3 Diagnostics

There are two types of diagnostics: ROM diagnostics that consist of a program that is an integral part of the hardware, and a floppy disk program contained on a single floppy disk.

Boot ROM Program

The primary function of the boot ROM program is to boot a program that loads the operating system from the hard disk drive or the diagnostics from the floppy disk drive. In addition, the boot ROM includes a number of diagnostic tests. It tests ROM, RAM memory, and video memory; it also programs the initial status of the memory management hardware and various peripheral controller chips.

During execution, the boot ROM program turns a set of LEDs on and off in appropriate binary number patterns as it completes its tests. The LEDs are visible through the ventilation slots on the left side of the system. (The ROM test executes so fast that the blinking on and off of the LEDs is not be noticed.) The successful completion of each test starts the next. If a test fails, the binary number pattern of that test continues to be displayed as long as the power is on.

Pressing the Reset button or turning on the power causes the boot ROM diagnostics program to execute.

The following descriptions of boot ROM program steps include address and data information for setting up a logic analyzer to trace program execution.

Bootstrap Jump

When the Reset button is released, the initial value of the stack pointer is loaded from addresses 800000 and 800002. The address of the first executable instruction is loaded into the program counter from reset vector locations 800004 and 800006. Then the program:

- Writes a data word with D15 high to address E40000, which sets IC7K (pin 7) high (ROMLMAP*, sheet 6).
- Turns off the LEDs by writing 0F00 to address 4A0000. (The LEDs are controlled by the miscellaneous control register, shown on sheet 15 of the schematics in Chapter 5.)

Initializing the System

To initialize the system, the program writes 0700 to address 4A0000, which sets the LEDs to binary 1. (Note: Data and address are in hexadecimal notation.)

Initializing the 7201 Serial Port Controller

To initialize channels A and B of the 7201 serial port controller, the program:

- Sets LEDs to 1.
- Resets the error registers by writing 18 to addresses E50004 and E50006.
- Writes F0 to addresses E50004 and E50006.

Initializing the Keyboard

To initialize the keyboard, the program:

- Resets the keyboard controller by writing 0300 to E70000.
- Sets the keyboard controller to eight bits per character with one stop bit by writing 9500 to E70000.

Initializing the Modem

To initialize the modem, the program:

- Disconnects the modem from lines 1 and 2 by writing 8000 to E44000 and E45000.
- Resets the modem by writing 0001 and then 0000 to E60000.

Initializing the Telephone Line Control

To initialize the telephone line control, the program:

- Enables the handset by writing 4000 to 490000
- Selects line 1 by writing 4000 to 491000
- Reads the telephone status register at 450000.

If bit D0 is a 0, the handset is onhook, so the program writes 4000 to 492000 and 496000 to open relay 1 and line 1.

If bit D0 is a 1, the handset is offhook, so the program writes 0 to 494000 and 495000, and then 4000 to 493000 and 497000, to maintain line 1.

Clearing the Printer Interrupt

To clear the printer interrupt, the program writes 0000 to 4F0000.

Clearing the Dialer Chip

To clear the dialer chip, the program writes 0 to 4B0400 and 4B0800.

Resetting the Disk DMA

To reset the disk DMA, the program writes 0000 to 460000 and 4D0000.

Testing Video RAM

To test video RAM, the program:

- Sets LEDs to 2
- Writes 0000 to 420000, the lowest video address
- Reads the same address (420000)

If the contents are not the same, the program jumps to an error loop. If the contents are the same, the program writes 0001 to address 420002 and continues to read and test.

- Continues incrementing address and data until the last video memory address (427FFF) has been tested.
- Reads back each address and checks to see that the contents are correct.

After each address is read, it is written again in case writing to it will affect the contents of the next address.

- Writes 0000 to all video addresses to clear the screen.

Testing Map RAM Memory

To test map RAM memory, the program:

- Sets LEDs to 3
- Performs the same tests on map RAM memory that were performed on video memory from addresses 400000 through 4007FF
- Sets the unity map by writing to all map RAMs, starting with 400000 and ending with 4007FF
- Writes A000 to address 400000 to map page 0

This sets page status bits D15, D14, and D13 = to 101, which corresponds to page status of page present, write enabled, not yet written to, and unity-mapped.

- Increments the address and data and writes to pages 1, 2, 3, and so on until all pages have been declared present, write enabled, not written to, and unity mapped.

Testing RAM

To test RAM, the program:

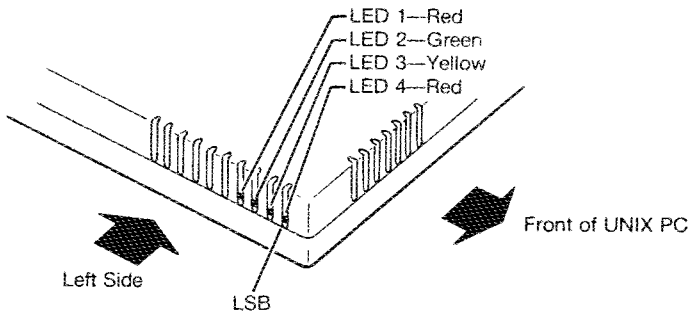
- Sets LEDs to 4.
- Performs the same memory test as before from addresses 000000 through 7FFFFF
- Sets LEDs to 5
- Sets LEDs to 6
- Puts a small inverse video block on the upper-left corner of the screen
- Searches for a loader program

The floppy disk is searched first. If no loader is found or if an error is found, the hard disk is searched. If the hard disk has no loader or has an error (for example, not ready), an inverse video block is added to the screen, and the process is repeated until a loader is found.

Jumping to the Loader Program

When the program finds the loader, it advances the LEDs to 7 and the processor jumps to the loader program.

Figure 3-1 shows the LED locations and off/on patterns for each test.



ROM Tests

	LED Status			
Test Number	**4**	**3**	**2**	**1**
Test 1	Off	Off	Off	On
Test 2	Off	Off	On	Off
Test 3	Off	Off	On	On
Test 4	Off	On	Off	Off
Test 5	Off	On	Off	On
Test 6	Off	On	On	Off
Test 7	Off	On	On	On

Figure 3-1 LED Locations and Off/On Patterns

Listed below is the status of the logic board for each LED test number:

Test 1: Failed telephone initialization
Test 2: Failed video RAM test
Test 3: Failed map RAM test
Test 4: Failed to set map RAM to unity map
Test 5: Failed dynamic RAM test
Test 6: Failed initialization
Test 7: Failed to find loader on disk

Note

Any other failure codes indicate a failure to execute the loader program.

Floppy Disk Diagnostics

The floppy disk diagnostics are in two parts, a Main Menu set which exercises the main components of the system, and a subsystem set.

Main Menu Diagnostics Summary

The following list summarizes how the various diagnostic tests from the UNIX PC Diagnostics Main Menu are divided into subtests or sequences. Tests that require operator evaluation are labeled interactive. The others, which run without user intervention, are labeled automatic. The automatic tests report failures with error messages. To abort any test still in process, with the exception of formatting tests, type "shift-break".

Test 1: Full System Test (Interactive)

This test consists of the following subsystem tests:

Note

These tests are from the Subsystem Menu, selection 6 of the Main Menu. They are run in order as listed.

- Test 2: Floppy disk
- Test 1: Hard disk
- Test 5: Memory and parity
- Test 9: Processor
- Test 11: Real Time Clock
- Test 7: Modem

Test 2: Initialize Hard Disk (Interactive)

- Requests type of drive
- Formats

Test 3: Enter Bad Blocks (Automatic)

- Modifies bad block table
- Displays VHB

Test 4: Park Disk Heads (Interactive)

- Parks disk

Test 5: Remote Diagnostics (Interactive)

- Allows diagnostics to be run from a remote site

Test 6: Goto Subsystem Menu

- These are a series of tests for subsystem checks

Selection 7: Reboot System

- Reboots the system

Diagnostic Subtest Summary

The following list summarizes the diagnostic tests from selection 6 of the Main Menu. These tests are also divided into subtests or sequences. Tests that require operator evaluation are labeled interactive. The others, which run without user intervention, are labeled automatic. The automatic tests report failures with error messages.

Test 1: Hard Disk (Automatic)

- Recalibration of hard disk
- Random seek of hard disk

Test 2: Floppy Disk (Interactive)

- Formats the floppy disk
- Random seek of floppy disk

Test 3: Keyboard (Interactive)

- Determines whether the keyboard and mouse work properly

Test 4: Video (Interactive)

- Tests all screen parameters

Test 5: Memory & Parity (Interactive)

- Performs data test
- Performs address test
- Performs random pattern test

Test 6: Communications (RS232 Ports) (Automatic)

- Self-test of the RS232 Expansion Ports
- Transfer test at 300 Baud through 19200 Baud

Test 7: Modem (Automatic)

- Internal loopback sequence test used to check:
  - 1200 or 300 Baud
  - no parity, odd parity or even parity
  - 7-bit or 8-bit characters

Test 8: Dialer (Interactive)

- Tests touch-tone pulses
- Tests rotary-dial pulses

Test 9: Processor (Automatic)

- Map RAM test:
  - Data
  - Address
  - Random pattern
- Parity test:
  - Read/write
  - Execution test
- Map translation test:
  - Subtest 1
  - Subtest 2
- Page fault test
- User I/O interrupt test:
  - Subtest 1
  - Subtest 2
- Clock test
- Page protection test

Test 10: Parallel Printer (Automatic)

- Subtest 1 (status test)
- Subtest 2 (transfer test)

Test 11: Real Time Clock (Interactive)

- Read Write test
- Operation test

Test 12: Return to Main Menu

- Returns you to the Main Menu diagnostics

Diagnostic Test Descriptions

The following descriptions include test algorithms, error messages, and screen displays for the floppy disk diagnostic tests.

Memory Test

This test has three subtests: data, address, and random pattern. The same algorithm is used to test map RAM, dynamic RAM, and video memory.

Memory Subtest: Data Test

This test writes a walking ones pattern to a memory location and then reads it back. It tests all memory locations. If the data read back is incorrect, it reports the address, data written, and data read back in the following error message:

Memory error at Address X; Wrote X; Read Back X

Memory Subtest: Address Test

The address test has the following three-step algorithm for each address tested:

- Writes data to a memory location.
- Reverses one bit in the address and writes different data.

- Checks the first address to see if there is a change. If data changes, the test generates an error message reporting the address line that has changed and the memory bank being tested. For example, it writes AAAA to addresss 000000 and then writes 5555 to address 000001. It then reads the contents of address 000000. If the contents are not still AAAA, address line A0 is bad. The following error message appears:

Memory error: Connection on address line X is bad at Bank X

Memory Subtest: Random Pattern Test

During this test, a random number generator sequence generates a 64KB random pattern of 16-bit words. This pattern is then written to memory, repeating every 64KB. Then the random number generator is invoked again, and the pattern is read back from memory and compared with the function generator output.

Memory Test Error Message

If the two patterns differ, the address data written and data read are displayed in the following error message:

Memory error at Address X; Wrote X; Read Back X

Memory Test Screen Display

```
MEMORY TEST

      DATA TEST
      Memory test will begin at 2F000, end at 7FFFF
      Testing.....................................
      ADDRESS TEST
      Memory test will begin at 2F000, end at 7FFFF
      RANDOM PATTERN TEST
      Memory test will begin at 2F000, end at 7FFFF
EXPANSION MEMORY TEST
```

Note

The preceding message is displayed if expansion memory is not present. If it is present, the preceding memory tests are repeated on expansion memory.

Parity Test

This test forces bad parity on each location and checks for an error during each memory access.

It first writes 8000 to address E40000 to set EE+ high at IC7K (pin 4) high. Next it writes 8000 to address E41000 to set PIE+ high (IC7K, pin 5). Then memory is written to and read to see if an interrupt results. If a parity error does occur, the following message appears:

Unexpected parity error at location X

Then the BP+ at IC7K (pin 6) is set low by writing 0 to address E42000. This causes bad parity to be written during any access to memory. Next, a selected memory location is written. Then that location is read. A level 7 interrupt results during the read. This interrupt causes the current address and data to be stored in bus status registers BS0 at address 430000 and BS1 at 440000 and the error bit status to be stored in the general status register at 410000.

Parity Test Error Messages

If the interrupt does not occur, the following messages appear:

No Parity Interrupt at location X
BSR incorrect after parity error at location X
BSR0 = X, BSR1 = X

Parity Test Screen Display

Parity Test

```
PARITY TEST - READ/WRITE TEST
Memory test will begin at 2F000, end at 7FFFF
     Reached Address 30000
     Reached Address 40000
     Reached Address 50000
     Reached Address 60000
     Reached Address 70000
PARITY TEST - SUBTEST 2 EXECUTION TEST
```

Map Translation Test

This test performs the following steps:

- Initializes all pages to one-to-one correspondence between logical and physical address space.
- Checks status bits of map registers. First it writes page present to all addresses and then reads them all. The status bits should show all pages present and read. Next it writes to every location in memory and checks to see that every page shows that it has been written to.
- Interchanges physical and logical page addresses and verifies that content has been interchanged.

The map translation test checks the status bits and page table entry swapping. The subtests are:

- Read/write memory test (checks access and dirty bits)
- Map test (checks page table entry swapping)

Subtest 1 (read/write memory test) performs the following steps:

- Sets status register of a page to valid (01)
- Reads from a memory location on that page
- Reads the status of the page and verifies that it has changed to read (10)
- Writes to a memory location on that page
- Reads the status of the page and verifies that it has changed to written to (11).

Subtest 2 checks mapping to the correct physical memory location as follows:

- Writes to two locations on different pages.
- Swaps the page table entries for the two pages.
- Reads back the two locations and verifies that the values are swapped.

Map Translation Test Error Messages

Subtest 1 can return the following error messages:

Page access bit not set for page number
Page access bits wrong: page number and page bits are X
Page mapping error
Page dirty bit not set for page number

Map Translation Test Screen Display

MAP TRANSLATION TEST
MAP TRANSLATION SUBTEST 1
MAP TRANSLATION SUBTEST 2

Page Protection Test

The page protection test has the following three parts:

o CPU page fault

During this portion of the test, a page is declared not present and the processor generates bus errors by reading and writing to it.

The following error messages may occur during this test:

No page fault received on write
Map add = X, Map = X, Mem add = X

No page fault received on read
Map add = X, Map = X, Mem add = X

Bus Error when none expected
Map add = X, Map = X, Mem add = X

o Writing to page, not write enabled, by user

This portion of the test sets the processor to user mode and writes to a memory page that is write-disabled. It accesses the first location of every page except those used by diagnostics.

The following error messages may occur during this test:

No Bus Error received on write, after page write being disabled
Map add = X, Map = X, Mem add = X

o Access below 512 KB by user

This portion of the test sets the 68010 to user mode and writes to every location below 512 KB except that contained by diagnostics. It verifies BSR0 for MMU, BSR0 and BSR1 for faulted address, and RAM for write-disabled.

The following error messages may occur during this test:

No Bus error detected, while user accessing below 512k memory
Disabled Ram writing failed at mem loca X

BSR incorrect after Bus error at location X
BSR0 = , BSR1 =

Page Protection Test Screen Display

PAGE PROTECTION TEST

Format Disk Test

This test has the following three parts:

- Reads volume home block (VHB) and bad block table (BBT)

This portion of the test reads the VHB and BBT and calculates a check sum. If the check sum is correct, it saves the VHB and BBT to rewrite them later. If the check sum is incorrect or the VHB is not present, new VHB and BBT are written after formatting is complete. Whether check sum is correct or not, it continues formatting.

- Formats all sectors

This portion of the test formats the disk one track at a time. After each track is formatted, it reads the status register to see if an error has occurred.

If the status register reports an error, the following message appears:

Error during Disk Format: Response = XX

(XX is the hexadecimal contents of the status register.)

After the disk is formatted, the VHB and BBT are written and read back and a check sum is calculated. If the check sum is incorrect, the following message appears:

VHB write failed. Disk needs to be re-initialized

If the BBT check sum is bad, the following message appears:

Bad block table write failed. Disk needs to be re-initialized

Format Disk Test Screen Display

FORMAT DISK
Formatting cylinder xx

Recalibration Test

This test has the following three parts:

- Seeks to track 0
- Reads status register for error

If an error is present, the following message appears:

Can't Recal: Response = XX

- Reads VHB and BBT and calculates the check sum

If the check sum is incorrect, the following error message appears:

Recal Failed

Recalibration Test Screen Display

Recal Disk

Surface Test

This test writes 6DB6DB6D to all byte locations in one track, 16 sectors. While writing this pattern, it checks the status register for errors.

If an error occurs, the test determines the number of the sector that has the error by writing 6DB6DB6D to one sector at a time. After each sector is written, the status register is checked. When the error is found, the BBT is updated with the number of the sector that generated the error.

If no errors are generated while the track is being written, the test reads the contents of all 16 sectors and checks the data for any errors.

If an error is found, it reads the sectors one at time to determine the one containing the error and updates the BBT. If no error is found, it checks the next track.

If all tracks are written without error, the test reads all tracks one at a time and checks for error. If an error occurs, it reads each sector to find the one that contains the error and adds it to the BBT.

Surface Test Error Messages

Can't Write the new VHB: Response = XX
Can't Write the new Bad Block Table: Response = XX
Error on Write: Response = XX, Start Block = XX
Error on Re-Read: Response = XX, Start Block = XX
Re-Read Data Fail: Start Block = XX, Byte = XX,
Received XX, Expected XX
Error on Check-Read: Response = XX, Start Block = XX
Check-Read Data Fail: Response = XX, Start Block = XX
Initiating Check Read for pass XX
Bad Block Table Overflow when adding Sector XX
Bad Block Table: Multiple use of alternate XX

Surface Test Screen Display

```
                Surface test
Volume Name: FLOPPY
Pass 1
Testing blocks xxx...yyy
```

User I/O Interrupt Test

This test tries to create a bus fault by having a user program attempt to access a register outside its space. If this does not cause a bus fault, the following error message appears:

No bus error when user access I/O address X

User I/O Interrupt Test Screen Display

```
USER I/O INTERRUPT TEST
     USER I/O INTERRUPT SUBTEST 1
     USER I/O INTERRUPT SUBTEST 2
```

Clock Test

This test sets up a clock interrupt so that a digit appears on the screen every second and counts down from 9. The screen shows 9, 8, 7, 6, ..., 1.

The following error message may appear during the clock test:

```
Time out while waiting for 60 Hz interrupt
```

Clock Test Screen Display

```
CLOCK TEST
     Subtest 1 - Timer 1/Counter 2 TEST
          9 8 7 6 5 4 3 2 1 0
```

Printer Test

The printer test has two parts. First the status register is read, and then a barber pole printing pattern is sent tc the printer.

Printer Test Screen Display

```
LINE PRINTER TEST
     LINE PRINTER SUBTEST 1, Status test
Line printer is selected.
     LINE PRINTER SUBTEST 2, Transfer test
Line printer is selected.
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

Communications Test

This test does a loopback test and checks combinations of different baud rates, number of bits, and types of parity as listed below:

- Baud rate is 300, 1200, 2400, 4800, 9600, or 19,200
- Bit count is 5, 6, 7, or 8
- Parity is none, odd, or even

Communications Test Error Messages

```
Overrun error
Framing error
Parity and framing error
```

Communications Test Screen Display

```
COMMUNICATION TEST (Self-test & Transfer test)

SELF-TEST
TRANSFER TEST

300 Baud
7 bits/character    no parity
7 bits/character    odd parity
7 bits/character    even parity
8 bits/character    no parity

1200 Baud
7 bits/character    no parity
7 bits/character    odd parity
7 bits/character    even parity
8 bits/character    no parity

2400 Baud
7 bits/character    no parity
7 bits/character    odd parity
7 bits/character    even parity
8 bits/character    no parity

4800 Baud
7 bits/character    no parity
7 bits/character    odd parity
7 bits/character    even parity
8 bits/character    no parity

9600 Baud
7 bits/character    no parity
7 bits/character    odd parity
7 bits/character    even parity
8 bits/character    no parity

19200 Baud
7 bits/character    no parity
7 bits/character    odd parity
7 bits/character    even parity
8 bits/character    no parity
```

After 19,200 baud, the program returns to the main diagnostics menu.

Modem Test

During this test, an internal loopback test sequence checks all possible combinations of the following parameters:

- 300 or 1200 baud
- 7- or 8-bit characters
- No parity, odd parity, or even parity

Modem Test Error Messages

```
SELF TEST failure detected
MODEM TRANSFER TEST, receive error - parity error,
     expected XX
```

Modem Test Screen Display

```
MODEM TEST (Self-test & Transfer test)
     SELF-TEST(B212 mode) at 300 baud..
          Test passed with no error detected.
     SELF-TEST (B212 mode) at 1200 baud..
          Test passed with no error detected.

     DATA TRANSFER TEST at 300 baud..
7 bits/character              no parity
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********

7 bits/character              odd parity
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********

7 bits/character              even parity
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********

8 bits/character              no parity
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
```

```
        8 bits/character                odd parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********

        8 bits/character                even parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********

             Test passed with no error detected.


DATA TRANSFER TEST at 1200 baud
        7 bits/character                no parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********

        7 bits/character                odd parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********

        7 bits/character                even parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********

        8 bits/character                no parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********

        8 bits/character                odd parity
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
        ******** ******** ******** ******** ******** ********
```

```
8 bits/character                    even parity
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
******** ******** ******** ******** ******** ********
```

Test passed with no error detected.

Dialer Test

This interactive test generates either touch tones or rotary dial pulses.

Connect outside telephone lines. Each number is dialed as you enter it, and you can hear it on the speaker. To connect the handset to the telephone line when answering a call, type **c** after the dialed number answers. To disconnect, type **q**.

Note

You must have the handset off hook before you type **c**. You then type **h** to put the line on hold.

Select Test 10 from the floppy disk main diagnostics menu. The following screen appears (DTMF means dual-tone multifrequency):

```
-- INTERACTIVE DTMF TEST --
Select line 1 or line 2 for testing:
```

When you select a line, the following message appears:

Enter digit 0-9;*,# for dialing. To connect the hand set for conversation, enter C/c. To toggle line between hold and active, enter H/h. To quit enter Q/q:

When you select a line and enter a number, you hear the touch tones as the number is transmitted, and the number is displayed on the screen. If you enter a series of numbers faster than they can be transmitted, they are stored and transmitted.

When you type **q**, the preceding message returns. When you select a line and enter numbers for the second time, dial pulses are generated.

Note

Pulse dialing may not work on certain internal electronic telephone systems.

When you type **q** the second time, the following message appears:

```
--      AUTO ANSWER  --
Waiting for incoming call! type Reset/Break to exit!
```

At this point, dial the number of the telephone that is connected to your **UNIX** PC from another telephone.

Video Test

This interactive test displays the following menu of test patterns:

```
VIDEO TEST
----------

 0) All black
 1) All white
 2) Half tone
 3) Vertical bars
 4) Horizontal bars
 5) Mosquito net
 6) Black pattern
 7) White pattern
 8) All m's
 9) Exit

At the end of a test, press any key to get back to this menu.

Please select test number 0 - 9 :(Return is 0)
```

Screen Descriptions

Test 0 makes the screen solid black.
Test 1 makes the screen solid light.
Test 2 displays a fine dot pattern.
Test 3 displays a sequence of vertical bars.
Test 4 displays a sequence of horizontal bars.
Test 5 displays a combination of Tests 4 and 5.
Test 6 displays a cross for checking screen alignment.
Test 7 is the same as Test 7 with inverse video.
Test 8 fills the screen with the letter m.

Trap Errors

Trap errors are general errors that can occur during any test.

Type = XX, GSR = XX, BSR0 = XX, BSR1 = XX
Repeated fault
Bad GSR after page fault
Bad GSR after execution Page fault
BSR not correctly set after Page Fault
BSR0 = XX, should be XX BSR1 = , should be XX
Unexpected bus error, GSR = XX, PC = XX, RPS = X
Unexpected NMI, PC = XX, RPS = XX
GSR = XX BSR0 = XX BSR1 = XX
Unexpected interrupt from level XX
PC = XX RPS = XX GSR = BSR0 = XX BSR1 = XX
HD or DMA interrupt from level XX, PC = XX, RPS = XX
GSR = XX BSR0 = X BSR1 = XX
XX Interrupt level XX, PC = XX, RPS = XX
User I/0 flag in GSR not set during User I/0 test
LWT in GSR set during User I/0 fault
Bus Grant set to DMA cycle during User I/0 fault
BSR not correctly set after User I/0 fault
BSR0 = XX, should be XX BSR1 = XX, should be XX

Expert Mode Diagnostics Program

The expert mode diagnostics program contains more subtests than are available from the floppy disk diagnostics main menu. In addition, it allows selection of specific subtests without running the complete test.

To enter expert mode diagnostics, type **s4test** and press <Return> instead of selecting a test number from the floppy disk diagnostics main menu. To exit from expert mode diagnostics, type: u and press <Return>.

A command> prompt is displayed to show that expert mode diagnostics has been entered.

To see a menu of expert mode diagnostics, type ? and press <Return>. The following menu scrolls on the screen:

```
Commands formatted at follows:
     [<repeat-count>]<command-letter>[L | C]      or
     [<repeat-count>]:<test-number>[,<subtest-number>][L |C]
           P[E | M] toggles Parity Enable | Page Mode
           LE toggles Parallel Printer Echo Mode
           V for Diagnostic Version,     U to return to User Menu
           Multiple commands may be separated by ;
Command    Description
1    Full System test.
2    Initialize Hard disk.
3    Enter Bad blocks.
4    Park Disk Heads.
7    Reboot System.
11    Hard Disk test.
12    Floppy Disk test.
13    Key board & Mouse Test.
14    Video test.
15    Memory & Parity test.
16    Communication test.
17    Modem Test.
18    Dialer test.
19    Processor test.
20    Parallel Printer test.
21    Real time clock test.
31    Interactive device test.
```

Note that entering page mode prevents the top items from rapidly scrolling off the screen. To enter page mode, type **PM** and press <Return> before typing **?**. Press <Page> to see the next page of the menu. Type **PM** and press <Return> again to exit page mode.

Disk Drive Commands

When the **command>** prompt appears, type **i** (Initialize command) and press <Return>. The **disk>** prompt appears. To return to the **command>** prompt, type **q**.

To select the drive that you want to test, type **dr n**, where n is either 0 (default) for the floppy disk or 1 for the hard disk. Then issue the Initialize (i) command or manually turn on the motor, select the drive, and reset the drive.

To include more than one test in a single command, use semicolons. For example:

disk> step 10; rd 1; sb; step - 10; rd 1

To repeat a command, type **;** r at the end of the command. To terminate the repeat option, press any key.

Read and Write Commands

All Read and Write commands operate on the specified sector(s) on the current track. To read from another track, first execute a Seek or a Step command.

Table 3-1 Read/Write Command Format

Command	Short Form	Syntax	Comment
Read	rd	rd [[sector] sector]	
Write	wr	wr [[sector] sector]	
Read ID	ri	ri	FD only
Read Track	rt	rt	FD only
Write Track	wt	wt	FD only

The Read and Write commands use the sector buffer.

The Read ID command uses the ID buffer.

The Read Track and Write Track commands use the track buffer.

Seek and Step Commands

Restore and Seek commands read the ID field at the destination track. The Step command issues step pulses without verifying.

Table 3-2 Seek and Step Command Format

Command	Short Form	Syntax	Comment
Restore	rs	rs	
Recal	rc	rc	
Home	h	h	
Seek	s	s track	
Step	(none)	step number	FD only

Register Commands

Register commands with no argument show the register. Register commands with an argument write to the register.

Table 3-3 Register Command Format

Command	Short Form	Syntax	Comment
Register	rg	rg	
Status	st	st [number]	
Command Register	cr	cr [number]	
Track Register	tr	tr [number]	FD only
Sector Register	sr	sr [number]	FD only
Data Register	dr	dr [number]	FD only
Sector Count Register	sc	sc [number]	HD only
Sector Number Register	sn	sn [number]	HD only
Cylinder Low Register	cl	cl [number]	HD only
Cylinder High Register	ch	ch [number]	HD only
Size Drive Head Register	sdh	sdh [number]	HD only

Buffer Commands

Buffer commands with no arguments show the buffer. Buffer commands with an argument fill the whole buffer with that number. The sector buffer holds 512 bytes, the ID buffer holds 6 bytes, and the track buffer holds approximately 8 KB.

Table 3-4 Buffer Command Format

Command	Short Form	Syntax
Sector Buffer	sb	sb [number]
Track Buffer	tb	tb [number]
ID Buffer	ib	ib [number]

Miscellaneous Commands

These commands are floppy and hard disk control commands.

Table 3-5 Miscellaneous Command Format

Command	Short Form	Syntax	Comment
Drive	dr	dr [number]	0=FD, 1=Internal Hard Disk
Initialize	i	i	
Mtron	(none)	mtron	FD only
Mtroff	(none)	mtroff	FD only
Format	fm	fm track	
Force Interrupt	fi	fi	FD only
Help ?	?		
Radix	(none)	radix [number]	
Quit	q	q	
Head	hd	hd [number]	
Reset	(none)	reset	
Repeat	r	r	

The Mtron (motor on) and Mtroff (motor off) commands control the floppy motor.

The Head command selects a head for further operations.

The Format and Radix commands do not work.

Reading/Writing to Hard Disk with Expert Mode

Use the following sequence of commands and prompts to read and write data to a sector of the hard disk drive using expert mode diagnostics.

1 Load floppy disk diagnostics.

2 When the test select prompt below the diagnostics main menu appears, type **s4test** and press <Return>.

The **command>** prompt appears.

3 Type **i** (the Initialize command) and press <Return>.

The **disk>** prompt appears.

4 Type **dr 1** and press <Return> to select the hard disk.

The **disk>** prompt reappears.

5 Type **i** and press <Return> to initialize the drive.

The LED on the drive lights up.

From this point, use the Sector Buffer (sb), Read Sector (rd), and Write Sector (wr) commands to fill the sector buffer with data, write the contents of the sector buffer to a sector, read the sector into the buffer, and then read the sector buffer contents.

For example, to fill sector 0 with 5555s and then read it back, issue the following sequence of commands after the **disk>** prompt:

1 Type **sb** to read the sector buffer.

The current contents of the sector buffer appear.

2 Type **sb 5555** to fill the sector buffer with 5555s.

3 Type **sb** to verify that the sector buffer contains 5555s.

4 Type **wr 0** to write the contents of the buffer to sector 0.

5 Type **sb 0** to fill the sector buffer with 0s, so you can see that the buffer contents have changed when you read sector 0 from the drive into the buffer.

6 Type **sb** to verify that the sector buffer contains all 0s.

7 Type **rd 0** to read from sector 0 and put the contents into the sector buffer.

8 Type **sb** to read the sector buffer, which now shows 5555s.

4 Logic Board Test Procedures

This section is a collection of test procedures for component-level troubleshooting of the **UNIX** PC logic board. In addition to standard test equipment, the following items are required:

- A set of debugger EPROMs
- A set of map RAM test EPROMs
- A set of RAM test EPROMs
- A set of silent loader EPROMs
- A logic analyzer
- An unintelligent terminal
- A master set of PALs
- A supply of the three semicustom gate arrays

The following reference books will also be useful:

- Motorola's MC68010 16-Bit Virtual Memory Microprocessor
- Motorola's MC68000 Programmer's Reference Manual
- Western Digital's Storage Management Products Handbook
- Intel's Microprocessor and Peripheral Handbook

Information provided with the board usually indicates to an experienced technician the tests that should be used. If the nature of the problem is not known, however, the first step in troubleshooting is to determine the highest level of diagnostics that will run. The higher the level, the larger the portion of the board that must be working and thus can be ruled out as the cause of the problem. Procedures associated with that level are then used to pinpoint the cause of the problem. (Some of these procedures have training value as well as testing value, so it is recommended that they be practiced on a known good board.)

The diagnostic levels are listed below in ascending order:

- Map RAM EPROM
- RAM EPROM
- Silent loader
- Debugger program
- Floppy disk diagnostics

MAP RAM EPROM Program

The map RAM EPROM program executes the following tasks:

- Performs data test
- Performs address test

Data Test

The map RAM EPROM tests the map RAMs and outputs data to the LEDs as listed in Table 4-1, (LED 4 is the one nearest the corner of the board):

Table 4-1 Logic Board LED States

Function	LED 4	LED 3	LED 2	LED 1
Testing data	On	Off	Off	Off
Testing address	On	On	Off	Off
Data error	On	Off	Off	On
Address error	On	On	On	Off

If a data error pattern appears, the EPROM is executing a loop that continuously reads and writes to an address where the data read back differs from the data written. This memory cycle can be observed using a logic analyzer. Set the analyzer to trace from the beginning of this loop. In the EPROM currently being used, this loop is located at 80007A hexadecimal, the address of the first instruction in the loop. The loop contains the following assembly instructions:

```
1$    MOVEW     D1,A0@
      MOVEW     A0@,D0
JMP   1$
```

The MOVEW instruction moves a 16-bit word from internal 68010 register D1 to the memory location whose address is contained in 68010 address register A0. Thus the first instruction is a memory write instruction. The second instruction is similar except that it reads from memory to register D0. The third instruction is a jump that causes the first instruction to repeat. Thus, once this loop is entered, these instructions will repeat as long as the power is on.

When the logic analyzer is set to trace starting at the location of the first instruction, it shows the address and data that were written during the first instruction and read during the second. The data read must differ from the data written for the program to have entered the loop. An example of address and data follows:

Address	Data	Type of Cycle
400000	0000	Data write
400000	0800	Data read

In this example, data bit D11 is a 0 during a write and a 1 during a read. Each static RAM chip contributes 4 bits to the data bus. The bit assignment is as follows:

Data Bits	SRAM Reference Designator
D08 & D13-D15	19C
D12-D09	20C
D04-D07	22C
D00-D03	21C

For the example above where D11 fails, device 20C is the most likely cause. Other causes could be the 74F245 buffers which connect the system data bus (D00-D15) with the static RAM data pins. Also, a bit stuck at 1 or at 0 at the first memory location, address 400000 could be caused by the static RAM and the buffers. A bit stuck at only one address (400020 for instance) and no others, suggests the static RAM and not on the buffer is at fault since its data bit is correct for other addresses.

Address Test

In the address test, the program first writes various data to all memory locations. It then checks all memory to see if there are any locations that do not have proper data. If an address line is faulty, shorted, open, or has some other problem, the program assumes that, at some point in the process of writing to all locations, a write in some location (call it new) changes the contents of a previous location (call it old). When it reads all locations and finds one that is incorrect, the program considers this location to be old. The old location is stored.

The next phase of the test determines the new location. To do this, the program starts a second writing operation to all memory locations. Each time it does a write, the program checks the old location to see if the contents have changed. When the program detects a change in the contents of the old address, it stores the address written to, because this must be new.

Error Loops

The following tables show the addresses of machine code that executes when the map RAM EPROM program finds an address or data error. These serve as trace addresses when a logic analyzer is used to determine the error addresses and data.

Data error (LEDs 1 and 4 on; LEDs 2 and 3 off):

Address	Register Notation	Description
800084:	MOVW D1, (A0)	Writes contents of D1 to address in A0
800086:	MOVW (A0), D0	Reads contents of address in A0
800088:	JMP 800084	Returns to top of loop

Both error addresses found (LEDs 4, 3, and 2 on; LED 1 off):

Address	Register Notation	Description
8000FA:	MOVW (A4), D0	Reads memory address in A4
8000FC:	MOVW D3, (A4)	Writes to memory; A4 holds address that causes contents of memory address in A0 to change when it is written to
8000FE:	MOVW (A0), D0	Reads memory address in A0
800100:	MOVW D7, (A0)	Writes to address in A0
800102:	JMP 8000FA	Returns to top of loop

Second error address not found (LEDs 4, 3, 2 on; LED 1 off):

If an address error occurs but the program cannot find the second error address, the following instruction loop is entered:

```
800110:   MOVW (A0), A0
800112:   JMP 800110
```

Data failures for data bits D10-D15 will be found in the address test instead of the data test as these bits are not fully exercised in the data test.

RAM EPROM Program

The RAM EPROM program executes the following tasks after setting unity map:

- Performs data test
- Performs address test

Data Test

The program performs a walking ones and walking zeros test of 512KB of memory. Each address is written to and then the contents are read back. If the contents read differ from those written, the program jumps to the data error loop.

Address Test

The address test of RAM is similar to the test of map RAM above except that several different data patterns are written. First a data pattern of 0-256 is written to all memory addresses. Then memory is divided into eight 64K blocks. Each block is then tested by dividing it into 256 subblocks of 256 bytes each and writing a different data word to each subblock. As in the map RAM address test, there are two error loops, one when both error addresses have been found and one when the second address cannot be found.

Error Loops

The following addresses are for machine code that executes when the RAM EPROM program finds an address or data error. These serve as trace addresses when a logic analyzer is used to determine the error addresses and data.

Data error loop (LEDs 1 and 4 on; LEDs 2 and 3 off):

```
8000A4:   MOVW D1, (A0)
8000A6:   MOVW (A0), D0
8000A8:   JMP 8000A4
```

Both error addresses found (LEDs 4, 3, 2 on; LED 1 off):

```
8001EE:   MOVB (A4), D0
8001F0:   MOVB D3, (A4)
8001F2:   MOVB (A1), D0
8001F4:   MOVB D7, (A1)
8001F6:   BRA 8001EE
```

Second error address not found (LEDs 4, 3, 2 on; LED 1 off):

```
800202:   MOVB (A1), D0
800204:   BRAL 800202
```

Debugger Program

The debugger is a breakpoint monitor program. With the debugger, the user can modify memory and run programs. That is, the user can enter and run programs, disassemble instructions, and set breakpoints. The debugger includes a load command that allows downloading diagnostics into memory from another computer. This can be used to load diagnostics from a machine that can load diagnostics from a floppy into one that cannot.

To set up the debugger program:

1 Replace the boot ROMs with the debugger ROMs.

2 Connect the **UNIX** PC to an unintelligent terminal using the RS-232-C port.

3 Push the Reset button.

The following message appears:

```
S4      MC68010      ROM      DEBUGGER      V1.0

COPYRIGHT 1984 BY CONVERGENT TECHNOLOGIES INC.

SR=XXXXXXXX PC=00800F66 SP=00800F66 UP=XXXXXXXX
D0=XXXXXXXX D1=XXXXXXXX D2=XXXXXXXX D3=XXXXXXXX
D4=XXXXXXXX D5=XXXXXXXX D6=XXXXXXXX D7=XXXXXXXX
A0=XXXXXXXX A1=XXXXXXXX A2=XXXXXXXX A3=XXXXXXXX
A4=XXXXXXXX A5=XXXXXXXX A6=XXXXXXXX A7=XXXXXXXX
00800F66          bras   0x00800F64
DBG>
```

The program counter (PC) shows the address of the next instruction to be executed. SP and UP are supervisory and user stack pointers, respectively, D0-D7 are data registers, A0-A7 are address registers, and SR is the status register.

After the debugger (DBG>) prompt, type **he** (for the Help command). The following message appears, listing the debugger program commands:

```
COMMANDS:  BR, BC, BO, DB, DF, DI, DM, DR, DW, GO, HE, LO,
           MB, MM, MR, MW, WM, WW, WB, TR
```

These commands are entered after the DBG> prompt is displayed. The first group, the breakpoint group, consists of BR and BC. BR followed by an address sets a breakpoint at that address; BC followed by an address clears the breakpoint at that address.

For example, to set and clear a breakpoint at address 40000:

1 After the DBG> prompt, type **br 40000** and press <Return>.

The following message appears:

```
1 BREAKPOINTS SET AT:  00040000
```

2 Type **bc 40000** and press <Return> to clear the breakpcint.

Using the debugger ROMs, you can set and clear as many breakpoints as necessary depending on your application. If you try to clear a breakpoint at an address where one does not exist, the following message appears:

```
NO BPT AT THAT ADDR
```

When a program containing several breakpoints stops at any given breakpoint, type **go** and press <Return> after the DBG> prompt to continue program execution to the next breakpoint.

The next group of commands, the display group, includes DB, DF, DM, DR, and DW. When followed by an address, command DB (display byte) causes the next 16 bytes of data to be displayed. Pressing <Return> causes 16 more bytes of data to be displayed; typing / and pressing <Return> recalls the DBG> prompt.

For example, to display 16 bytes starting at address 40000:

1 After the DBG> prompt, type **db 40000** and press <Return>.

The following message appears, displaying the 16 bytes:

00040000: XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX

2 Press <Return> again.

The screen displays the next 16 bytes:

00040010: XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX

3 Type / and press <Return> to recall the DBG> prompt.

The DF (display all registers) command presents all registers internal to the 68010. To use this command, simply type df and press <Return> to display all registers.

The DM (display 32-bit word) command, when followed by an address, displays the 32-bit word starting at that address. Pressing <Return> gives the next 32-bit word; typing / and pressing <Return> recalls the DBG> prompt.

The DR (display register) command displays each of the 68010 registers just as the DF command does.

The DW (display 16-bit word) command, when followed by an address, displays the 16-bit word located there. Pressing <Return> gives the next 16-bit word; typing / and pressing <Return> recalls the DBG> prompt.

The BO (boot the **UNIX** PC) command first checks to see if there is a bootable floppy inserted. If not, it boots from the hard disk.

The GO command, when followed by an address, causes program execution to begin at that address.

HE is the Help command, which has already been discussed.

LO, the Load command, downloads a program into memory from the RS-232-C port.

The next group of debugger commands, the modification group, includes MB, MM, MR, and MW. With this group, the user can inspect and change the contents of various addresses and registers in the **UNIX** PC.

The MB (modify byte) command, when followed by an address, displays the 16-bit word at that address and enables the user to modify the word. The least significant bit of the address indicates whether the upper or lower half of the 16-bit word can be modified. If it is clear (0), the upper half of the 16-bit word can be modified; if it is set (1), the lower half can be modified. Pressing <Return> allows the next byte to be modified; typing / and pressing <Return> recalls the DBG> prompt.

The MM (modify 32-bit word) command enables the user to modify an entire 32-bit word at a specified address. Pressing <Return> allows the next 32-bit word to be modified; typing / and pressing <Return> recalls the DBG> prompt.

The MR (modify register) command displays each of the 68010 registers so the user can find the one that needs to be modified. It then allows the user to modify that register. Pressing <Return> displays the next register; typing / and pressing <Return> recalls the DBG> prompt.

The MW (modify word) command lets the user modify the 16-bit word at the specified address. Pressing <Return> gives the next word; typing / and pressing <Return> recalls the DBG> prompt.

The write commands, including WM, WW, and WB, enable the user to change the contents of various addresses just as the modify commands do, but the write commands do not display the previous contents. Pressing <Return> lets you write to the next address; typing / and pressing <Return> recalls the DBG> prompt.

The TR (trace trap) command allows single-step execution through the user program. TR followed by an address starts the trace at that address.

Machine Language Programming

Table 4-2 on the next page, contains a few instructions for writing short programs, which can be loaded and run using the debugger program.

Definitions

Dn and An refer to address and data registers inside the 68010, where n is any number from 0 through 7. (An) refers to the contents of memory pointed to by the address in address register An.

Instructions consist of one op code word followed by extension words, if used.

2An in the op code instruction means 2 times the number of the address register. For example, to use the move instruction below (3/2An/8/Dn) to write the contents of data register D2 to the memory location contained in address register A1, the op code is 3282.

Table 4-2 Program Instructions

Machine Code	Register Notation	Description
3/2An/8/Dn	MOVE Dn to (An)	Writes contents of data register Dn to memory location contained in An.
3/2Dn/3/C [ext. word]	MOVE #word to Dn	Loads extension word into data register Dn.
2/2An/7/C [00XX]	MOVEA <ea>, An	Moves the two extension words into address register An. The first extension word contains the most significant address digits.
5/3/4/Dn	Dn = Dn - 1	Subtract Quick. Decrements data register Dn.
B/2Dn/7/C [immed. data]	CMP <ea>, Dn	Compares immediate data to Dn. Subtracts source operand from the specified data register and sets condition codes according to the result; the data register is not changed.

Table 4-2 Program Instructions (Continued)

Machine Code	Register Notation	Description
6/6/X/X	BNE <label>	Branch not equal. If this instruction is preceded by a CMP instruction (above) and the data words compared are not equal, then program execution continues at a location obtained by adding the displacement represented by <label> and the address of the next instruction. The displacement is a 2's complement form.
6/0/X/X	BRA <label>	Program execution continues at the address of the next instruction plus the displacement. The displacement is a 2's complement form.

To calculate the 2's complement displacement for a BNE or BRA instruction (see preceding table):

1 Count the number of bytes back to the opcode in the instruction that is to be branched to.

2 Convert this number to 2's complement by:

- Writing it as an 8-bit binary number
- Changing each 0 to 1 (binary 1's complement)
- Adding 1 (binary 2's complement)
- Converting to hexadecimal

Walking Ones Test Program

The following program sets a pattern of walking ones in the first 1/2KB of memory:

```
070000: 303C   MOVE 1h to D0
070002: 0001

070004: 307C   MOVEA 0100h to A0
070006: 0100

070008: 3100   MOVE D0, -(A0)  (loop)

07000A: E358   ROL D0

07000C: B0FC   CMPA (compare A0 with 0)
07000E: 0000

070010: 66F6   BNE (branch if not equal to loop)

070012: 4EF9   JMP (return to debugger)
070014: 0080
070016: 0EE0
```

Read/Write Loop Program

The use of an oscilloscope in troubleshooting the logic board is limited because most signals, such as address and data, do not have repeating waveforms that an oscilloscope can synchronize on. This problem can be solved with a short machine language program that continuously reads and writes the same address in an endless loop.

The program in Table 4-3 can be loaded into memory using the debugger memory write (MW) command and executed by typing go followed by the starting address of the program. Once the program is executing, an oscilloscope connected to a chip-select pin being addressed by the program synchronizes easily because the chip-select pulse repeats about every 2 micro-seconds. A second oscilloscope channel is used to measure logic levels on address and data lines when the chip-select is active.

To see how this works, execute the following procedure on a known good logic board. In this example, data 0F00, which turns off the LEDs, is written to address 4A0000.

1 Load the program shown below using the MW command.

Register notation and description are given to aid understanding. Enter address and data only.

Table 4-3 Read/Write Loop Program

Machine Code	Register Notation	Description
070200:303C 070202:0F00	MOVE 0F00	Moves 0F00 to data register D0.
070204:207C 070206:004A 070208:0000	MOVEA.1	Loads address register A0. Two most significant address digits 4A. Four least significant address digits 0000.
07020A:3080	MOVE D0 to (A0)	Writes D0 to memory location in A0.
07020C:3210	MOVE (A0) to D1	Reads contents of memory location in A0 and puts it in D1.
07020E:60FA	BRA	Branch always--returns program execution to data write instruction at 07020A.

2 Verify correct loading by typing **di** (the disassemble command) followed by **070200**, the starting address of the program.

The instructions disassemble as shown below. Press <Return> to show the next instruction.

```
070200: #0F00, D0
070204: #004A00, A0
07020A: MOVEW D0, A0@
07020C: MOVEW A0@, D1
07020E: BRA 0x007020A
```

3 Run the program by typing **go 070200** <Return>.

The program executes and the DBG> prompt disappears from the screen.

4 Observe the waveforms by connecting an oscilloscope to IC27B (pin 11).

5 Stop program execution by pushing the Reset button.

The DBG> prompt reappears.

Note that the starting address to load this program is arbitrary. You can load it anywhere you like in memory without changing the program code. The data word in the instruction at 070202 and the address data in the instruction at 070206 are also arbitrary. The first two digits of the address at 070206 are always set to 0, because it takes two words, or eight hex digits, to load an address that uses only six digits; thus, the first two are not used.

Turn On/Turn Off Program

The following program writes data to the same address twice, so that a device can be turned on and then turned off. A delay of 64K machine cycles occurs between turning on and turning off.

```
303C  MOVE # to D0
XXXX  Immediate turn on data

323C  MOVE # to D1
XXXX  Immediate turn off data

343C  MOVE # to D2
XXXX  Immediate delay data

207C  MOVE address to A0
00XX
XXXX

3080  MOVE D0 to (A0)

5342  Decrement D2

B47C  Compare immediate data to D2
XXXX  Immediate data set to 0

66F8  BNE (branch not equal)

3081  MOVE D1 to (A0)

60F2  BRA (branch always)
```

Parity Test Procedure

This procedure verifies the ability of the parity circuits to generate a level 7 interrupt when a memory location containing bad parity is read. In addition, it tests the ability of the bus status registers to store the address at which the error occurred.

First, the starting address of the debugger is loaded into the level 7 interrupt vector location so that, when the interrupt occurs, the program returns to the debugger. Then the error enable EE+ and PIE+ bits are set high to enable the circuits that generate the interrupt. Next a program that writes bad parity to address 000000 is loaded and run. Then address 000000 is read, causing the parity error circuit to generate a parity error. The parity error causes the address of the error, 000000, to be stored in the bus status registers and a level 7 interrupt to be generated. The bus status registers are then read to verify that they were loaded properly.

1 Set up the interrupt vector by writing the following data to level 7 interrupt vector address locations 00007C and 00007E:

```
00007C: 0080
00007E: 0EE0
```

Note

800EE0 is the starting address of the debugger.

2 Enable errors and interrupts by writing the following data to enable the EE+ and PIE+ bits:

```
E40000: 8000   Sets EE+ high; IC7K (pin 4) goes high
E41000: 8000   Sets PIE+ high; IC7K (pin 5) goes high
```

3 Load the following program, which writes bad parity to address 000000 (register notation and comments are included to aid understanding):

```
070300: 303C   MOVE #00008000, D0
070302: 8000

070304: 4241   CLR, D1

070306: 207C   MOVEA.1 #00E42000, A0
070308: 00E4
07030A: 2000
```

Loads address to which bad parity is written in A1:

```
07030C: 227C   MOVEA.1 #000000, A1
07030E: 0000
070310: 0000
```

Sets BP+ high so that bad parity is written:

```
70312: 3080   MOVE D0, A0@
(Writes D0 to memory location in A0)
```

Writes to test address to cause parity error:

```
070314: 3280   MOVE D0, A1@
```

Resets BP+ low:

```
070316: 3081   MOVE D1, A0@
```

Jumps back to debugger:

```
070318: 4EF9   JMP 0x00800EE0
07031A: 0080   Returns to debugger
07031C: 0EE0
```

4 Run the program by typing **go 70300** <Return>.

The program performs the following steps:

- Writes 8000 to address E42000, which sets the BP+ bit at IC7K (pin 6) high.
- Writes to address 000000, which sets bad parity at address 000000.

- Resets the BP+ bit low so that bad parity is not written to any other address after the program returns to the debugger.
- Jumps back to the debugger.

  The debugger message scrolls back on the screen after the program has run.

5 Use the debugger to read address 00000.

This causes the parity error circuit to latch the parity error signal and load the address at which the error occurred into bus status registers 0 and 1. The parity error also produces a level 7 interrupt, which causes the debugger message to scroll on the screen.

6 Read BSR0 430000 and BSR1 440000.

These bus status registers contain FC00 and 00000, showing that address 000000 is the address that caused the error.

Note

BSR0s least two significant digits contain the two most significant address digits. BSR1 contains the four least significant address digits.

7 Check PERR* bit at IC20E (pin 3), which is latched low.

To repeat the test, press the Reset button to clear the latched bits and repeat the preceding steps, omitting step 3.

Page Status Test

This test verifies proper operation of the map RAM page status bits PS0 and PS1. First, the status of page 0 is set to page present, write-enabled, not read. Then page 0 is read and the page status checked to see that it has changed from not read to read. Finally, page 0 is written to and the page status is checked to verify that it has changed to written.

1 Write A000 to address 400000.

This sets status of page 0.

2 Read but do not write to address 0.

3 Read address 400000.

It contains C000, which indicates that page 0 has been read.

4 Write any data to address 0.

5 Read address 400000.

It now contains E000, which means that page 0 is present and written to.

MMUERR* Test

This test procedure checks the memory management unit error (MMUERR*) signal. In summary, it:

- Declares page 0 not present
- Sets EE+ high to enable the MMUERR* signal
- Reads page 0, producing the MMUERR* signal

To test the MMUERR* signal:

1 Connect a oscilloscope to the bus error pin (22) of the 68010.

2 Write 400000: 0000.

This sets the status of page 0 to not present.

3 Write 8000 to address E40000.

This sets EE+, IC7K (pin 4), high.

4 Read page 0 by reading address 000000.

A negative pulse appears on the oscilloscope, indicating that a MMUERR* signal has generated a bus error to the 68010. The debugger program crashes.

The following list summarizes possible page status conditions:

```
400000: 00000   Page 0 not present
400000: 20000   Page 0 present, not write-enabled, not read
400000: A0000   Page 0 present, write-enabled, not read
400000: C0000   Page 0 present, write-enabled, read
400000: E0000   Page 0 present, written, and read
```

Dummy DMA Test Procedure

The dummy DMA test procedure confirms that the disk bus interface logic can write to a RAM memory location and change its contents. Using the debugger program, follow these steps:

1 Set the contents of memory location 000000 to some arbitrary value--for example, FFFF--by typing **mw0** and pressing <Return>.

The system responds with the following address and its current contents:

000000:4EF9

4EF9 is the current contents of location 0. To change the contents, type **FFFF** and press <Return>. The system responds with:

000000:FFFF

2 Escape from the current location and get back to the DBG> prompt by typing a period and pressing <Return>.

3 When the DBG> prompt appears, write 0 to the following locations (note that, because these are write-only registers, they show FFFF each time they are read):

4d0000	Disk DMA address register
4d4000	Disk DMA address register
4a0000	Miscellaneous control register

4 Write 0 to location 460000, the DMA count register (it shows C000); then write 8001 to it (it shows C001).

You have toggled the least significant bit.

5 Examine location 0; it will now be different.

If dummy DMA fails, check the following three signal groups:

- IDMAR/W, DCNTCS, DCNTWR
- DMAREN, MCRWR, DCNTCS
- DMAEN, DKRQ, DKBG, DMA DATA

Floppy Disk Recalibration Procedure

Refer to schematic page 13 for ICs referred to in this section.

This procedure uses the debugger program to manually check the operation of the floppy disk system when the floppy disk drive cannot boot diagnostics.

The procedure verifies the ability of the disk controller to execute the Restore and Seek commands. Data for these commands is shown in two forms: seek with and without verify. When the verify form is used with a formatted floppy disk, the WD2797 disk controller reads the track number from the disk and compares it with the number in its track register, thus verifying the ability of the controller to read data from the floppy disk. The no verify form can be used with an unformatted floppy disk.

1 Adjust the VCO capacitor and RPW potentiometer.

- Check the 1-MHz clock on the disk controller chip (pin 24).
- Reset the floppy disk controller 22M (pin 5) by setting MR (master reset) low and then setting it back high.

To do this, use the debugger program to write the following data to the disk control register:

Address	Data	MR Status (22M, Pin 19, WD2797)
4E0000	0000	Low (resets disk controller)
4E0000	0080	High (clears reset)

- Ground the test pin.

Put a jumper across the two pins marked E4 and E5 next to the disk controller at 22M. This grounds pin 22, the test pin on the WD2797. For CPU boards numbered 60-00225, jumper E4 and E5 does not exist. Use test clipo to connect 22M pin 22 to ground.

- Adjust the frequency.
- Adjust the read pulse width.

Put an oscilloscope on pin 29 (RG43) and adjust R141 for a pulse width of 500 ns.

Put an oscilloscope on pin 16 (DIRC) and adjust the trimmer cap for a frequency of 250 kHz. After this is done, remove the jumper.

2 Verify that you can write to and read back from controller registers.

Using the following addresses, write data to each address and then read it back to check the disk controller's ability to store data in its internal registers. If this does not work, check the data and the read/write lines at the controller.

Register	Address
Track	E10002
Data	E10006

3 Put a scratch floppy disk in the drive, select the drive, and turn on the motor by writing data FFE0 to address 4E0000.

The motor comes on and FDRIVE0, 17H (pin 6), will be high.

4 Issue the Restore command by writing FF0F (with verify) or FF03 (no verify) to address E10000.

If the head is not at track 0 already, it is driven there. After the command is executed, read the track register (E10002), which shows 0 (FF00).

Check that the controller (pin 39) generated an interrupt when the Restore command was completed.

5 Drive head to track 40 (inside track).

- Load hexadecimal 28 (decimal 40) into the data register by writing FF28 to address E10006.
- Issue the Seek command by writing FF13 (no verify) or FF1F (with verify) to address E10000.

During execution, a step pulse occurs at pin 15 each time the head moves one track.

When the Seek command is completed, the track register (E10002) contains 28 hex and a positive pulse appears on pin 39, indicating interrupt at command completion.

Table 4-4 lists the address, data, and WD2797 controller chip pins checked during this procedure:

Table 4-4 WD2797 Pin Listing

Address	Data	Pin	Notes
E10000	FF03	--	Restore command, no verify
E10000	FF0F	--	Restore command, with verify
E10000	FF13	--	Seek command, no verify
E10000	FF1F	--	Seek command, with verify
E10002	XXXX	--	Track register
E10006	XXXX	--	Data register
4E0000	FFE0	--	Data to select drive and motor
4E0000	FF00	--	Data to deselect drive, motor off
		39	Interrupt at end of command
		15	Step pulses during execution of Seek command

3 Put a scratch floppy disk in the drive, select the drive, and turn on the motor by writing data FFE0 to address 4E0000.

The motor comes on and FDRIVE0, 17H (pin 5), is high.

4 Issue the Restore command by writing FF0F (with verify) or FF03 (no verify) to address E10000.

If the head is not at track 0 already, it is driven there. After the command is executed, read the track register (E10002), which shows 0 (FF00).

Check that the controller (pin 39) generated an interrupt when the Restore command was completed.

5 Drive head to track 40 (inside track).

- Load hexadecimal 28 (decimal 40) into the data register by writing FF28 to address E10006
- Issue the Seek command by writing FF13 (no verify) or FF1F (with verify) to address E10000.

During execution, a step pulse occurs at pin 15 each time the head moves one track.

When the Seek command is completed, the track register (E10002) contains 28 hex and a positive pulse appears on pin 39, indicating interrupt at command completion.

Floppy Disk Read Track Procedure

This procedure uses the debugger program to read information from a floppy disk for analyzing problems involving booting from the floppy disk drive.

1 Initialize the system by loading the following data at the addresses shown:

4A0000:	0	Sets DMA R/W* bit low (see sheet 16 of schematics)
4E0000:	0	Resets the floppy disk controller
4E0000:	E0	Selects the floppy disk drive and turns on the motor

This step selects the disk drive and sets up the disk PAL for a floppy disk write to memory from disk. This has to be done only once unless you turn off the power or push the Reset button.

2 Load DMA address and word counters by writing the following data to the addresses shown:

4D0000:	0	
4D4000:	0	Sets starting address at 0
460000:	0	Shows C000 when 0 is written to this address
460000:	8000	

This enables a DMA write to memory. The transfer to memory starts from address 000000 in RAM.

Note

460000 shows C001 after you write 8000 to this address. This indicates that the DMAEN+ bit is set and the first byte has been moved. If you read address 0 at this point, it is different. These counters should be set up before any command that reads or writes information from a track or sector.

3 Issue the Read Track command to read either side 0 by writing E0 or side 1 by writing E2 to address E10000.

This causes track 0, side 0 or side 1, to be moved into the first 8KB of memory. The highest address that contains data from the track is about 187A (hex). The disk controller starts reading from the first index pulse after the command is issued and continues until the next index pulse.

IBM System 34 Format

Number of Bytes	Hex Value of Byte Written
80	4E
12	00
3	F6 (writes C2)
1	FC (index mark)
50	4E
----*	
12	00
3	F5 (writes A1)
1	FE (ID address mark)
1	Track number (0-4C)
1	Side number (0 or 1)
1	Sector number (1-1A)
1	01 (sector length)
1	F7 (2 CRCs written)
22	4E
12	00
3	F5 (writes A1)
1	FB (data address mark)
256	DATA
1	F7 (2 CRCs written)
54	4E
----*	
598	4E

* Bracketed field is written 26 times.

4 Examine memory (see preceding table) starting at address 0 and look for the pattern of 80 4Es and 12 00s that precedes the index address mark of FC.

Note

The 4E pattern may look like 27, 96, or 39 at first, because the disk controller has not yet synchronized its data separator, which tells it which bit a byte starts with.

6 Next find the ID fields for the first sector. Look for an A1A1FB pattern.

The next byte after FB is the first byte of the six bytes in the ID field. The contents of the fields are:

Byte 1: Track address 0-40 (decimal)
Byte 2: Side number 0 or 1
Byte 3: Sector address 1-26 (decimal)
Byte 4: Sector length code
Byte 5: CRC
Byte 6: CRC

Seek Command

If you want to modify the preceding procedure to read a different track, simply load the data register with the track you want and issue the Seek command before issuing the Read Track command. Note that there are two forms of the Seek command: with and without verify. With verify, the command reads the track information from the disk and compares it with its internal track register. The maximum track number is 28 hex.

Read Address Command

If you want only to read the ID field, you can substitute the Read Address command for the Read Track command. With this command, you can read the first ID field after the index pulse. Because this command writes only six bytes, you can repeat the command without reloading the DMA counters. The data is six bytes higher in memory because the address counter advances with each execution.

Summary of addresses and data used in the foregoing procedure:

4A0000:	0	Sets DMA R/W*
4E0000:	0	Resets controller
4E0000:	E0	Selects controller and turns on motor
4D0000:	0	DMA address counter
4D4000:	0	DMA address counter
460000:	8000	DMA word counter, DMAEN+, IDMAR/W*
E10002:		Track register
E10004:		Sector register
E10006:		Data register

Summary of commands used to write to the floppy disk controller command register at address E10000:

1B:	Seek, no verify
1F:	Seek, with verify
E4:	Read Track side 0
E6:	Read Track side 1
C4:	Read Address side 0
C6:	Read Address side 1
8C:	Read Sector side 0
8E:	Read Sector side 1
AC:	Write Sector side 0
AE:	Write Sector side 1

Guidelines for using commands:

All commands:	Set DMA R/W*, reset controller, and select drive and head before executing first command and after any reset.
Seek:	Load data register with desired track number first.
Read Track:	Set up DMA counters first.
Read Address:	Set up DMA counters first.
Read/Write	Sector:Set up DMA counters and load sector number into sector register first.

Sector numbers start at 1. Tracks are 0-40 decimal or 28 hex.

Floppy Write Sector Procedure

This procedure writes data from the lowest 1/2 KB of memory to sector 1, track 0, side 0 of the floppy disk. Enter the following data at the addresses shown:

4A0000:	4000	Sets DMAR/W- high to read memory and write to disk
400000:	0	Resets the floppy disk controller
4E0000:	E0	Selects floppy drive 0 and turns on motor
4D0000:	0	A1-A8 correspond to least significant 8 bits of DMA address
4D4000:	0	Sets starting address to 0; A1-A13 correspond to most significant 13 bits of DMA address
460000:	0	Shows C000
460000:	C000	Sets DMAEN D15 and IDMAR/W high; initializes DMA word count to 0 E10000: AC. Writes sector side 0

Initialize Floppy Program

This machine language program initializes the DMA bus interface and floppy disk controller to read sector 1, track 0, side 0 from a formatted floppy disk. To use the program, perform the following steps:

1 Load the program as shown below. Verify that it is entered correctly using the debugger DI command.

Note

The program is stored in video memory so that it is not overwritten by disk DMA.

```
420008:   33FC
42000A:   0000 Resets disk drive
42000C:   004E
42000E:   0000

420010:   33FC
420012:   00E0 Selects disk drive
420014:   004E
420016:   0000

420018:   33FC
42001A:   0000 Sets DMA address least significant
               bits to 0

42001C:   004D Uses address bits A1-A8
42001E:   0000

420020:   33FC
20022:    0000 Sets starting address most
               significant bits to 0

420024:   004D Uses address bits A1-A13
420026:   4000

420038:   4EF9
42003A:   0080 Returns to debugger
42003C:   0EE0
```

Note

This program starts storing data from sector 1 at RAM address 000000. You can change the starting address by changing the address data shown at 42026. For example, to change the starting address to 70000, change data for address 420026 to 4700. The third digit of this word is the fifth digit of the DMA address.

2 Execute the program by typing go 420000.

The debugger message scrolls on the screen.

3 Read sector 1 by writing 8C (the Read Sector command) to E10000.

4 Examine memory to see if the command has been executed.

Note

If you issue the Read Sector command a second time, the sector is read again but it is stored in the next 1/2 KB word of memory. You can fill as much memory as you like by repeating the Read Sector command. Each time the Read Sector command is repeated, the DMA word counter at 460000 increments by 1/4 KB.

DMA Looping Program

The following machine language program uses a loop instruction to test the DMA bus interface. It produces a continuous disk bus request, grant, and acknowledge waveform. The program fills the 2 MB of memory that can be accessed by DMA with the same data.

```
420100:   33FC
420102:   0000
420104:   004A          Sets DMA R/W bit low to write
                        to memory
420106:   0000

420108:   33FC          Sets DMA enable low
42010A:   0000
42010C:   0046
42010E:   0000

420110:   33FC          Toggles DMA enable high
420112:   8000
420114:   0046
420116:   0000
420038:   60EE          Branch always
```

Hard Disk Registers

This section provides general information that is to be used in the procedures involving the hard disk drive. Addresses 4A0000 and 4E0000 are the addresses of the disk control and miscellaneous control registers. These registers must be set up before any command is written to the disk controller.

Summary of addresses and data written to the disk control and miscellaneous registers:

4A0000:	0000	Sets bit 14 low for DMA read disk and write to memory
4A0000:	8000	Sets bit 14 high for DMA read memory and write to disk (see sheet 15 of schematics for circuits)
4E0000:	FF00	Resets hard disk controller (see sheet 11)
4E0000:	FF1F	Selects drive 0, head 0 (see sheet 11) and clears reset

To set the DMA address counter to start a DMA transfer at memory address 0, write 0 to addresses 4D0000 and 4D4000. Do this before each Read or Write Sector command.

To set the DMA word counter and enable DMA write, write the following data to the address shown:

460000:	0	Sets enable low
460000:	8000	Sets DMAEN high and selects DMA write

To initialize the WD1010 internal registers, write the following data to the addresses shown:

E00004:	01	Sector count (number of sectors to be transferred)
E00006:	0000	Sector number (sector to be transferred)
E00008:	FF64	Cylinder, least significant 8 bits
E0000A:	FF02	Cylinder high (only bits 0 and 1 are used)
E0000C:	20	Size drive head (SDH), head 0, drive 1, 512 bytes/sector

WD1010 Command Register (E0000E)

Table 4-5 lists the WD1010 command register bit definitions:

Table 4-5 Bit Number

Command	7	6	5	4	3	2	1	0
Restore	0	0	0	1	R3	R2	R1	R0
Seek	0	1	1	1	R3	R2	R1	R0
Read Sector	0	0	1	0	I	M	0	T
Write Sector	0	0	1	1	0	M	0	T
Scan ID	0	1	0	0	0	0	0	T
Write Format	0	1	0	1	0	0	0	0

R3-R0 = Step rate for head movement
F = 7.5 ms
0 = 35 us

I = Interrupt enable
I = 0 Interrupts at BDRQ time
I = 1 Interrupts at end of command

M = Multiple sector flag
M = 0 Transfers one sector
M = 1 Transfers multiple sectors

T = 0 Enables retries
T = 1 Disables retries

Commands in Hexadecimal

The left or most significant digit selects the command. The right or least significant digit selects the options described in the command register table above and in the following tables.

Restore = 1X, where X is a value from 0 through F depending on step rate
Seek = 7X

Read Sector = 2X

Table 4-6 Read Sector Commands

Read Sector	I	M	T
20	0	0	0
21	0	0	1
24	0	1	0
25	0	1	1
28	1	0	0
29	1	0	1
2C	1	1	0
2D	1	1	1

Requirements to successfully complete the Read Sector command:

- Ready input must be asserted from the disk drive.
- Seek complete must be asserted at the end of a Seek command. A rising edge on this input informs the WD1010 when head settling time has expired.
- Cylinder and head numbers must match.
- No CRC error or bad block must be detected.
- Data address mark must be found.

Write Sector = 3X

Table 4-7 Write Sector Commands

Write Sector	M	T
30	0	0
31	0	1
34	1	0
35	1	1

Scan ID = 40 or 41, where 40 enables retries up to 10 revolutions. 41 aborts after 2 revolutions of drive.

Table 4-8 SDH Byte Register (E0000C, Read/Write)

SDH Byte Register

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

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

Bit Number			Size Selected
6	5		
0	0		256 bytes/sector
0	1		512 bytes/sector
1	0		1024 bytes/sector
1	1		128 bytes/sector

Bit 7, the extension bit, extends the data field by seven bytes when ECC codes are used. CRC is not appended to the end of the data field. The SDH byte is written into the ID field during format.

Write Precompensation Register (E00002, Write Only)

The value 00-FF loaded into this register is internally multiplied by 4 to specify the cylinder where RWC is asserted. A value of FF always causes RWC to be low, no matter what cylinder number is used.

Error Register (E00002, Read Only)

Bit 7: Bad block detect, used for bad sector mapping.
Bit 6: CRC error.
Bit 5: Forced to 0.
Bit 4: ID not found. This bit is set to indicate that the correct cylinder, head, sector number, or size parameter could not be found or that CRC error occurred on the ID field. This bit is set on the first failure and remains set even if the error is recovered on a retry. When recovery is unsuccessful, the error status bit is set also.
Bit 3: Forced to 0.
Bit 2: Aborted command. This bit is set if the command is issued while the DRDY is deasserted or the WF is asserted. The aborted command bit is also set if an undefined command code is written into the command register.
Bit 1: Track 0 error. This bit is set only by the Restore command. It indicates that TK000 has not gone active after the issuance of 1KB of stepping pulses.
Bit 0: Data address mark not found. This bit is set during a Read Sector command if the data address mark is not found after the proper sector ID is read.

Status Register (E0000E, Read Only)

Bit 7: Busy. This bit is set whenever the WD1010 is accessing the disk. Commands should not be loaded into the command register while the busy is set.
Bit 6: Ready.
Bit 5: Write fault.
Bit 4: Seek complete.
Bit 3: Data request.
Bit 2: Reserved, forced to 0.
Bit 1: Command in progress.
Bit 0: Error. This bit indicates that a nonrecoverable error has occurred. If the host reads the status and finds this bit set, it must read the error register (E00002) to determine the type of error written into the error register.

Hard Disk Recalibration Procedure

The following procedure uses the debugger program to verify the ability of the hard disk controller to execute the Restore and Seek commands and the ability of the data separator to produce a proper clock and data stream.

1 Write the following data in the sequence listed using the debugger program:

```
Disk control register, reset:          4E0000: FF00
Disk control register, select drive:   4E0000: FF1F
```

2 After writing this data, verify the following signal conditions:

- o At chip 17K (pin 9), a logic low indicates that drive 0 has been selected.
- o At chip 17K (pin 3,5 & 7), a logic low indicates that hard disk drive head 0 has been selected.
- o At chip 18K (pin 5), index pulses are present.
- o At chip 18K (pin 11), a logic low indicates that the drive is ready.
- o At chip 16M (pin 9), separated data from the data separator is present.
- o At chip 13K (pin 3), data from the hard disk drive is present.

3 Issue the Restore command by writing FF1F to address E0000E.

This command causes the stepping motor on the drive to bring the head back to cylinder 0 if it is not already there.

After this data is written to the WD1010 command register, the data you read back on the screen is FF50. This is the contents of the disk controller's status register, and it indicates that D6, the ready bit, and D4, the seek complete bit, are both high. When D6 is high, it indicates that the disk controller is receiving a logic high on its RDY pin 28. When D4 is high, it indicates that a seek initiated by the last command has been completed successfully.

4 Write the following reset and drive select data to the disk-control register to reset the controller:

```
Disk control register, reset:          4E0000: FF00
Disk control register, select drive:   4E0000: FF19 (head 1)
```

5 Load the cylinder register as shown below (these are read/write registers, so you should read back the same data you write):

Cylinder number low (614): E00008: FF64

This register holds the eight least significant bits of the cylinder number.

Cylinder number high (614): E0000A: FF02

Bits 0 and 1 are the two most significant bits of the cylinder number.

6 Issue the Seek command, which causes the stepping motor to rotate until the head is at cylinder 614.

Seek command: E0000E: FF7F

Procedures for Reading/Writing Data to Any Sector

Using the debugger program, the next two procedures make it possible to write any desired data to any selected sector on the hard disk and read it back to verify that it was written properly. These procedures use the Read Sector, Write Sector, Seek, and Scan ID commands of the WD1010 controller. The error register bits generated by the WD1010 are listed as well as the signals to check when a board failure prevents these procedures from executing properly.

Read Data to Any Sector

1 Set the DMA address counter:

```
4D0000: 0
4D4000: 0
```

2 Set DMA write, select the drive, and restore the head to track 0:

```
4A0000: 0    Sets bit 14 low, DMA write to memory from disk
4E0000: 0    Resets the disk controller
4E0000: 19   Selects drive 0, head 1
E0000E: 1F   Restore command, returns the head to track 0
```

3 Set the DMA word counter and toggle DMA enable:

```
460000: 0       Reads C000
460000: 8000    Reads C001
```

4 Clear memory by putting 0s in the first 16 words of memory at addresses 0h-2h and 1F0-1FF.

5 Load the hard disk controller registers by writing the following data to the addresses shown:

E00002:	0	Write compensation register
E00004:	1	Sector count register (selects the number of sectors to read)
E00006:	0	Sector number register (selects the sector to read)
E00008:	FF	Least significant byte of cylinder number (cylinder 255 selected as an example)
E0000A:	0	D0 and D1 are the most significant bits of the cylinder number

6 Issue the Seek command by writing 7F to address E0000E.

You read back D2 and the stepping motor rotates. D2 indicates that the command has executed with no errors.

7 Issue the Scan ID command to update the SHD register by writing 41 to address E0000E.

You read back D2, indicating successful completion.

8 Check the SDH register by reading address E0000C.

It contains FF21, which indicates that 512 bytes per sector and head 1 have been selected.

9 Write 28 (Read Sector command) to address E0000E.

It reads back D2. Check the contents of memory to see that the data has been moved from disk. If you do not read D2 after issuing this command, read E00002, the error register, and determine the error. If you write 28 to E0000E again, the command is executed again, but the data is transferred to the next 1/2 KB of memory from address 000200 to 000300. Each time the command is repeated, the DMA count register at 460000 shows another 1/2 KB of data moved. After the first execution, it shows C101; after the second execution, it shows C201, and so on.

Write Data to Any Sector

This procedure can be used to write data to any selected sector of the hard disk drive from the first 1/2 KB of memory. To execute the procedure, use the debugger program to write the data to the addresses shown.

1 Set the DMA write bit, select the drive, and restore the head to track 0:

4A0000:	4000	Sets bit 14 high, DMA write to disk from memory
4E0000:	0	Resets disk controller
4E0000:	19	Selects drive 0, head 1
E0000E:	1F	Restore command

2 Set the DMA address counter to 0:

4D0000:	0
4D4000:	0

3 Set the DMA word counter and toggle DMA enable:

460000:	0	Reads C000
460000:	C000	Reads C001, bit 15 is DMAEN, bit 14 is R/W*

4 Load an arbitrary data pattern in the first 16 words of memory at addresses 0h-2h.

5 Load the hard disk controller registers:

E00002:	0	Write compensation register
E00004:	1	Sector count register (selects the number of sectors to read)
E00006:	0	Sector number register (selects the sector to read)
E00008:	FF	Least significant byte of the cylinder number (cylinder 255 selected as an example)
E0000A:	0	D0 and D1 are most significant bits of the cylinder number

6 Issue the Seek command:

E0000E:	7F

It reads back D2 and the stepping motor rotates. D2 indicates that the command has executed with no errors.

7 Issue the Scan ID command to update the SDH register:

E0000E: 41

It reads back D2, indicating successful completion.

8 Check the SDH register by reading E0000C.

It contains FF21, which indicates that 512 bytes per sector and head 1 have been selected.

9 Issue the Write Sector command by writing 35 to address E0000E, and then use the read sector procedure above to verify that the data has been moved to disk.

After the Write Sector command has been issued, address E0000E shows DA if the command executes without error. If it does not execute properly, read address E00002, the error register, to determine the error.

Write fault and DRDY lines are checked by the controller during command execution.

Signals to Check if Procedures do not Execute

DRDY (IC21H, pin 28) must be asserted after the drive is selected in step 1.

SC (IC21H, pin 32) must be asserted from the drive when a Seek command is completed.

Unseparated data from the drive must appear at IC13K (pin 3) and separated data must appear at IC16M (pin 9) during a Read Sector command.

WFAULT (IC21H, pin 30) must not be asserted from the disk drive.

INTRQ (IC21H, pin 3) must be asserted at the end of a command and must be latched into the line printer status register at IC15P (pin 15).

TK000 (IC21H, pin 31) must be asserted after the Restore command is executed.

INDEX pulses (IC21H, pin 29) from the drive must appear after the drive is selected.

Disk RE (IC21H, pin 6) and WE (IC21H, pin 7) must be asserted during command execution.

TFER (IC24H, pin 18) must occur for each byte transferred. For every other TFER, there must be a DKBG (IC25B, pin 17) and a DKBGA (IC26F, pin 8).

Bit D0 of the WD1010 status register at address E0000E must be low after any command is issued. If it is not, read E00002, the error register, to determine the error.

Hard Disk Data Separator Test

This test verifies proper operation of the hard disk data separator and its component circuits. The test requires a running system with debugger PROMs installed, a serial terminal connected to the serial port, an oscilloscope, and a frequency counter. The hard disk data separator is shown on sheet 12 of the schematics.

The circuits that make up the data separator are:

- Voltage-controlled oscillator (output at IC14N, pin 10)
- Phase detector (outputs at IC18M, pins 11 and 8)
- Loop filter (output at IC17N, pin 1)
- Data run one shot (output at IC13N, pin 13)
- Data separation PAL (outputs at IC14M, pins 19 and 12)

To perform this test:

1 Select the hard disk drive.

Before the data separator can be tested, the hard disk drive must be selected to provide 5-MHz data and clock input to data separation PAL IC14M (pin 3). To select the drive and cause the drive to send data, push the Reset button and then use the debugger program to write 19 hex to address 4E0000. This causes pins 9 and 3 of IC17K (shown on sheet 11) to go low. These signals are sent to the hard disk drive, which responds by turning on its LED and generating the following signals (shown on sheet 11):

- DRDY from the drive goes low at IC18K (pin 11)
- INDEX pulses appear at IC18K (pin 5)
- TK000 goes low at IC18K (pin 3)
- SC goes low at IC18K (pin 13)

2 Check the data separator input and output.

Use a oscilloscope to check the input to the data separation PAL on sheet 12 at IC14M (pin 3). It contains 5-MHz data and clock pulses coming from the hard disk drive. Then check the output of the data separator at IC16M (pin 9), which shows 5 MHz. The DRUN signal at IC13N (pin 13) is high. RCLK at IC14M (pin 12) shows 5 MHz.

If these results are not obtained, test the individual circuits that make up the data separator using the following procedures (steps 3-5):

3 Check the voltage-controlled oscillator.

The voltage-controlled oscillator is IC14N. First check pin 11, the enable input; it is low to enable the oscillator. Next ground pin 1, the voltage control input, forcing the output frequency to a minimum of 3-4 MHz. Then connect pin 1 to 5 volts, forcing the output frequency to a maximum of 17-18 MHz.

4 Check the pullup and pulldown circuits and the loop filter.

To test pullup, ground pin 11 of IC18M, the pullup circuit. This saturates Q7. Its collector goes to +5 volts, causing the inverting input of IC17N (pin 6) to be +5 volts and the noninverting input to be +3.75 volts. This +3.75 volts results from the voltage drop across R67 to the voltage divider made up of R96 and R97. Thus the output at pin 7 drops to -12 volts. The section of 17N with output at pin 1 is connected as an inverting amplifier with a gain equal to the ratio of R65 to RP24, or about -2, and its input is fed from IC17N (pin 7). With an input of -12 volts, the output is driven to the positive supply voltage of +5 volts. This drives the output of the VCO IC14N (pin 10) to a maximum of about 17 MHz. Enable pin 11 must be low for the VCO to work.

To test pulldown, ground the output of the pulldown circuit at IC18M (pin 8). This causes the inverting input at IC17N (pin 6) to drop to 0 and the noninverting input to be +1.25. Thus the output at pin 7 is +5 volts, causing the output of the inverting amplifier to be -12 volts. This output results in -0.66 volts at IC14N (pin 1) of the VCO. The -0.66 volts is the output of voltage divider R64 and RP24. This voltage divider applies 1/3 of the -12 volts from the inverting amplifier output at IC17N (pin 1) plus 2/3 of the +5 volts connected to RP24. The total is -0.66 volts. The result is that the VCO outputs a minimum frequency of 3-4 MHz.

During normal operation, the capacitor C253 provides an integrating action. To check this, connect an oscilloscope on the output at IC17N (pin 7) when the VCO is locked on a 10-MHz output. The input to the VCO is about +2 volts. To obtain this, the output of the integrator at IC17N (pin 7) is about -4 volts. Each time a pullup pulse is generated, the voltage across C253 starts to swing negative at a rate calculated by dividing the current through R68 by the value of C253, or about 2.5 mA/0.01 uFd = 250,000 volts per second. Since typical pullup or pulldown pulses last only 10-20 ns, the output can change by about 5 mV for each pullup or pulldown pulse.

5 Check the phase detector.

The phase detector has a 5-MHz input from IC14M (pin 1) of the data separator PAL and a 5-MHz input from IC14M (pin 19). The 50-ns delay circuit together with the data flipflop output IC15K (pin 9) generates a 25-ns pulse at IC17M (pin 11). Pulses of 10 MHz are applied to IC16M (pin 11).

Dialer Test

The following addresses and data turn on the tone generator in the dialer chip and send tones to the speaker in the monitor. To turn the tone off, write the second address twice.

4B0560:	0	
4B0800:	0	Turns on maximum volume ringing
4B0500:	0	
4B0802:	0	Dual tone to speaker
4B0540:	0	
4B0802:	0	Single tone to speaker
4B0548:	0	
4B0802:	0	Single tone to speaker

Waveform Timing Analysis Procedure

This procedure checks signal generation and signal timing during various bus cycles of the logic board.

Special tools required include:

- HP1630D logic analyzer
- Piggyback extender cables
- Floppy diagnostics disk

Logic Analyzer Setup

1 Disassemble the **UNIX** PC to gain access to the logic board.

2 Install external cables to the piggyback board (if present).

3 Set up the HP1630D logic analyzer as follows:

- Apply power by depressing the line switch in the upper-right corner. The logic analyzer comes up in the system mode.
- In system mode, select timing mode by moving the down-arrow cursor key to the second line (- 16).
- Depress the format key and create the following labels:

LABEL	POL	POD1	POD0
1PCK	+	....	.......*
LDS	+	....	......*.
UDS	+	....	.....*..
DTACK	+	....	....*...
AS	+	....	...*....
R/W	+	....	..*.....
ADDEN	+	....	.*......
DBEN	+	....	*.......
RAMEN	+	.......*	....

- Go to trace mode and set the trace on the AS signal by putting a 0 under AS.

- Connect the signal probes from POD1 and POD0 to the following locations on the logic board:

Signal	Chip	Pin	Schematic
1PCK	14E	15	Sheet 5
LDS	14E	8	Sheet 5
UDS	14E	7	Sheet 5
DTACK	14E	10	Sheet 5
AS	14E	6	Sheet 5
R/W	14E	9	Sheet 5
ADDEN	17F	19	Sheet 5
DBEN	13D	9	Sheet 5
RAMEN	12C	19	Sheet 20

4 When the connections are made and the trace is set, insert the floppy diagnostics disk and power up the system.

5 When the disk has booted, depress the Run button on the logic analyzer.

This causes a waveform to be stored.

Typical Waveforms

The following waveforms show examples of a fast-cycle read operation. The first example shows the timing of AS-, which is approximately 130 ns after the beginning of the machine cycle.

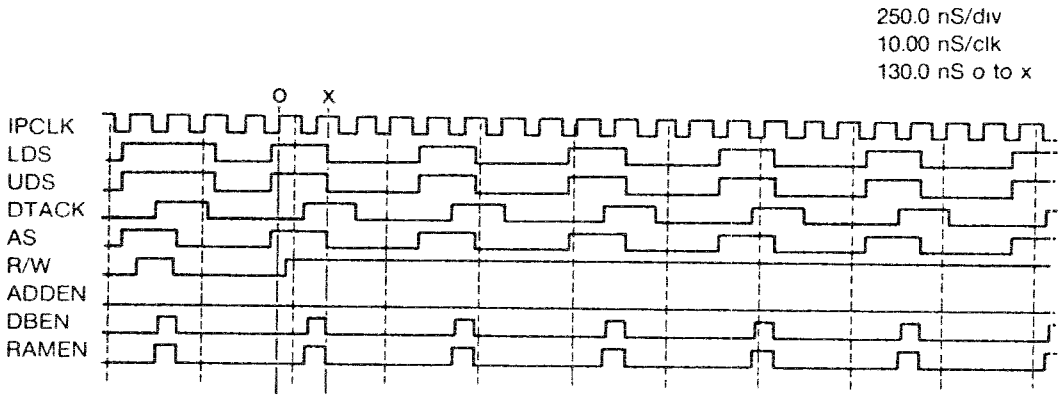


Figure 4-1 Timing of AS- (130 ns)

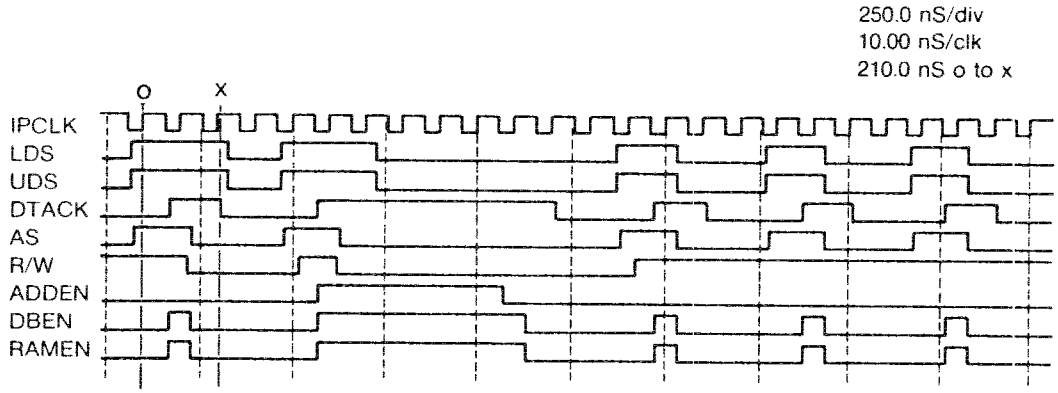


Figure 4-2 Timing of DTACK (210 ns)

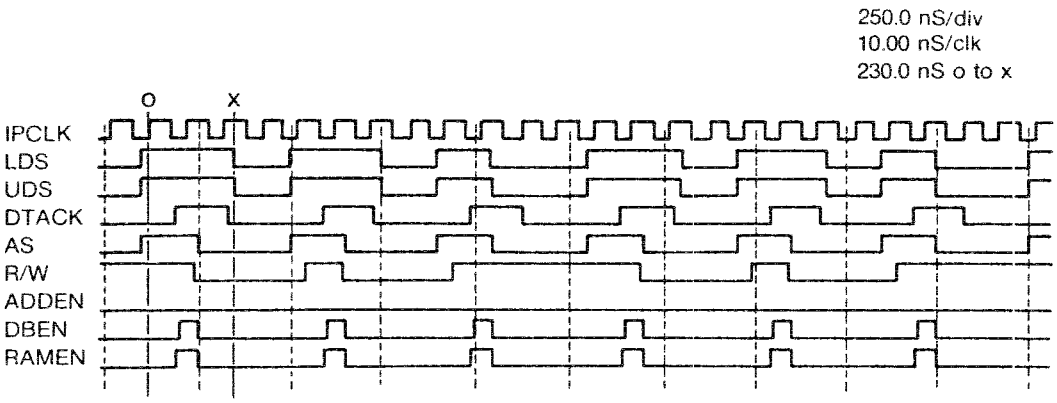


Figure 4-3 Timing of LDS and UDS (230 ns)

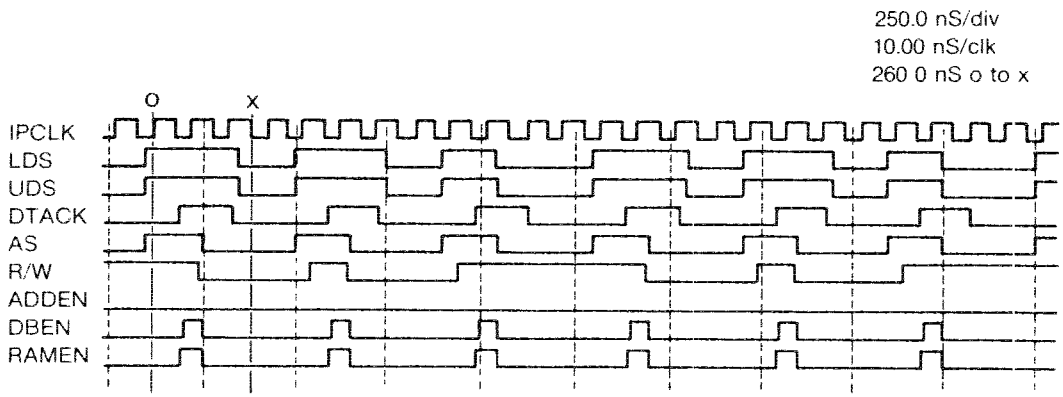


Figure 4-4 DTACK Latched by Processor (260 ns)

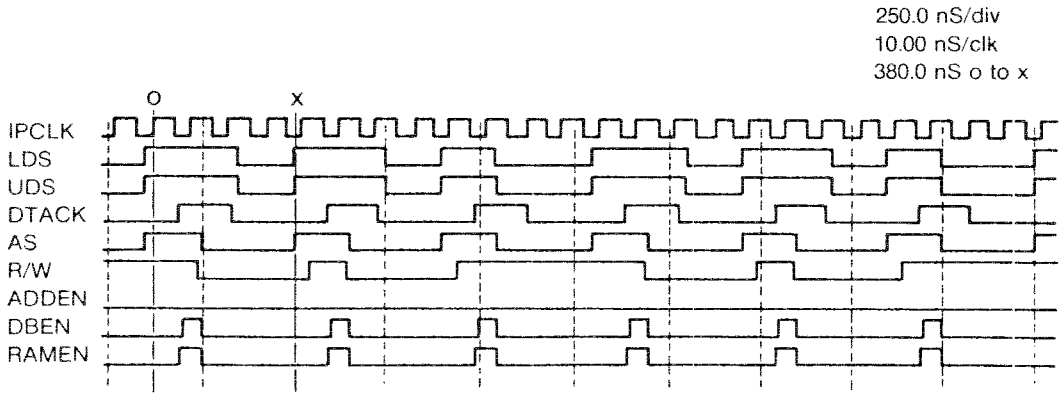


Figure 4-5 AS, LDS, and UDS Deasserted by Trailing Edge of Next 1PCK (380 ns)

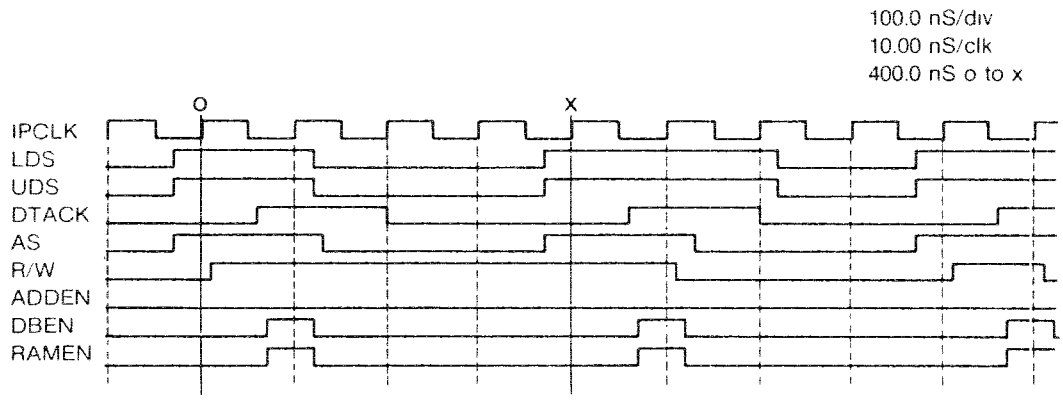


Figure 4-6 Length of Machine Cycle (400 ns)

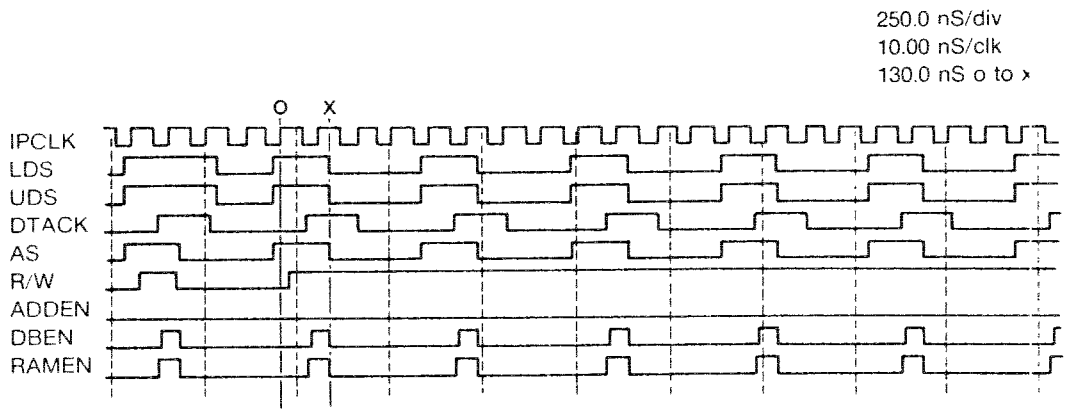


Figure 4-7 Timing of LDS and UDS During a Read Operation (130 ns)

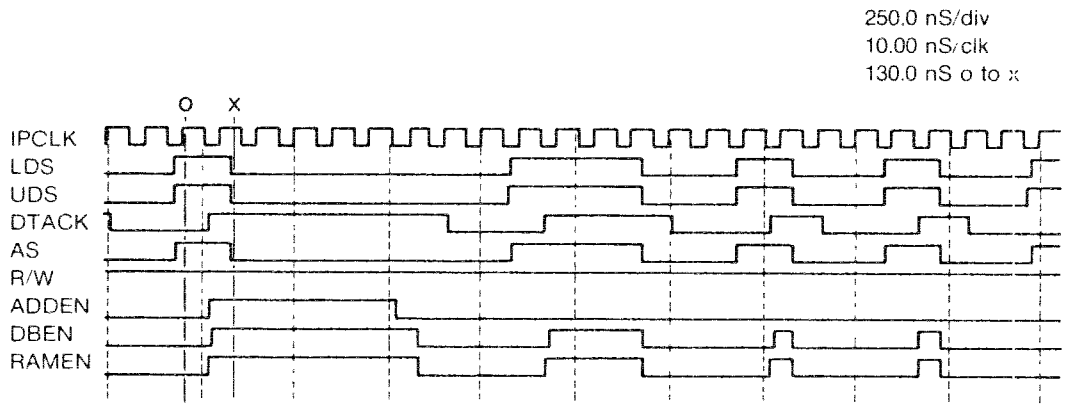


Figure 4-8 Slow Cycle Timing of AS (130 ns)

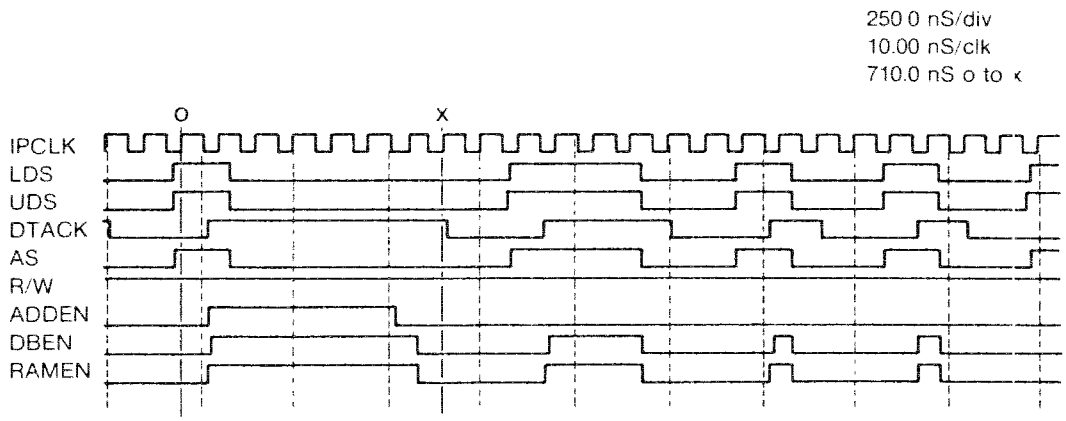


Figure 4-9 Timing of DTACK (710 ns)

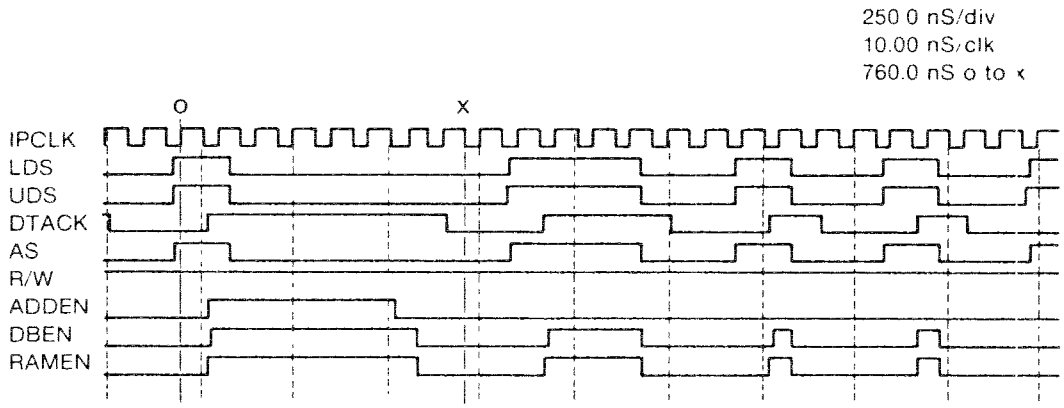


Figure 4-10 DTACK Latched by Processor (760 ns)

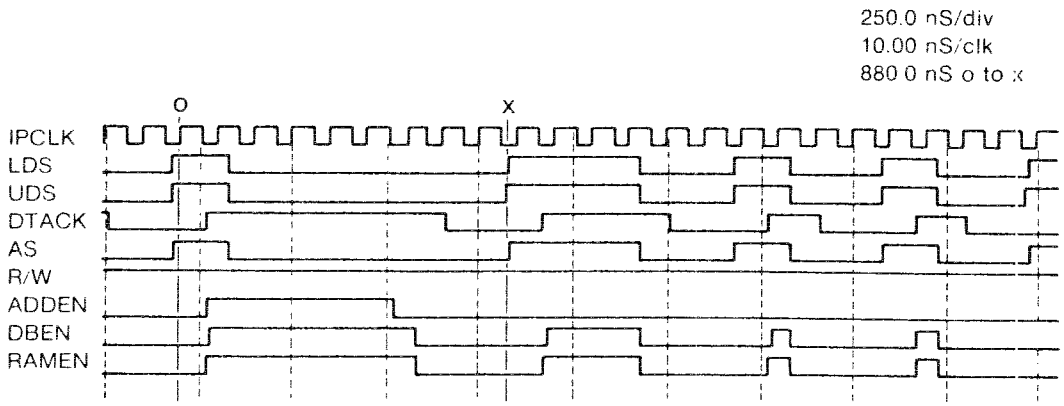


Figure 4-11 AS, LDS, and UDS Deasserted by Trailing Edge of Next 1PCK (880 ns)

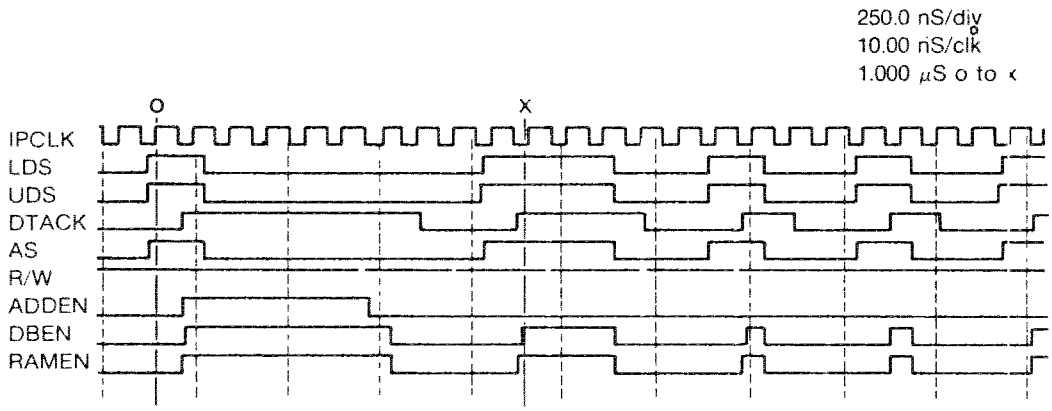


Figure 4-12 Slow Cycle Ends (Complete Cycle = 1.0 us)

68010 Timing Summary

The following waveforms show examples of a 68010 write cycle and a 68010 read-cycle summary.

The 68010 to DRAM write-cycle timing summary as shown in Figure 4-13 is as follows:

1 DTACK goes low before the trailing edge of state 4. This is a high-to-low clock pulse. From this reference point, the rest of the timing states are identified. The complete cycle takes place in 4 clock cycles of 100 ns each. Address line A23 goes low.

2 Address is tristate during S0 and becomes valid during state 1. A23 is valid at the end of state 1.

3 During state 2, the address strobe goes low and r/w (read-write) goes low.

4 When A23 goes low, it causes the following timing events to occur.

- o On sheet 3 of the schematic, 19K pin 6 goes low at next 2 PCK+ going hi. This is the start of state 2.
- o 21E pin 5 goes low generating ENRAS*. The propogation delay 19K and 21E causes ENRAS* going low to be delayed up to 12 ns after the rising edge of S1.
- o 28 to 32 ns after ENRAS* goes low T30* goes low.

5 On sheet 17 of the schematic, when T30* goes low, 3 to 8 ns later the RAS input to the DRAMs goes low. RAS going low strobes the row address from A3 to A11 into the DRAMs. 57 to 63 ns after ENRAS* goes low T60* goes low.

6 On sheet 18 of the schematic, when T60* goes low, the select inputs to the 74F258 multiplexers switch the address input to DRAMs from row address to column address. 4 to 11 ns after T60* goes low, the output of the multiplexers will be valid. Column address comes from the MAP RAMs.

The MAP RAMs receive valid address 2.5 to 6 ns after the start of state 2. The access time of the MAP RAMs is 35 ns.

The MA12-MA21 address output of the MAP RAM is valid at the input to the multiplexers 37.5 to 41 ns after the start of state 2.

7 On sheet 3 of the schematic, 87.5 to 100.5 ns after ENRAS* goes low, 19F9 goes hi. This causes 27F8 to go hi after 27N5 goes low generating DTACK* to the 68010. The propogation delay of the gates generating DTACK and variations in delay adds about 13 to 31 ns to the 90 ns output of the delay line. Thus, DTACK goes low at the 68010 input pin 10 between 100 and 126.5 ns after the start of state 2.

8 On sheet 5 of the schematic, DTACK* arrives at the processor about 24.5 to 50 ns before the falling edge of S4. DTACK* must go low before the falling edge of S4 to prevent the 68010 from inserting wait states. Output data from the processor is valid before the end of S3.

9 On sheet 16 of the schematic, T120* goes low 114 to 126 ns after NRAS* goes low. A low T120* causes 25K pin 6 to go hi which in turn makes 26M pin 11 hi and further 25F18 hi causing ENCAS+ to be hi. The delay of these gates, makes ENCAS+ hi between 13 and 37 ns after T120* goes low. Thus, the CAS input to the DRAMs goes low between 136 and 176 ns after the start of state 2.

Data from the 68010 is stable at the latest by the rising edge of state 4. It takes a maximum of 14 ns to pass data through the 74F245 data buffers. Data from the processor arrives at least 29 ns before it is strobed into the DRAMs by CAS.

10 On sheet 3 of the schematic, T120 going low causes 19F pin 7 to go hi and 28F pin 6 goes hi, k18G pin 8 goes low and 16K pin 13 goes hi. The rising edge of state 6 clocks MMUWREN+ hi. Variation in propogation time causes 16K pin 13 to go hi 122 to 145 ns after the rising edge of state 1.

16K pin 13 is hi a minimum of 55 ns before the rising edge of S6. MMUWREN+ going hi sets 18H4 hi. The falling edge of S7 clocks 18H3 hi resetting MMUWREND+.

11 On sheet 2 of the schematic, during the time that MMUWREN+ is hi, 25C pin 6 is low. This generates MMUWR*. MMUWR* enables tristate buffers 24F which output the new page status for the MAP RAMs.

12 On sheet 16 of the schematic, MMUWR* puts a low on the W* input to the 19C and 20C MAP RAMs. This causes the new page status to be written to the MAP RAMs during S6.

13 On sheet 3 of the schematic, MMUWREN+ hi causes 16K pin 13 to go low. The rising edge of S7 then clocks 18H pin 3 hi setting 21E pin 4 hi. If A23 is low on the next rising clock pulse, the memory cycle can start over.

14 On sheet 2 of the schematic, during a 68010 access to I/O address space, PA22 is hi and CASEN+ is not hi.

68010 to DRAM write cycle summary:

A23 before the leading edge of S2
ENRAS* low 6 to 13 ns after S2s leading edge
T30* goes low 37 to 51 ns after S2s leading edge
DRAM RAS 37-51 ns after S2s leading edge
DRAM ROW ADDRESS 5 to 12 ns after S2s leading edge
DRAM ROW ADDRESS worst case setup time 23 ns
T60* 63 to 75 ns after S2s leading edge
T90* 91.5 to 106.5 ns after S2s leading edge
DTACK* 100 to 126.5 ns after S2s leading edge
T120* 120 to 139 ns after S2s leading edge
DRAM CAS 136 to 176 ns after S2s leading edge
DRAM COLUMN ADDRESS to MUX input 37.5 to 41 ns after S2s leading edge
COLUMN ADDRESS set up before select input to MUX 22
COLUMN ADDRESS TO DRAM 67 to 86 ns after S2s leading edge
COLUMN ADDRESS SET UP before CAS 55 ns (write cycle)
DATA to DRAM 205 to 213 ns after S2s leading edge
DRAM DATA SET UP 29 ns.

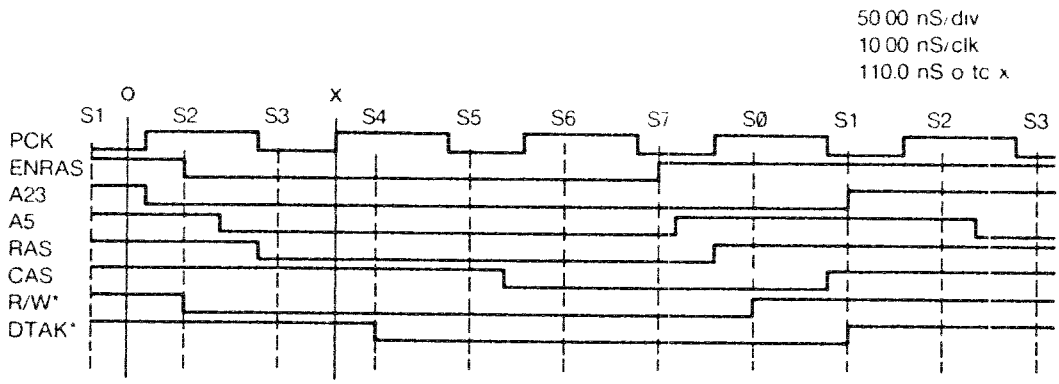


Figure 4-13 68010 to DRAM Write Cycle

The 68010 read cycle timing summary as shown in Figure 4-14 is as follows:

1 During a read cycle CAS to the DRAMs is generated earlier than during a write cycle because data from the DRAM chips is not valid until 60 ns after CAS or 120 ns after RAS.

2 CAS must go low before DTACK to allow time for data from DRAM to be stable at the input of the 68010. During a write, the opposite is true. CAS must not go low until data from the 68010 data bus is stable at the input to the DRAMs.

Read Cycle Summary:

CAS to DRAM 94.5 to 115.5 ns after S2 leading edge.
DRAM read column address set up 14.5 ns.
Data from DRAM valid 156.5 to 189.5 ns.
68010 data read set up time 60 ns.
Minimum row address hold 21 ns - 15 ns chip = 6ns.

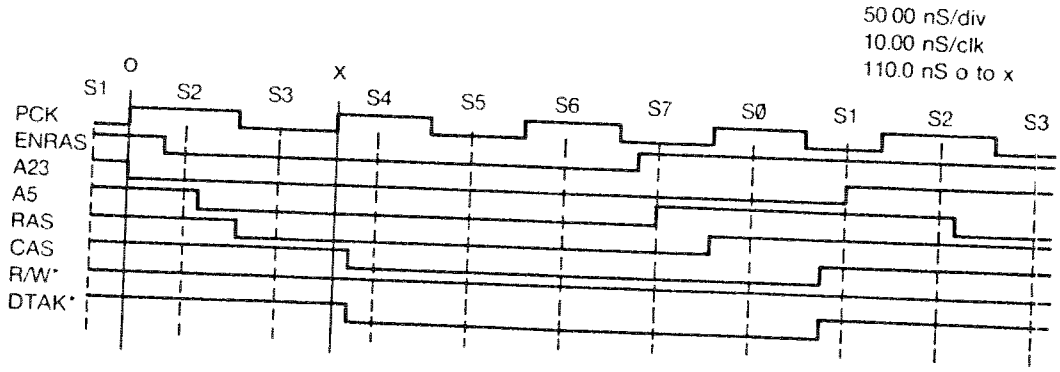


Figure 4-14 68010 Read Cycle Summary