Symbol 9
BESOS

DESIGN NOTEBOOK

APPENDIX Ø

July 8, 1965

## Plan For Using The 6.36

Following is a brief description of the steps involved in making a test run on the 6.36. The sequence is planned such that the programmer will be remotely located and will use a 1050 console in CTSS. The description of each step includes an indication of any new programs which will need to be written and on which machine these will operate. The documents required by the user or operator at each step (in addition to the CTSS manual) are also indicated. A 6.36 run may be composed of any number of symbolic BSA segments for assembly and any number of binary segments (with their linkage sections). The job may consist only of assemblies; of assemblies and a load-and-execute; or of only a load-and-execute. The programmer may request that the results of an assembly (the segment text, linkage section, debug tables, and the listing file), as well as a binary core image at abort time, be made abailable as CTSS files. He may then interrogate these files from his 1050 using CTSS commands.

1. From a 1050, using the CTSS command TYPSET, the user creates (or edits) a file $\alpha$ (MEMO) which contains a segment, named $\alpha$, in symbolic BSA. This file is a string of 12-bit characters (packed three per word). The user will need a write up of the BSA. No new programs are needed.

2. From a 1050, using the CTSS command ED, the user creates (or edits) a file σ GECOS containing the GECOS control cards for a 6.36 run. A pseudo control card INSERT β γ will cause the file β γ to be inserted in place of the control card when this file is processed (see step 3 below). A writeup of GECOS control cards and 6.36 job make up, etc., will be required. The user is responsible for inserting the required copies of BSA, the 636 simulator, etc. No new programs are required. A sample 6.36 run is shown in the Appendix.

3. From the 1050, the user issues the CTSS command,

   R 6.36 σ

   which, using the file σ GECOS, writes a GECOS input tape. A new program, in CTSS is required which scans the control cards in σ GECOS and merges in any files referred to by INSERT cards, converts from 12-bit characters to 7-bit characters (packed four per word, right adjusted in 9-bit sub-fields) any files with class name (MEMO), and writes a tape (possibly having to trick CTSS) in GECOS input tape format. A brief note on the use of 6.36 SAVED is required.

   Convert 6-bit → 6-bit for σ GECOS file.

4. The user telephones the 7094 machine room and the operator dismounts the tape from the 7094. The tape is then mounted on the 635. It becomes input to GECØs (in lieu of its normal card input) by giving a command from the 635 operators console. A writeup for the computer operator is needed.

5. The symbolic input for BSA and the binary text and linkage sections of previously assembled segments are written on the disk as permanent files, with names α BSA, β LINK, respectively, since each of these programs expect to find their input in disk files. A 635 GECOS program is needed to read binary cards from the input tape and write the contents (without check sums, etc.) on the disk in a permanent file.

6. BSA running in GECOS assembles a segment taking its input from the file α BSA created in step 5. The assembly output is left in four files on the disk, namely; α TEXT, α LINK, α LIST, α DEBUG. The program needed here is the Bootstrap Assembler written in 635 GECOS.

7. After all assemblies, if execution is requested, the GECOS job will call the loader / linker (LDRLNK). LDRLNK is a 635 GECOS program which reads a load list from a file on the disk. The load list contains a list of segments which are to be loaded, linked completely, and simulated by the 636 simulator. The first word in the load list is the execution time limit which is handed to the 636 simulator when it is called by LDRLNK. The purpose of LDRLNK is to create a 636 process. The following segments are set up ( in addition to those given in the load list): descriptor, linkage, linkage section boundary, stack, and concealed stack. All segment descriptors, and page tables are created, the descriptors put in the descriptor segment, and all links in the linkage segment are established.

8. After LDRLNK is finished it calls the 636 simulator. The simulator in a 635 GECOS program. In order to properly simulate I/Ø interrupts, the time limit on courtesy calls in GEIØS will need to be extended, A writeup on how to use GEIØS when running in the 636 simulator, especially the calls for I/Ø when referring to the pseudo-disk and pseudo-drum. A single large file on the disk and on the drum will be used as pseudo-disk and pseudo-drum on the 6.36.

9. After the test program aborts, the next activity in the run is a dump program written in 635 GECOS which dumps, in binary, the core image and machine conditions onto the 7094 output tape. In addition, the dumper consults a dump list in a disk file and dumps any additional files listed in the dump list (such as the BSA output) onto the 7094 output tape. The format of this tape is such that it is suitable for a CTSS disk update run, i.e., each file (including the core dump) written on the tape is in the form of 7-punch cards ( a CTSS binary card format used to load arbitrary files) preceeded by a CTSS file input card.

10. Dismount the 7094 output tape from the 635.

11. Mount this tape on the 7094 and immediately run a CTSS disk update run, which will copy the files on the tape into the CTSS file system, making appropriate entries in the user's file directory.

12. From the 1050, the user may then use any one of a number of analysis routines (which will need to be written) to debug the results. These programs will be such things as symbolic dump of portions of the

segments in the core image (this would be the minimum in debugging aids) or a trace back of control by unthreading the stack.

In summary, each of the above steps requires careful documentation. The following programs need writing:

1. A 7094 program to generate a GECOS input tape.
2. A 635 program to create disk files from binary cards on the input tape.
3. BSA, the bootstrap assembler (on the 635).
4. LDRLNK, the leader/linker (on the 635).
5. The 636 simulator (on the 635).
6. A 635 program to dump core and write a CTSS disk update tape.
7. A 7094 symbolic segment dump program.
8. Other debugging aids such as a subroutine call trace.

Test run of segment $\alpha$ which is in BSA and segment $\beta$ which was previously assembled.

Control Card File

```
$SNUMB
$IDENT
$OBJECT
     binary of file writter program
$DKEND
$EXECUTE
$LIMITS
$DATA  FW
```

```
**α BSA
     binary of BSA of segment α
**β TYPE
     binary of text of segment β
**β LINK
     binary of linkage section of segment β
$OBJECT
     binary of BSA
$DKEND
$EXECUTE
$LIMITS
$DISC  IN,A1,R1
$DISC  TX,A2,R1
$DISC  LK,A3,R1
$DISC  LS,A4,R1
$DISC  DB,A5,R1
$DATA
  α
$OBJECT
     binary of LDRLNK
$DKEND
$OBJECT
     binary of 636 simulator
$DKEND
$EXECUTE
```

```
$LIMITS
$DATA LL
     load list
$OBJECT
     binary of dumper
$DKEND
$EXECUTE
$LIMITS
$DATA  DL
     dump list
```