I) preliminary info

The sub process structure at the command level of a process contains 4 sub processes. The dead short, which is a restage of the old basic interpreter, handles all of the old band calls, and user errors and interrupts. G4O services, which does the simulation of the old band calls, as well as other services, The line collector, which feeds to the teleotypes. The command process, which handles primary conversation with the user's teletypewriter, control of the process and which this document describes.

The command processor actually contains 3 separate programs. The command processor proper, which is used for calling sub processes; Services, which is used for a number of utility functions; and the break short, which is called by the dead short sub process to handle errors and interrupts.

With one exception, each of these programs uses the same form of command line, a verb followed by zero or more parameters. The verb and parameters are separated by one or more blanks, and the command line is terminated by one 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 on object or the location of a datum or object.
II) Command processor

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

Notes: in describing standard commands, I state the verb as typed followed by the format which 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.

ii) USERBEX

calls Bruces debugger

ii) JPPROC

causes simulated JPPROC Forum, should be done exactly once per call of XIRPROC.
iii) Keith

makes debug call on Keith's loop/pump/recovery

iv) Bill

Identifer

Identifer

makes debug call on Bill's loop debugger simulator

(first a void type 06=0 call on void struct)

v) Crunch

Corres 06=1 void 06=4 call on 4th void

shift, 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

This command starts with a standard parameter naming

a file containing a subprocess description for the desired

subprocess. This is followed by 0, 1, or 2 blank type parameters

separated by blanks. The line is terminated by 0 or more

blanks followed by a carriage return. A blank type parameter

is either an identifier or an integer. (See standard

parameters.)
III) Services and Bead ghost,

All command lines here are of the same form, a verb followed by one or more parameters, which 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 bead ghost; 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 sub process is in the middle of an xf,
Then that xf will be reentered; otherwise same as return.

1.3) RETURN (bead ghost only)

The sub process calling the bead ghost continues
at its next instruction
1.5) **kill proc**

reduces process sub-process call stack to initial value, deletes all user sub-processes.

2.1) **p. ASCII**

changes mode of `PORT4H` to

4 bits, 7 bits, ..., 26 bits

2.2) **p. FULL**

changes mode of `PORT4H` to 60 bits

2.3) **p. INST**

changes mode of `PORT4H` to

15 bits, 15 bits, 15 bits, 15 bits

3.1) **IN. OCT**

causes all integers without trailing '0' to be read in octal.

3.2) **IN. DEC**

causes all integers without trailing '0' to be read in decimal.
4.1) PRINT Datum

prints the datum in current print mode

4.2) PRINT Datum,LOC0 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) MOVE Datum Datum,LOC

moves datum to given datum,loc, 1 word only.

5.2) MCAP object object,loc

moves object to given object,loc, 2 objects only.
If the object is a disystem 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) `viewk` claim

prints out the contents of the 'sown' of a subprocess

call stack entry, not affected by current printmode.

The call stack entry would depend upon whether it's in `bodgnot` or `svl` services.

A) services

0  service itself (pointer and address load)
1  service's current service
2  higher

etc.

B) `bodgnot`

-1  `bodgnot` itself (pointer under address load) (partitioning in command processor)
0  `bodgnot` subprocess
1  calling subprocess

etc.
Section 5.1) 
\texttt{NEWON 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.

Section 5.2) 
\texttt{NEWON 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 size.
I now list actions which are in fact test purposes only. Then in the final version they will either disappear 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 only per call of `xroot`.

T.2) **DEATH**

Destroys this user process.

T.3) **BU**

Calls `good with --stop` (the real `good` is an ancestor of `xroot`)

T.4) **CRUNCH**

Causes a `good --stop call (OE6=4)` to be made on some dead host. Used for debugging parts of the command processor. Even more dangerous here than under the command processor.