Published: 04/15/68

Identification

Procedure to produce printable information from the SLT.
slt_statistics
S. Ohayon

Purpose

The Procedure gives some information which is selected from the Segment Loading Table (SLT) (BL.2.01) and some information which is selected from the Major Module Configuration Table (MMCT) (BK.4.04). These two kinds of information give a picture at a given time of the hardcore ring, the current version of Multics and the hardware configuration.

Discussion

I. Information to be selected from the MMCT for output:

1. System identification (sys_name). This item identifies unambiguously the version of multics currently loaded.

2. Calendar time of the issue of the Multics System Tape (sys_date). This date will be set by the program which creates a new Multics System Tape.

3. Calendar time (start_date) the currently running Multics System was loaded (initialized). This date will be set by the Multics Initializer and current hardware configuration (CPU, GIOC, Clock, Drum, Memories, Interlace).

II. Information to be selected from the SLT for output:

Total amount of wired-down core.

For each segment:

a) segment no. - segment number in octal
b) name(s)
c) dirname - directory path name
d) max. size - maximum segment size in 1024-word units.
e) current segment size in 1024-word units (obtained by calling entry_status is described in BY.2.10, except for temporary initialization segment or permanent segments we use the size contained in the SLT).

f) status (normal, active, loaded, or wired-down)
g) access (slvacc, slvprc, wpermt, masprc, data or execute only)
h) whether or not segment has been deleted
i) ID (Procedure Identification) will be created from the date last modified of the source, the date of the compilation (assembly) and the versions of the assembler and compiler used.

The procedure identification is a future inclusion. An implementation method is proposed below. Two dates will be stored by the compiler (assembler) in each text, link and symbol segment: the date last modified of the source and the date and time that the compilation or assembly takes place. The versions of the assembler (and compiler) being used will also be stored.

The procedure ID of each segment is a combination of these dates and version ids. Slt_statistics calls get_calendar$brief to obtain a character-string representation of each date. The date (mm/dd/yy - giving month, day and year) and the time (xxxx.x in local time) are selected from the character string. The following items are output in the format:

S:mm/dd/yy xxxx.x O:mm/dd/yy xxxx.x A:version id C:version id

where S, O, A and C stand for date modified of source and object and version of assembler and compiler respectively. (The date modified of the object program is identical to the date and time of compilation or assembly).

Usage

The user calls the procedure by

```
call slt_statistics (segname);

dcl segname char(*);
```

where segname is the name of the segment where information from the SLT will be gathered. The procedure slt_statistics creates the segment segname in the working directory of the user.

Implementation

The following actions are done when slt_statistics is called.

1. creates the segment "segname" in the user's working directory

```
call smm$set_name_status (segname, (wdir), segname,
     null, "0110101"b, 0, "01011" b, segptr, " ",
     status);                                   (BD.3.02)
```

2. gets the date and time

a) `call get_calendar$full ((clock_), full); (BY.15.03)`

b)
```
call write_cs        /*to write in segname the title
                      and the date*/    (BY.3.01)
```

3. gets from MMCT the items of the discussion and write the items into the segment segname

4. gets the pointer to the SLT

```
call smm$initiate ("slt", ">initializer", "slt", 1, sltp,
     stat);
```

For both supervisor and initialization segments:

5. gets the first (bseg) and the last (tseg) segment number from the SLT.

6. We get the segment number, name(s), etc. for all the segments between <u>bseg</u> and <u>tseg</u>.

7. Call write_cs to write out each line of information.

From the SLT we get the information mentioned in part II of the discussion except the current segment size which is obtained by calling entry_status. Some items are converted for output.

1. Items which are put directly into the segment <u>segname</u>:

   a) name(s)
   b) directory name

2. Items which are converted into character strings before the output is put into segname by write_cs:

   a) segment number is transformed into an octal number.

   b) max_size is transformed into a decimal number.

   c) current_size is transformed into a decimal number.

   d) status is transformed respectively from 100, 000, 001, 010 (which are the values returned by SLT) into the character strings "wired-down", "normal", "active", "loaded", respectively.

   e) access is transformed into one or more of the following items, separated by commas: (slvacc, slvrc, wpermt, masprc, data or exonly).

   f) If the value of deleted returned by SLT is 0, deleted becomes "yes" else "no".

   g) Procedure ID.

Some items in Part I of the Discussion gathered from the MMCT need transformation for the output. The transformation will be made similarly to those made for the SLT items.

Output Format

Title: slt_statistics day-time

Identification __________ Created__________ Started__________

Configuration: CPU GIOC Clock Drum Memories Interface

Total amount of wired-down core

Supervisor segments

Segment no: Name(s)

ID

Dirname

max. size current size status access deleted

```
                                   Initialization segments
Segment no: name(s)

                -----------------------------------------------------

                ID

                Dirname

                max. size  current size  status  access  deleted
```

Two arrays will be developed: one for the supervisor segments and one for the initialization segments. All the names for each segment are written (because a multi-named segment may be the result of binding) and may take several lines.