PWB/UNIX Operations Manual

M. E. Pearlman

Bell Laboratories
Piscataway, New Jersey 08854

ABSTRACT

This manual contains a complete description of console operations, as well as step-by-step instructions for normal operator functions.

The information in this manual was gathered from personal experience, *PWB/UNIX* User's Manual* pages, and Digital Equipment Corporation hardware manuals and microprogram flow diagrams.

This manual is intended to be as general as possible, in order to serve for all PWB/UNIX systems. It is suggested that each location add specific information about:

a. Hardware configuration.
b. Telephone line configuration.
c. Specific logging and record-keeping practices.
d. Contacts for hardware and software problems.
e. Site-dependent diagnostic procedures.

* UNIX is a Trademark/Service Mark of the Bell System.

HARDWARE OPERATION — PDP 11/45, /70

PDP 11/45, /70 CONSOLES

HARDWARE OPERATION — PDP 11/45, /70

CONSOLE DESCRIPTION AND OPERATION

The following documentation is primarily intended to describe the PDP® 11/70 console and its operation. Differences in the PDP 11/45 appear within brackets ‘‘[ ]’’. Those cases that are applicable to only one of the two systems are clearly labeled as such.

PDP 11/45, /70 CONSOLE DESCRIPTION

The PDP 11/45 and 70 consoles are composed of the following:

1. Power Key Switch (OFF/POWER/LOCK)
2. Mapping Lights (<u>11/70 only</u>).
3. Seven Execution Indicator Lights displaying the following Central Processor (CPU) States: RUN, PAUSE, MASTER, USER, SUPERVISOR, KERNEL, DATA.
4. A 22-bit [18-bit][1] Address Register Display.
5. A 16-bit Data Register Display.
6. An Addressing Error (ADRS ERR) Indicator Light.
7. Parity Error (PAR ERR) Indicator Light (<u>11/70 only</u>).
8. High and Low Parity Indicator Lights (<u>11/70 only</u>).
9. A 22-bit [18-bit] Switch Register (CSW).
10. Control Knobs:

    Address Display Select which can be set to any of 8 positions:

    a. USER I
    b. USER D
    c. SUPERVISOR I
    d. SUPERVISOR D
    e. KERNEL I
    f. KERNEL D
    g. PROGRAM PHYSICAL
    h. CONSOLE PHYSICAL

    Data Display Select which can be set to the following 4 positions:

    a. DATA PATHS
    b. BUS REGISTER
    c. FPP µ ADRS/CPU µ ADRS
    d. DISPLAY REGISTER

11. Lamp Test Switch
12. Control Switches which are labelled:

    a. LOAD ADRS (load address)
    b. EXAM (examine)
    c. CONT (continue)
    d. ENABL/HALT (enable)
    e. S-INST/S-BUS CYCLE (single instruction/bus cycle)
    f. START

[1] This documentation shows differences for the PDP 11/45 within brackets ‘‘[ ]’’.

g. DEP (deposit)
h. REG EXAM (register examine, 11/45 only)
i. REG DEP (register deposit, 11/45 only)

CONSOLE OPERATION — SWITCH REGISTER

The Switch Register consists of 22 [18 on the 11/45] switches labelled 0 through 21[17] (numbers correspond to bit positions). They are used to manually enter both addresses and data into the processor.

To enter an address such as 173020_8, the bits must be divided into groups of three starting from the right. Bits 00-02 in the first group, bits 03-05 in the second group, 06-08 in the third, 09-11 the fourth, 12-14 in the fifth, etc. In each group of 3 bits an octal digit is indicated as follows:

zero	All 3 switches down
one	Lowest numbered switch (right-most bit) up.
two	Middle switch up.
three	Right and middle switches up.
four	Left switch only, in group of three, up.
five	Left and right switches up.

six	Left and middle switches up.
seven	All 3 switches up.

HARDWARE OPERATION — PDP 11/45, /70

The arrows on the following two diagrams indicate which switches should be up to set 173020_8 in both the 11/70 and the 11/45.

one seven three zero two zero

one seven three zero two zero

CONSOLE OPERATION — LIGHT TEST SWITCH

Depressing the Lamp Test switch should light all lamp indicators on the console. Any burnt-out lamps should be replaced at the next preventive maintenance session.

CONSOLE OPERATION — POWER KEY SWITCH

The Power Key controls CPU power in the following manner.

OFF	Power is off for CPU.
POWER	Power is on for CPU. <u>This is the normal position</u> to enable use of all console controls.
LOCK	Power is on for CPU. Disables all control switches except for the Switch Register, Data Select, Address Select, and Power Key.

CONSOLE OPERATION — STARTING AND STOPPING

Starting
Once the power is on, execution can be started by setting ENABL/HALT toggle up, putting the starting address in the Switch Register, and depressing the LOAD ADRS Switch. Verify in the Address Register Display lights that the address was entered correctly, then depress START. The computer system RESETS and begins execution of the program instructions at the address specified by the current contents of the Address Display. Depressing start again has no effect, when the CPU is in the RUN state.

When START is depressed with the ENABL/HALT toggle in the halt position (down) the system will RESET, but not commence execution.

Stopping
Depressing the ENABL/HALT toggle will HALT execution.

Continuing
After the computer has been halted, execution can be resumed at the same point it was when halted by depressing the CONT switch. If still in the halted state CONTINUE will single cycle (either by CPU instruction or by Bus[2] cycle) with each depression (depending on the setting of the S-INST/S-BUS CYCLE switch). If ENABLE is set, normal execution will resume.

CONSOLE OPERATION — CONTROL SWITCH FUNCTIONS

LOAD ADRS loads the contents of the Switch Register into the CPU and displays it in the Address Display. The address displayed in the Address Display depends on the position of the Address Select knob.

The EXAM Switch being depressed causes the contents of the location specified in the Address Display to be displayed in the Data Display lights when the Data Select knob is in the DATA PATHS position.

The DEP switch being raised causes the current contents of the Switch Register to be deposited into the address specified by the current contents of the Address display.

CONT being depressed after the system is halted, causes the CPU to resume execution as described in the previous section on STARTING AND STOPPING.

ENABL/HALT is a two position switch whose functions also have been described in the previous section.

S/INST-S/BUS affects only the operation of the CONTINUE switch. It controls whether execution stops after a single INSTRUCTION or a single BUS CYCLE. This switch has no effect when CPU is in the RUN state.

START starts execution or RESETS depending on the setting of the ENABL/HALT switch, as previously described.

REG EXAM (11/45 only) depressed causes the contents of the General Purpose Register specified by the low order 4 bits of the Bus Address Register to be displayed in the Data Display lights.

REG DEP (11/45 only) raised causes the contents of the Switch Register to be deposited into the General Purpose Register specified by the current contents of the CPU Bus Address Register.

FOR FURTHER INSTRUCTIONS SEE THE PDP 11/45 PROCESSOR HANDBOOK.

CONSOLE OPERATION — ADDRESS SELECT KNOB AND DISPLAY REGISTER

Address Select provides an interpretation for the Address Display Register when set as follows:[3]

[2] The *Bus* (or UNIBUS) is the primary control and communications path connecting most of the PDP 11 system's components and peripherals.

VIRTUAL — (6 positions for USER, SUPERVISOR, and KERNEL) Indicates the current address as a 16-bit Virtual address (when the Memory Management Unit is turned on, i.e. UNIX is running, otherwise it indicates the true 16-bit Physical Address). FOR DEBUGGING ONLY

PROGRAM PHYSICAL — Displays a true 22-bit [18-bit] Physical Address for the current address reference. ALSO FOR DEBUGGING

CONSOLE PHYSICAL — Display a 22-bit [16-bit] Physical Address to be used for ALL console operations such as LOAD ADRS, EXAM and DEP.

CONSOLE OPERATION — DATA SELECT KNOB AND DISPLAY REGISTER

The Data Select Knob controls the contents of the 16-bit Data Display Register according to the following settings:

DATA PATHS — The normal mode when performing console operations. Shows examined or deposited data.

DISPLAY REGISTER — Displays the current contents of the 16-bit write only ‘‘Switch Register’’. This is the normal position for the Data Select Switch when UNIX is running.

BUS REG — Displays the contents of the Bus Register.

µ ADRS FPP/CPU — Bits 15-08 display the current ROM[4] address of the Floating Point Processor microprogram.
Bits 07-00 display the current ROM address of the CPU microprogram.

CONSOLE OPERATION — DISPLAYING DATA AT A SPECIFIC ADDRESS

To display the data contained at address X_8:

1. Set Power Key to POWER position.
2. Set Address Select Knob to CONSOLE PHYSICAL.
3. Set Data Select Knob to DATA PATHS.
4. Enter X_8 into the Switch Register.
5. Depress LOAD ADRS.
6. Depress EXAM.
7. The Data Display lights will contain the data from address X_8.
8. Depressing EXAM again will display the data from address $X+2_8$.

CONSOLE OPERATION — EXECUTION INDICATOR LIGHTS

Lighting of each display indicates the following:

RUN — The processor is executing instructions, including WAIT instructions, but is not in a Pause cycle.

[3] This documentation shows differences for the PDP 11/45 within brackets ‘‘[ ]’’.

[4] *ROM* is an abbreviation for ‘‘read only memory’’.

PAUSE — The processor is in a Bus Pause or Interrupt Pause Cycle, waiting for a Unibus device or Memory.
Or, the CPU has been halted from the console.

MASTER — The CPU is in control of the Unibus or during Console operations.

KERNEL SUPER USER — These lights show where the current memory reference is taking place.

DATA — Shows whether I (program) or D (data) space is being used in the current memory reference. It is on when D space is used and off when I space is used.

CONSOLE OPERATION — ADDRESS ERROR INDICATOR

ADRS ERR is on when an addressing error occurs. (e.g., non-existent memory, access control violation, reference of unassigned memory pages).

CONSOLE OPERATION — PARITY AND PAR ERR INDICATORS (11/70 only)

PARITY indicators display the parity bits associated with the HIGH and LOW bytes of the word currently read from Cache memory. Indicators are off during write operation.

PAR ERR indicator is lit when a Unibus or a memory parity error is flagged.

CONSOLE OPERATION — MAPPING INDICATORS (11/70 only)

The 16, 18, and 22 indicate the bits used for Memory Management mapping for each cycle.

TAPE DRIVE OPERATION — TU10

Please file Chapter 3 of DEC®MAINTENANCE MANUAL DEC-00-TU10S-DC here.

TAPE DRIVE OPERATION — TU16

Please file Sections 1.7 and 1.8 of DEC MAINTENANCE MANUAL EK-TU16-MM-001 here.

HARDWARE OPERATION — PDP 11/45, /70

DISK DRIVE OPERATION — RP03

Please file Section 3 of DEC ISS MANUAL UD002341-1 here.

HARDWARE OPERATION — PDP 11/45, /70

DISK DRIVE OPERATION — RP04

Please file Section 3 of DEC ISS MANUAL UD002511-1 here.

HARDWARE OPERATION — PDP 11/45, /70

DISK DRIVE OPERATION — RP05

Please file ‘‘operation’’ section of DEC ISS MANUAL for RP05 disk drive here.

DISK DRIVE OPERATION — RP06

Please file ‘‘operation’’ section of DEC ISS MANUAL for RP06 disk drive here.

BOOT PROCEDURES

BEFORE BOOTING

Make sure the system is *idle.*[5] If *UNIX* is running, the system should be cleared via the UNIX SHUTDOWN PROCEDURE.

The normal boot performed is a ROM BOOT. The TAPE BOOT is performed (on the 11/45 only) when loading a special system, or when a ROM BOOT is impossible.

When performing any boot procedure the Power Key Switch should be in the *POWER* position, the Address Select Knob should be in the *CONSOLE PHYSICAL* position, and the Data Select Knob should be in the *DATA PATHS* position.

Before performing any procedure, please read that procedure completely.

PDP 11/45 DISK BOOT

A. Halt system (set *ENABL*/*HALT* toggle down).

B. Depress *START* (this resets the system).

C. Set CONSOLE ADDRESS SWITCHES (*CSW*) to the address of your ROM or Boot Strap Loader (e.g., 173020_8).[6]

[5] The system is *idle* when no *user* processes are running.

[6] See *romboot*(VIII) in the PWB/UNIX User's Manual for further details.

D. Depress *LOAD ADDRESS* toggle.

E. Set *ENABL/HALT* toggle up.

F. Set console switches to appropriate settings.

   1. For *SINGLE-USER*, set 173030_8 → *CSW* .

   2. For *MULTI-USER*, leave *CSW* at original settings.

G. Depress *START* .

H. At console, enter:[7]

```
# 00 = unixⒸ
```

You enter the two zeroes, the system name,[8] and the Ⓒ, all on the same line as the # .

[7] Information or commands entered by the operator appear underlined. The symbol Ⓒ is used throughout this procedure to indicate a *carriage return*.

[8] If you are booting a special system, the name of that special system may be substituted for ‘‘unix’’. See *diskboot*(VIII) for complete boot instructions.

BOOT PROCEDURES

PDP 11/70 ROM BOOT

A. Halt system (set *ENABL/HALT* toggle down).

B. Depress *START* (this resets the system).

C. Set CONSOLE ADDRESS SWITCHES (*CSW*) to 165000_8.

D. Depress *LOAD ADDRESS* toggle.

E. Set *ENABL/HALT* toggle up.

F. Set *CSW* to 000000_8 (all CONSOLE ADDRESS SWITCHES down).[9]

G. Depress *START*.

H. Wait for console to respond with a # .

[9] If the default boot address on your ROM is not set to the correct boot device, entering zeroes will fail. See *11/70boot*(VIII) for complete instructions.

BOOT PROCEDURES

I. Set console switches to appropriate settings:

1. For *SINGLE-USER*, set 173030_8 → *CSW* .

2. For *MULTI-USER*, leave console switches at original settings.

J. At console, enter:

00 = unixⒸ

You enter the two zeroes, the system name,[10] and the Ⓒ all on the same line as the # .

[10] If you are booting a special system, the name of that special system may be substituted for ‘‘unix’’. See *diskboot*(VIII) for complete boot instructions.

BOOT PROCEDURES

11/45 TAPE BOOT

A. After the system is halted (*ENABL/HALT* toggle down), depress *START* (this resets the system).

B. Mount the appropriate tape on the tape drive (make sure tape is ready at *load point*).

C. Set the address of your ROM or Boot Strap Loader in the console switches (e.g., 173000_8).

D. Depress *LOAD ADDRESS* toggle.

E. Set *ENABL/HALT* toggle up.

F. Set console switches to appropriate settings:

   1. For *SINGLE-USER*, set 173030_8 → *CSW*.
   2. For *MULTI-USER*, leave console switches at original settings.

G. Depress *START* (system will give you =).

H. At console, enter:

   1. If loading *UNIX*

      unixⒸ

   2. If loading a special system

      ‘‘filename’’Ⓒ
      (where *filename* is the name of the system being loaded, e.g.,‘‘pwbx0409’’)

BOOT PROCEDURES

GENERAL PROCEDURES AFTER BOOTING

After the boot is completed, UNIX will print a ‘‘restricted rights’’ message and tell you the System Name and the amount of memory available. The latter should be checked against a previous boot to be sure that all of memory is available. Otherwise, serious problems can occur.

A. If booting as part of *SYSTEM RECOVERY*, continue SYSTEM RECOVERY PROCEDURE.

B. If bringing up *UNIX MULTI-USER*, perform UNIX INITIALIZATION PROCEDURE.

OPERATOR COMMANDS

MOUNT AND UMOUNT COMMANDS

The **mount** and **umount** commands are used to logically **mount** or **unmount** filesystems. The **mount** command without any arguments is used to list which filesystems are *mounted* and on which *device* they are located. This aspect of the command will not give accurate data when performed on a SINGLE-USER system.

The format for these commands is:

```
# mount
```

Output will be formatted:
``*fsname* on *devname permissions* on *day date time year*''

An example would be:

```
``/a3 on /dev/rp5 read/write on thu may 15 17:43:51 1976''
```

or

```
# mount devname fsname option
```
[11]

or

```
# umount devname
```

devname = the logical device name (e.g.,``/dev/rp12'')

fsname = the filesystem name that you wish this device to be called (e.g.,``/f3'' or ``/bck'')

permissions = *read only* or *read/write* permissions are granted.

option = the optional ``–r'' for *read only* (this field is left blank for write capabilities)

Before **umount**ing any filesystems, it is a good practice to perform a **mount**, without arguments, to know which devices you will have to re-mount, and what they must be named.

If the **umount** command complains that the device is *busy*, it means that someone is using the filesystem (maybe you). Change directory (**chdir**(I)) to ``/'' and try again.

[11] NOTE: Before **mount**ing a filesystem, be sure that a filesystem with the same name (i.e., *fsname*) is not already mounted .

DEVNAME (PHYSICAL VERSUS BLOCK)

Block device names are used in commands which must have data passed through the system buffers (e.g., **mount**, **umount**). They are formatted:

‘‘/dev/rp12’’ or ‘‘/dev/mt0’’

Physical device names should always be used for *file maintenance* commands (e.g., **vc10**, **vc88**, **vc50** and **check**). Use of Physical device names makes the transfer of data ‘‘device-to-device’’, thereby shortening execution time considerably. These are formatted:

‘‘/dev/rrp12’’ or ‘‘/dev/rmt0’’

(note the extra ‘‘r’’ in ‘‘rrp12’’)

Physical device names should be used only in referencing logically **umount**ed filesystems, since changes made to the physical device may not be reflected in the systems in-core image of the device. The exception to this rule is the ‘‘root’’ device, which is never **umount**ed.

OPERATOR COMMANDS

CHECK AND ICHECK

The **check** command is used to check filesystems for possible file damage. For example, **check** should always be used after a system crash.

The **check** and **icheck** commands are formatted:

```
# check option devname devname ...
```

and

```
# icheck option devname
```

Leaving the the *devname* field blank for **check** will default to a **check** of all filesystems listed in the file ‘‘/etc/checklist’’.

option = option field.[12]

The – *s* (for salvage) is used with **icheck** to create a new *free list* for the filesystem specified; **icheck – sX** (where ‘‘X’’ is 3 for RP03s, and 4 for RP04s and RP05s) should always be performed on an **umount**ed filesystem (except for the root which is always mounted); **icheck – sX** of the root must be followed by an IMMEDIATE reboot.

devname = physical device name

For an example, the *root* filesystem for a system might be located on /dev/rrp0. If you wanted to check only that particular filesystem you would type:

```
# check /dev/rrp0
```

On the other hand:

```
# check
```

would check all of them.
An example of what the normal output of check should look like is:

```
# check /dev/rrp0

    /dev/rrp0:
    spcl        155
    files       253
    large        85
    direc        18
    indir        85
    used       2849
    last       6681
    free       3698
```

[12] An explanation of all possible options can be found in the description of *check* (VIII) and *icheck* (VIII)

The two examples below illustrate errors of moderate impact and also non-disastrous errors called *100s* and *201s*. They are not particularly dangerous, and the system can run with them present, but they should be repaired before opening the system to users.

```
# check /dev/rrp0

    /dev/rrp0:
       372 100
    spcl           155
    files          253
    large           85
    direc           18
    indir           85
    used          2849
    last          6681
    free          3698
```

```
# check /dev/rrp0

    /dev/rrp0:
       372 100
       465 201
       379 201
    spcl           155
    files          253
    large           85
    direc           18
    indir           85
    used          2849
    last          6681
    free          3698
```

The following errors should be repaired before users log on via **icheck**; they indicate errors in the *free list*. These types of errors leave the system virtually useless.

```
# check /dev/rrp0

    /dev/rrp0:
    Bad freeblock
    15 dups in free
    2005 missing
    spcl           155
    files          253
    large           85
    direc           18
    indir           85
    used          2849
    last          6681
    free          3698
```

These errors illustrate *Errors of Severe Impact* (*377s* and *177s*) and if encountered, they should be repaired immediately via **patchup**. It is disastrous to run with such errors outstanding, since these types of errors will spread throughout the filesystem.

```
#  check /dev/rrp0

   /dev/rrp0:
     675 377
     496 177
   spcl         155
   files        253
   large         85
   direc         18
   indir         85
   used        2849
   last        6681
   free        3698
```

There are many other types of *serious* errors that can appear in your **check** output. The repair of these errors is too complicated to be described in this document.[13]

Phone numbers to call when problems arise can be found in the CONTACTS section of this book.

[13] For more details on repairing file system errors, see *Repairing Damaged PWB/UNIX File Systems* by P. D. Wandzilak.

UNIX SYSTEM RECOVERY PROCEDURES

AFTER ALL SYSTEM CRASHES

A. Note all error indications, hardware or software:

for example;

1. CPU looping or halted.
2. Power on?
3. Error messages on console.
4. RP04 *DISK UNSAFE.*

B. Shutdown the System to all users.

1. Halt System, by pressing *halt* switch down.
2. Busy out (push buttons in) dial up lines for that system (check labels on *data-set* consoles).

C. Attempt System Core Dump Procedure.

1. Make sure the *KEY* is in the *POWER* position (not *LOCK*)
2. Mount dump tape with write-ring on the tape drive. Make sure the tape is *ONLINE* and ready at *load point.*
3. Depress *START*.
4. Set 044_8 → *CSW*.
5. Depress *LOAD ADDRESS* switch down.
6. Set *ENABL/HALT* switch up.
7. Depress *START* (dump will be written to tape).
8. When dump tape has rewound, unmount dump tape; remove write-ring; and label the tape with the following information:

   a) "CORE DUMP"

   b) date

   c) time

   d) System

   e) Console messages (e.g., KA6 # , APS # , and trap type).

D. Reboot system *SINGLE-USER*, via ROM BOOT PROCEDURE.

E. Do a *CHECK* of the filesystems, via **check**.

```
# check
```

F. Perform FILE PATCHING PROCEDURE.

G. Reboot the system for ‘MULTI-USER’ (via either the *ROM BOOT* Procedure or changing the console switches to any other value other than the previous setting, followed by typing a ‘‘CTRL/d’’).

H. Bring up *UNIX* via UNIX INITIALIZATION PROCEDURE.

I. Record downtime in SYSTEM DOWN LOG.

UNIX SHUTDOWN PROCEDURE

TO SHUTDOWN THE SYSTEM

A. Make sure system is *idle* (all users logged-off). This can be accomplished by executing the **shutdown** program.[14]

```
login: root

password: !;%&?

# shutdown
```

If the shutdown program fails, the following steps can be used instead.

1. *BROADCAST* a message until all users, who can, are logged-off.

```
    # wall FILENAME
    (to broadcast the contents of a file)
OR

    # wall
    The body of you message should be typed here.
    ‘‘CTRL/d’’
```

2. Busy out (push in buttons) the dial-up lines for the appropriate system(s).

3. Halt *HASP* or *HASP2*, if applicable.

```
# setuid hasp

% /usr/hasp/hasphalt

# setuid hasp2

% /usr/hasp2/hasp2halt
```

(Systems will respond with either *hasp halted* or *hasp2 halted* or both)

4. On systems with UNIVAC RJE.

```
# setuid uvac

% /usr/uvac/uvachalt
```

(system will respond with *uvac halted*)

5. Do a **ps a** to find out the process number of the *CRON*, then, **kill** the *CRON*, if it is active, and any other active processes other than the **ps a** and your terminal process.

```
# ps a
```

(system will give you a list of active processes)

```
# kill -9 process#
```

(where *process#* is the decimal process number of each active process)

6. Enter

```
# sync;sync;sync
```

B. Halt system (set *HALT* toggle down).

C. If turning off power, make sure the disk drives have been stopped before turning CONSOLE POWER KEY to the OFF position.

[14] Commands actually entered by the operator appear underlined.

FILE PATCHING PROCEDURE

After a system crash, there are several types of damage that can disable a file. Some of these errors must be repaired immediately before the disease spreads throughout the entire filesystem. A complete analysis of these errors can be found in the explanation of **check** in the COMMANDS section of this manual.

ERRORS OF SEVERE IMPACT

Errors such as *377s*, *177s*, and those with a format of:

nnn complaint; i = xxxx (class)[15]

should be treated immediately. Errors in the *root* should be handled first, because the commands *you* execute are in the *root.*

377s and *177s* can be handled by the program **patchup**.

```
# patchup
```

The *complaint* types of errors are too complicated to have established handling procedures, so an immediate telephone call to SYSTEM SUPPORT is recommended.

ERRORS OF MODERATE IMPACT

Errors of the following formats are often found after a crash, and indicate problems in the *super block* :

n dups in free

bad freeblock

n missing

After all other errors are repaired, these must be remedied by:

```
# icheck – sX devname        (physical device name)
```

where X is 3 for RP03s and 4 for RP04s and RP05s.

Icheck – sX must always be performed on an unmounted filesystem, except in the special case of the *root* device, which is always mounted. Errors in the *super block* of *root* MUST be repaired first and be followed by an IMMEDIATE reboot.

NON-DISASTROUS ERRORS

201s and *100s* are fairly innocuous errors. They can be repaired by the **patchup** program as described for *377s* above.

[15] A full explanation of this type of error can be found in the description of *check* (VIII).

UNIX INITIALIZATION PROCEDURE

TO OPEN THE SYSTEM TO MULTIPLE USERS

A. If running *SINGLE-USER:*

1. Do a **CHECK** of the filesystems via **check** (if the filesystems are not clean, refer to FILE PATCHING PROCEDURE).

```
# check
```

2. Set date to the correct time.

3. Reboot by changing the position of the *CSW* to any position other than the original setting, and entering *CTRL/d*.[16]

B. When running *MULTI-USER* log on as *SUPER USER*.

- login: root
- password: !;%&?   (system will give you a #)

If you are running *SINGLE-USER* you automatically get back the # .

C. ONLY IF the filesystems are clean, to open the system for general use (i.e., to allow users to log on):

1. Start **/ETC/STARTUP**.

```
# startup
```

This command alone will activate the processes for each of the systems. The file ''/etc/startup'' should contain any commands and cleanup to be done on each reboot.

2. Do a **ps a** to make sure *HASP*, *CRON*, *HASP2*, and *UVAC* (if applicable) are active.

```
# ps a
```

3. Unbusy the appropriate dial-up lines for each system.

4. Log-off (*CTRL/d*).

[16] Do not do a *CTRL/d* after an *icheck -sX* of the *ROOT* device; instead the system MUST be halted and completely rebooted.

FILESAVE PROCEDURES

HOW TO MAKE BACKUP PACKS

A. Before initiating **filesave**, check the *mail* for the *root* to make sure there are no files to be restored. Restore any files at this time, because when you execute the filesave you may be destroying the only good backup you have for a file. For instructions on how to restore a file, see the FILE RESTORE PROCEDURE section of this manual.

B. Make sure the system is *idle* (no user processes active). This can be accomplished by executing the **shutdown** program.[17]

```
login: root

password: !;%&?

# shutdown
```

If the shutdown program fails, an explanation of the steps that can replace it can be found in the UNIX SHUTDOWN PROCEDURE section of this book.

C. Halt the system (set *HALT* toggle down).

D. Reboot the system *SINGLE-USER* via ROM BOOT PROCEDURE.

E. Perform file maintenance.

1. Make any necessary patches to the filesystems (see FILE PATCHING PROCEDURE).

2. Get packs and tapes for the File Save from the tape/disk cabinets using the oldest dated disk packs first. To determine the appropriate disk and tape, check your Disk Log and Tape Log.

3. Start FILESAVE *shell* procedure.

```
# filesave.n
```

(where *n* is the system identification letter, e.g., *a*)

Allow the filesave to run to completion. If any part of the file save fails, re-execute starting from that point.

4. Record all disk packs and tapes used on the log sheets.

F. Bring up UNIX MULTI-USER via the UNIX INITIALIZATION PROCEDURE.

[17] Commands actually entered appear underlined.

TAPE BACKUP PROCEDURES

In addition to the backup packs created in the **filesave**, tape backups for each filesystem should be created once a week. One filesystem fits on each tape used, provided you are using your tape media at 1600 bpi. These tapes serve as long-term storage for files.

The best way to make a backup tape is to make a copy of a Backup disk. This way tapes can be made while the system is available to the users.

HOW TO MAKE A BACKUP TAPE

A. A backup pack for the filesystem of which you wish to make a tape copy must be physically mounted on the spare drive. If no spare drive is available, tapes must be made directly from the filesystem that is normally on-line, on a *single-user* system. This can be done immediately after the **filesave.**

B. Tapes and packs for each backup should be found ahead of time. Tapes should not be reused before approximately 6-8 weeks. The correct backup pack and tape to use can be determined by checking the Disk and Tape Log(s).

C. To create a backup tape for a filesystem a **vc10** is performed (see **volcopy**(VIII)).[18] The type of device you copy to depends on the type of tape drive used:

For a 800 bpi(TU10) tape drive, use:

```
vc10 fsname /dev/rrpX n packid /dev/rmt0 tapeid
```

For a 1600 bpi(TU16) tape drive, use:

```
vc10 fsname /dev/rrpX n packid /dev/rmt1 tapeid
```

- *fsname* is the filesystem name (e.g., *a1* or *root*) .
- *X* is the number of the disk drive on which the *fsname* or its backup is mounted.
- *n* is the number of the section of the pack on which the filesystem resides (e.g., *0*, *1*, *2*, *3*, *4*, *5*, or *6*).
  Examples:

/dev/rrp14	section 4 on drive 1
/dev/rrp41	section 1 on drive 4
/dev/rrp0	section 0 on drive 0
/dev/rrp65	section 5 on drive 6
/dev/rrp64	section 4 on drive 6

- *packid* is the five-character external pack label (e.g., p0452).
- *tapeid* is the five-character external tape label (e.g., t0352).

Example:

```
vc10 root /dev/rrp0 p0450 /dev/rmt1 t0432
```

D. After backup tapes are created, packs should be returned to original locations and the tape identification numbers should be recorded in your Tape Log.

[18] During the creation of a backup tape errors may occur. If ‘‘read errors’’ occur, the reliability of the disk pack is questionable. In any case the backup tape will be bad. Detailed description of these errors may be found below under RESTORING FILES FROM TAPE.

FILE RESTORE PROCEDURE

During a system crash, files may be lost. Users may also accidently destroy their own files. In either event, the user will probably call or mail to the *root* to request that his/her file(s) be restored. When the user calls, make sure you get the complete *pathname*, the time and date of the desired restoral source, and the approximate size of the file. *Pathnames* are composed as follows:

/fsname/dirname/filename

fsname = the *filesystem* name (e.g., *b3*).

dirname = the complete directory name, including all sub-directories between the file and the *fsname* (e.g., *source/s2* or *actg*).

filename = the actual file name.

The operator should perform any outstanding file restores before all filesaves that could destroy the only good restoral source.

There are, at present, two basic methods of restoring files: (1) from tape, or (2) from disk. The latter method is preferred, if you have a pack that contains a copy of the file.

RESTORING FILES FROM DISK

A. Make the file accessible on an available spare disk drive, by performing the following steps:

1. Carefully, physically mount the backup pack (found by checking your Disk Log) on the drive and bring it to *READY*.

2. **Mount**, as *read only*, the device on which the backup filesystem resides. In the **mount** command *devname* is the block device name of the filesystem being mounted (e.g., */dev/rp12*). The filesystem should be **mounted** as */bck* or some other currently unused mount point directory.

```
# mount devname /bck –r
```

3. Change your working directory to ‘‘/bck/*dirname*’’ (explanation of *dirname* is the same as above).

```
# chdir /bck/dirname
```

B. Check to make sure a copy of the file exists.

```
# ls –l filename
```

The date the file was last written to and the number of characters it contains are pertinent information in determining that particular validity of the copy.

C. Copy the file into the desired *pathname* (the pathname given by the user). This may be accomplished as follows:

```
# cpx filename /fsname/dirname/filename
```

D. Do a **ls – l** of the copied file.

```
# ls –l filename
```

Check this against the original **ls – l** of the file. If the number of characters agree, the copy was successful. Check for the owner and the modes of the original file.

E. Notify the user either through the UNIX mail command or by telephone.

F. Return the system to its original configuration:

1. Logically **unmount** the filesystem copy (*devname* for **umount** is also block, e.g., */dev/rp12*).

```
# chdir /
# umount devname (watch for error messages, if any occur restart from preceding step)
# mount (to make sure the spare filesystem is no longer mounted)
```

2. Log-off (*CTRL*/*d*)
3. Remove the pack using the following steps:

   a) Turn off the disk drive containing the unmounted filesystem, by pressing the button labelled *STOP*.

   b) Wait until the pack has completely stopped spinning.

   c) Carefully remove the disk pack from the drive.

RESTORING FILES FROM TAPE

A. Copy the file from tape onto a disk.

1. Find an available device name on disk, where the data from tape can be read and stored (i.e. a *devname* that is logically **unmount**ed where there is no valuable data). This usually involves physically mounting a scratch disk pack on a spare drive.
2. Find the appropriate tape determined by checking the date on the tape labels.
3. Mount the tape on the tape drive on the system where the *devname* is available.
4. **Login** as *root* and **read** the tape onto the available *devname* (physical *devname*, e.g., */dev/rrp10*).

```
    login: root

    password: !;%&?

    # vc10 fsname /dev/rmt0 txxxx devname pxxxx (for 800 bpi tapes)
or

    # vc10 fsname /dev/rmt1 txxxx devname pxxxx (for 1600 bpi tapes)
```

   Messages like *read error block N*[19] can *not* be ignored, and watch for fatal errors such as *read error – 1*; **vc10** should say *END* and tell you the number of blocks copied, when it is finished.

B. Now you have a copy on disk of the filesystem in which your file resides. You can now perform steps *A2* through *F2* of RESTORING FILES FROM DISK (without physically removing any packs that are already mounted).

[19] When this error occurs on the input device, the 10-block record currently being copied is zeroed out (because the error is probably caused by a bad block on the tape). The probability is small, but this may occur in the very file you are attempting to restore, making this tape useless for backup.

CONTACTS

This section should contain the names, addresses, telephone numbers, etc., of your installation Customer Engineers, PWB/UNIX system experts, etc.

LOCAL PROCEDURES

This section is reserved for your installation’s local procedures.

CONTENTS