Published: 07/19/67

Identification

Hash table for the APT
A. Evans

Purpose

Since a process is identified uniquely in Multics only by its process identification, all interprocess references are by process id. In general, referencing another process involves accessing its APT entry, and doing so requires knowing the index in the APT of the desired process. Thus there is need for a reasonably fast mechanism for getting the index, given the process id.

Discussion

The Active Process Table (see BJ.1.01) is a global data base containing, in addition to general information, an entry for each active process. In general, referring to a process requires referencing its APT entry, and doing this requires knowing the index in the APT of the relevant entry. Clearly, an effective scheme to deduce the APT index of a process, given its id, would be to iterate through the APT checking the process ids of the successive entries. This solution has the difficulty that it is slow, having an execution time roughly proportional to the number of processes allowed on the system. Since this lookup is done quite frequently in Multics, it is important that it be done efficiently. Thus the APT hash table module is provided. Using standard hashing techniques, the module is able to return the APT index of a process quite efficiently.

Usage

The APT hash table module has three entries: enter, delete, and lookup. (There is also a fourth entry, init, used to initialize the hash table at bootload time.) At a certain point in activating a process, its id and its index in the APT are known. Then executing

```
call apt_hash$enter (id, index)
```

will associate the id with the index, so that a subsequent execution of

```
index = apt_hash$lookup (id)
```

will return the desired index. When the process is deactivated, executing

```
call apt_hash$delete (id)
```

will delete the associated entry. Subsequent calls to lookup for that id will return the value -1 as an indication that the process is inactive. A suitable declaration is

```
declare id bit (36), index fixed,
        (apt_hash$enter, apt_hash$delete) ext entry,
         apt_hash$lookup ext entry returns (fixed);
```

Method

A standard hashing technique is used, similar to that described in Multics Repository Document M0082 (as amended by M0089).

Comment

Since this module may be called by interrupt handlers, both the module and the hash table must be in wired-down core. The hash table is in segment tc_data.