Command processor
preliminary document.

Bill

I) preliminary info

The sub process structure at the command level of A process contains 4 subprocesses. The Bead ghost, which is a vestage of ~~the~~ the old bead, intercepts all of the old bead calls, ~~and~~ user errors and interrupts. BEAD Services, which does the simulation of the old bead calls, as well as other services, The line collector, which talks to the teletypes. The command processor, which handle primary control of the process in conversation with the users teletype, and which this document describes.

The command processor actually contains 3 separate programs. The command processor proper, which is used for calling subprocesses; Services, which is used for a number of utility functions; and the bead ghost, which is called by the Bead ghost subprocess to handle errors and interrupts.

2-0857

With one exception, each of these programs uses The same form of command line; a verb followed by 0 or more parameters. The verb and parameters are seperated by 1 or more blanks, and The command line is terminated by 0 or more blanks followed by a carriage return. The line may have initial blanks which are ignored,

With few exceptions, The parameters have a common structure, described below, which designate either a datum, an object or The location of a datum or object.

II) Command processor

The command processor accepts ~~one of~~ 2 types of command lines. The first type is the ~~command line~~ standard command line of a verb with 0 or more parameters. These cause special actions. First I list those which will appear in the final version.

note: in describing standard commands, I state the verb as typed, followed by ~~the type of~~ what is expected of its parameters, if any.

i) Services

Causes control to go to Services

Next I list those used in the test version only.

i) USERBUG

Calls Bruces debugger

ii) JPROC

Causes simulated JPROC return, should be done exactly once per call of XIPROC.

iii) Keith

makes debug call on Keiths LOAD/DUMP/RECOVER

iv) Bill ~~Bead type, param~~ Identifier ~~Bead type, param~~ Identifier

makes debug call on Bill Bridges Bead simulator,
(actually a Bead ~~type~~ BG=0 call on Bead ghost)

v) Crunch

Causes [illegible] a Bead BG=4 call on Fake Bead ghost, used for debugging parts of command processor subprocess. dangerous to use.

The Second type of command accepted by The command processor is a subprocess call, ~~This consists of a standard~~ This command starts with a standard parameter, naming a file containing a subprocess descriptor for The desired subprocess, This is followed by 0, 1 or 2 bead type parameters, seperated by blanks. The line is terminated by 0 or more blanks followed by a carriage return, A bead type parameter is either an identifier or an integer. (see standard parameters.)

II) Services and Bead ghost.

All command lines here are of the same form, a verb followed by 0 or more parameters. what follows is a list of the verbs, and what they expect to be provided by their parameters. Most parameters are standard. Most verbs are common to services and beadghost; some are used by one program only and are so indicated. All verbs are to be typed as written.

1.1) FIN (Services only)

returns control to the command processor

1.2) RETRY (Bead ghost only)

If the bead ghost was called by an error or interrupt, and the calling subprocess is in the middle of an XJ, Then that XJ will be repeated; otherwise same as return.

1.3) RETURN (Bead ghost only)

The subprocess calling the bead ghost continues at it's next instruction

1.5) ~~Purge~~ PURGE

reduces ~~pr~~ ~~process~~ subprocess call stack to initial value, deletes all user subprocesses.

2.1) P.ASCII

changes mode of PDATA to ~~[illegible]~~
4 bits, 7 bits, ..., 7 bits

2.2) P.FULL

changes mode of PDATA to 60 bits

2.3) P.INST

changes mode of PDATA to
15 bits, 15 bits, 15 bits, 15 bits

3.1) IN.OCT

causes all integers ~~[illegible]~~ without trailing 'D' to be read in octal.

3.2) IN.DEC

Causes all integers without trailing 'B' to be read in Decimal.

4.1) PDATA Datum

prints The datum in current print mode

4.2) PDATA Datum.LOC@ Count

prints several datum words in current print mode. An interrupt will stop The printing with no damage. (except in current test mode, while printing from a disk file.)

4.3) PCAP object

prints in octal The contents of The 2 words of The capability.

5.1) MDATA Datum Datum.LOC

moves datum to given datum loc, 1 word only.

5.2) MCAP object object.loc

moves object to given object loc, 1 object only. If The object is a disk system object, and The object.loc is a directory loc, it forms a hard link.

6.1) NEWV Identifier

Creates a variable of given name, maximum of 8 characters in the identifier, Current maximum number of variables is 10. A variable can hold either objects or data.

6.2) KILLV Identifier

Destroys named variable

7.1) ~~8)~~ VIEW ~~VIEW~~ datum

~~If doing broadcast~~
~~[illegible]~~

prints out the contents of the 3 words of a subprocess callstack entry. ~~not affected by current print mode~~
The call stack entry named depends upon whether in broadcast or services

A) Services

0	Services itself (pointer and XJ address bad)
1	~~[illegible]~~ Beads, which called services
2	builder
etc.	

B) Broadcast

-1	~~[illegible]~~ Broadcast itself (pointer and XJ address bad) (portion living in command processor)
0	The broadcast subprocess
1	calling subprocess
etc	

~~8)~~ 8.1) NEWDF Directory.Loc

Creates a disk file, of current shape, in the given directory with the given name. The access key part of the directory.loc is ignored. Makes a non scratch entry.

8.2) NEWDR Directory.Loc Datum Datum

Creates a new directory, of size given in first datum, with given name. The access key part of the directory loc is ignored. Makes a non scratch entry. The second datum is the accounting block flag.

I now list actions which are in for test purposes only. ~~Ther~~ In the final version, they will either disapear entirely or appear in heavily modified form.

T.1) USER Identifier (services only)

Sets the running user name and creates a temporary directory. Should be called once only per call of XROOT.

T.2) DEATH

destroys this user process

T.3) BUB

Calls Bead with ..STOP, (The real Bead, ~~a~~ ancestor of XROOT)

T.4) CRUNCH

Causes a Bead ..STOP call (B6=4) to be made on fake Bead ghost. Used for debugging parts of the command processor. Even more dangerous here than under the command processor.