SCIENCE AND ENGINEERING RESEARCH COUNCIL
RUTHERFORD APPLETON LABORATORY

COMPUTING DIVISION

DISTRIBUTED INTERACTIVE COMPUTING NOTE 746

PERQ UNIX IMPLEMENTATION NOTE # 43
Accent/Unix: Boot Sequence

Issued by
J.C.Malone

26 January 1983

---

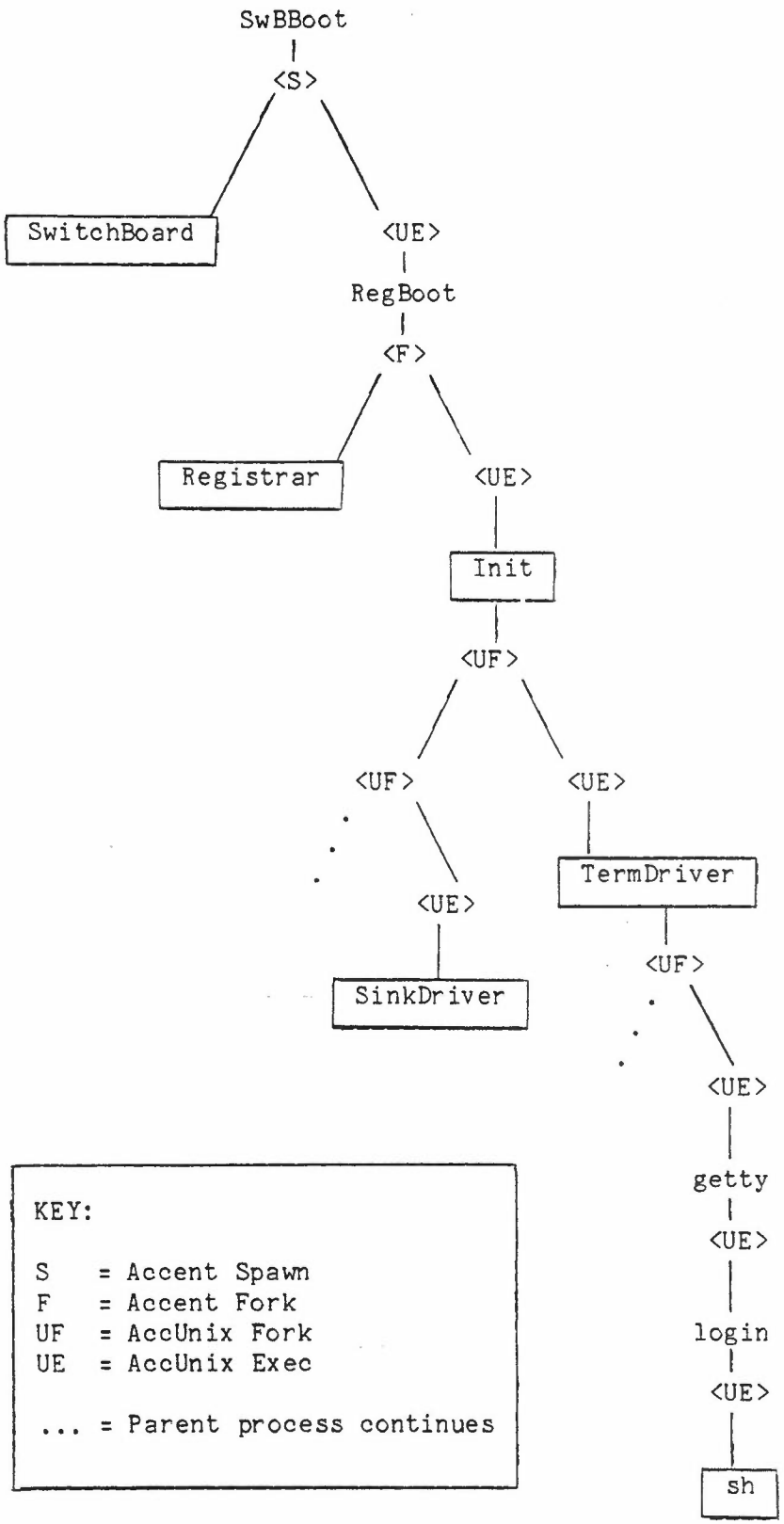
DISTRIBUTION:
R W Witty
K Robinson
C Prosser
A S Williams
L O Ford
T Watson
E V C Fielding
J C Malone
P J Smith
A J Kinroy
J M Loveluck
C P Wadsworth
Martin Ritchie (ICL Dalkeith)
P Palmer (ICL Dalkeith)
RL Support/PERQ/Unix Implementation Notes file

## Starting Up

The following diagram shows the sequence involved in bringing up the shell, after starting the AccUnix program.

Boxed processes are those which remain when the shell is running:

SwitchBoard
Registrar
Init
TermDriver
SinkDriver
sh


SwBBoot
<S>
SwitchBoard
<UE>
RegBoot
<F>
Registrar
<UE>
Init
<UF>
<UF>
<UE>
<UE>
SinkDriver
TermDriver
<UF>
<UE>
getty
<UE>
login
<UE>
sh
KEY:
S = Accent Spawn
F = Accent Fork
UF = AccUnix Fork
UE = AccUnix Exec
... = Parent process continues

SwBBoot and RegBoot are used to set up the separate processes of SwitchBoard [2] , Registrar [1] and Init.

Init starts all device drivers and remains in the background, the parent of all further processes and ready to restart any devices which terminate.

The Terminal Driver [3] starts the login session by starting Getty.

Getty sets up the root and current directories and also opens the standard I/O channels. It then tries to exec login. If login cannot be found then it execs the shell directly.

## Maintaining The System

Init repeatedly issues Wait requests, checking returned process ids to see if a driver has died and needs restarting. The Wait requests also serve to clear dead processes from Registrar's process table.

The terminal driver also checks dead children, to see whether a new login session must be started. However, since it must not hang up on a Wait call, this is done differently, see [3].

Note that at start up the device drivers and login session each request a new process group so that it is possible to kill the whole process group without killing the parent.

## Shutting Down

Init and TermDriver set up signal handlers [5] for the hang up signal, SIGHUP, to enable them to shut down cleanly.

On receiving SIGHUP, Init sends SIGHUP to each device driver. When it is sure that they have finished any clearing up (Init receives their PId in answer to a Wait call), Init sends SIGKILL to any remaining processes and then terminates.

When the Terminal Driver receives SIGHUP, it must close down all dependent processes before terminating. This is to avoid the races which would otherwise occur: on Exit processes with files open to the terminal will close them; it is possible for the Terminal Driver to receive the request, but die before answering it - this would leave the child hanging on the receive. So the Terminal Driver must reply to all requests until being told that it has no more children before terminating.

Registrar keeps a count of active processes, returning to RegBoot when there are none left. RegBoot then terminates so that the Registrar process is removed.

When SwitchBoard receives an emergency message to say that its port to Registrar has been deleted it returns to SwBBoot. This then terminates, returning the user to the POS-like environment.

Thus it is possible to bring the system down cleanly using

Kill -1 1 [4].

However, if there are any background processes with files open to the terminal, these will not be killed before the Terminal Driver has Exited - so the system will hang as these wait for their Close reply, as explained above.

REFERENCES


1. L.O. FORD, "Perq Unix Implementation Note # 42 - Accent/Unix: Registrar", DIC Note # 745, Rutherford Appleton Laboratory (December 82). [ to be published ].

2. J.C. MALONE, "Perq Unix Implementation Note # 41 - Accent/Unix: Switchboard", DIC Note # 744, Rutherford Appleton Laboratory (January 83).

3. J.C. MALONE, "Perq Unix Implementation Note # 44 - Accent/Unix: TermDriver", DIC Note # 747, Rutherford Appleton Laboratory (January 83).

4. A.S. WILLIAMS, "Perq Unix Implementation Note # 31 - UNIX System Call Specification", DCS Note # 727, Rutherford Appleton Laboratory (November 1982). [In Preparation].

5. A.S. WILLIAMS, "Perq Unix Implementation Note # 50 - Accent/Unix: Signals and deadports", DIC Note # 755, Rutherford Appleton Laboratory (December 82). [ to be published ].