number (Ø thru 15)
CB	Compare read data buffer to data previously written
DT	Disc test-perform all functions of the controller
DE	Disc exerciser-read random sectors and verify contents (verifies data written during "DT")
DR	Read sector Ø on every track
IN	Perform on board internal processor diagnostics
RA	Perform on board RAM diagnostics
*	Specify number of repetitions
+	Increment last specified value after each operation
-	Decrement last specified value after each operation
EX	Exit floppy diagnostic

DATA FORMATS (DA) Specify data to be written

Ø	All Ø's
1	All 1's
2	0 thru 9 over and over
3	A thru Z over and over
4	Any character typed in by user
5	Text typed in by user
6	Track and sector numbers over and over

EXAMPLES:

Unit X Head X Track X Sector X

WRDAØ	Will write all Ø's
WRDA4T	Will write all T's
RDCB*3	Will read previously written data and compare buffers in memory 3 times.
RDUNØHD1TR44SC3	Will read sector 3 on track 44, unit Ø, head 1

```
          MOVE.B    NXT_SEC,D0          Get next sector pointer
          BRA       RETURN.I            Return
*
*          Error Table
*
ERR_TBL   DC.B      0                   00 - No error
          DC.B      -6                  01 - Undefined
          DC.B      -19                 02 - No seek complete signal
          DC.B      -6                  03 - Undefined
          DC.B      -19                 04 - Not ready (after select)
          DC.B      -6                  05 - Undefined
          DC.B      -19                 06 - Track 00 not found
          DC.B      -16                 07 - Door open on drive
          DC.B      -19                 08 - No head loaded
          DC.B      -6                  09 - Undefined
          DC.B      -6                  0A - Undefined
          DC.B      -6                  0B - Undefined
          DC.B      -6                  0C - Undefined
          DC.B      -6                  0D - Undefined
          DC.B      -6                  0E - Undefined
          DC.B      -6                  0F - Undefined
          DC.B      -14                 10 - ID CRC error
          DC.B      -14                 11 - Write fault
          DC.B      -5                  12 - Write protect
          DC.B      -6                  13 - Undefined
          DC.B      -14                 14 - Sector not found
          DC.B      -17                 15 - Seek error
          DC.B      -14                 16 - Format track timeout error
          DC.B      -14                 17 - Format track not complete
```



```
          DC.B      -6                  18 - Undefined
          DC.B      -17                 19 - Two sided error
          DC.B      -14                 1A - Wrong data mark found
          DC.B      -18                 1B - Transfer length error
          DC.B      -6                  1C - Undefined
          DC.B      -18                 1D - Lost data in FDC
          DC.B      -14                 1E - Data CRC error
          DC.B      -18                 1F - FDC busy error
          DC.B      -17                 20 - Invalid command
          DC.B      -17                 21 - Illegal disk address
          DC.B      -17                 22 - Configure data error
          DC.B      -17                 23 - Invalid interleave
          DC.B      -6                  24 - Undefined
          DC.B      -6                  25 - Undefined
          DC.B      -6                  26 - Undefined
          DC.B      -6                  27 - Undefined
          DC.B      -6                  28 - Undefined
          DC.B      -6                  29 - Undefined
          DC.B      -6                  2A - Undefined
          DC.B      -6                  2B - Undefined
          DC.B      -6                  2C - Undefined
          DC.B      -6                  2D - Undefined
          DC.B      -6                  2E - Undefined
          DC.B      -6                  2F - Undefined
          DC.B      -20                 30 - RAM error
          DC.B      -20                 31 - Program memory checksum error
```

Magnetic Tape Diagnostics

Quick Data Test

This test is entered by typing "3" on the main menu then "1" on the tape diagnostics submenu. This test will rewind the tape drive to the load point and then write a specified byte of data, a specified number of times per record, and a specified number of records in the file. The program will then rewind the drive and read the data back comparing it to what was written. The program provides for modifications of the default parameters which will not change unless the user specifically changes them or the user exits the Quick data test submenu to the magnetic tape test main menu. The default parameters and their ranges are:

Name	Default value	Range
Data byte	-127	-127 ... 127
#of bytes per record	256	0... 1024
# of records per file	20	0... 250

Typing a value outside these values will result in the default value.

The quick data test provides error messages and gives the operation which was being attempted when the error occurred.

Examples:

1) unit not on line
   error during rewind operation

2) No write ring detected error during mag tape write operation

3) Data read does not match data written error during data compare

If no errors occur the message will read:

No errors in quick data test

Using the quick data test is as simple as selecting "1" from the Quick data test submenu and pressing return. To alter the parameters type "2" at this level, the program will then prompt you for the data byte, followed by the prompt for the number of bytes per record, and finally the number of records per file. After entering the number of records the test will automatically run. Repeated rests using the same parameters can be run by typing "1" followed by a return. In each prompt the range of acceptable values is displayed. Typing a value outside the accepted range will cause the default value to be used.

Examples:

1) Quick data test

1. Run test with present parameter values
2. Alter parameter values

99. Exit

Please enter a menu choice 1

This will cause the program/test to run with the present parameter values. These will be the default values or the user inputed values.

2) Quick Data test

1. Run the test with present parameter values
2. Alter parameter values

99. Exit

Please enter a menu choice 2

Type in the data byte (-127...127)

-1

Type in the number of bytes per record (...1024)

1024

Type in the number of records in the file (....250)

200

This sequence will cause the system to write and verify 200 records of 1024 bytes each with each byte being equal to -1.

P-1206

RATIO (R WORD)

BD0-BD15

RATIO BUFFER

RATIO REGISTER

ADDER

16 BIT ACCUMULATOR

$\overline{LDR}$
$\overline{CLR}$

M13
$\overline{XFB}$

$\overline{XFBR}$

MOVF
MK
EOL

MJP

RDVF
MME

÷B

MAJOR PULSE SMOOTHER

MINOR PULSE SMOOTHER

DED

$\overline{SMJP}$

MJA

1A 1
2B
MOTOR PULSE MULTIPLEXER
1B
2A 2

ONE SHOT

XMP

ONE SHOT

YMP

SMNP

NMV

VELOCITY (M WORD)

BD0-BD7

LOWER M BUFFER

M0-M7

LOWER M REGISTER

ADDER

8 BIT ACCUMULATOR

$\overline{LDM}$
$\overline{CLR}$

BD8-BD15

UPPER M BUFFER

M8-M15

UPPER M REGISTER

XSIGN
YSIGN
MJA
PWD
MME
NMV

$\overline{LDM}$
$\overline{CLR}$

M14
$\overline{XFB}$

$\overline{XFBM}$

SEQUENCER

BLDW
DMAEN

$\overline{A1}$

M13

$\overline{CLR}$

D Q $\overline{Q}$

A1
A2

A
B
1 OF 4 DECODER
0 1 2 3

$\overline{LDM}$

$\overline{LDR}$

$\overline{LDD}$

1 OF 8 DECODER
0 1 2 3 4 5 6 7

$\overline{A2}$

BA1
BA2
BA3

A
B
C

CLOCK

HALT CLOCK REGISTER

CLOCK DIVIDER REGISTER

BD0-BD15

B-15

0-7

$\overline{LDC2}$

$\overline{CLR}$

UP COUNTER

UP COUNTER

÷2

16 MHZ CLOCK

MMK

HLT

UP/DOWN COUNTERS

BINARY RATE MULTIPLIER

MASTER CLOCK FLIP-FLOP

SK
$\overline{SK}$
MK
$\overline{MK}$

DISTANCE (D WORD)

BD0-BD15

DISTANCE BUFFER

$\overline{LDD}$
$\overline{CLR}$

DOWN COUNTERS

A
B
C
D

EOL

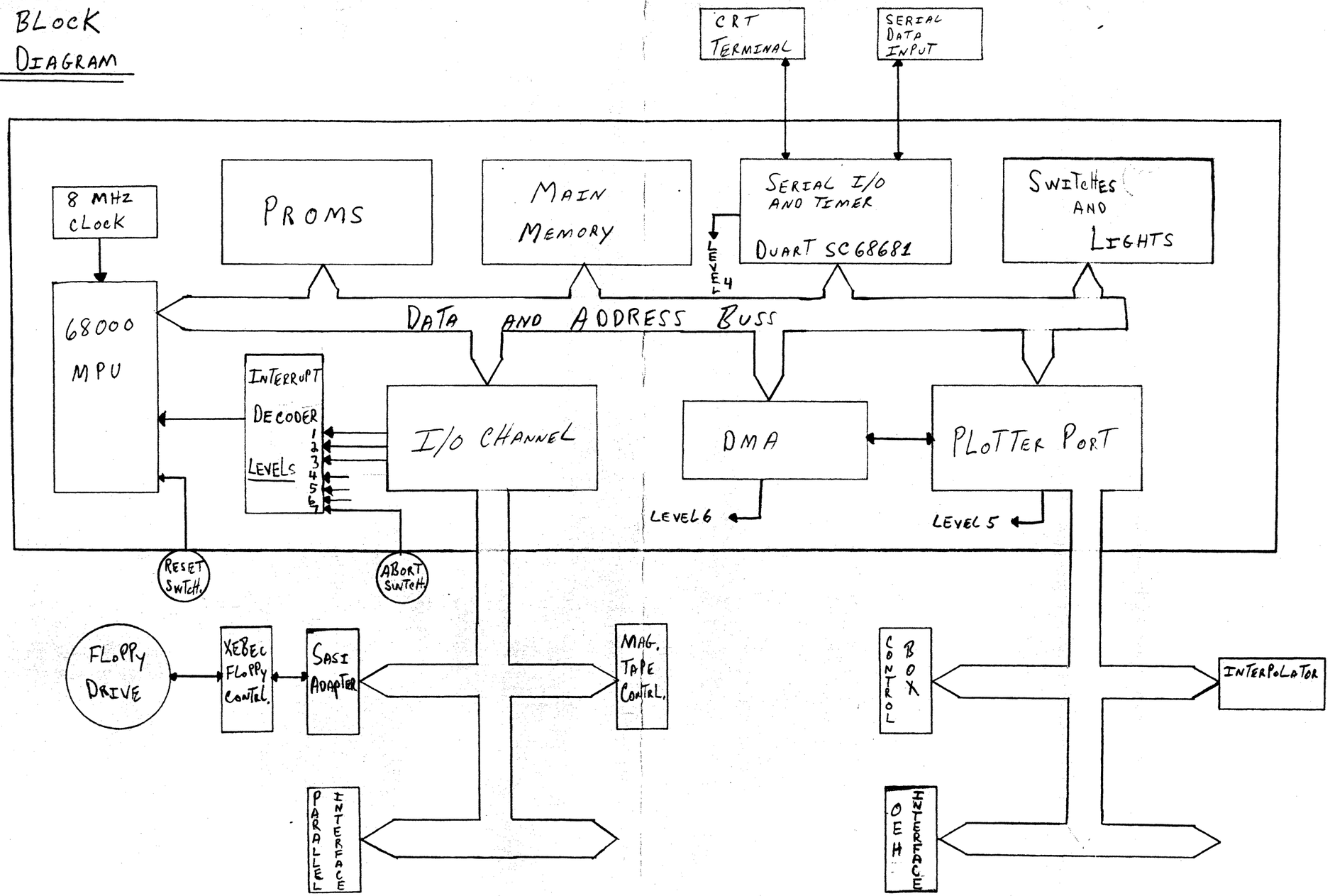
DED

$\overline{XFB}$
$\overline{CLR}$

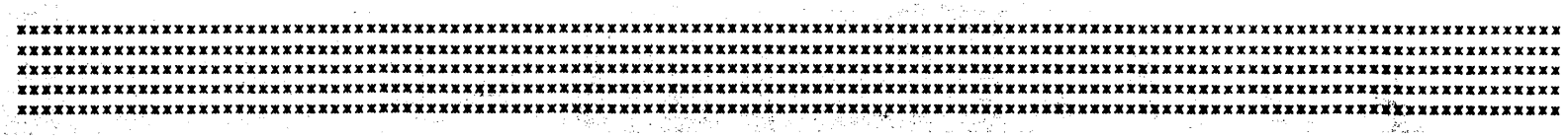
BD0-BD15 (BUFFERED DATA BUS)

RATIO BUFFER
DØ-D15
RATIO REGISTER
DØ-D15
ADDER
ACC.+ADDER OUTPUT
ACCUMULATOR 16 BIT
LDR
CLR
CLR
OVF
CLR
M13 (NOR WD)
XFB
XFBR
MJH (X=Ø, SEC. H)
+5V
SEC
1A
2B
MOTOR 1 PULSE
1B
2A
MUX 2
STROBE
ONE SHOT
XMP
ONE SHOT
YMP
MME (45°)
OVF
NMV (NO MOVE)
SMJP
XSIGN
YSIGN
SIGN REG.
XSIGN
YSIGN
SCLK
MJP
SMJP
BUS DØ-D15
DATA
DISTANCE BUFFER
DØ-D15
DOWN COUNTERS
MI
N/MA
X
EOL DETECT
EOL
LDD
CLR
XFB
CLR
SMJP
16 MHz
M/M HALT
HALT (COUNT DOWN)
HALT CLOCK REGISTER
8-15
UP
COUNTERS
CLOCK DIVIDER REGISTER
0-7
LDC2
CLR
÷2
COUNTER
BRM
Y
MASTER CLOCK F/F
SK
SK
MK
MK
MK
OVF
÷ 8
QA
QB
QC
EOL
SMJP
OVF
M BUFFER
0-7
M REGISTER
DØ-D7
ADDER
Acc.+ADDER OUTPUT
ACCUMULATOR 8 BIT
MK
LDM
CLR
CLR
XFBM
XFBM
XFB
M14 (TOOL SELECT)
XFBM
DATA BUS DØ-D15

3100 BLOCK DIAGRAM



```
JOB ID= 154809  USER  15  M JEAN
FILENAME= @EA2005F.M9
TASKNAME= RASM
SESSION=  01F2
DATE= 3/08/1984
TIME= 16:38:56
```



```
---------- NPC 3100 MAGTAPE DRIVER ----------

                        *
                                OPT     CRE
                        *
00000009                        SECTION 9
                        *
                                XDEF    MAGDVR,MAGINTRP
                                XDEF    MAGDVR1
                                XDEF    MAGDVR.I
                                XDEF    MAGDVR.T
                        *
                        *        XREF   SETTMR
                        *        XREF   STARTTMR
                        *        XREF   STOPTMR
                        *
                        ********************************************************************************
                        *                                                                              *
                        *                                                                              *
                        *        NAME:    MAGDVR                                                     *
                        *        ----                                                                 *
                        *                                                                              *
                        *        DATE:    05-06-83                                                   )
                        *        ----                                                                 *
                        *                                                                             )
                        *        AUTHOR:  M. JEAN                                                    )
                        *        ------                                                               *
                        *                                                                             )
                        *        REVISION: 1.2                                                       )
                        *        --------                                                             *
                        *                                                                             )
                        *        MAJ  07-15-83   Added software to handle the case of file           )
                        *        searching over blank tape and gracefully terminating the driver.     *
                        *                                                                             )
                        *        MAJ  07-25-83   Added time-out software.                            )
                        *                                                                             )
                        *                                                                             )
                        *        MIC  08-29-83  Added software to handle unvalid interrupts (MAGINTRP)*
                        *                                                                             )
                        *                                                                             )
                        *        MODULE DESCRIPTION:                                                 )
                        *        ------------------                                                   *
                        *                                                                             )
                        *                                                                             )
                        *    This module controls all I/O functions between a Motorola 68000          )
                        *    I/O channel and a CDC magnetic tape drive and formatter, inter-          )
                        *    faced by a Motorola mag tape adapter (MTA).                              *
                        *                                                                             )
                        *                                                                             )
                        *    +------------+                          +------------+                   )
                        *    |            |          +--------+      |            |                   )
                        *    |  68000     |--------->|        |--------->|    CDC     |              )
                        *    |   CPU      |          |  MTA   |      | MAG  TAPE  |                   )
                        *    |            |<---------|        |<---------|   DRIVE    |              )
                        *    |            |          +--------+      |            |                   )
                        *    +------------+                          +------------+                   )
                        *                                                                             )
                        *                                                                             )
                        *    The MTA will use the buffered transfer method.  The I/O channel          )
                        *    master can buffer up to 4k of data in the on-board FIFO before           *
```

```
                        *       transferring the data from I/O channel master to tape (Tape Write)
                        *       or vice versa (Tape Read).  This eliminates generating an inter-
                        *       rupt for every byte transfered and thus reducing overhead.
                        *
                        *       The MTA provides the following functions:
                        *           1. Read forward 1 record
                        *           2. Read reverse 1 record   (not used at this time)
                        *           3. Write forword 1 record
                        *           4. Write file mark
                        *           5. Search file mark forward
                        *           6. Search file mark reverse
                        *           7. Space forward 1 record
                        *           8. Space reverse 1 record
                        *           9. Rewind the Magnectic tape to the BOT
                        *          10. Error handling
                        *
                        *       Formatter commands are divided into three categories:
                        *           1. Group commands
                        *           2. Discrete commands
                        *           3. Transport/density select command
                        *
                        *       The interface lines of group commands are:
                        *           ------------------------------------------------------------
                        *           |   Formatter commands     |    D7  D6  D5  D4  D3  D2  D1  D0   |
                        *           |----------------------------------------------------------|
                        *           | Read forward record      |                0   0   0   0   0   |
                        *           | Read reverse record      |                0   0   0   0   1   |
                        *           | Write forward record     |                0   0   0   1   0   |
                        *           | Write file mark          |                0   0   1   1   0   |
                        *           | Search file mark forward |                1   0   1   0   0   |
                        *           | Search file mark reverse |                1   0   1   0   1   |
                        *           | Space forward record     |                1   0   0   0   0   |
                        *           | Space reverse record     |                1   0   0   0   1   |
                        *           ------------------------------------------------------------
                        *
```

```
 96                     *
 97                     *      Formatter discrete commands are:
 98                     *          ------------------------------------------------------------------
 99                     *          |       Command             |    D7  D6  D5  D4  D3  D2  D1  D0    |
100                     *          |------------------------------------------------------------------|
101                     *          | Rewind                    |                    0   0   0   1    |
102                     *          | Off line                  |                    0   0   1   0    |
103                     *          | Formatter disable         |                    0   1   0   0    |
104                     *          | MTA reset                 |                    1   0   0   0    |
105                     *          ------------------------------------------------------------------
106                     *
107                     *     The MTA status register consists of two, 8-bit registers STAT1 & STAT2:
108                     *     (The status register is a read only register)
109                     *          ------------------------------------------------------
110                     *          |    STAT1                  |    STAT2                 |
111                     *          |----------------------------------------------------|
112                     *          | Bit 0  :  READY           | Bit 0  :  ST1            |
113                     *          | Bit 1  :  ON LINE         | Bit 1  :  ST2            |
114                     *          | Bit 2  :  LDPNT           | Bit 2  :  ST3            |
115                     *          | Bit 3  :  EOT             | Bit 3  :  ST4            |
116                     *          | Bit 4  :  FNBSY           | Bit 4  :  FIFUL          |
117                     *          | Bit 5  :  RWNDG           | Bit 5  :  FIHAL          |
118                     *          | Bit 6  :  FLPRT           | Bit 6  :  FIEMP          |
119                     *          | Bit 7  :  NRZI            | Bit 7  :  DNBSY          |
120                     *          ------------------------------------------------------
121                     *
122                     *         NOTE: ST1-ST4 cause interrupts when true.  These bits
123                     *               are reset when either STAT1 or STAT2 are read.
124                     *               Because they are reset STAT2 should always be
125                     *               read in first.
126                     *
127                     *         ST1 = LP, EOT, CCG
128                     *         ST2 = FM, FNBSY
129                     *         ST3 = DURING DMA 1/2 FULL READ OR WRITE
130                     *         ST4 = TAPE WRITE ERROR OR READ AFTER WRITE
131                     *
132                     *
133                     *
134                     *          Offsets of I/O addresses as follows:
135                     *          ------------------------------------------------------
136                     *          |        DATIO        |       1(40000)                 |
137                     *          |        GRCMD        |       8(40000)                 |
138                     *          |        DSCMD        |      16(40000)                 |
139                     *          |        TRSEL        |      24(40000)                 |
140                     *          |        STAT1        |      33(40000)                 |
141                     *          |        STAT2        |      41(40000)                 |
142                     *          |        LSWRD        |      48(40000)                 |
143                     *          |        DIRSL        |      56(40000)                 |
144                     *          ------------------------------------------------------
145                     *
```

```
147                             *       CALLING SEQUENCE
148                             *       ------- --------
149                             *
150                             *       The calling sequence of this driver will be as follows:
151                             *
152                             *           MAGDVR(COMMAND,STATUS,BUFFER)
153                             *                  ^       ^      ^
154                             *                  |       |      |------ call by name  (long word)
155                             *                  |       |------------- call by name  (long word)
156                             *                  |--------------------- call by value (long word)
157                             *
158                             *       An alternate entry point (MAGDVR1) is provided to allow the
159                             *       PASCAL programs to use the driver for two different data types
160                             *       (e.g. character or numeric).
161                             *
162                             *
163                             *       COMMAND
164                             *       -------
165                             *
166                             *       The COMMAND parameter is a long word value.  The command
167                             *       code byte is contained in bits 0-7.
168                             *
169                             *        31                    24 23                    16
170                             *       ------------------------------------------------
171                             *       |                       |                      |
172                             *       ------------------------------------------------
173                             *       |                       |    COMMAND  BYTE     |
174                             *       ------------------------------------------------
175                             *        15                    8 7                     0
176                             *
177                             *       The following is a list of the command codes and operations
178                             *       this driver recognizes:
179                             *
180                             *            0 -> INITIALIZE
181                             *            1 -> READ DATA
182                             *            2 -> WRITE DATA
183                             *            3 -> REWIND
184                             *            4 -> FILE SEARCH
185                             *            5 -> WRITE FILE MARK
186                             *            6 -> SKIP RECORDS
187                             *
188                             *
189                             *
190                             *       STATUS
191                             *       ------
192                             *
193                             *       The STATUS parameter is returned by the driver and indicates
194                             *       what happened or is happening with the driver.  When an
195                             *       operation is started STATUS will be set busy (-7) and no
196                             *       further commands will be processed until the operation
197                             *       completes and goes not busy.  STATUS will be returned as
198                             *       follows:
199                             *
200                             *            0 -> NO ERROR (READY)
201                             *           -1 -> EOF
202                             *           -2 -> BOT ERROR
203                             *           -3 -> EOT ERROR
204                             *           -4 -> NOT ON LINE
```

```
205                             *             -5 -> WRITE PROTECTED (NO WRITE RING)
206                             *             -6 -> HARD ERROR
207                             *             -7 -> COMMAND REJECT (BUSY)
208                             *             -8 -> RECORD SIZE EXCEEDS BUFFER CAPACITY
209                             *            -10 -> TIME OUT
210                             *            -20 -> BLANK TAPE ERROR
211                             *
212                             *
213                             *         BUFFER
214                             *         ------
215                             *
216                             *         This is a long word address of the I/O buffer to be used
217                             *         for data transfers with the mag tape.  The first word in the
218                             *         buffer is used as a character count.  When doing a READ
219                             *         operation it indicates the maximum number of characters to
220                             *         be placed in the buffer.  When doing a WRITE operation it
221                             *         indicates tha actual number of characters in the buffer to
222                             *         be written.  The character count will be updated when the
223                             *         operation being performed is complete to reflect the actual
224                             *         number of characters transfered.  The following is a repre-
225                             *         sentation of the buffer:
226                             *
227                             *         |    word 1     |     word  2     |     word  3     |
228                             *         -------------------------------------------------------
229                             *         | buffer count  | char 1 | char 2 | char 3 | char 4 | etc...
230                             *         -------------------------------------------------------
231                             *
232                             *         When file searching or skipping records the buffer count
233                             *         is used to indicate what file to search for or how many
234                             *         records to skip.  When file searching the actual file number
235                             *         desired is passed.  Load point is defined as file number 1.
236                             *         When skipping records the buffer count contains the relative
237                             *         number of records to skip from the present position of the
238                             *         tape.  Positive numbers indicate forward record skipping and
239                             *         negative numbers are used to indicate reverse record skipping.
240                             *
241                             *
242                             *
```

```
244                             *
245                             *   INTERNAL OPERATIONS
246                             *
247                             *           GROUP COMMANDS
248                             *
249             00000000        READFWD  EQU      0
250             00000001        READREV  EQU      1
251             00000002        WRTFWD   EQU      2
252             00000006        WRTEOF   EQU      6
253             00000014        FILEFWD  EQU      20
254             00000015        FILEREV  EQU      21
255             00000010        SKPFWD   EQU      16
256             00000011        SKPREV   EQU      17
257                             *
258                             *           DISCRETE COMMANDS
259                             *
260             00000001        REWND    EQU      1
261             00000002        OFFLINE  EQU      2
262             00000004        FMTDSBL  EQU      4
263             00000008        MTA_RST  EQU      8
264                             *
265                             *
266                             *           I/O ADDRESS OFFSETS
267                             *
268             00000001        DATIO    EQU      1
269             00000008        GRCMD    EQU      8
270             00000010        DSCMD    EQU      16
271             00000018        TRSEL    EQU      24
272             00000021        STAT1    EQU      33
273             00000029        STAT2    EQU      41
274             00000030        LSWRD    EQU      48
275             00000038        DIRSL    EQU      56
276                             *
277                             *           REGISTER SAVE DEFINITION
278                             *
279 9 00000000 00               MAXINVAL DC.B     0                    Invalid interrupt count
280 9 00000001 00               MAGINTRP DC.B     0                    Valid interrupt flag
281                             REGLIST  REG      D0-D5/A0-A4          Register save list
282                             *
283 9 00000002 0000002C         REGSAVE  DS.L     11                   Register save area
284                             *
```

```
 286                            ********************************************************************************
 287                            *                                                                              *
 288                            *                                                                              *
 289                            *         MAG TAPE DRIVER ENTRY POINT BY SUBROUTINE CALL                       *
 290                            *                                                                              *
 291                            *                                                                              *
 292                            ********************************************************************************
 293                            *
 294 9          0000002E        MAGDVR   EQU      *                    Driver entry point
 295 9          0000002E        MAGDVR1  EQU      *                    Alternate driver entry point
 296                            *
 297 9 0000002E 40F9000005D8            MOVE.W   SR,PSWSAVE           Save current proc. status word
 298 9 00000034 46FC2400                MOVE.W   #$2400,SR            Turn off interrupts (to level 4)
 299 9 00000038 48F91F3F0000            MOVEM.L  REGLIST,REGSAVE      Save registers
                0002
 300                            *
 301 9 00000040 285F                    MOVE.L   (SP)+,A4             Save the return address
 302 9 00000042 245F                    MOVE.L   (SP)+,A2             Get address of I/O buffer
 303 9 00000044 225F                    MOVE.L   (SP)+,A1             Get address of status parameter
 304 9 00000046 201F                    MOVE.L   (SP)+,D0             Get the command
 305                            *
 306 9 00000048 23CA000005E0            MOVE.L   A2,BUF_ADR           Save buffer address
 307 9 0000004E 23C9000005DC            MOVE.L   A1,STAT_ADR          Save status address
 308 9 00000054 2079000005D4            MOVE.L   MAG_IO,A0            Get base address for MAG I/O
 309                            *
 310                            *         IF NOT ON-LINE - RETURN
 311                            *
 312 9 0000005A 082800010021            BTST.B   #1,STAT1(A0)         Are we on-line?
 313 9 00000060 6608                    BNE.S    CMND.1               Yes, OK to proceed
 314 9 00000062 22BCFFFFFFFC            MOVE.L   #-4,(A1)             No, STATUS = on-line error
 315 9 00000068 6048                    BRA.S    RETURN.S             Return
 316                            *
 317                            *         IF NOT READY - REJECT COMMAND (READY OR FNBSY = 0)
 318                            *
 319 9 0000006A 082800000021 CMND.1     BTST     #0,STAT1(A0)         Is unit ready?
 320 9 00000070 6708                    BEQ.S    CMND.2               No, unit is busy
 321 9 00000072 082800040021            BTST     #4,STAT1(A0)         Is formatter busy?
 322 9 00000078 6608                    BNE.S    CMND.3               No, OK to continue
 323 9 0000007A 22BCFFFFFFF9 CMND.2     MOVE.L   #-7,(A1)             STATUS = busy
 324 9 00000080 6030                    BRA.S    RETURN.S             Return
 325                            *
 326                            *         SET UP TIME OUT TIMER
 327                            *
 328                            *         MOVE.W #10,-(SP)          Set time out value to 5 seconds
 329                            *         PEA.L  MAGDVR.T           Time out handler address
 330                            *         JSR    SETTMR             Go program timer
 331                            *
 332                            *         DETERMINE COMMAND
 333                            *
 334 9 00000082 E500            CMND.3   ASL.B    #2,D0               Adjust for long word addressing
 335 9 00000084 13C0000005DB            MOVE.B   D0,COMMAND           Save adjusted command
 336 9 0000008A 47F900000096            LEA.L    CMNDS.S,A3           Get jump table address
 337 9 00000090 D6C0                    ADDA     D0,A3                Add index
 338 9 00000092 2653                    MOVE.L   (A3),A3              Get address of command instructions
 339 9 00000094 4ED3                    JMP.L    (A3)                 Go do command
 340                            *
 341                            *         TABLE OF ENTRY POINT ADDRESSES
 342                            *
```

```
 343 9 00000096 00000164      CMNDS.S  DC.L      INIT.S
 344 9 0000009A 0000017E               DC.L      READ.S
 345 9 0000009E 00000262               DC.L      WRITE.S
 346 9 000000A2 00000300               DC.L      REWIND.S
 347 9 000000A6 0000034C               DC.L      SEARCH.S
 348 9 000000AA 0000048A               DC.L      FILMRK.S
 349 9 000000AE 000004E8               DC.L      SKPREC.S
 350                           *
 351                           *          RESTORE REGISTERS & RETURN
 352                           *
 353 9 000000B2 2F0C          RETURN.S MOVE.L    A4,-(SP)              Put return address on stack
 354 9 000000B4 4CFA1F3FFF4A           MOVEM.L   REGSAVE,REGLIST       Restore registers
 355 9 000000BA 46F9000005D8           MOVE.W    PSWSAVE,SR            Restore status word & interrupts
 356 9 000000C0 4E75                   RTS                             Return
 357                           *
```

```
 359                            ********************************************************************************
 360                            *                                                                              *
 361                            *                                                                              *
 362                            *          MAG TAPE DRIVER ENTRY POINT BY INTERRUPT                            *
 363                            *                                                                              *
 364                            *                                                                              *
 365                            ********************************************************************************
 366 9           000000C2       MAGDVR.I EQU      *                       Interrupt service routine
 367                            *
 368 9 000000C2 48F91F3F0000            MOVEM.L   REGLIST,REGSAVE         Save registers
                0002
 369 9 000000CA 2079000005D4            MOVE.L    MAG_IO,A0               Get base address of MAG I/O
 370 9 000000D0 2479000005E0            MOVE.L    BUF_ADR,A2              Get buffer address
 371 9 000000D6 2279000005DC            MOVE.L    STAT_ADR,A1             Get external status address
 372 9 000000DC 4280                    CLR.L     D0
 373 9 000000DE 4281                    CLR.L     D1
 374 9 000000E0 10280029                MOVE.B    STAT2(A0),D0            Get status byte 2
 375 9 000000E4 12280021                MOVE.B    STAT1(A0),D1            Get status byte 1
 376 9 000000E8 0C3900010000            CMPI.B    #1,MAGINTRP             Valid interrupt ?
                0001
 377 9 000000F0 672A                    BEQ.S     CMND.4                  YES, continue processing.
 378 9 000000F2 08010002                BTST.L    #2,D1                   Are we at load point?
 379 9 000000F6 661A                    BNE.S     VI
 380 9 000000F8 08000003                BTST.L    #3,D0                   Hard error?
 381 9 000000FC 675E                    BEQ.S     RETN1.I                 NO, return
 382 9 000000FE 22BCFFFFFFFA            MOVE.L    #-6,(A1)                YES, store error status
 383 9 00000104 317C00080010            MOVE.W    #MTA_RST,DSCMD(A0)      Reset MTA
 384 9 0000010A 117C00040010            MOVE.B    #FMTDSBL,DSCMD(A0)      Reset formattor 0
 385 9 00000110 6044                    BRA.S     RETURN.I                Return from interrupt
 386 9 00000112 33FC00010000 VI         MOVE.W    #1,ACTFILE              YES, reset current file value.
                05E6
 387 9 0000011A 603A                    BRA.S     RETURN.I                Return from interrupt.
 388                            *
 389                            *          IF BUSY FLAG IS NOT SET - IGNORE INTERRUPT
 390                            *
 391 9 0000011C 4283            CMND.4  CLR.L     D3                      Busy?
 392 9 0000011E 1600                    MOVE.B    D0,D3
 393 9 00000120 0203004B                AND.B     #$4B,D3
 394 9 00000124 6736                    BEQ.S     RETN1.I                 No, ignore interrupt
 395                            *
 396                            *          GO BACK TO WHAT WE WERE WORKING ON
 397                            *
 398 9 00000126 4282            CMND.5  CLR.L     D2
 399 9 00000128 1439000005DB            MOVE.B    COMMAND,D2              Get the adjusted command
 400 9 0000012E 47F90000013A            LEA.L     CMNDS.I,A3              Get the jump table address
 401 9 00000134 D6C2                    ADDA      D2,A3                   Add the offset
 402 9 00000136 2653                    MOVE.L    (A3),A3                 Get the address of command instructions
 403 9 00000138 4ED3                    JMP.L     (A3)                    Go do it
 404                            *
 405 9 0000013A 0000017A        CMNDS.I DC.L      INIT.I
 406 9 0000013E 000001BC                DC.L      READ.I
 407 9 00000142 000002CA                DC.L      WRITE.I
 408 9 00000146 0000032E                DC.L      REWIND.I
 409 9 0000014A 000003D4                DC.L      SEARCH.I
 410 9 0000014E 000004B4                DC.L      FILMRK.I
 411 9 00000152 0000054C                DC.L      SKPREC.I
 412                            *
 413                            *          RESTORE REGISTERS AND RETURN FROM EXCEPTION.
```

```
 414                            *            RETURN POINT RETN1.I IS USED FOR COMMANDS SUCH AS
 415                            *            FILE SEARCH THAT REQUIRE MULTIPLE INTERRUPTS.  FOR
 416                            *            THESE, THE VALID INTERRUPT FLAG WILL STAY SET UNTIL
 417                            *            THE COMPLETION OF THE LAST COMMAND.
 418                            *
 419 9 00000156 423900000001 RETURN.I CLR.B    MAGINTRF              Clear valid interrupt flag
 420 9 0000015C 4CFA1F3FFEA2 RETN1.I  MOVEM.L  REGSAVE,REGLIST       Restore the registers
 421 9 00000162 4E73                  RTE                           Return from exception
```

```
 423                            *
 424                            ********************************************************************************
 425                            *         INITIALIZE MAG TAPE UNIT  (SUBROUTINE)                               *
 426                            ********************************************************************************
 427                            *
 428 9           00000164       INIT.S   EQU      *
 429 9 00000164 317C00080010             MOVE.W   #MTA_RST,DSCMD(A0)     Reset MTA
 430 9 0000016A 117C00000018             MOVE.B   #0,TRSEL(A0)           Select formatter 0
 431 9 00000170 117C00040010             MOVE.B   #FMTDSBL,DSCMD(A0)     Reset formatter 0
 432 9 00000176 60000188                 BRA      REWIND.S               Go rewind unit
 433                            *
 434                            ********************************************************************************
 435                            *         INITIALIZE MAG TAPE UNIT  (INTERRUPT)                                *
 436                            ********************************************************************************
 437                            *
 438 9 0000017A 600001B2        INIT.I   BRA      REWIND.I               Do rewind completion
 439
```

```
                                ********************************************************************************
                                *          READ A RECORD              (SUBROUTINE)                             *
                                ********************************************************************************
                                *
0000017E      READ.S    EQU     *
0000017E 317C00080010           MOVE.W  #MTA_RST,DSCMD(A0)      Reset FIFO to empty
00000184 13FC000A0000           MOVE.B  #10,RETRY               Initialize retry counter
         05DA
0000018C 33D2000005E4           MOVE.W  (A2),BUF_CNT            Save buffer length
00000192 082800030021           BTST.B  #3,STAT1(A0)            Is it EOT?
00000198 670A                   BEQ.S   READ.S1                 No, continue
0000019A 22BCFFFFFFFD           MOVE.L  #-3,(A1)                Yes, STATUS = EOT error
000001A0 6000FF10               BRA     RETURN.S                Return
                                *
000001A4 117C00000008 READ.S1   MOVE.B  #READFWD,GRCMD(A0)      Issue read record command
000001AA 22BCFFFFFFF9           MOVE.L  #-7,(A1)                STATUS = busy
000001B0 13FC00010000           MOVE.B  #1,MAGINTRP             Set valid interrupt flag
         0001
000001B8 6000FEF8               BRA     RETURN.S                Return
                                *
                                ********************************************************************************
                                *          READ A RECORD              (INTERRUPT)                              *
                                ********************************************************************************
                                *
000001BC      READ.I    EQU     *
000001BC 08000000               BTST    #0,D0                   Is it MR, 1600 PE interrupt?
000001C0 669A                   BNE.S   RETN1.I                 YES, ignore him and go home.
000001C2 08000001               BTST    #1,D0                   Is it a FNBSY interrupt?
000001C6 6754                   BEQ.S   READ.I3                 No, must be error
000001C8 4282                   CLR.L   D2                      Clear byte counter
000001CA 08010004               BTST    #4,D1                   EOF?
000001CE 661A                   BNE.S   READ.I1                 No, get data
000001D0 22BCFFFFFFFF           MOVE.L  #-1,(A1)                STATUS = EOF
000001D6 5279000005E6           ADDQ.W  #1,ACTFILE              Increment file counter
000001DC 4252                   CLR.W   (A2)                    Return with length = 0 bytes
000001DE 082800040021 A         BTST    #4,STAT1(A0)            Wait for FNBSY bit
000001E4 67F8                   BEQ.S   A                       No, try again
000001E6 6000FF6E               BRA     RETURN.I                Return
                                *
000001EA 082800060029 READ.I1   BTST    #6,STAT2(A0)            Is FIFO empty?
000001F0 6622                   BNE.S   READ.I2                 Yes, all done reading
                                *        MOVE.B DATIO(A0),2(A2,D2) Get byte of data
                                
                                
000001F2 16280001               MOVE.B  DATIO(A0),D3            Get byte of data
000001F6 4EB90000025E           JSR     ROTCHAR                 Rotate character
000001FC 15842002               MOVE.B  D4,2(A2,D2)             Put byte in buffer
                                
                                
00000200 5242                   ADD.W   #1,D2                   Increment byte count
00000202 B479000005E4           CMP.W   BUF_CNT,D2              Reached max record size yet?
00000208 6DE0                   BLT.S   READ.I1                 Get more data
0000020A 22BCFFFFFFF8           MOVE.L  #-8,(A1)                Yes, STATUS = buffer overflow error
00000210 6000FF44               BRA     RETURN.I                Return
                                *
00000214 3482           READ.I2 MOVE.W  D2,(A2)                 Return # of bytes read
00000216 4291                   CLR.L   (A1)                    STATUS = ready
00000218 6000FF3C               BRA     RETURN.I                Return
```

```
 497                            *
 498 9 0000021C 08000003       READ.I3  BTST     #3,D0                 Is it a hard error?
 499 9 00000220 6732                    BEQ.S    READ.I6               No, different error
 500 9 00000222 082800040021    READ.I7  BTST     #4,STAT1(A0)          Wait for read to complete
 501 9 00000228 67F8                    BEQ.S    READ.I7               Not yet - try again
 502 9 0000022A 117C00110008            MOVE.B   #SKPREV,GRCMD(A0)     Backspace record
 503 9 00000230 5339000005DA            SUB.B    #1,RETRY              Decrement retry counter
 504 9 00000236 6F12                    BLE.S    READ.I5               If no more retries - hard error
 505 9 00000238 082800040021    READ.I4  BTST.B   #4,STAT1(A0)          Wait for reverse record to complete
 506 9 0000023E 67F8                    BEQ.S    READ.I4               Not yet - try again
 507 9 00000240 117C00000008            MOVE.B   #READFWD,GRCMD(A0)    Issue read command again
 508 9 00000246 6000FF14                BRA      RETN1,I               Return
 509                            *
 510 9 0000024A 22BCFFFFFFFA    READ.I5  MOVE.L   #-6,(A1)             STATUS = hard error
 511 9 00000250 6000FF04                BRA      RETURN,I              Return
 512                            *
 513 9 00000254 22BCFFFFFFFD    READ.I6  MOVE.L   #-3,(A1)             STATUS = EOT erorr
 514 9 0000025A 6000FEFA                BRA      RETURN,I              Return
 515                            *                                                                  *
 516                            ************************************************************************************
 517                            *                                                                  *
 518 9 0000025E 1803            ROTCHAR  MOVE.B   D3,D4
 519 9 00000260 4E75                    RTS
 520                            *                                                                  *
 521                            ************************************************************************************
 522                            *                                                                  *
 523
```

```
 525                            ********************************************************************************
 526                            *          WRITE A RECORD              (SUBROUTINE)                        *
 527                            ********************************************************************************
 528                            *
 529 9           00000262       WRITE.S  EQU      *
 530 9 00000262 317C00080010            MOVE.W   #MTA_RST,DSCMD(A0)     Reset FIFO to empty
 531 9 00000268 33D2000005E4            MOVE.W   (A2),BUF_CNT           Save buffer length
 532 9 0000026E 082800060021            BTST.B   #6,STAT1(A0)           Write enabled?
 533 9 00000274 670A                    BEQ.S    WRITE.S1               Yes, continue
 534 9 00000276 22BCFFFFFFFB            MOVE.L   #-5,(A1)               No, STATUS = write protect error
 535 9 0000027C 6000FE34                BRA      RETURN.S               Return
 536                            *
 537 9 00000280 082800030021 WRITE.S1 BTST.B   #3,STAT1(A0)           Is it EOT?
 538 9 00000286 670A                    BEQ.S    WRITE.S2               No, continue
 539 9 00000288 22BCFFFFFFFD            MOVE.L   #-3,(A1)               Yes, STATUS = EOT error
 540 9 0000028E 6000FE22                BRA      RETURN.S               Return
 541                            *
 542 9 00000292 7401            WRITE.S2 MOVE.L   #1,D2                 Clear byte counter
 543 9 00000294 B479000005E4    WRITE.S4 CMP.W    BUF_CNT,D2            All done?
 544 9 0000029A 6E10                    BGT.S    WRITE.S3               Yes, go issue last word command
 545
 546
 547                            *         MOVE.B 1(A2,D2),DATIO(A0) Get byte from buffer and output it
 548
 549 9 0000029C 16322001                MOVE.B   1(A2,D2),D3            Get byte from buffer
 550 9 000002A0 4EBAFFBC                JSR      ROTCHAR                Reverse bits
 551 9 000002A4 11440001                MOVE.B   D4,DATIO(A0)           Move data into FIFO
 552
 553
 554
 555 9 000002A8 5242                    ADD.W    #1,D2                  Increment byte counter
 556 9 000002AA 60E8                    BRA      WRITE.S4               Do it again
 557                            *
 558 9 000002AC 317C00000030 WRITE.S3 MOVE.W   #0,LSWRD(A0)           Send last word to MTA
 559 9 000002B2 117C00020008            MOVE.B   #WRTFWD,GRCMD(A0)      Issue write record command
 560 9 000002B8 22BCFFFFFFF9            MOVE.L   #-7,(A1)               STATUS = busy
 561 9 000002BE 13FC00010000            MOVE.B   #1,MAGINTRP            Set valid interrupt flag
                0001
 562 9 000002C6 6000FDEA                BRA      RETURN.S               Return
 563                            *
 564                            ********************************************************************************
 565                            *          WRITE RECORD                (INTERRUPT)                         *
 566                            ********************************************************************************
 567                            *
 568 9           000002CA       WRITE.I  EQU      *
 569 9 000002CA 08000001                BTST     #1,D0                  Is it a FNBSY interrupt?
 570 9 000002CE 660A                    BNE.S    WRITE.I3               Yes, continue
 571 9 000002D0 08000006                BTST     #6,D0                  No, is it a FIFO empty interrupt?
 572 9 000002D4 6600FE86                BNE      RETN1.I                Yes, ignore (wait for FNBSY)
 573 9 000002D8 600C                    BRA.S    WRITE.I1               No, must be error
 574 9 000002DA 34B9000005E4 WRITE.I3 MOVE.W   BUF_CNT,(A2)            Yes, return # of bytes written
 575 9 000002E0 4291                    CLR.L    (A1)                   STATUS = ready
 576 9 000002E2 6000FE72                BRA      RETURN.I               Return
 577                            *
 578 9 000002E6 08000003        WRITE.I1 BTST     #3,D0                  Is it a hard error?
 579 9 000002EA 670A                    BEQ.S    WRITE.I2               No, different error
 580 9 000002EC 22BCFFFFFFFA            MOVE.L   #-6,(A1)               Yes, STATUS = hard error
 581 9 000002F2 6000FE62                BRA      RETURN.I               Return
```

---------- NPC 3100 MAGTAPE DRIVER ----------

```
 582                            *
 583 9 000002F6 22BCFFFFFFFD WRITE.I2 MOVE.L   #-3,(A1)            STATUS = EOT error
 584 9 000002FC 6000FE58              BRA      RETURN.I            Return
 585                            *
```

```
 587                            ********************************************************************************
 588                            *          REWIND                    (SUBROUTINE)                             *
 589                            ********************************************************************************
 590                            *
 591 9           00000300       REWIND.S EQU      *
 592 9 00000300 0828000020021            BTST.B   #2,STAT1(A0)          Are we at load point?
 593 9 00000306 6618                     BNE.S    REWND.S1              Yes, go return
 594 9 00000308 317C00010010             MOVE.W   #1,DSCMD(A0)          Issue rewind command
 595 9 0000030E 22BCFFFFFFF9             MOVE.L   #-7,(A1)              STATUS = busy
 596 9 00000314 13FC00010000             MOVE.B   #1,MAGINTRP           Set valid interrupt flag
                0001
 597 9 0000031C 6000FD94                 BRA      RETURN.S              Return
 598                            *
 599 9 00000320 4291            REWND.S1 CLR.L    (A1)                  STATUS = ready
 600 9 00000322 33FC00010000             MOVE.W   #1,ACTFILE            Reset file counter (load point = file 1)
                05E6
 601 9 0000032A 6000FD86                 BRA      RETURN.S              Return
 602                            *
 603                            ********************************************************************************
 604                            *          REWIND                    (INTERRUPT)                              *
 605                            ********************************************************************************
 606                            *
 607 9           0000032E       REWIND.I EQU      *
 608 9 0000032E 08010002                 BTST     #2,D1                 Check for load point
 609 9 00000332 660A                     BNE.S    REWND.I1              Yes, continue
 610 9 00000334 22BCFFFFFFFE             MOVE.L   #-2,(A1)              No, STATUS = BOT error
 611 9 0000033A 6000FE1A                 BRA      RETURN.I              Return
 612                            *
 613 9 0000033E 4291            REWND.I1 CLR.L    (A1)                  STATUS = ready
 614 9 00000340 33FC00010000             MOVE.W   #1,ACTFILE            Reset file counter (load point = file 1)
                05E6
 615 9 00000348 6000FE0C                 BRA      RETURN.I              Return
 616                            *
```

```
                                ********************************************************************************
                                *         FILE SEARCH                (SUBROUTINE)                              *
                                ********************************************************************************
                                *
0000034C                SEARCH.S EQU     *
0000034C 3212                    MOVE.W  (A2),D1             Get requested file #
0000034E 0C410001                CMP.W   #1,D1               Is it file #1?
00000352 6E0A                    BGT.S   SERCH.S7            No, continue
00000354 13FC000C0000            MOVE.B  #12,COMMAND         Yes, set adjusted command to rewind
         05DB
0000035C 60A2                    BRA     REWIND.S            Go do rewind
                                *
0000035E 33C1000005E8   SERCH.S7 MOVE.W  D1,REQFILE          Save for later
00000364 9279000005E6            SUB.W   ACTFILE,D1          Determine number of files to search and
0000036A 33C1000005EC            MOVE.W  D1,FILECNT          direction to search
00000370 6F32                    BLE.S   SERCH.S2            Go search backwards
                                *
                                *         SEARCH FORWARD
                                *
00000372 082800030021            BTST.B  #3,STAT1(A0)        Is it EOT?
00000378 670A                    BEQ.S   SERCH.S1            No, continue
0000037A 22BCFFFFFFFD            MOVE.L  #-3,(A1)            Yes, STATUS = EOT error
00000380 6000FD30                BRA     RETURN.S            Return
                                *
00000384 117C00140008   SERCH.S1 MOVE.B  #FILEFWD,GRCMD(A0)  Issue forward file command
0000038A 13FC00010000            MOVE.B  #1,DIR_FLG          Set direction flag to forward
         05EA
00000392 22BCFFFFFFF9            MOVE.L  #-7,(A1)            STATUS = busy
00000398 13FC00010000            MOVE.B  #1,MAGINTRP         Set valid interrupt flag
         0001
000003A0 6000FD10                BRA     RETURN.S            Return
                                *
                                *         SEARCH BACKWARDS
                                *
000003A4 082800020021   SERCH.S2 BTST.B  #2,STAT1(A0)        At load point?
000003AA 670A                    BEQ.S   SERCH.S3            No, continue
000003AC 22BCFFFFFFFE            MOVE.L  #-2,(A1)            Yes, STATUS = BOT error
000003B2 6000FCFE                BRA     RETURN.S
                                *
000003B6 117C00150008   SERCH.S3 MOVE.B  #FILEREV,GRCMD(A0)  Issue reverse file command
000003BC 4239000005EA            CLR.B   DIR_FLG             Set direction flag to reverse
000003C2 22BCFFFFFFF9            MOVE.L  #-7,(A1)            STATUS = busy
000003C8 13FC00010000            MOVE.B  #1,MAGINTRP         Set valid interrupt flag
         0001
000003D0 6000FCE0                BRA     RETURN.S            Return
                                *
                                ********************************************************************************
                                *         FILE SEARCH                (INTERRUPT)                               *
                                ********************************************************************************
                                *
000003D4                SEARCH.I EQU     *
000003D4 08000000                BTST    #0,D0               Is it 1600 bpi (PE)?
000003D8 6600FD82                BNE     RETN1.I             YES, then just ignore 1st interrupt
000003DC 08010004                BTST    #4,D1               Is FNBSY set?
000003E0 66000094                BNE     SRCH.I10            Check for blank tape error
000003E4 5379000005EC            SUBQ.W  #1,FILECNT          Decrement file search
000003EA 4A39000005EA            TST.B   DIR_FLG             Determine direction
000003F0 673C                    BEQ.S   SERCH.I5            Search backwards
```

```
 672                            *
 673                            *          SEARCH FORWARD
 674                            *
 675 9 000003F2 08010003      SERCH.I1 BTST     #3,D1                 Is it EOT?
 676 9 000003F6 670A                   BEQ.S    SERCH.I2              No, continue
 677 9 000003F8 22BCFFFFFFFD           MOVE.L   #-3,(A1)              Yes, STATUS = EOT error
 678 9 000003FE 6000FD56               BRA      RETURN.I              Return
 679                            *
 680 9 00000402 5279000005E6 SERCH.I2 ADDQ.W   #1,ACTFILE            Increment file counter
 681 9 00000408 3439000005E8          MOVE.W   REQFILE,D2           Get requested file #
 682 9 0000040E B479000005E6          CMP.W    ACTFILE,D2           More to search?
 683 9 00000414 6712                  BEQ.S    SERCH.I4             No, continue
 684 9 00000416 082800070029 SERCH.I3 BTST     #7,STAT2(A0)         Wait for
 685 9 0000041C 67F8                  BEQ.S    SERCH.I3             Try again
 686 9 0000041E 117C00140008          MOVE.B   #FILEFWD,GRCMD(A0)   Yes, issue forward file command
 687 9 00000424 6000FD36              BRA      RETN1.I              Return
 688                            *
 689 9 00000428 4291          SERCH.I4 CLR.L    (A1)                 STATUS = ready
 690 9 0000042A 6000FD2A               BRA      RETURN.I             Return
 691                            *
 692                            *          SEARCH BACKWARDS
 693                            *
 694 9 0000042E 08010002      SERCH.I5 BTST     #2,D1                Is it BOT?
 695 9 00000432 670A                   BEQ.S    SERCH.I6             No, continue
 696 9 00000434 22BCFFFFFFFE           MOVE.L   #-2,(A1)             STATUS = BOT error
 697 9 0000043A 6000FD1A               BRA      RETURN.I             Return
 698                            *
 699 9 0000043E 5379000005E6 SERCH.I6 SUBQ.W   #1,ACTFILE            Decrement file counter
 700 9 00000444 3439000005E8          MOVE.W   REQFILE,D2           Get requested file #
 701 9 0000044A B479000005E6          CMP.W    ACTFILE,D2           More to search?
 702 9 00000450 6E12                  BGT.S    SERCH.I7             No, continue
 703 9 00000452 082800070029 SERCH.I8 BTST     #7,STAT2(A0)         Wait for DNBSY
 704 9 00000458 67F8                  BEQ.S    SERCH.I8
 705 9 0000045A 117C00150008          MOVE.B   #FILEREV,GRCMD(A0)   Issue backspace file command
 706 9 00000460 6000FCFA              BRA      RETN1.I              Return
 707                            *
 708 9 00000464 13FC00010000 SERCH.I7 MOVE.B   #1,DIR_FLG           Set direction flag to forward
                05EA
 709 9 0000046C 082800040021 SERCH.I9 BTST     #4,STAT1(A0)         Wait for FNBSY (changing direction)
 710 9 00000472 67F8                  BEQ.S    SERCH.I9             Busy - try again
 711 9 00000474 60A0                  BRA      SERCH.I3             No, move to other side of file mark
 712                            *
 713                            *          If FILECNT >= 0 then drive stopped with more files to search and
 714                            *          is therfore a BLANK TAPE ERROR (searched past end of written tape)
 715                            *
 716 9 00000476 4A79000005EC SRCH.I10 TST.W    FILECNT              Check file counter
 717 9 0000047C 6F00FCD8              BLE      RETURN.I             If count<=0  ignore interrupt
 718 9 00000480 22BCFFFFFFEC          MOVE.L   #-20,(A1)            STATUS = Blank tape error
 719 9 00000486 6000FCCE              BRA      RETURN.I             Return
 720                            *
```

```
 722                            ********************************************************************************
 723                            *         WRITE END-OF-FILE MARK    (SUBROUTINE)                               *
 724                            ********************************************************************************
 725                            *
 726 9           0000048A       FILMRK.S EQU      *
 727 9 0000048A 082800060021             BTST.B   #6,STAT1(A0)          Is there a write ring?
 728 9 00000490 670A                     BEQ.S    FILMK.S1              Yes, continue
 729 9 00000492 22BCFFFFFFFB             MOVE.L   #-5,(A1)              STATUS = file protect error
 730 9 00000498 6000FC18                 BRA      RETURN.S              Return
 731                            *
 732 9 0000049C 117C00060008 FILMK.S1    MOVE.B   #WRTEOF,GRCMD(A0)     Issue file mark command
 733 9 000004A2 22BCFFFFFFF9             MOVE.L   #-7,(A1)              STATUS = busy
 734 9 000004A8 13FC00010000             MOVE.B   #1,MAGINTRP           Set valid interrupt flag
                0001
 735 9 000004B0 6000FC00                 BRA      RETURN.S              Return
 736                            *
 737                            ********************************************************************************
 738                            *         WRITE END-OF-FILE MARK    (INTERRUPT)                                *
 739                            ********************************************************************************
 740                            *
 741 9           000004B4       FILMRK.I EQU      *
 742 9 000004B4 08000001                 BTST     #1,D0                 Correct interrupt?
 743 9 000004B8 661A                     BNE.S    FILMK.I2              Yes, continue
 744 9 000004BA 08010003                 BTST     #3,D1                 No - error, is it EOT?
 745 9 000004BE 670A                     BEQ.S    FILMK.I1              No, must be hard error
 746 9 000004C0 22BCFFFFFFFD             MOVE.L   #-3,(A1)              Yes, STATUS = EOT error
 747 9 000004C6 6000FC8E                 BRA      RETURN.I              Return
 748                            *
 749 9 000004CA 22BCFFFFFFFA FILMK.I1    MOVE.L   #-6,(A1)              STATUS = hard error
 750 9 000004D0 6000FC84                 BRA      RETURN.I              Return
 751                            *
 752 9 000004D4 08010004      FILMK.I2   BTST     #4,D1                 Is it FNBSY?
 753 9 000004D8 6700FC82                 BEQ      RETN1.I               No, ignore interrupt
 754 9 000004DC 4291                     CLR.L    (A1)                  Yes, STATUS = ready
 755 9 000004DE 5279000005E6             ADDQ.W   #1,ACTFILE            Increment file counter
 756 9 000004E4 6000FC70                 BRA      RETURN.I              Return
 757                            *
```

```
 759                            ********************************************************************************
 760                            *          SKIP RECORD(S)              (SUBROUTINE)                        *
 761                            ********************************************************************************
 762                            *
 763 9           000004E8       SKPREC.S EQU      *
 764 9 000004E8 3412                    MOVE.W    (A2),D2              Get repeat count
 765 9 000004EA 33C2000005EC            MOVE.W    D2,REPEAT            Save for later
 766 9 000004F0 4A42                    TST.W     D2                   Determine direction
 767 9 000004F2 6700FBBE                BEQ       RETURN.S             Repeat = 0, don't move
 768 9 000004F6 6D2A                    BLT.S     SKPRC.S2             Go skip backward
 769                            *
 770                            *          SKIP RECORD(S) FORWARD
 771                            *
 772 9 000004F8 082800030021            BTST.B    #3,STAT1(A0)         Is it EOT?
 773 9 000004FE 670A                    BEQ.S     SKPRC.S1             No, continue
 774 9 00000500 22BCFFFFFFFD            MOVE.L    #-3,(A1)             Yes, STATUS = EOT error
 775 9 00000506 6000FBAA                BRA       RETURN.S             Return
 776                            *
 777 9 0000050A 117C00100008 SKPRC.S1   MOVE.B    #SKPFWD,GRCMD(A0)    Issue skip record forward command
 778 9 00000510 22BCFFFFFFF9            MOVE.L    #-7,(A1)             STATUS = busy
 779 9 00000516 13FC00010000            MOVE.B    #1,MAGINTRP          Set valid interrupt flag
                0001
 780 9 0000051E 6000FB92                BRA       RETURN.S             Return
 781                            *
 782                            *          SKIP RECORD(S) REVERSE
 783                            *
 784 9 00000522 082800020021 SKPRC.S2   BTST.B    #2,STAT1(A0)         Is it BOT?
 785 9 00000528 670A                    BEQ.S     SKPRC.S3             No, continue
 786 9 0000052A 22BCFFFFFFFE            MOVE.L    #-2,(A1)             Yes, STATUS = BOT error
 787 9 00000530 6000FB80                BRA       RETURN.S             Return
 788                            *
 789 9 00000534 117C00110008 SKPRC.S3   MOVE.B    #SKPREV,GRCMD(A0)    Issue skip record reverse command
 790 9 0000053A 22BCFFFFFFF9            MOVE.L    #-7,(A1)             STATUS = busy
 791 9 00000540 13FC00010000            MOVE.B    #1,MAGINTRP          Set valid interrupt flag
                0001
 792 9 00000548 6000FB68                BRA       RETURN.S             Return
 793                            *
 794                            ********************************************************************************
 795                            *          SKIP RECORD(S)              (INTERRUPT)                         *
 796                            ********************************************************************************
 797                            *
 798 9           0000054C       SKPREC.I EQU      *
 799 9 0000054C 3439000005EC            MOVE.W    REPEAT,D2            Get repeat count
 800 9 00000552 4A42                    TST.W     D2                   Determine direction
 801 9 00000554 6D34                    BLT.S     SKPRC.I4             Skip backwards
 802                            *
 803                            *          SKIP RECORD(S) FORWARD
 804                            *
 805 9 00000556 08010003                BTST      #3,D1                Is it EOT?
 806 9 0000055A 670A                    BEQ.S     SKPRC.I1             No, continue
 807 9 0000055C 22BCFFFFFFFD            MOVE.L    #-3,(A1)             Yes, STATUS = EOT error
 808 9 00000562 6000FBF2                BRA       RETURN.I             Return
 809                            *
 810 9 00000566 5342            SKPRC.I1 SUBQ.W   #1,D2                Decrement repeat count
 811 9 00000568 33C2000005EC            MOVE.W    D2,REPEAT            Save new repeat count
 812 9 0000056E 4A42                    TST.W     D2                   More to skip?
 813 9 00000570 6F12                    BLE.S     SKPRC.I3             No, continue
 814 9 00000572 082800070029 SKPRC.I2   BTST      #7,STAT2(A0)         Is DNBSY?
```

```
 815 9 00000578 67F8                    BEQ.S    SKPRC.I2              No, try again
 816 9 0000057A 117C00100008            MOVE.B   #SKPFWD,GRCMD(A0)     Yes, issue skip forward record command
 817 9 00000580 6000FBDA                BRA      RETN1.I               Return
 818                            *
 819 9 00000584 4291            SKPRC.I3 CLR.L    (A1)                  STATUS = ready
 820 9 00000586 6000FBCE                BRA      RETURN.I              RETURN
 821                            *
 822                            *         SKIP RECORD(S) BACKWARDS
 823                            *
 824 9 0000058A 08010002        SKPRC.I4 BTST     #2,D1                 Is it BOT?
 825 9 0000058E 670A                    BEQ.S    SKPRC.I5              No, continue
 826 9 00000590 22BCFFFFFFFE            MOVE.L   #-2,(A1)              Yes, STATUS = BOT error
 827 9 00000596 6000FBBE                BRA      RETURN.I              Return
 828                            *
 829 9 0000059A 5242            SKPRC.I5 ADDQ.W   #1,D2                 Increment repeat count
 830 9 0000059C 33C2000005EC            MOVE.W   D2,REPEAT             Save new repeat count
 831 9 000005A2 4A42                    TST.W    D2                    More to skip?
 832 9 000005A4 6CDE                    BGE.S    SKPRC.I3              No, continue
 833 9 000005A6 082800070029 SKPRC.I6 BTST     #7,STAT2(A0)          Is DNBSY?
 834 9 000005AC 67F8                    BEQ.S    SKPRC.I6              No, try again
 835 9 000005AE 117C00110008            MOVE.B   #SKPREV,GRCMD(A0)     Yes, issue skip reverse command
 836 9 000005B4 6000FBA6                BRA      RETN1.I               Return
 837                            *
```

```
 839                            ********************************************************************************
 840                            *          TIME OUT HANDLER                                                    *
 841                            ********************************************************************************
 842                            *
 843 9           00005B8     MAGDVR.T EQU      *
 844 9 000005B8 48F91F3F0000          MOVEM.L  REGLIST,REGSAVE       Save the registers
                0002
 845 9 000005C0 2279000005DC          MOVEA.L  STAT_ADR,A1           Get the status address
 846 9 000005C6 22BCFFFFFFF6          MOVE.L   #-10,(A1)             STATUS = time out
 847 9 000005CC 4CFA1F3FFA32          MOVEM.L  REGSAVE,REGLIST       Restore registers
 848 9 000005D2 4E73                  RTE                            Return
 849                            *
```

```
 851                            ********************************************************************************
 852                            *          CONSTANTS AND STORAGE                                               *
 853                            ********************************************************************************
 854                            *
 855 9 000005D4 00040000       MAG_IO    DC.L      $40000              Base address of MAG I/O
 856 9 000005D8 00000002       PSWSAVE   DS.W      1                   Processor status word
 857 9 000005DA 00000001       RETRY     DS.B      1                   Re-read counter
 858 9 000005DB 00000001       COMMAND   DS.B      1                   Copy of present command
 859 9 000005DC 00000004       STAT_ADR  DS.L      1                   Current status address
 860 9 000005E0 00000004       BUF_ADR   DS.L      1                   Buffer address
 861 9 000005E4 00000002       BUF_CNT   DS.W      1                   Byte count of buffer
 862 9 000005E6 00000002       ACTFILE   DS.W      1                   Actual file number
 863 9 000005E8 00000002       REQFILE   DS.W      1                   Requested file number
 864 9 000005EA 00000001       DIR_FLG   DS.B      1                   Direction flag
 865 9 000005EC 00000000       FILECNT   DS.W      0                   File counter for searching
 866 9 000005EC 00000002       REPEAT    DS.W      1                   Skip record counter
 867                            *
 868                                      END

****** TOTAL ERRORS      0--
****** TOTAL WARNINGS    0--
```

SYMBOL TABLE LISTING

SYMBOL NAME		SECT	VALUE		CROSS-REF (LINENUMBERS)											
A		9	000001DE	-474	475											
ACTFILE		9	000005E6	-862	386	472	600	614	630	680	682	699	701	755		
BUF_ADR		9	000005E0	-860	306	370										
BUF_CNT		9	000005E4	-861	448	489	531	543	574							
CMND.1		9	0000006A	-319	313											
CMND.2		9	0000007A	-323	320											
CMND.3		9	00000082	-334	322											
CMND.4		9	0000011C	-391	377											
CMND.5		9	00000126	-398												
CMNDS.I		9	0000013A	-405	400											
CMNDS.S		9	00000096	-343	336											
COMMAND		9	000005DB	-858	335	399	626									
DATIO			00000001	-268	483	551										
DIRSL			00000038	-275												
DIR_FLG		9	000005EA	-864	642	655	670	708								
DSCMD			00000010	-270	383	384	429	431	446	530	594					
FILECNT		9	000005EC	-865	631	669	716									
FILEFWD			00000014	-253	641	686										
FILEREV			00000015	-254	654	705										
FILMK.I1		9	000004CA	-749	745											
FILMK.I2		9	000004D4	-752	743											
FILMK.S1		9	0000049C	-732	728											
FILMRK.I		9	000004B4	-741	410											
FILMRK.S		9	0000048A	-726	348											
FMTDSBL			00000004	-262	384	431										
GRCMD			00000008	-269	454	502	507	559	641	654	686	705	732	777	789	816	835
INIT.I		9	0000017A	-438	405											
INIT.S		9	00000164	-428	343											
LSWRD			00000030	-274	558											
MAGDVR	XDEF	9	0000002E	-294	-7											
MAGDVR.I	XDEF	9	000000C2	-366	-9											
MAGDVR.T	XDEF	9	000005B8	-843	-10											
MAGDVR1	XDEF	9	0000002E	-295	-8											
MAGINTRP	XDEF	9	00000001	-280	-7	376	419	456	561	596	644	657	734	779	791	
MAG_IO		9	000005D4	-855	308	369										
MAXINVAL		9	00000000	-279												
MTA_RST			00000008	-263	383	429	446	530								
OFFLINE			00000002	-261												
PSWSAVE		9	000005D8	-856	297	355										
READ.I		9	000001BC	-463	406											
READ.I1		9	000001EA	-478	470	490										
READ.I2		9	00000214	-494	479											
READ.I3		9	0000021C	-498	467											
READ.I4		9	00000238	-505	506											
READ.I5		9	0000024A	-510	504											
READ.I6		9	00000254	-513	499											
READ.I7		9	00000222	-500	501											
READ.S		9	0000017E	-445	344											
READ.S1		9	000001A4	-454	450											
READFWD			00000000	-249	454	507										
READREV			00000001	-250												
REGLIST	REG	*		-281	299	354	368	420	844	847						

```
REGSAVE          9  00000002  -283   299   354   368   420   844   847
REPEAT           9  000005EC  -866   765   799   811   830
REQFILE          9  000005E8  -863   629   681   700
RETN1.I          9  0000015C  -420   381   394   465   508   572   666   687   706   753   817   836
RETRY            9  000005DA  -857   447   503
RETURN.I         9  00000156  -419   385   387   476   492   496   511   514   576   581   584   611   615   678
                              690   697   717   719   747   750   756   808   820   827
RETURN.S         9  000000B2  -353   315   324   452   457   535   540   562   597   601   639   645   652   658
                              730   735   767   775   780   787   792
REWIND.I         9  0000032E  -607   408   438
REWIND.S         9  00000300  -591   346   432   627
REWND                00000001  -260
REWND.I1         9  0000033E  -613   609
REWND.S1         9  00000320  -599   593
ROTCHAR          9  0000025E  -518   484   550
SEARCH.I         9  000003D4  -664   409
SEARCH.S         9  0000034C  -622   347
SERCH.I1         9  000003F2  -675
SERCH.I2         9  00000402  -680   676
SERCH.I3         9  00000416  -684   685   711
SERCH.I4         9  00000428  -689   683
SERCH.I5         9  0000042E  -694   671
SERCH.I6         9  0000043E  -699   695
SERCH.I7         9  00000464  -708   702
SERCH.I8         9  00000452  -703   704
SERCH.I9         9  0000046C  -709   710
SERCH.S1         9  00000384  -641   637
SERCH.S2         9  000003A4  -649   632
SERCH.S3         9  000003B6  -654   650
SERCH.S7         9  0000035E  -629   625
SKPFWD               00000010  -255   777   816
SKPRC.I1         9  00000566  -810   806
SKPRC.I2         9  00000572  -814   815
SKPRC.I3         9  00000584  -819   813   832
SKPRC.I4         9  0000058A  -824   801
SKPRC.I5         9  0000059A  -829   825
SKPRC.I6         9  000005A6  -833   834
SKPRC.S1         9  0000050A  -777   773
SKPRC.S2         9  00000522  -784   768
SKPRC.S3         9  00000534  -789   785
SKPREC.I         9  0000054C  -798   411
SKPREC.S         9  000004E8  -763   349
SKPREV               00000011  -256   502   789   835
SRCH.I10         9  00000476  -716   668
STAT1                00000021  -272   312   319   321   375   449   474   500   505   532   537   592   636   649
                              709   727   772   784
STAT2                00000029  -273   374   478   684   703   814   833
STAT_ADR         9  000005DC  -859   307   371   845
TRSEL                00000018  -271   430
VI               9  00000112  -386   379
WRITE.I          9  000002CA  -568   407
WRITE.I1         9  000002E6  -578   573
WRITE.I2         9  000002F6  -583   579
WRITE.I3         9  000002DA  -574   570
WRITE.S          9  00000262  -529   345
WRITE.S1         9  00000280  -537   533
WRITE.S2         9  00000292  -542   538
WRITE.S3         9  000002AC  -558   544
```

---------- NFC 3100 MAGTAPE DRIVER ----------

WRITE.S4	9	00000294	-543	556
WRTEOF		00000006	-252	732
WRTFWD		00000002	-251	559