IOtest

Version 1.0 8/25/99
Version 1.1 9/12/99
Version 1.2 Updated for IOtest version 2.31 3/19/01 rbc

1. IOtest Overview

IOtest is a Unix-based software package featuring a suite of tools that enable systems administrators to measure performance, test reliability, and troubleshoot any SCSI disk device. IOtest objectively compares the performance of various storage devices such as hard disks, RAID systems, or solid state file cache systems. IOtest also helps to ensure that all storage devices are installed properly and performing at peak levels. The IOtest software package includes three distinct tools; (1) a benchmarking tool; (2) an exerciser/reliability tool; and (3) a maintenance/diagnostic tool.

Benchmarking Tool

The benchmarking application is a useful analysis tool that measures relative performance of any SCSI disk device. The benchmark test can be run in three different modes: read, write, or a combination of reads and writes. The benchmark test objectively demonstrates performance by measuring transfer rates and I/Os per second while the system is being loaded. This enables systems administrators to run the same tests on a variety of storage devices and compare the results of each to determine which storage devices are providing optimum performance.

Exerciser/Reliability

The exerciser tool tests the reliability of any SCSI disk device by performing random reads and writes across the entire disk using varying transfer sizes and random data. The exerciser tool's purpose is not to measure performance-it is a brute exerciser that checks the reliability of the device by saturating it with I/Os. Periodic

status messages describe the tests that are running and the results, allowing the systems administrator to easily monitor progress.

Maintenance /Diagnostic

The maintenance tool includes a variety of tests for troubleshooting. These tests can be run while the system remains online, enabling the systems administrator to diagnose the state of the disk device without disrupting the system and impacting users. The array of diagnostic tests includes a reliability test, quick read test, write block test, scan block test, hex block dump, and byte editor.

2. Loading and Installation

The IOtest program can be downloaded from Solid Data's ftp server at ftp.soliddata.com. Use "anonymous" for the login, and an email address as the password. CD to the "iotest" directory. Here you will find tarfiles of the current source code packages. Pre-compiled binaries are located in the "binaries" sub-directory. Make sure you transfer in "binary" mode, and make the downloaded binary executable by:

```
#chmod +x IOtestXX-XXX-binary
```

CURRENT VERSIONS
The current versions are as follows:

> IOtest version 2.31: This version uses a menu, and should test devices of any size. It is written in ANSI C, but uses STDC_EXT extensions for large file support.

COMPILING

The IOtest source code should compile using an ANSI "C" compliant compiler such as "gcc."

The following compiler options are supported. Change the "CFLAGS" line in the makefile as needed.

`-D_LARGEFILE64_SOURCE`	Needed to support > 2.1 GB devices.
`-D_NO_ASK_ABOUT_WRITE`	When run on the command line with "-q," the "ask about writing" question is inhibited.
`-D_LOOP_BM`	A question will be asked during the benchmark test startup about looping the test forever. This feature is useful for demos.
`-D_BDRS`	Enables Bus Device Reset test under the Maintenance Menu. **NOTE:** Only under HPUX!
`-D_ALL_SIZES`	Forces the Maintenance/reliability test to use all transfer sizes from 512 to 65536 bytes.
`-D_DO_QUICKIE`	Suppresses transfer size, process count, sync, and block "0" questions. **NOTE:** No menus! Runs only a 8192 byte write and read benchmark.

-D_NEW_OUTPUT_FORMAT Revised benchmark output format for easy spreadsheet loading.

```
Beginning 2 process Read/Write test of /dev/rdsk/c12t0d0s2
Bytes  ET-Secs   IOPS    MB/Sec
  512    0.565  10619    5.310
 1024    0.509   9430    9.430
 2048    0.507   7573   15.147
 4096    0.567   5417   21.672
 8192    0.703   3493   27.948
16384    0.968   2028   32.462
32768    1.418   1107   35.430
65536    2.168    579   37.077
```

STARTING USING THE MENU

```
#./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Tue Mar 27 05:31:29 2001

Sysname: HP-UX    Release: B.11.00    Machine: 9000/800


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number:
```

STARTING USING COMMAND LINE INTERFACE

Read Benchmark	`# ./IOtest –r –d /raw_device_name –p 6`
Write Benchmark	`.# /IOtest –w –d /raw_device_name –p 6`
Read/write Benchmark	`.# /IOtest –a 50 –d /raw_device_name –p 6`
Exerciser	`.# /IOtest –x –d /raw_device_name`
Maintenance Tests	`.# /IOtest –m –d raw_device_name`

IOtest has the following command line options:

-r	;read benchmark
-w	;write benchmark
-a XX	;XX is the percentage of reads vs. writes for the random read/write benchmark
-x	;exerciser
-m	;maintenance tests
-f filename	;logfile name for results
-i iterations	;iteration count for benchmarks, default is 10000
-p process count	;maximum number of processes for benchmark tests

3. README

README file for IOtest version 2.31

This package of benchmarks, utilities, and exercisers should run on any SCSI disk. It's designed to be run under the "root" account on a raw device.

1. IOtest version 2.31: This version uses a menu, and should test devices of any size. It is written in ANSI C, but uses STDC_EXT extensions for large file support.

INSTALLATION
The program consists of four "C" language source code files, and a ".h" file. Edit the ".h" file as required.

COMPILING
Edit the makefile, removing the "#" for the appropriate line. Build the program by typing "make" to build a dynamically linked binary, or "make static" to build a static binary:

COMPILE FLAGS:

-D_LARGEFILE64_SOURCE	Needed for > 2.1 GB support
-D_NO_ASK_ABOUT_WRITE	When run on the command line with "-q," the "ask about writing" question is inhibited.
-D_DEFS	Enables printout of various OS specific flags.
-D_ALL_SIZES	Allows the Maintenance/reliability test to use all transfer sizes from 512 to 65536 bytes.
-D_LOOP_BM	A question will be asked during the benchmark test startup about looping the test forever. This feature is useful for demos.
-DDEBUG	Enables extended messages while the various tests are running.

-D_BDRS	Enables Bus Device Reset test under the Maintenance Menu. **NOTE:** Only under HPUX!
-D_DO_QUICKIE	Supresses transfer size, process count, sync, and block "0" questions. No menus! Runs only a 8192 byte write and read benchmark. This is designed as an "installation verification quick test"
-D_NEW_OUTPUT_FORMAT	Revised benchmark output format for easy spreadsheet loading.

```
#make                        ;(sample output)
gcc -I. -O2 -c IOtest.c
gcc -I. -O2 -c exercise.c
gcc -I. -O2 -c maint.c
gcc -I. -O2 -c benchmks.c
gcc -I. -D NODEBUG -O2 -o IOtest IOtest.o exercise.o maint.o benchmks.o
#
```

The IOtest executable can be copied to /usr/local/bin; or wherever.

RUNNING IOTEST

NOTE: The random write benchmark, read/write benchmark, exerciser, and some of the maintenance tests will WRITE on the device under test, and are guaranteed to destroy all data! IOtest is designed to run on a raw device.

1. IOtest version 2.31 Menu versions

Login as "root" and run the program as follows:

```
#./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Sun Aug 22 12:08:54 1999

Sysname: UNIX    Release: 7.6    Machine: 6350

                Main Menu


Quick Test                          1
Random Read Benchmark               2
```

```
Random Write Benchmark            3
Random Read/Write Benchmark       4
Exerciser                         5
Maintenance tests                 6
Exit                              7


Enter test number:2
```

The program will prompt for the raw device name, and any other parameters.

2. IOtest can also by run from the command line:

Login as "root" and run the program as follows:

```
USAGE: IOtest -a XX|-r|-w|-x|-m|-v -d device -p processes [-f logfile]
          [-i iterations]
```

MANDATORY COMMAND LINE OPTIONS:

```
One of the following must be specified:
-r      read benchmark          random read benchmark
                                (1 to 6 processes)
-w      write benchmark         random write benchmark
                                (1 to 6 processes)
-a XX   read/write benchmark    read/write with adjustable duty cycle
-x      exercise                random read/write exerciser
-m      maintenance             various maintenance tests
                        1 reliability
                        2 quick read
                        3 write block numbers
                        4 scan block numbers
                        5 continuous write/scan block
                        6 block dump
                        7 byte editor
                        8 exit
```

NOTE:
The "-a" "-w" "-x" and "-m" options should only be used by a responsible adult. They will destroy ALL data on the device under test. No checking is done to see if the device is mounted, has a file system on it, or has any valid data whatsoever.

-r: A random read test is run on the drive. In default mode, the system is loaded with 1 to 6 processes. The device must be specified with the "-d" option. Program output can be directed to a file using "-f filename."

Sample output:

```
Beginning 1 process read test on /dev/sdd [10000 iterations]
512  Byte reads ET= 7.145 secs IOs/second = 1399  Data Rate = 0.716 MBs
1024 Byte reads ET= 6.365 secs IOs/second = 1256  Data Rate = 1.287 MBs
2048 Byte reads ET= 6.038 secs IOs/second = 1059  Data Rate = 2.170 MBs
```

-w: A random write test is run on the drive. In default mode, the system is loaded with 1 to 6 processes. The device must be specified with the "-d" option. Program output can be directed to a file using "-f filename." This is a destructive test. All data on the device will be lost.

Sample output:

```
Beginning 1 process write test on /dev/sdd [10000 iterations]

512  byte writes ET= 6.789 secs IOs/second= 1472  Data Rate = 0.754 MBs
1024 byte writes ET= 5.401 secs IOs/second= 1481  Data Rate = 1.516 MBs
2048 byte writes ET= 4.320 secs IOs/second= 1481  Data Rate = 3.034 MBs
```

-a A random read/write test is run on the drive. On the command line after the "-a", you must specify the read/write duty cycle. Valid values are from 10 to 90, indicating the percentage of read vs. writes. For example:

```
./IOtest -a 75 -d /dev/sdd -i 5000 -p 6
```

This will run the read/write test with an initial iteration count of 5000 and a read/write duty cycle of 75%. The 75% means that there is a 75% chance that the next I/O will be a read and a 25% chance that the next I/O will be a write. Due to "randomness", there will be approximately 3750 reads and 1250 writes. The total I/O count will be 5000.

Sample output:

```
Beginning 1 process Read/Write test on /dev/sdd
512  byte rdwr  ET=  6.250 secs  IOs/second= 1600  Data Rate= 0.819 MBs
1024 byte rdwr  ET=  5.658 secs  IOs/second= 1413  Data Rate= 1.447 MBs
2048 byte rdwr  ET=  5.379 secs  IOs/second= 1189  Data Rate= 2.436 MBs
```

-x: The exerciser starts a write/read/compare test on the device under test. Transfer sizes are random between the MIN_TRANSFER_SIZE and

MAX_TRANSFER_SIZE limits. These limits are set in IOtest.h. This test runs until terminated with ^C. All data on the device will be lost. Errors are detected by comparing the data read back from the device to the original write buffer. The bad block in the read buffer is dumped in HEX. The bad block is then re-read from the device and dumped a second time.

-m: The maintenance package includes several functions used to troubleshoot a SCSI disk:

```
#./IOtest -m -d /dev/rdsk/c1t1d0s2
```

These tests will also run on a file:

```
#./IOtest -m -d testfile
```

The write block and reliability tests may extend the size of the file under test to complete the last write.

DETAILED TEST DESCRIPTIONS

TEST 1 RELIABILITY: The drive is written with 0's, read back, and the data compared to what was originally written. The operation is repeated with 1's, A's, and 5's. The test will run until interrupted with ^C. Transfer sizes are dependent on the MIN_TRANSER_SIZE and MAX_TRANSFER_SIZE definitions in IOtest.h. Normally, all transfers are at MAX_TRANSFER_SIZE bytes. To use all values from MIN_TRANSFER_SIZE to MAX_TRANSFER_SIZE, set _ALL_SIZES to "1" and re-compile. To do this, add "-D_ALL_SIZES" to the CFLAGS line in the makefile.

TEST 2 QUICK READ: All blocks on the drive are read one time.

TEST 3 WRITE BLOCK NUMBERS: Every block on the device under test is written with its block number. Each 32 bit word in the block gets written with the same number. Use test 3 in conjunction with test 4 to identify addressing problems. The block number pattern has changed in version 2.31.

TEST 4 SCAN BLOCKS: All blocks on the device are checked for block numbers. The device must be written with TEST 3 before SCAN BLOCKS is run.

TEST 5 CONTINUOUS WRITE/SCAN BLOCK: Alternating write block/scan block sequences.

TEST 6 BLOCK DUMP: This test will dump any block on the device or file in HEX or ASCII. When prompted, enter "n" for the next block, "p" for the previous block, or "q" to quit. Toggle between ASCII and HEX by pressing "a" or "h."

TEST 7 BYTE EDITOR: This test will dump any block in HEX and allow you to change any byte. When prompted, press <enter> or <return> to repeat the same block, or enter a negative block number to quit.

-d device: Use the raw device name for the physical device.

EX:	/dev/rdsk/c1t0d0s2	SUN
	/dev/rdsk/c12t0d0	HP
	/dev/rrz10c	DEC UNIX
	/dev/sdd	LINUX

The device MUST be specified on the command line.

OPTIONAL COMMAND LINE OPTIONS:

-f logfile	Name of a logfile in the current directory for the benchmark results. Handy when you are running on the system console; or when you want to print the results Only the "-r", "-a", and "-w" tests can be logged.
-p processes	Number of processes to load the system. The range is from 1 to 30. If the process number is not specified, it defaults to 6. If you want to go higher than 30, change MAX_PROC_COUNT in IOtest.h, and re-compile. Implemented on "-r" and "-w" only.
-i iterations	The number of iterations for the read and write benchmarks can be set with "-i.” The range is 100 to 500000. The default is set to 10000 in IOtest.h.
-q	Inhibit the "do you want to write" question. If you use this option and write on the wrong drive by mistake, you're on you own.

EXAMPLES:

```
# ./IOtest -r -d /dev/rdsk/c1t0d0s2 -p 20 -f results.log
```

This will run a random read benchmark on /dev/rdsk/c1t0d0s2. The maximum process load will be 20, and a copy of the results will go in ./results.log.

```
# ./IOtest -w -d /dev/rdsk/c1t0d0s2 -i 1000
```

This will run a random WRITE test on /dev/rdsk/c1t0d0s2. The test will size the drive, and test up to (INT_MAX) bytes. The maximum process load

defaults to 6. The iteration count for each test is set to 1000. All data on /dev/rdsk/c1t0d0s2 will be destroyed.

```
# ./IOtest -a 60 -d /dev/sdd1 -i 2000 –p 6
```

This will run the random read/write test on device /dev/sdd1. The initial iteration count is set to 2000. There will be 1200 reads (2000 x 60%) and 800 writes (2000 x 40%). Transfer sizes from MIN_TRANSFER_SIZE to MAX_TRANSFER_SIZE will be used. The process load is 6.

```
# ./IOtest -x -d /dev/sdd
```

This will run the random read/write test on LINUX device /dev/sdd. The entire drive will be tested, and all data on the drive will be destroyed. The test will run until terminated with ^C.

```
# ./IOtest -m -d /dev/sdd
```

This will start the maintenance mode test on /dev/sdd. At this time, reliability, quick read, write block number, scan block number, block dump, and a byte editor are available.

MISCELLANEOUS NOTES:

1. SUN

If you run the write block/maintenance option under Solaris, you will wipeout the label in block 0. You will get the following error when you re-start the test:

```
I/O error (Error opening device RDWR): No such device or address.
```

You MUST re-label the drive using "format" before you can access the drive again. The driver uses the label information when you access the "raw" device.

2. Data General

Don't use the raw device under DGUX. Use the "registered" device name!

3. If you see the following error when compiling with the HP ANSI C compiler, add a "+DAportable" to the cc command line:

```
sut8:/des> make
        cc -I. -c IOtest.c
        cc -I. -c exercise.c
```

```
        cc -I. -c maint.c
        cc -I. -c benchmks.c
        cc -I. -D NODEBUG  -o IOtest IOtest.o exercise.o maint.o
benchmks.o
/usr/ccs/bin/ld: (Warning) At least one PA 2.0 object file (IOtest.o) was
detec.
sut8:/des>
```

4. Try adding "-Ofast=IP27" to the makefile for the IRIX MIPSPro C Compiler. "-n32" can be used to force a 32 bit binary.

4. Detailed Test Descriptions

QUICK TEST

```
#./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Sun Mar 19 09:51:59 2000

Sysname: HP-UX    Release: B.11.00    Machine: 9000/800


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number: 1
Enter device name: /dev/rdsk/c3t0d0
device = /dev/rdsk/c3t0d0
Reading drive to determine size...
Drive size = 2096129 blocks (1073218048 bytes)
  8192 byte read   ET=    0.167 secs  IOs/second =  2994  Data Rate = 23.951 MBs


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number:
```

RANDOM READ BENCHMARK

The random read benchmark sizes the device and does random reads of different transfer sizes under varying process load. The transfer sizes and process load can be set through the menu.

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 16:14:49 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                   1
Random Read Benchmark             2
Random Write Benchmark            3
Random Read/Write Benchmark       4
Exerciser                         5
Maintenance tests                 6
Exit                              7


Enter test number: 2
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter iteration count (100-50000 [10000]: 5000
itercount = 5000
Enter logfile name:
logfile name =
Enter minimum process count [1]:
minimum process count = 1
Enter maximum process count (1-40 [8]):
maximum process count = 8
Enter minimum transfer size (512-65536 [512] bytes):
minimum transfer size = 512
Enter maximum transfer size (512-65536 [65536] bytes):
maximum transfer size = 65536
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)
Starting iteration count = 5000
Transfers will vary between 512 and 65536 bytes

Beginning 1 process random read test on /dev/rdsk/c12t0d0s2

   512 byte read   ET=    0.811 secs  IOs/second =  6165  Data Rate =  3.082 MBs
```

```
  1024 byte read   ET=    0.637 secs  IOs/second =  6279  Data Rate =  6.279 MBs
  2048 byte read   ET=    0.718 secs  IOs/second =  4456  Data Rate =  8.913 MBs
  4096 byte read   ET=    0.657 secs  IOs/second =  3896  Data Rate = 15.586 MBs
  8192 byte read   ET=    0.746 secs  IOs/second =  2745  Data Rate = 21.962 MBs
 16384 byte read   ET=    0.928 secs  IOs/second =  1765  Data Rate = 28.241 MBs
 32768 byte read   ET=    1.282 secs  IOs/second =  1021  Data Rate = 32.698 MBs
 65536 byte read   ET=    1.903 secs  IOs/second =   550  Data Rate = 35.245 MBs

Beginning 2 process random read test on /dev/rdsk/c12t0d0s2

   512 byte read   ET=    0.909 secs  IOs/second = 11001  Data Rate =  5.500 MBs
  1024 byte read   ET=    0.831 secs  IOs/second =  9626  Data Rate =  9.627 MBs
  2048 byte read   ET=    0.830 secs  IOs/second =  7710  Data Rate = 15.421 MBs
  4096 byte read   ET=    0.928 secs  IOs/second =  5517  Data Rate = 22.068 MBs
  8192 byte read   ET=    1.159 secs  IOs/second =  3534  Data Rate = 28.272 MBs
 16384 byte read   ET=    1.605 secs  IOs/second =  2041  Data Rate = 32.657 MBs
 32768 byte read   ET=    2.358 secs  IOs/second =  1111  Data Rate = 35.555 MBs
 65536 byte read   ET=    3.611 secs  IOs/second =   580  Data Rate = 37.148 MBs

Beginning 4 process random read test on /dev/rdsk/c12t0d0s2

   512 byte read   ET=    1.770 secs  IOs/second = 11299  Data Rate =  5.649 MBs
  1024 byte read   ET=    1.619 secs  IOs/second =  9882  Data Rate =  9.882 MBs
  2048 byte read   ET=    1.631 secs  IOs/second =  7847  Data Rate = 15.695 MBs
  4096 byte read   ET=    1.826 secs  IOs/second =  5607  Data Rate = 22.431 MBs
  8192 byte read   ET=    2.297 secs  IOs/second =  3566  Data Rate = 28.530 MBs
 16384 byte read   ET=    3.181 secs  IOs/second =  2059  Data Rate = 32.955 MBs
 32768 byte read   ET=    4.701 secs  IOs/second =  1114  Data Rate = 35.669 MBs
 65536 byte read   ET=    7.209 secs  IOs/second =   581  Data Rate = 37.215 MBs

Beginning 6 process random read test on /dev/rdsk/c12t0d0s2

   512 byte read   ET=    2.692 secs  IOs/second = 11144  Data Rate =  5.571 MBs
  1024 byte read   ET=    2.474 secs  IOs/second =  9700  Data Rate =  9.700 MBs
  2048 byte read   ET=    2.473 secs  IOs/second =  7763  Data Rate = 15.527 MBs
  4096 byte read   ET=    2.761 secs  IOs/second =  5563  Data Rate = 22.252 MBs
  8192 byte read   ET=    3.467 secs  IOs/second =  3544  Data Rate = 28.354 MBs
 16384 byte read   ET=    4.796 secs  IOs/second =  2049  Data Rate = 32.787 MBs
 32768 byte read   ET=    7.070 secs  IOs/second =  1111  Data Rate = 35.575 MBs
 65536 byte read   ET=   10.822 secs  IOs/second =   581  Data Rate = 37.186 MBs

Beginning 8 process random read test on /dev/rdsk/c12t0d0s2

   512 byte read   ET=    3.579 secs  IOs/second = 11176  Data Rate =  5.588 MBs
  1024 byte read   ET=    3.295 secs  IOs/second =  9711  Data Rate =  9.711 MBs
  2048 byte read   ET=    3.299 secs  IOs/second =  7759  Data Rate = 15.520 MBs
  4096 byte read   ET=    3.687 secs  IOs/second =  5554  Data Rate = 22.218 MBs
  8192 byte read   ET=    4.616 secs  IOs/second =  3549  Data Rate = 28.395 MBs
 16384 byte read   ET=    6.394 secs  IOs/second =  2049  Data Rate = 32.790 MBs
 32768 byte read   ET=    9.422 secs  IOs/second =  1112  Data Rate = 35.593 MBs
 65536 byte read   ET=   14.433 secs  IOs/second =   580  Data Rate = 37.177 MBs


                Main Menu
```

```
Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number:
```

RANDOM WRITE BENCHMARK

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 16:18:57 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number: 3
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter iteration count (100-50000 [10000]: 5000
itercount = 5000
Enter logfile name:
logfile name =
Enter minimum process count [1]:
minimum process count = 1
Enter maximum process count (1-40 [8]):
maximum process count = 8
Enter minimum transfer size (512-65536 [512] bytes):
minimum transfer size = 512
Enter maximum transfer size (512-65536 [65536] bytes):
maximum transfer size = 65536
Do you want writes to be synchronous (yes,no [no])?
Writes will be asynchronous

This test will WRITE on device /dev/rdsk/c12t0d0s2.
 Do you want to continue (yes,no [no])?  y
Writing on /dev/rdsk/c12t0d0s2 enabled.
Asynchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)
Starting iteration count = 5000
Transfers will vary between 512 and 65536 bytes

Beginning 1 process random write test on /dev/rdsk/c12t0d0s2
```

```
   512 byte write  ET=  0.819 secs  IOs/second =  6105  Data Rate =  3.052 MBs
  1024 byte write  ET=  0.707 secs  IOs/second =  5657  Data Rate =  5.657 MBs
  2048 byte write  ET=  0.650 secs  IOs/second =  4923  Data Rate =  9.846 MBs
  4096 byte write  ET=  0.657 secs  IOs/second =  3896  Data Rate = 15.586 MBs
  8192 byte write  ET=  0.747 secs  IOs/second =  2741  Data Rate = 21.933 MBs
 16384 byte write  ET=  0.936 secs  IOs/second =  1750  Data Rate = 28.000 MBs
 32768 byte write  ET=  1.287 secs  IOs/second =  1017  Data Rate = 32.571 MBs
 65536 byte write  ET=  1.898 secs  IOs/second =   552  Data Rate = 35.338 MBs

Beginning 2 process random write test on /dev/rdsk/c12t0d0s2

   512 byte write  ET=  0.954 secs  IOs/second = 10482  Data Rate =  5.240 MBs
  1024 byte write  ET=  0.863 secs  IOs/second =  9269  Data Rate =  9.270 MBs
  2048 byte write  ET=  0.863 secs  IOs/second =  7415  Data Rate = 14.831 MBs
  4096 byte write  ET=  0.952 secs  IOs/second =  5378  Data Rate = 21.512 MBs
  8192 byte write  ET=  1.179 secs  IOs/second =  3474  Data Rate = 27.793 MBs
 16384 byte write  ET=  1.617 secs  IOs/second =  2025  Data Rate = 32.415 MBs
 32768 byte write  ET=  2.370 secs  IOs/second =  1105  Data Rate = 35.375 MBs
 65536 byte write  ET=  3.619 secs  IOs/second =   579  Data Rate = 37.066 MBs

Beginning 4 process random write test on /dev/rdsk/c12t0d0s2

   512 byte write  ET=  1.833 secs  IOs/second = 10911  Data Rate =  5.455 MBs
  1024 byte write  ET=  1.672 secs  IOs/second =  9569  Data Rate =  9.569 MBs
  2048 byte write  ET=  1.678 secs  IOs/second =  7628  Data Rate = 15.256 MBs
  4096 byte write  ET=  1.862 secs  IOs/second =  5499  Data Rate = 21.997 MBs
  8192 byte write  ET=  2.332 secs  IOs/second =  3512  Data Rate = 28.103 MBs
 16384 byte write  ET=  3.206 secs  IOs/second =  2043  Data Rate = 32.698 MBs
 32768 byte write  ET=  4.715 secs  IOs/second =  1111  Data Rate = 35.562 MBs
 65536 byte write  ET=  7.223 secs  IOs/second =   580  Data Rate = 37.144 MBs

Beginning 6 process random write test on /dev/rdsk/c12t0d0s2

   512 byte write  ET=  2.784 secs  IOs/second = 10775  Data Rate =  5.388 MBs
  1024 byte write  ET=  2.555 secs  IOs/second =  9393  Data Rate =  9.393 MBs
  2048 byte write  ET=  2.542 secs  IOs/second =  7553  Data Rate = 15.105 MBs
  4096 byte write  ET=  2.818 secs  IOs/second =  5450  Data Rate = 21.802 MBs
  8192 byte write  ET=  3.527 secs  IOs/second =  3483  Data Rate = 27.871 MBs
 16384 byte write  ET=  4.839 secs  IOs/second =  2030  Data Rate = 32.495 MBs
 32768 byte write  ET=  7.096 secs  IOs/second =  1107  Data Rate = 35.445 MBs
 65536 byte write  ET= 10.845 secs  IOs/second =   579  Data Rate = 37.107 MBs

Beginning 8 process random write test on /dev/rdsk/c12t0d0s2

   512 byte write  ET=  3.731 secs  IOs/second = 10720  Data Rate =  5.360 MBs
  1024 byte write  ET=  3.410 secs  IOs/second =  9384  Data Rate =  9.384 MBs
  2048 byte write  ET=  3.385 secs  IOs/second =  7562  Data Rate = 15.125 MBs
  4096 byte write  ET=  3.751 secs  IOs/second =  5459  Data Rate = 21.839 MBs
  8192 byte write  ET=  4.697 secs  IOs/second =  3488  Data Rate = 27.905 MBs
 16384 byte write  ET=  6.451 secs  IOs/second =  2031  Data Rate = 32.501 MBs
 32768 byte write  ET=  9.460 secs  IOs/second =  1107  Data Rate = 35.450 MBs
 65536 byte write  ET= 14.469 secs  IOs/second =   579  Data Rate = 37.084 MBs
```

Main Menu

```
Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number:
```

RANDOM READ/WRITE BENCHMARK

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 16:22:46 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                   1
Random Read Benchmark             2
Random Write Benchmark            3
Random Read/Write Benchmark       4
Exerciser                         5
Maintenance tests                 6
Exit                              7


Enter test number: 4
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter percentage of reads vs writes (5-95 [50]): 75
read percentage = 75
Enter iteration count (100-50000 [10000]: 5000
itercount = 5000
Enter logfile name:
logfile name =
Enter minimum process count [1]:
minimum process count = 1
Enter maximum process count (1-40 [8]):
maximum process count = 8
Enter minimum transfer size (512-65536 [512] bytes):
minimum transfer size = 512
Enter maximum transfer size (512-65536 [65536] bytes):
maximum transfer size = 65536
Do you want writes to be synchronous (yes,no [no])?
Writes will be asynchronous

This test will WRITE on device /dev/rdsk/c12t0d0s2.
 Do you want to continue (yes,no [no])?  y
Writing on /dev/rdsk/c12t0d0s2 enabled.
Asynchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)
Starting iteration count = 5000
```

```
Transfers will vary between 512 and 65536 bytes
There will be ~3750 reads and ~1250 writes per test

Beginning 1 process Read/Write test of /dev/rdsk/c12t0d0s2

   512 byte rdwr   ET=  0.840 secs  IOs/second =  5952  Data Rate =  2.976 MBs
  1024 byte rdwr   ET=  0.725 secs  IOs/second =  5517  Data Rate =  5.517 MBs
  2048 byte rdwr   ET=  0.661 secs  IOs/second =  4841  Data Rate =  9.682 MBs
  4096 byte rdwr   ET=  0.663 secs  IOs/second =  3861  Data Rate = 15.444 MBs
  8192 byte rdwr   ET=  0.749 secs  IOs/second =  2734  Data Rate = 21.874 MBs
 16384 byte rdwr   ET=  0.936 secs  IOs/second =  1750  Data Rate = 28.000 MBs
 32768 byte rdwr   ET=  1.288 secs  IOs/second =  1017  Data Rate = 32.546 MBs
 65536 byte rdwr   ET=  1.895 secs  IOs/second =   553  Data Rate = 35.394 MBs

Beginning 2 process Read/Write test of /dev/rdsk/c12t0d0s2

   512 byte rdwr   ET=  0.919 secs  IOs/second = 10881  Data Rate =  5.440 MBs
  1024 byte rdwr   ET=  0.839 secs  IOs/second =  9535  Data Rate =  9.535 MBs
  2048 byte rdwr   ET=  0.837 secs  IOs/second =  7646  Data Rate = 15.292 MBs
  4096 byte rdwr   ET=  0.933 secs  IOs/second =  5487  Data Rate = 21.950 MBs
  8192 byte rdwr   ET=  1.167 secs  IOs/second =  3509  Data Rate = 28.078 MBs
 16384 byte rdwr   ET=  1.609 secs  IOs/second =  2036  Data Rate = 32.576 MBs
 32768 byte rdwr   ET=  2.365 secs  IOs/second =  1107  Data Rate = 35.450 MBs
 65536 byte rdwr   ET=  3.613 secs  IOs/second =   580  Data Rate = 37.128 MBs

Beginning 4 process Read/Write test of /dev/rdsk/c12t0d0s2

   512 byte rdwr   ET=  1.770 secs  IOs/second = 11299  Data Rate =  5.649 MBs
  1024 byte rdwr   ET=  1.615 secs  IOs/second =  9907  Data Rate =  9.906 MBs
  2048 byte rdwr   ET=  1.626 secs  IOs/second =  7872  Data Rate = 15.744 MBs
  4096 byte rdwr   ET=  1.835 secs  IOs/second =  5580  Data Rate = 22.321 MBs
  8192 byte rdwr   ET=  2.307 secs  IOs/second =  3550  Data Rate = 28.407 MBs
 16384 byte rdwr   ET=  3.186 secs  IOs/second =  2056  Data Rate = 32.903 MBs
 32768 byte rdwr   ET=  4.708 secs  IOs/second =  1112  Data Rate = 35.615 MBs
 65536 byte rdwr   ET=  7.211 secs  IOs/second =   581  Data Rate = 37.205 MBs

Beginning 6 process Read/Write test of /dev/rdsk/c12t0d0s2

   512 byte rdwr   ET=  2.713 secs  IOs/second = 11057  Data Rate =  5.528 MBs
  1024 byte rdwr   ET=  2.492 secs  IOs/second =  9630  Data Rate =  9.630 MBs
  2048 byte rdwr   ET=  2.500 secs  IOs/second =  7680  Data Rate = 15.359 MBs
  4096 byte rdwr   ET=  2.781 secs  IOs/second =  5523  Data Rate = 22.092 MBs
  8192 byte rdwr   ET=  3.482 secs  IOs/second =  3529  Data Rate = 28.231 MBs
 16384 byte rdwr   ET=  4.803 secs  IOs/second =  2046  Data Rate = 32.739 MBs
 32768 byte rdwr   ET=  7.075 secs  IOs/second =  1110  Data Rate = 35.550 MBs
 65536 byte rdwr   ET= 10.833 secs  IOs/second =   580  Data Rate = 37.148 MBs

Beginning 8 process Read/Write test of /dev/rdsk/c12t0d0s2

   512 byte rdwr   ET=  3.617 secs  IOs/second = 11058  Data Rate =  5.529 MBs
  1024 byte rdwr   ET=  3.330 secs  IOs/second =  9609  Data Rate =  9.609 MBs
  2048 byte rdwr   ET=  3.310 secs  IOs/second =  7734  Data Rate = 15.468 MBs
  4096 byte rdwr   ET=  3.704 secs  IOs/second =  5529  Data Rate = 22.116 MBs
  8192 byte rdwr   ET=  4.639 secs  IOs/second =  3531  Data Rate = 28.254 MBs
 16384 byte rdwr   ET=  6.405 secs  IOs/second =  2045  Data Rate = 32.734 MBs
```

```
 32768 byte rdwr   ET=  9.432 secs  IOs/second =  1111  Data Rate = 35.555 MBs
 65536 byte rdwr   ET= 14.438 secs  IOs/second =   580  Data Rate = 37.164 MBs


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number:
```

EXERCISER

The exerciser performs 25,000 write/read/compare operations on the device under test. The starting block of each transfer is between block # 1 and the end of the device. Block “0” is not tested. The transfer sizes are random between 512 and 65536 bytes (default). The exerciser process is single threaded. Each pass of 25,000 iterations takes approximately 1 to 2 minutes depending on the host system and the transfer rate of the SCSI bus.

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 16:26:10 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number: 5
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter logfile name:
logfile name =
Enter minimum transfer size (512-65536 [512] bytes):
minimum transfer size = 512
Enter maximum transfer size (512-65536 [65536] bytes):
maximum transfer size = 65536

This test will WRITE on device /dev/rdsk/c12t0d0s2.
 Do you want to continue (yes,no [no])?  y
Writing on /dev/rdsk/c12t0d0s2 enabled.
Synchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)
Transfers will vary between 512 and 65536 bytes
Starting exerciser on device /dev/rdsk/c12t0d0s2 at 16:26:24
1611393 blocks written, read and compared
End of pass 1 on device /dev/rdsk/c12t0d0s2 at 16:27:32
1603965 blocks written, read and compared
```

```
End of pass 2 on device /dev/rdsk/c12t0d0s2 at 16:28:39
1620099 blocks written, read and compared
End of pass 3 on device /dev/rdsk/c12t0d0s2 at 16:29:47
1613684 blocks written, read and compared
End of pass 4 on device /dev/rdsk/c12t0d0s2 at 16:30:55
^C#
```

Sample error from exerciser test:

The exerciser wrote 54 blocks of random data starting at block number 679693. An error was detected in block 679738 when the data was read back and compared to the original write buffer. The expected and received data are printed in HEX starting at the first failing 32 bit integer in the bad block.

The failing block is then dumped in HEX from the read buffer. The failing block is then re-read from the disk and dumped in HEX a second time.

```
.
.
Error comparing write and read buffers
54 block transfer starting at block number 679693
Error is in 32 bits starting at byte 0 in block number 679738
Expected data = CE0F9A20  Received data = C512B656

HEX dump of block 679738 --  read from read buffer
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| C512B656 374CB370 2145E379 C89A5F38 A8338307 EA43BE6F 01E8A83C 65EA0C7C
 32| 124BEE69 2BA04645 60F09A01 7359A96F E3558521 536D7F68 81896A23 1D280938
 64| DB83BD2F F97D6F13 DF971576 4C53672E 29FF9B15 B05C4C5A C1F4AB08 3551E412
 96| 9262AF23 F5579333 7289FE08 0CC50236 3A28B231 212AF465 23A45201 D67B3621
128| 13FB2808 98DF9637 5D803F69 5C244D6A DF745468 04BDFE5D 249D0125 3B543D35
160| E3727C18 BB07E660 A51A6F49 188CDF62 2ACB933C 3EB20A39 9CF9F15F F20C5E4A
192| 73CC1508 DB0E2264 46B62001 9070E351 B8926B02 72CA2929 FD585B13 19175D0E
224| 5BDABD59 53912E76 75B6DA2C C29B2539 F09B7811 47E08436 9139813F A2A23C0E
256| A15A901C A5A3767C E3301804 A1717441 42AF6D14 A566B05B BF391A60 8EE6D279
288| F239E252 B3F4623C FA025B44 517C6A27 6AC45220 F86E3A26 45416625 35314962
320| E5FEF431 FEF7190C 204FF30B 6840E613 ACE13B33 6B371B46 7B92E849 7DC0D44C
352| 3D5DF407 EC17C350 F061C14E 2D59905D 17A3C422 2E41D142 97C35A6E 1E621A6C
384| 24F15A68 07F0A504 86FFD632 DCDCF01B C4693D5F B1EE113D E69F0D23 E4DBCD4F
416| 73227B45 358F9E11 B2187D1A DF08B523 16331741 E6A91738 0EF18C17 EC1B3E5B
448| E72F8958 05C27F02 40E8C26C 1536E676 F917BB5A 8192B65D 5030FB4E E4309F58
480| 6B622328 E1641C68 38785B4D 0F15F36A 26749A01 3CB4800C C81F3A31 500A0308


Re-reading block number 679693
HEX dump of block 679738 -- read from /dev/rdsk/c12t0d0s2
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| C512B656 374CB370 2145E379 C89A5F38 A8338307 EA43BE6F 01E8A83C 65EA0C7C
 32| 124BEE69 2BA04645 60F09A01 7359A96F E3558521 536D7F68 81896A23 1D280938
 64| DB83BD2F F97D6F13 DF971576 4C53672E 29FF9B15 B05C4C5A C1F4AB08 3551E412
 96| 9262AF23 F5579333 7289FE08 0CC50236 3A28B231 212AF465 23A45201 D67B3621
128| 13FB2808 98DF9637 5D803F69 5C244D6A DF745468 04BDFE5D 249D0125 3B543D35
160| E3727C18 BB07E660 A51A6F49 188CDF62 2ACB933C 3EB20A39 9CF9F15F F20C5E4A
192| 73CC1508 DB0E2264 46B62001 9070E351 B8926B02 72CA2929 FD585B13 19175D0E
224| 5BDABD59 53912E76 75B6DA2C C29B2539 F09B7811 47E08436 9139813F A2A23C0E
256| A15A901C A5A3767C E3301804 A1717441 42AF6D14 A566B05B BF391A60 8EE6D279
288| F239E252 B3F4623C FA025B44 517C6A27 6AC45220 F86E3A26 45416625 35314962
320| E5FEF431 FEF7190C 204FF30B 6840E613 ACE13B33 6B371B46 7B92E849 7DC0D44C
352| 3D5DF407 EC17C350 F061C14E 2D59905D 17A3C422 2E41D142 97C35A6E 1E621A6C
```

```
384| 24F15A68 07F0A504 86FFD632 DCDCF01B C4693D5F B1EE113D E69F0D23 E4DBCD4F
416| 73227B45 358F9E11 B2187D1A DF08B523 16331741 E6A91738 0EF18C17 EC1B3E5B
448| E72F8958 05C27F02 40E8C26C 1536E676 F917BB5A 8192B65D 5030FB4E E4309F58
480| 6B622328 E1641C68 38785B4D 0F15F36A 26749A01 3CB4800C C81F3A31 500A0308


Do you want to continue (yes,no [no])?  no


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number:
```

5. MAINTENANCE PROGRAMS

RELIABILITY TEST

The reliability test writes the drive with a specific pattern, reads the drive, and does a compare in memory of the expected and received data. The process is repeated for the following patterns:

```
unsigned long data[] = {
    0x00000000,
    0xFFFFFFFF,
    0xaaaaaaaa,
    0x55555555,
    0x0F0F0F0F,
    0xF0F0F0f0,
    0xFF00FF00,
    0x00FF00FF
  };
```

When an error is detected, the byte location of the bad data within the failing block is printed out. The received data is dumped in HEX as well as the failing block re-read from the device. Block “0” is not tested.

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 16:49:30 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number: 6
Enter device name: /dev/rdsk/c12t0d0s2
```

```
device = /dev/rdsk/c12t0d0s2
Enter logfile name:
logfile name =
Enter transfer size [65536 bytes]:
transfer size = 65536
Synchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)

        Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12

Enter test number: 1

This test will WRITE on device /dev/rdsk/c12t0d0s2.
 Do you want to continue (yes,no [no])?  y
Writing on /dev/rdsk/c12t0d0s2 enabled.
Starting reliability test on device /dev/rdsk/c12t0d0s2 at 16:49:41
Block zero(0) will not be tested

Transfer size = 65536 bytes
writing with 0x00000000's   100%
reading and comparing 0x00000000's    100%
writing with 0xffffffff's   100%
reading and comparing 0xffffffff's    100%
writing with 0xaaaaaaaa's   100%
reading and comparing 0xaaaaaaaa's    100%
writing with 0x55555555's   100%
reading and comparing 0x55555555's    100%
writing with 0x0f0f0f0f's   100%
reading and comparing 0x0f0f0f0f's    100%
writing with 0xf0f0f0f0's   100%
reading and comparing 0xf0f0f0f0's    100%
writing with 0xff00ff00's   100%
reading and comparing 0xff00ff00's    100%
writing with 0x00ff00ff's   100%
reading and comparing 0x00ff00ff's    100%
End of pass 1 on /dev/rdsk/c12t0d0s2 at 16:54:48

Transfer size = 65536 bytes
writing with 0x00000000's   100%
reading and comparing 0x00000000's      60%^C#
#
```

Error detected by reliability test.

In this case, a bit was dropped in the 32 bits starting at byte number 4 at block 10000.

```
Starting reliability test on device /dev/rdsk/c12t0d0s2 at 16:56:37
Block zero(0) will not be tested

Transfer size = 65536 bytes
writing with 0x00000000's   100%
reading and comparing 0x00000000's    100%
writing with 0xffffffff's   100%
reading and comparing 0xffffffff's      0%
Error comparing write and read buffers
block = 10000
Error is in 32 bits starting at byte 4
Expected data = 0xFFFFFFFF  Received data = 0xFFFFFFEF

HEX dump of block 10000 --  read from read buffer
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| FFFFFFFF FFFFFFEF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
 32| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
 64| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
 96| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
128| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
160| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
192| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
224| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
256| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
288| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
320| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
352| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
384| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
416| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
448| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
480| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF


Re-reading block number 10000
HEX dump of block # 10000 -- read from /dev/rdsk/c12t0d0s2
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| FFFFFFFF FFFFFFEF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
 32| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
 64| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
 96| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
128| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
160| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
192| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
224| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
256| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
288| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
```

```
320| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
352| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
384| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
416| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
448| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
480| FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
#
```

QUICK READ TEST

The drive is read one time from block “0” to the end.

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 17:01:08 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number: 6
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter logfile name:
logfile name =
Enter transfer size [65536 bytes]:
transfer size = 65536
Synchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)

        Maintenance mode tests
Reliability                     1
Quick read                      2
Write block numbers             3
Scan block numbers              4
Continuous write/scan block     5
HEX block dump                  6
Byte editor                     7
Exit                           12

Enter test number: 2
Starting quick read at 17:01:16
100%
Done! 1045504 blocks read at 17:01:31
```

```
        Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12

Enter test number:
```

WRITE BLOCK TEST

The write block test is used in conjunction with the scan block test to check for addressing problems. Each block on the device, including block "0", is written with a recognizable block number pattern. If the correct pattern is not read back during the scan block test, the original data may have been written to the wrong location.

The first three blocks on the device are written with “0’s.” The second three block group is written with “1’s” in a rotating three byte pattern.

```
block number 3
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
-------------------------------------------------------------------------------
  0| 01000001 00010000 00000100 01000001 00010000 00000100 01000001 00010000
 32| 00000100 01000001 00010000 00000100 01000001 00010000 00000100 01000001

Block number 4
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
-------------------------------------------------------------------------------
  0| 00000100 01000001 00010000 00000100 01000001 00010000 00000100 01000001
 32| 00010000 00000100 01000001 00010000 00000100 01000001 00010000 00000100

Block number 5
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
-------------------------------------------------------------------------------
  0| 00010000 00000100 01000001 00010000 00000100 01000001 00010000 00000100
 32| 01000001 00010000 00000100 01000001 00010000 00000100 01000001 00010000
```

The next three blocks are written with “2’s” in the same three byte rotating pattern.

```
Block number 6
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
-------------------------------------------------------------------------------
  0| 02000002 00020000 00000200 02000002 00020000 00000200 02000002 00020000
 32| 00000200 02000002 00020000 00000200 02000002 00020000 00000200 02000002

Block number 7
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
-------------------------------------------------------------------------------
  0| 00000200 02000002 00020000 00000200 02000002 00020000 00000200 02000002
 32| 00020000 00000200 02000002 00020000 00000200 02000002 00020000 00000200

Block number 8
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
-------------------------------------------------------------------------------
  0| 00020000 00000200 02000002 00020000 00000200 02000002 00020000 00000200
 32| 02000002 00020000 00000200 02000002 00020000 00000200 02000002 00020000
```

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 17:02:43 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                   1
Random Read Benchmark             2
Random Write Benchmark            3
Random Read/Write Benchmark       4
Exerciser                         5
Maintenance tests                 6
Exit                              7


Enter test number: 6
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter logfile name:
logfile name =
Enter transfer size [65536 bytes]:
transfer size = 65536
Synchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)

        Maintenance mode tests
Reliability                     1
Quick read                      2
Write block numbers             3
Scan block numbers              4
Continuous write/scan block     5
HEX block dump                  6
Byte editor                     7
Exit                           12

Enter test number: 3

This test will WRITE on device /dev/rdsk/c12t0d0s2.
 Do you want to continue (yes,no [no])?  y
Writing on /dev/rdsk/c12t0d0s2 enabled.
This test will write on block 0
Do you want to continue (yes,no [no])?  y
Block 0 will be tested

Starting write block at 17:02:54
```

```
100%
Done! 1045504 blocks written at 17:03:15

        Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12

Enter test number:
```

SCAN BLOCK TEST

NOTE: The drive must be written with the Write Block test before running the Scan Block! The Scan Block tests reads the device looking for the three byte, three block rotating block number pattern written by the Write Block test.

```
# ./IOtest

         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 17:16:10 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


               Main Menu


Quick Read Test                 1
Random Read Benchmark           2
Random Write Benchmark          3
Random Read/Write Benchmark     4
Exerciser                       5
Maintenance tests               6
Exit                            7


Enter test number: 6
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter logfile name:
logfile name =
Enter transfer size [65536 bytes]:
transfer size = 65536
Synchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)

        Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12
```

```
Enter test number: 4
Starting block number scan at 17:17:05
100%
Done! 1045504 blocks read at 17:17:31

        Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12

Enter test number:
```

Scan Block Error

In this case, byte “0” at block 120000 read back as “41.” The Scan Block test expected “40.” The expected data is dumped in HEX followed by the received data and the data re-read from the drive.

```
Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12

Enter test number: 4
Starting block number scan at 17:18:38
 10%
Error comparing expected and received data at block 120000
Error is in 32 bits starting at byte 0
Expected data = 40009C40  Received data = 41009C40

HEX dump of block 120000 --  read from write buffer
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
 32| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
 64| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
 96| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
128| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
160| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
192| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
224| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
256| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
288| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
320| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
352| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
384| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
416| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
448| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
480| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C

HEX dump of block 120000 --  read from read buffer
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| 41009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
 32| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
 64| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
 96| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
128| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
160| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
```

```
192| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
224| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
256| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
288| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
320| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
352| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
384| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
416| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
448| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
480| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C


Re-reading block number 120000
HEX block dump of block 120000
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
--------------------------------------------------------------------------------
  0| 41009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
 32| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
 64| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
 96| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
128| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
160| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
192| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
224| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
256| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
288| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
320| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
352| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
384| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
416| 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40
448| 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000
480| 40009C40 9C40009C 009C4000 40009C40 9C40009C 009C4000 40009C40 9C40009C
#
```

HEX BLOCK DUMP

The HEX dump will display any block on the device in HEX.

Press:

<cr>	repeat the same block
“n”	next block
“p”	previous block
“a”	display in ascii
“h”	display in HEX
“q”	quit

```
384| 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
416| 00000000 FF031815 00000000 01000000 0200FD03 80000800 00000000 00000000
448| 00000100 40000000 00000100 00000000 00F40F00 00000000 00000000 00000000
480| 00000000 00000000 00000000 80000000 00F40D00 00000000 00000000 581CBEDA
Enter block number (0 to 1045503):  a
Block number 0
Byte   0  1  2  3   4  5  6  7   8  9 10 11  12 13 14 15  16 17 18 19
--------------------------------------------------------------------------------
  0|   -  S  E  D   S  0  0  8   r  t  l  U   S  C  S  a   S  H  -  I
 20|   c     1  0   1     l  y      1  2  0      t  l  a   d  h     2
 40|   s     8      1     c  e  \0 \0  8  2  \0 \0 \0 \0  \0 \0 \0 \0
 60|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
 80|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
100|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
120|  \0 \0 \0 \0  \0 \0 \0 \0    \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
140|    \0 \b \0    \0 \0 \0    \0   \0  \0 \0   \0  \0 \0 \0 \0
160|  \0 \0 \0 \0    \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
180|  \0 \0 \0 \0  \0 \0 \0 \0   î  Þ \r  `  \0 \0 \0 \0  \0 \0 \0 \0
200|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
220|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
240|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
260|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
280|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
300|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
320|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
340|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
360|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
380|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
400|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
420|   ÿ        \0 \0 \0 \0    \0 \0 \0    \0  ý     € \0 \b \0
440|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0   \0   @ \0 \0 \0  \0 \0   \0
460|  \0 \0 \0 \0  \0  ô   \0  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0
480|  \0 \0 \0 \0  \0 \0 \0 \0  \0 \0 \0 \0   € \0 \0 \0  \0  ô \r \0
500|  \0 \0 \0 \0  \0 \0 \0 \0   X    ¾  Ú  <--- byte 512
Enter block number (0 to 1045503):
```

BYTE EDITOR

The Byte Editor allows you to display a block in HEX and change any byte.

WARNING: There is no limit to how much damage you can cause with this test if written on the wrong device.

```
#
         Solid Data Systems IOtest V2.31

Starting at: Mon Mar 26 19:37:04 2001

Sysname: SunOS    Release: 5.8    Machine: sun4u


                Main Menu


Quick Read Test                  1
Random Read Benchmark            2
Random Write Benchmark           3
Random Read/Write Benchmark      4
Exerciser                        5
Maintenance tests                6
Exit                             7


Enter test number: 6
Enter device name: /dev/rdsk/c12t0d0s2
device = /dev/rdsk/c12t0d0s2
Enter logfile name:
logfile name =
Enter transfer size [65536 bytes]:
transfer size = 65536
Synchronous writes enabled
Reading drive to determine size...
Drive size = 1045504 blocks (535298048 bytes)

        Maintenance mode tests
Reliability                     1
Quick read                      2
Write block numbers             3
Scan block numbers              4
Continuous write/scan block     5
HEX block dump                  6
Byte editor                     7
Exit                           12
```

```
Enter test number: 7

This test will WRITE on device /dev/rdsk/c12t0d0s2.
 Do you want to continue (yes,no [no])?  y
Writing on /dev/rdsk/c12t0d0s2 enabled.
IOt 1 do_log = 0
Write a byte // Enter `q` to quit.

Enter block number (0 to 1045503): 300

Block number = 300
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| 03020100 07060504 0B0A0908 0F0E0D0C 13121110 17161514 1B1A1918 1F1E1D1C
 32| 23222120 27262524 2B2A2928 2F2E2D2C 33323130 37363534 3B3A3938 3F3E3D3C
 64| 43424140 47464544 4B4A4948 4F4E4D4C 53525150 57565554 5B5A5958 5F5E5D5C
 96| 63626160 67666564 6B6A6968 6F6E6D6C 73727170 77767574 7B7A7978 7F7E7D7C
128| 83828180 87868584 8B8A8988 8F8E8D8C 93929190 97969594 9B9A9998 9F9E9D9C
160| A3A2A1A0 A7A6A5A4 ABAAA9A8 AFAEADAC B3B2B1B0 B7B6B5B4 BBBAB9B8 BFBEBDBC
192| C3C2C1C0 C7C6C5C4 CBCAC9C8 CFCECDCC D3D2D1D0 D7D6D5D4 DBDAD9D8 DFDEDDDC
224| E3E2E1E0 E7E6E5E4 EBEAE9E8 EFEEEDEC F3F2F1F0 F7F6F5F4 FBFAF9F8 FFFEFDFC
256| 03020100 07060504 0B0A0908 0F0E0D0C 13121110 17161514 1B1A1918 1F1E1D1C
288| 23222120 27262524 2B2A2928 2F2E2D2C 33323130 37363534 3B3A3938 3F3E3D3C
320| 43424140 47464544 4B4A4948 4F4E4D4C 53525150 57565554 5B5A5958 5F5E5D5C
352| 63626160 67666564 6B6A6968 6F6E6D6C 73727170 77767574 7B7A7978 7F7E7D7C
384| 83828180 87868584 8B8A8988 8F8E8D8C 93929190 97969594 9B9A9998 9F9E9D9C
416| A3A2A1A0 A7A6A5A4 ABAAA9A8 AFAEADAC B3B2B1B0 B7B6B5B4 BBBAB9B8 BFBEBDBC
448| C3C2C1C0 C7C6C5C4 CBCAC9C8 CFCECDCC D3D2D1D0 D7D6D5D4 DBDAD9D8 DFDEDDDC
480| E3E2E1E0 E7E6E5E4 EBEAE9E8 EFEEEDEC F3F2F1F0 F7F6F5F4 FBFAF9F8 FFFEFDFC

Enter byte number to change (0-511): 0

Enter byte (HEX 00-FF):  ff

Block number = 300
Byte  0 1 2 3  4 5 6 7  8 91011 12131415 16171819 20212223 24252627 28293031
------------------------------------------------------------------------------
  0| FF020100 07060504 0B0A0908 0F0E0D0C 13121110 17161514 1B1A1918 1F1E1D1C
 32| 23222120 27262524 2B2A2928 2F2E2D2C 33323130 37363534 3B3A3938 3F3E3D3C
 64| 43424140 47464544 4B4A4948 4F4E4D4C 53525150 57565554 5B5A5958 5F5E5D5C
 96| 63626160 67666564 6B6A6968 6F6E6D6C 73727170 77767574 7B7A7978 7F7E7D7C
128| 83828180 87868584 8B8A8988 8F8E8D8C 93929190 97969594 9B9A9998 9F9E9D9C
160| A3A2A1A0 A7A6A5A4 ABAAA9A8 AFAEADAC B3B2B1B0 B7B6B5B4 BBBAB9B8 BFBEBDBC
192| C3C2C1C0 C7C6C5C4 CBCAC9C8 CFCECDCC D3D2D1D0 D7D6D5D4 DBDAD9D8 DFDEDDDC
224| E3E2E1E0 E7E6E5E4 EBEAE9E8 EFEEEDEC F3F2F1F0 F7F6F5F4 FBFAF9F8 FFFEFDFC
256| 03020100 07060504 0B0A0908 0F0E0D0C 13121110 17161514 1B1A1918 1F1E1D1C
288| 23222120 27262524 2B2A2928 2F2E2D2C 33323130 37363534 3B3A3938 3F3E3D3C
320| 43424140 47464544 4B4A4948 4F4E4D4C 53525150 57565554 5B5A5958 5F5E5D5C
352| 63626160 67666564 6B6A6968 6F6E6D6C 73727170 77767574 7B7A7978 7F7E7D7C
384| 83828180 87868584 8B8A8988 8F8E8D8C 93929190 97969594 9B9A9998 9F9E9D9C
416| A3A2A1A0 A7A6A5A4 ABAAA9A8 AFAEADAC B3B2B1B0 B7B6B5B4 BBBAB9B8 BFBEBDBC
448| C3C2C1C0 C7C6C5C4 CBCAC9C8 CFCECDCC D3D2D1D0 D7D6D5D4 DBDAD9D8 DFDEDDDC
480| E3E2E1E0 E7E6E5E4 EBEAE9E8 EFEEEDEC F3F2F1F0 F7F6F5F4 FBFAF9F8 FFFEFDFC
```

```
Enter block number (0 to 1045503): q



        Maintenance mode tests
Reliability                    1
Quick read                     2
Write block numbers            3
Scan block numbers             4
Continuous write/scan block    5
HEX block dump                 6
Byte editor                    7
Exit                          12

Enter test number:
```