THE BEAD

THE BEAD IS THE FIRST SUBPROCESS CREATED IN EVERY PROCESS WHEN THE SYSTEM IS INITIALIZED. ITS FUNCTION IS TO ACT AS AN INTERIM MOVING PACKAGE TO COORDINATE FILE ACTIVITY AND NAMING, ELEMENTARY UTILITIES, AND BE A COMMAND PROCESSOR FOR THE USER.

EVERY BEAD IS GIVEN A CLIST WITH CAPABILITIES FOR ITS TELETEYPE AND FOR THE SYSTEM DIRECTORY. THE SYSTEM DIRECTORY CONTAINS NAMES AND TYPES OF OBJECTS IN A STANDARD FILE AND A GIANT CLIST WITH CAPABILITIES FOR THESE OBJECTS. THE NAME FILE HAS ENTRIES FORMATTED LIKE THIS:


<table>
<thead>
<tr>
<th>NAME OF OBJECT</th>
<th>NAME OF OWNER</th>
<th>TYPE/STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE/STATUS:</td>
<td>OBJECT NUMBER</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCOPE FILE INFO</th>
<th>SCOPE FILE INFO</th>
<th>CURRENT FILE POINTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOCK SIZE</td>
<td>CURRENT LENGTH OF FILE</td>
<td>NEXT BLOCK IS CREATED AT THIS ADDRESS</td>
</tr>
</tbody>
</table>

**TYPE/STATUS Fields**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSY</td>
<td></td>
<td>CURRENT ACTION</td>
<td></td>
<td>TYPE OF OBJECT</td>
<td></td>
</tr>
</tbody>
</table>

SYNTAX: set to 2 if SYSTEKT file

**CURRENT ACTION Codes:**

- 3 - WRITING  }
  used only by SCOPE
- 4 - READING  }
- 5 - REWOUND
- 6 - AT END OF INFORMATION
CALLS ON THE BEAD: (case register first)

X1 - OBJECT NAME
X2 - USER NAME or zero if current user name is requested
B1 - ADDRESS OF 4 WORD AREA
     FOR ENTRY
B7 - LIST INDEX for entry capability
B6 - ACTION

Actions

0 - LOCATE NAMED OBJECT, make BUSY.
   PUT ENTRY AT B1 AND
   CAPABILITY AT B7. IF THE OBJECT IS
   BUSY, GIVE BUSY NOTIFICATION.

1 - UPDATE OBJECT FROM
   OBJECT NUMBER field in entry
   AT B1

2 - DELETE OBJECT FROM
   DIRECTORY. OBJECT NUMBER?
   FROM ENTRY AT B1

3 - CALL SUBPROCESS. GET
   DESCRIPTOR FROM FILE NAME
   AND USER NAME in X1 and X2
Commands to the READ:

{CALL} file, user, Pan 1, Pan 2

A subprocess descriptor is obtained from the named file and is created. A CALL is then made to transfer control with the two parameters in X4 and X5 left justified display code.

{SNATCH} file, user

The named file is made NOT Busy.

{Delete} file, user

The named file is deleted.

Continue

Entered to continue anyway after a Busy notification is made.

TRY

Entered to TRY AGAIN and check if the file is still Busy after a Busy notification is made.

RESTORE

Undo STOP
**SubProcess descriptors**

The first words of a binary output file from the assembler contain the following information:

<table>
<thead>
<tr>
<th>Process in assembly listing</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

CLIST SPECIFIERS contain the 2-word names of objects. N appears in the first part of the local CLIST.
### MAP SPECIFIERS

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>File specifier - filename/username</td>
</tr>
<tr>
<td>2</td>
<td>File address</td>
</tr>
<tr>
<td>3</td>
<td>CM address</td>
</tr>
<tr>
<td>4</td>
<td>Word count</td>
</tr>
<tr>
<td>5</td>
<td>Read only flag</td>
</tr>
</tbody>
</table>

If the file name is zero, the local scratch file will be used.

Note that the assembly addresses and the file addresses differ by 17 because of the 50 table and the 77 table and the scope logical record word at the beginning of the file.
Names of special objects in the directory:

System operations have user names of 'OPERATE'. The list includes:

READ
WRITE
SEND
HANG
GETE
ECLIST - create c list
cfile - create file
cblk
cproc
cyuch
CSPROC
CCC - create class code
SAVE - save registers
RESTORE
DSPCAP - Display capability
DSPARE - Display any capability
MVICAP - move cap in full c list
cap in - move cap into full c list
cap out - move cap out of full c list
RSHAPE - Read file shape
COPER - create operation
MBLOCK - move block
CONDITIONS AT INITIAL ENTRY TO A CALLED SUBPROCESS:

C-LIST 8

0 ALLOCATION BLOCK
1 TTY CAP 2 (TTY File)
2 TTY CAP 2 (CP to PP event channel)
3 TTY CAP 3 (PP to CP event channel)
4 CALL TO BEAD ←
5 CALL TO SELF ←
6 CAP of SELF ← class code
7 CAP of THIS C-LIST

1 BEAD

3 read
4 write
5 send
6 hang