Published: 11/20/67

Identification

clock addressing segment
clock_, nclocks
J. H. Saltzer

Purpose

To read or set a system clock, one must execute an RCCL or LACL instruction with an address which, after appending, turns out to lie in the system controller containing the clock to be read or set. The address needed depends in general upon the current hardware configuration. The clock addressing segment contains a series of ITS pointers which, if indirectly referenced, will produce proper absolute addresses for reading or setting the system clocks. The clock addressing segment, named clock_, is accessible to slave procedures in all rings, although of course the alarm clock setting instruction (LACL) may only be executed in Master Mode. The clock addressing segment is set up at initialization or reconfiguration time; its purpose is to allow the programmer to access the clock symbolically without the need to know the current hardware configuration.

Discussion

Addressing pointers and other data items which depend on the system hardware configuration are normally stored in the hardcore-ring segment scs (see BK.4.01). However, the clock addressing pointers are placed in a separate segment so that they may be accessible in any ring. The contents of the calendar clock are thereby available to any programmer by the execution of a single instruction. Even a compiler may provide a built-in function to read the clock, as described in BP.0.03.

The segment clock_ contains two entry points, clock_ and nclocks. The first entry point is the only one of general interest; it contains at all times an ITS pointer which will address the current primary system clock.

Thus, the 645 machine instruction

RCCL <clock_>|[clock_],*

will cause the AQ register to be loaded with the current calendar time, as measured by the primary system clock. The ITS pointer stored in entry point clock_ contains the segment number of the system controller addressing segment (see section BK.4.02) and the word number necessary to address the system controller containing the primary system clock.

Entry point nclocks is of interest only to test and diagnostic programs; it contains a Multics integer equal to the number of system clocks in the current configuration. Entry point clock_ is actually the base of an array of ITS pointers capable of addressing each of the system clocks. For example, to read system clock number four (assuming that nclocks indicates that there are at least four clocks), one could use the sequence

```
eax7    2*4     "use index of twice clock number.
rccl    <clock_>|[clock_],7*    "read clock.
```

System clocks are numbered from 1. By convention the zeroth array location contains the pointer to the current primary clock; the pointer in the zeroth location is a copy of one of the later pointers in the array.

The clock addressing segment is set up by

```
call clock_init;
```

performed at initialization or reconfiguration time. clock_init builds the clock addressing segment according to information it finds in the major module configuration table (BK.4.04), and then changes its descriptor to read-only access. The clock addressing segment must be "wired-down" since it is used by hardcore-ring procedures which cannot tolerate a missing-page fault.

Implementation

The segment name "clock_" is actually a synonym for the segment name "sys_info", a segment which contains miscellaneous constant information about the current operating system. sys_info is described in full in section BK.4.05.