1145

Bell Laboratories

subject: DEC Factory Acceptance Test

date: January 19, 1977

from: R. B. Brandt
C. D. Perez
MF-77-8234-001

MEMORANDUM FOR FILE

1. Introduction

During June and July, 1976, meetings were held between members of BTI Division 52 and representatives of Digital Equipment Corporation (DEC). [1,2] The primary purpose of these meetings was to discuss some of the problems encountered by Operations Support Systems (OSS's) in the field, especially those resulting in long outages. A number of proposals for improving the installation and maintenance of these OSS's were advanced. One such proposal was that BTL should make available to DEC a UNIX checkout package for use as a final factory acceptance test on DEC-supplied OSS hardware prior to shipment to the field site. As a result of this suggestion, it was decided that such a checkout package would be developed and used on a trial basis for approximately six months on Switching Control Center System (SCCS) configurations. This memorandum is a description of the checkout package that is being used in the trial.

2. Overview

The package that was provided to DEC in early December, 1976, consists of a UNIX "sysgen" tape and the documentation describing how to boot UNIX from the tape and run the tests. The "sysgen" tape is like those supplied by the UNIX Support Group (USG) to BTL projects; it contains a root file system image and the programs that are necessary to copy the tape to a disk and boot the UNIX system. Source code was not included, and the UNIX on the tape was configured for SCCS (11/70 with 128K memory, TU16, RP05, and four DH11's) at modification level 2.0j. A copy of the documentation that accompanied the "sysgen" tape is attached to this memorandum (Attachment A). The checkout system is to be run on all SCCS machines after all other DEC factory acceptance tests have been completed. Test results are to be documented so that an evaluation of the trial may be made at its completion. The checkout package is not supplied to the field site for use during actual system installation. See Reference [3] for a breakdown of test responsibilities for DEC and BTL.

Before proceeding with a detailed description of the tests

included in the checkout package, a synopsis of the design criteria and constraints is in order. First, the tests were to be made as "goof-proof" as possible, so that persons with little or no experience on UNIX systems could run them successfully after either reading the accompanying documentation or receiving a minimal amount of coaching. To this end, the number of commands necessary to run the entire test was minimized, as was the number of arguments to these commands; extensive argument validation is performed; error messages include an error number that may be used to easily look up additional information about an error (see the "Error Conditions" portion of the documentation); and the user is totally insulated from the notion of UNIX device names.

Although the initial trial of this package is for SCCS systems only, it was written to be easily extensible to other configurations. The test themselves are shell command files. Only three special purpose commands were written for use in these command files; all other commands are standard UNIX commands. There were a number of benefits that resulted from following this course of action. The time required to actually write and debug the tests was greatly reduced from what would have been required if the whole package had been written in C. The resultant product may be easily modified, even by those inexperienced in the C language. No recompilation of commands is necessary if the package is used on a new configuration. In fact, the tests were debugged on the USG 11/45 system; conversion to the SCCS configuration was readily accomplished by loading a new UNIX and creating less than a dozen (one line) files.

As was previously mentioned, the UNIX used is the standard USG system. No attempt is made to exercise devices or hardware features that are not supported by this system (e.g. DJ11's, Program Interrupt Register). Block devices (disk and tape) are tested both singly and in pairs. However, to preserve the generality of the tests and minimize the development effort, no effort is made by the tests to access every sector of every disk. Disk file system are all less than 2K blocks in size, except for the root file system (4K blocks).

Finally, since the purpose of these tests is to exercise the system as much as possible, speed of execution was not a major consideration in the design. That is, the tests were written to be efficient, but no time or effort was expended in optimizing execution times.

3. Future Work

Should this trial demonstrate that running a UNIX checkout package as the final step of DEC factory acceptance testing

is beneficial, it should be incorporated as normal procedure for all Bell System UNIX-based OSS machines. Even if this course of action is not taken, this package could be provided to all BTL UNIX projects as a tool to help resolve such questions as "Is my problem hardware or software?" and "How do I verify that a hardware problem has really been fixed?" In fact, this system has already been used at least twice to resolve these questions for the USG.

It is felt that the trial is somewhat incomplete, in that the checkout system is not also being run at the field site. Doing so may illuminate problems that are possibly being introduced during shipment and installation. Indeed, successful completion of the checkout package could be used as one of the criteria for acceptance of the hardware from DEC. Efforts should be made to work out the necessary legal arrangements necessary to include field site testing as a part of the trial.

The package itself was designed to be an exerciser, not a diagnostic. However, its ease of use would be greatly enhanced if command outputs were automatically checked against expected results. This would alleviate the need to visually examine the actual test outputs. Although this is a logical follow-on to the work already done, further investigation may reveal that it is not worth the additional investment that would be required.

4. Operation of the Tests

The actual mechanics of running the test package, including the transfer of the root file system from the "sysgen" tape to a disk, are provided in the attached documentation (see "Acceptance Test Procedures" and "Procedures for Installing UNIX on the PDP-11"). The purpose of this and succeeding paragraphs is to detail some of the inner workings of the shell command files. All of the shell files and special purpose commands necessary for the actual testing reside in the directory /accept.

4.1. Configuration Control

All needed information about the system's configuration, including the translation of generic device names to UNIX device names, is contained in files in subdirectories of the /configs directory. In particular, the directory "/configs/running" contains configuration information about the currently running system. Information about the configuration used by project xyz is contained in the directory /configs/xyz and is copied to /configs/running by the shell command files when a xyz configured system is being tested. By using this convention, it is possible to create a "sysgen" tape that can be used on more than one system

configuration. For example, if a single tape may be used for the configurations for project1, project2, and project3, the necessary configuration control information would be contained in the directories /configs/project1, /configs/project2, and /configs/project3. The /configs/running directory would contain the configuration data for whichever of those configurations was running at the time. The bootable UNIX files are handled in an analogous manner. That is, for the case above, the files /project1unix, /project2unix, and /project3unix would be the UNIX files that are booted when testing the configuration for project1, project2, and project3, respectively. The file /unix is a copy of the UNIX that is currently being run.

Within a configuration control directory (e.g. /configs/project1) there are two files, "root" and "swap", that contain, respectively, the UNIX special file name of the root and swap device for that particular UNIX system. In addition, the file "synjob" contains a number, which determines how many copies of the synthetic job are to be spawned when the memory test is run. There are also two subdirectories in this directory, "block" and "char". The "char" subdirectory is currently unused, but the "block" subdirectory contains files that define the block devices on the system. The names of the files vary, of course, from system to system, but are always formed by appending a unit number to a generic device name. See paragraph 7.2 of "Acceptance Test Procedures" in the attached documentation for a cross reference of block device types to generic device names. Thus, using the SCCS configuration as an example, the files "tape0" and "hp0" would exist within the subdirectory "block". The contents of these files are the UNIX special file names that are to be used when that particular device is being tested. In the case of SCCS, "tape0" contains "ht0" and "hp0" contains "hp6". Using these conventions, it is easy for the shell test files to determine if a system is configured for a device, and, if so, how to reference it. All references to devices by the user are made by the generic device name plus unit number. The existence of a file by that name in the "block" subdirectory confirms that the system is configured for such a device. The translation of that generic name to a UNIX special file name is readily accomplished by merely ascertaining the contents of the file. One disadvantage of this scheme is that it is possible for the actual UNIX file that is booted to not match the configuration defined to the shell files. However, since in the context of this package the configurations being tested are well defined and static, this is not seen as a major problem. In fact, it proved to be invaluable during the development and debugging of the test package, as the system on which the test was developed did not have a full complement of device types.

Some supplementary information is also contained in the "block" subdirectory. When a block device has multiple logical UNIX file systems, an additional file is created whose name is the generic device name plus the unit number plus the character "a". This file contains the UNIX special file name of an alternate file system on the physical device. In the case of SCCS, for example, the file "hp0" contains "hp6" and the file "hp0a" contains "hp7". This second file is used when a single block device is being tested to allow a second file system to be used on the same physical device.

4.2. The Startup Shell

The "startup" shell is the first test run after the system has been booted. Its purpose is to ensure that a satisfactory environment has been established before any further tests are run. There is one argument to this shell file, the project name (e.g. sccs). The shell file verifies that project is "known"; that is, it confirms that a UNIX file and configuration information are available for later reference by the tests themselves. Then the proper UNIX is placed in /unix, the configuration control information is copied to /configs/running, and the root file system is checked.

4.3. Testing Block Devices

All block devices are tested by invoking the "blkdev" shell as follows:

```
sh /accept/blkdev loop_count fromdev [todev]
```

The validity of the "loop_count" argument, which indicates the number of passes that are to be made through the test, is verified. The "fromdev" and, if present, "todev" argument, which are generic device names, are translated to UNIX special file names in the manner previously described. Note that if a "todev" was not specified and the physical device is able to have multiple logical file systems, the "todev" is filled in with the name of the alternate file system that is available for use on that device. After confirming that the appropriate UNIX special files exist with proper permissions and that the root and swap device will not be overwritten, either the shell file "tapecases" or "diskcases" is invoked to determine which shell file will be used to do the actual testing. Once this has been done, the shell file "passes" is called upon to actually run the test for the indicated number of passes.

The "diskcases" and "tapecases" shells are relatively straightforward; they determine the specific test that will be run as indicated in Attachment B. The only special

consideration is that "tapecases" must confirm that a scratch tape has been mounted on the appropriate drive(s) before returning.

Likewise, the "passes" shell file is straightforward, as it calls upon the actual test shell the number of times indicated by the "loopcount" argument. In those cases where a TU16 is being tested, this shell file also manipulates the entries in /dev so that in each pass through the test the tape drive is tested at both 800 bpi and 1600 bpi.

All of the tests use standard UNIX commands; these include /etc/mkfs, icheck, dd, dump, tp, and cp. Since some commands require super-user permission, a copy of "sh" is available that is owned by root and has the set-userid bit set. Whenever a file system is made, a proto for a 2000 block file system is specified to the mkfs command. Block devices are tested using both normal UNIX block I/O and raw I/O (block sizes of 1K, 10K, and 20K).

4.4. Testing DH11's

The test system is normally run in multi-user mode with one line activated (the console). When the shell file "dhtest" is invoked, the /etc/ttys file is modified to activate the console plus sixteen DH11 lines (ttya through ttyq, excluding ttyo). The /dev nodes for ttya through ttyq are then removed and recreated so that the minor device numbers correspond to the DH in question. Init is sent a hangup signal, which causes it to enable the lines on the DH. At this point the shell waits until it receives a response from the tester to indicate that the DH has been tested in the manner prescribed in the documentation. The lines to the DH are then deactivated by modifying /etc/ttys and once again sending init a hangup signal. This procedure is repeated for each DH on the system until they all have been tested.

4.5. Testing Memory

In order to test memory and the ability of the system to function under some loading, the "synthrc" shell command file is used to execute a number of simultaneous processes. The exact number of processes is obtained from the "synjob" file in the /configs/running directory. This value is set high enough to ensure that not all of the processes will fit into the available memory. The program used for this test is the UNIX synthetic job [4], which, as a portion of the test, is always recompiled. A wide range of options are available to users of the synthetic job; an arbitrary choice was made for the purposes of this test. In particular, each instance of the synthetic job process spawns one child. Both the parent and the child occupy 40,000 bytes of memory (80,000 bytes total) and make twenty five passes. Each pass

consists of a hundred iterations through a compute loop, five hundred read calls, and five hundred write calls.

4.6. Putting It Together

The system's complete configuration may be exercised (once the "startup" shell has been executed) by using the "fulltest" shell. This shell determines all combinations of block devices that are permissible to the tests and calls the "blkdev" test for each such combination. It then tests the DH's using "dhtest" and memory by using the synthetic job test "synthrc" . Verification that the test ran successfully is done by comparing the console output with the sample output provided in the documentation. If a device fails and adjustments are made to it, the tester may ensure that it is functioning properly by either rerunning "fulltest" or by running the specific test that failed. debugging, and final testing of this package.

R. B. Brandt

C. D. Perez

MH-8234-RBB-ds

Attachments
As Noted Above

Copy to
J. Balaguer - WECo Purchasing
R. M. Kuhner - AT&T - BR
B. O. Larsen - AT&T - BR

UNIX Support Group
D. S. DeJager
J. F. Maranzano
B. A. Tague

Copy to (w/o attachments)

R. M. Averill
G. L. Baldwin
J. H. Carran
D. H. Copp
A. D. Hall
G. W. R. Luderer
W. B. Smith
J. M. Malik
H. M. Zydney

References

[1] Averill, R. M., "Meeting Between Bell Telephone Laboratories and Digital Equipment Corporation", Conference Notes, July 2, 1976.

[2] Guyer, E. J., Meyer, H. C., and Averill, R. M., "Minutes of July 20th Meeting at Columbus BTL with DEC", July 26, 1976.

[3] Meyer, Howard C., "Minutes of SCC Factory Checkout Package Meeting of September 27, 1976", DEC Interoffice Memorandum, October 5, 1976.

[4] Bernstein, D. R., "Synthetic Process for UNIX", TM-76-8234-17, September 24, 1976.

ATTACHMENT A

Acceptance Test Procedures

1. Booting the System

Boot the UNIX system, following the procedures given in "Procedures for Installing UNIX on the PDP11". Note that there are two procedures specified. The first is used when the UNIX system has not yet been placed on the disk from the tape, and the second is used when a disk image of the UNIX tape has already been made.

2. Coming Up Multi-User

Set the console switches to 00000002 and type (simultaneously) control-d. This causes the system to come up multi user, and must be done in order to test the DH11's.

3. Some UNIX Conventions

At this point you are ready to use a UNIX system. Certain conventions will prevail throughout the remainder of the procedure. A brief description of them is given here.

UNIX responds to both upper and lower case. When typing input always use the case as it is presented in these instructions. A character mistyped may be erased by following it with a "sharp" sign (#). Multiple mistyped characters may be erased by typing the appropriate number of "sharp" characters. To erase an entire line of input follow it with the "at" sign (@). All input lines in the remaining procedure must be followed by a carriage return. Once hit, it is too late to erase anything and UNIX will attempt to execute the command you have typed. If you inadvertantly mistype a command to UNIX, you may terminate it by hitting the "delete" key. Retype the command correctly after you have been returned a prompt sign (%).

Most of the command lines to be typed begin with the command "sh". Part of the line is invariant; part of the line may be choices that give you some control over certain tests. The places where a choice is to be made are always shown within angle brackets (< >) and will be followed by a description of what constitutes a valid choice. All characters from "<" through ">" should be replaced with your choice. If for any reason you wish to terminate a command before its normal completion, hit the "delete" key. The command will abort and the system will respond with a prompt character.

You have been supplied with a sample of the output to be expected from the exercises. Keep it on hand to compare with the terminal output you receive. At this point in the test you should be able to see a similarity in the line

mem = 1016

on both outputs. This indicates how much memory UNIX believes is available for user programs. This number is expressed in units of tenths of K. For an SCCS configuration, any value other than the expected one of 1016 is an indication that the system was not able to access all available memory on the machine.

4. Logging In

Respond to the "login" message printed at the console by typing "dec". When logged in (as indicated by the "%" prompt sign), type the command:

stty -tabs -lcase

This causes tab characters to be translated into the appropriate number of spaces and allows for upper and lower case printouts.

5. Preparations for Running the Software Tests

Mount a scratch tape with a write ring on drive 0. Make sure that the drive is placed online. Run the startup procedure by entering the command:

sh /accept/startup <configuration>

The choice of configuration is currently limited to "sccs".

6. Running the Complete Test

Run the all-encompassing test by typing the command:

sh /accept/fulltest <loopcount> <number_dh>

This command causes all feasible combinations of the devices to be tested. The "loopcount" argument should indicate how many passes are to be made through each device test; "number_dh" should be the number of DH11's that are on the system. It should be noted that only one pass is made through the portion of the test that tests the DH11's, regardless of the value of "loopcount". Similarly, the synthetic job (which causes all available memory to be used) is also run independent of "loopcount". Expect one hour of testing for each increment of loopcount. This rule of thumb

applies to an 11/70 SCCS system.

The output from the various tests should be compared with the sample output to ascertain if there were any errors. In general, any deviations from the sample output (except as noted below) are indicative of errors. If any such errors are found in the printout, then the offending device(s) should be retested (see "Testing Specific Devices" below). If the errors persist, it is likely that there is a hardware problem. Deviations from the sample output are permitted on lines of the following type (N represents one or more digits):

```
N files  N blocks  N free

N files
N blocks
N.N tapes
N phase errors

N entries
N used
N last
```

6.1. Special Considerations for Tapes

Note that when the tape drive is being tested a reply from the console to verify that a scratch tape with a write ring is mounted on the indicated drive. The test will not proceed until this reply is made.

6.2. Special Considerations for DH11's

The procedure for testing the DH11's involves logging onto the system on every line of every DH11 that is on the system. This is done as follows. A message will be printed on the console

```
TEST DH NUMBER 1
HIT RETURN WHEN DONE TO PROCEED
```

At this point all of the lines on the first DH11 have been activated. Connect a terminal (set to EVEN parity and 300 baud) to the first line of the first DH11 with a null modem. The message

```
;login:
```

should appear on the terminal. Respond to this message by typing "dec". Once logged in (as evidenced by the prompt character %), the terminal may be connected to the second line of the first DH11. Continue with this procedure until all sixteen lines of the DH11 have been tested. You should

be able to login in on each of the lines of the first DH11 (only).

Once tne first DH11 has been completely tested, you may push the "return" key on the console. The system will then disable the first DH11 and enable the second. This is shown by the appearance on the console of the message

```
TEST DH NUMBER 2
HIT RETURN WHEN DONE TO PROCEED
```

Test the second DH11 as you did the first. Continue with these procedures until all of the DH11's on the system have been tested.

7. Testing Specific Devices

Facilities exist for testing a particular device or combination of devices only. This might be done, for example, if fixes are made to a device and verification that no further problems exist is to be made.

7.1. DH11 Testing

To test the DH11's only, type the command:

```
sh /accept/dhtest <number_dh>
```

The "number_dh" argument should be the number of DH11's that are on the system. Each DH11 is tested as indicated above. As before, a reply is required after each DH11 has been tested.

7.2. Disk and Tape Testing

To test any disk, tape, or combination of disk and tape, type the command:

```
sh /accept/blkdev <loopcount> <fromdev> <todev>
```

The "loopcount" argument indicates how many passes are to be made through the test. The "fromdev" argument specifies one of the devices to be used in the test. Note that the "todev" argument is optional. If no "todev" argument has been supplied, then insofar as possible, the test will be confined to the "fromdev". Both reads and writes will be done on the device. If the optional third argument "todev" has been supplied, however, the majority of the test will consist of reading the "fromdev" and writing the results on the "todev".

Disk and tape devices are specified by generic names followed by the applicable unit number. These generic names

A5

are as follows:

Device	Generic
RK05	rk
RP03	rp
RP04	hp
RP05	hp
RS03	hs
RS04	hs
TU10	tape
TU16	tape

For example, to test the RP04 on drive zero (making five passes), one could type the command:

```
sh /accept/blkdev 5 hp0
```

To cause data to be copied from RP04 drive zero to TU16 drive zero (making eight passes), type the command:

```
sh /accept/blkdev 8 hp0 tape0
```

7.3. Memory Testing

The swap device and all of memory are exercised by invoking the synthetic job. This is done by typing the command:

```
sh /accept/synthrc
```

PROCEDURES FOR INSTALLING UNIX ON THE PDP11

Introduction

Use the following set of procedures when initially installing the UNIX software on your PDP11. Follow the procedures appropriate to your machine model and type of tape drive or disk as indicated below.

First, copy the software from magnetic tape to a disk pack. Note that inscfar as these procedures are concerned, RP04 and RP05 are synonymous terms. If the system being tested has an RP05 disk, follow the procedures given below for an RP04.

1. Copy System From Magtape to Disk

1. Mount the magtape on drive 0 at load point.
2. Mount a formatted disk pack on drive 0.

   (Unformatted disk packs can be formatted using the DEC supplied software diagnostics.)

3. Set console control knob to CONS PHY position.
4. Set console control knob to DATA PATHS position.
5. Depress HALT switch.

6A. For a TU10 magtape drive and a PDP11/40, do the following:

   a. Set console switches to starting address 100000.
   b. Depress LOAD ADRS switch.
   c. Set console switches to:
      012700 and raise DEP switch
      172526 "
      010040 "
      012740 "
      060003 "
      000777 and raise DEP switch
   d. Set switches again to starting address 100000.
   e. Depress LOAD ADRS switch.
   f. Raise the ENABLE switch.
   g. Depress the START switch.

   (The tape will move and the CPU will loop. This is not the DEC bulk ROM for tape.)

or

A7

6B. For a TU10 magtape drive and a PDP11/45, do the following:
   a. Set console switches to 772522.
   b. Depress LOAD ADRS switch.
   c. Set switches to 060003.
   d. Raise the ENABLE switch.
   e. Raise the DEP switch.

   (The tape will move.)

or

6C. For a TU16 magtape drive and either machine model 11/45 or 11/70 do the following:
   a. Set console switches to 17772442.
   b. Depress LOAD ADRS switch.
   c. Set switches to 00000000.
   d. Raise DEP switch three times.
   e. Set switches to 17772472.
   f. Depress LOAD ADRS switch.
   g. Set switches to 001300.
   h. Raise DEP switch.
   i. Set switches to 17772440.
   j. Depress LOAD ADDRS switch.
   k. Set switches to 00000071.
   l. Raise the ENABLE switch.
   m. Raise the DEP switch.

   (The tape will move.)

7. Depress the HALT switch.
8. Set switches to 00000000.
9. Depress LOAD ADRS switch.
10. Raise the ENABLE switch
11. Depress the START switch.

   (The tape will rewind, and the character "=" will print on the console terminal.)

12. Now go to the console terminal. If you detect a typing error on the console terminal while entering information and before the carriage return is typed, you can make corrections as follows:

   a. For an incorrect character discovered immediately, type the sharp-character "#" immediately after the character in error to erase it, retype the character correctly and continue making the entry.

   b. For an incorrect character discovered within the

line, type the at-character, "@" to erase all characters typed thus far and then retype the line.

Proceed as follows on the console terminal; be sure to press carriage return after each entry except where indicated by "<NO CR>".

[you] list

[mach] m (for tm11); c (for tc11); u (for tu16)

(Type of Tape Controller)

[you] m or c or u <NO CR>

[mach] (The machine will list all the files on the front of the tape. The last line will look like: tape size = NN [decimal].)

[mach] =

[you] copy

[mach] p (for rp03); f (for rf); k (for rk); 4 (for rp04)

(Type of Disk Controller)

[you] p or f or k or 4 <NO CR>

[mach] m (for tm11); u (for tu16); c (for tc11)

(Type of Tape Controller)

[you] m or c or u <NO CR>

[mach] disk offset

[you] 0

[mach] tape offset

[you] (Type in the tape size number, NN, which the machine gave you earlier).

(The tape will move.)

[mach] count

[you] 4000

(Wait until the tape stops moving - about three minutes.)

[mach] =

13. Set console switches to 17773030

14. After the tape rewinds, return to the console terminal and continue. Press carriage return after each entry except where indicated by "<NO CR>".

[you] rpboot (for RP03) or rkboot (for RK05) or hpboot (for RP04/05)

(Type of Disk Drive)

[you] p or k or 4 <NO CR>

[mach] (carriage returned)

[you] /sccsunix

[mach] UNIX PG-1C300 Issue 3
mem = NNN
#

The above message indicates that UNIX is running on the disk pack. The number,NNN, indicates the amount of space available for user programs in units of tenths of K such that 12.0K words will be indicated by NNN=120.

2. Load and Execute UNIX from Disk

Use the following procedures only when the UNIX system needs to be replaced in core from a disk already containing the proper file system, boot program, special files, etc. These procedures assume that your machine has been powered up using the DEC procedures for your particular PDP11.

1. Turn console control knob to CONS PHY position.
2. Turn console control knob to DATA PATHS position.
3. Depress HALT switch.

4A. For an RP03 disk and a PDP11/40, do the following:
   a. Set console switches to starting address 100000.
   b. Depress HALT switch.
   c. Set console switches to:
      012700 and raise DEP switch.
      176726 "
      005040 "
      005040 "
      005040 "
      010040 "
      012740 "
      000005 "
      105710 "
      002376 "
      005007 and raise the DEP switch.

      (This program corresponds to the DEC ROM.)

   d. Set switches again to the starting address 100000.
   e. Depress LOAD switch.
   f. Raise ENABLE switch.
   g. Depress START switch.

4B. For an RK disk on a PDP11/40, do the following:
   a. Set console switches to starting address 100000.
   b. Depress LOAD ADRS switch.
   c. Set console switches to :
      012700 and raise DEP switch.
      177414 "
      005040 "
      005040 "
      010040 "
      012740 "
      000005 "
      105710 "
      002376 "
      005007 and raise DEP switch.

      (This program corresponds to the DEC bulk ROM.)

   d. Set switches again to starting address 100000.
   e. Depress LOAD ADRS switch.
   f. Raise ENABLE switch.
   g. Depress START switch.

A11

4C. For an RK disk on a PDP11/45, do the following:
   a. Set console switches to address 777404.
   b. Depress LOAD ADRS switch.
   c. Set switches to 000005.
   d. Raise ENABLE switch.
   e. Raise DEP switch.
   f. Depress HALT switch.
   g. Set switches to address 000000.
   h. Depress LOAD ADRS switch.
   i. Raise ENABLE switch.
   j. Depress START switch.

4D. For an RP03 disk on a PDP11/45, do the following:
   a. Set console switches to address 776716.
   b. Depress LOAD ADRS switch.
   c. Set switches to 000001.
   d. Raise DEP switch.
   e. Depress HALT switch.
   f. Set switches to 776714.
   g. Depress LOAD ADRS switch.
   h. Set switches to 000205.
   i. Raise HALT switch.
   j. Raise DEP switch.
   k. Depress HALT switch.
   l. Set switches to address 000000.
   m. Depress LOAD ADRS switch.
   n. Raise ENABLE switch.
   o. Depress START switch.

4E. For an RP04 or RP05 disk on a PDP11/70, do the following:
   a. Set console switches to address 17765000.
   b. Depress LOAD ADRS switch.
   c. Set switches to 00000070.
   d. Raise ENABLE switch.
   e. Depress START switch.

4F. For an RS04 disk on a PDP11/70, do the following:
   a. Set console switches to 17765000.
   b. Depress LOAD ADRS switch.
   c. Set switches to 00000100.
   d. Raise the ENABLE switch.
   e. Depress the START switch.

   (The CPU will loop.)

5. Set console switches to 17773030.
6. Return to console terminal and enter the following; be sure to press carriage return after each entry, except where indicated by "<NO CR>":

[you] p (for RP); k (for RK); 4 (for RP04/05); or s (for RS04) <NO CR>

(Type of Disk Drive)

[mach] (Carriage returned by machine)

[you] /sccsunix

[mach] UNIX PG-1C300 Issue 3
mem = NNN
#

(The above message indicates that the loading of the UNIX system from disk has been completed.)

Error Conditions

err001: no argument

The name of the project (e.g. sccs) is required as the first argument to the "startup" procedure.

err002: unknown project: XXX

The project name XXX that was supplied to the "startup" procedure as the first argument is invalid.

err003: missing directory

A file system directory that is required for execution of the "startup" procedure is missing. This "can not happen".

err004: configuration unknown for XXX

Information that the "startup" procedure needs about the configuration for project XXX is missing from the file system. This "can not happen".

err005: root unknown

Identification of the root file system is not contained in the system configuration. This "can not happen".

err006: premature termination

The test procedure has been prematurely terminated by the user.

err007: missing loop count

The loop count that must be specified as the first argument to the "blkdev" procedure was not given. This value is used to govern the number of passes that are made through each test.

err008: missing dev entry for XXX

A "special" file entry for device xxx is either missing or incorrectly specified in the file system. This "can not happen".

err009: invalid loop count XXX

The loop count that was specified as the first argument to the "blkdev" procedure is not a valid number (0 through 32000).

err010: missing from device

The second argument to the "blkdev" procedure specifies the "from" device or the only device to be tested. The procedure terminated because this argument was not supplied.

err011: system not configured for XXX

The system currently under test is not configured for device type XXX.

err012: can not copy XXX to YYY

The "blkdev" procedure will not permit device XXX to be copied to device YYY. This may happen if the same "from" and "to" device were specified to the "blkdev" procedure and that device does not have the capability to support multiple logical file systems (e.g. the rk).

err013: fatal configuration error

An internal sanity check in the "blkdev" procedure failed. This "can not happen".

err014: root/swap unknown

The configuration does not contain information about the location of the root and/or the swap device. This "can not happen" if the "startup" procedure runs successfully.

err015: can not overwrite root/swap

The "to" device specified to the "blkdev" procedure would cause either the root or the swap device to be overwritten.

err016: missing number of dh

The "dhtest" procedure requires that the number of dh's on the system be specified as the first argument.

A15

err017: invalid number XXX

The first argument to the "dhtest" procedure must be the number of dh's on the system. The value that was supplied was not a valid number.

err018: missing file for dh number XXX

The "dhtest" procedure can not locate a file that is necessary to properly test dh number XXX. Currently, this "can not happen" if the system contains five or less dh's.

Sample Output For SCCS

```
% sh /accept/startup sccs
Check Root Filesystem

                  /dev/hp0
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
Phase 5 - Check Free List
  596 files      3692 blocks        226 free

startup completed
% sh /accept/fulltest 1 4

Select Valid Device Pairs for Testing

Run Block Device Tests

Test Devices hp0

*** Number Of Passes:  1
*** First Device Is:   hp0
*** Output Type Is:   dcased1


*** Pass Number  1  ***


Making file system on /dev/hp6

Checking file system on /dev/hp6

                  /dev/hp6
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
Phase 5 - Check Free List
  168 files      1764 blocks        194 free

Making file system on /dev/hp7

Checking file system on /dev/hp7

                  /dev/hp7
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
Phase 5 - Check Free List
  168 files      1764 blocks        194 free
```

A17

```
Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 1000 records of size 1024 bytes
1000+0 records in
1000+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

Dump filesystem to a disk
/dev/hp6:
incremental dump from
the epoch
168 files
1749 blocks
0.8 tapes
0 phase errors

Make filesystem

*** Test Done

Test Devices hp0 tape0
MAKE SURE TAPE(S) ARE ON DRIVE(S) 0
HIT RETURN WHEN READY TO CONTINUE TEST


*** Number Of Passes:  1
*** First Device Is:   hp0
*** Second Device Is:  tape0
*** Output Type Is:    tcasent


*** Pass Number  1  ***


***    800 bpi

Making file system on /dev/hp6

Checking file system on /dev/hp6

                  /dev/hp6
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
```

```
Phase 5 - Check Free List
  168 files     1764 blocks       194 free

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

Dump filesystem
/dev/hp6:
incremental dump from
the epoch
168 files
1749 blocks
0.0 tapes
0 phase errors

Write files (tp)
  53 entries
 747 used
 809 last
END

***  1600 bpi

Making file system on /dev/hp6

Checking file system on /dev/hp6

                 /dev/hp6
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
Phase 5 - Check Free List
  168 files     1764 blocks       194 free

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
```

```
50+0 records out

Dump filesystem
/dev/hp6:
incremental dump from
the epoch
168 files
1749 blocks
0.0 tapes
0 phase errors

Write files (tp)
  53 entries
 747 used
 809 last
END

*** Test Done

Test devices tape0
MAKE SURE TAPE(S) ARE ON DRIVE(S) 0
HIT RETURN WHEN READY TO CONTINUE TEST


*** Number Of Passes:  1
*** First Device Is:   tape0
*** Output Type Is:    tcaset0


*** Pass Number  1  ***


***  800 bpi

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out
```

```
Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

***  1600 bpi

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

*** Test Done

Test Devices tape0 hp0
MAKE SURE TAPE(S) ARE ON DRIVE(S) 0
HIT RETURN WHEN READY TO CONTINUE TEST


*** Number Of Passes:  1
*** First Device Is:   tape0
*** Second Device Is:  hp0
*** Output Type Is:    tcasetd


*** Pass Number  1  ***


***  800 bpi

Get a Tape File of 2000 Records
2000+0 records in
2000+0 records out

Copy 2000 records of size 512 bytes
```

```
2000+0 records in
2000+0 records out

Get a Tape File of 100 Records
100+0 records in
100+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
100+0 records out

Get a Tape File of 50 Records
50+0 records in
50+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

Making file system on /dev/hp6

Checking file system on /dev/hp6

                  /dev/hp6
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
Phase 5 - Check Free List
  168 files     1764 blocks       194 free

Get files from tape (tp)
  58 entries
 855 used
 917 last
END
END

***  1600 bpi

Get a Tape File of 2000 Records
2000+0 records in
2000+0 records out

Copy 2000 records of size 512 bytes
2000+0 records in
2000+0 records out

Get a Tape File of 100 Records
100+0 records in
100+0 records out

Copy 100 records of size 10240 bytes
100+0 records in
```

```
100+0 records out

Get a Tape File of 50 Records
50+0 records in
50+0 records out

Copy 50 records of size 20480 bytes
50+0 records in
50+0 records out

Making file system on /dev/hp6

Checking file system on /dev/hp6

                /dev/hp6
Phase 1 - Check Blocks
Phase 3 - Check Pathnames
Phase 4 - Check Reference Counts
Phase 5 - Check Free List
  168 files     1764 blocks        194 free

Get files from tape (tp)
  58 entries
 855 used
 917 last
END
END

*** Test Done

Test DH lines

TEST DH NUMBER 1
HIT RETURN WHEN DONE TO PROCEED


TEST DH NUMBER 2
HIT RETURN WHEN DONE TO PROCEED


TEST DH NUMBER 3
HIT RETURN WHEN DONE TO PROCEED


TEST DH NUMBER 4
HIT RETURN WHEN DONE TO PROCEED


Test Memory
compile synthetic job
synthetic job ready
synthetic job completed
```

ACCEPTANCE TESTS COMPLETED
%

Attachment B

Disk And Tape Test Files

Disk

```
From Device != Root or Swap
        To Device Not Given          ==> dcased0
        To Device Given              ==> dcased1
From Device == Root
        To Device Not Given          ==> dcaser0
        To Device Given              ==> dcaser1
From Device == Swap
        To Device Not Given          ==> dcases0
        To Device Given              ==> dcases1
```

Tape

```
From Device != Tape
        From Device Not Root/Swap    ==> tcasent
        From Device Is Root          ==> tcasert
        From Device Is Swap          ==> tcasest
From Device == Tape
        To Device Not Given          ==> tcaset0
        To Device Not Tape           ==> tcasetd
        To Device Is Tape            ==> tcasett
```