Identification

The Active Process Table

Robert L. Rappaport, Michael J. Spier

Purpose

The Active Process Table (APT) is a systemwide data base in segment <tc_data>; the Traffic Controller maintains an APT entry for every ~~loaded~~ active (see BJ.00 & BJ.6) process in the system. A process' APT entry contains all the information about that process that needs be publicly known within the Traffic Controller.

The Traffic Controller also maintains a number of lists threaded through the APT, namely the empty-list, the ready-list, the blocked-list and the various event-lists.

The APT entry

following is an itemized description of an APT entry, preceeded by the entry's EPL declaration.

```
declare   1  apt_entry based(p),
          2  thread,
             3  forward bit(18),
             3  backward bit(18),
          2  level fixed bin(17),
          2  state fixed bin(18),
          2  timer_residue fixed bin(35),
          2  time_last_run fixed bin(71),
          2  process_id bit(36),
          2  load_state fixed bin(17),
          2  wakeup_waiting bit(1),
          2  quit_pending bit(1),
          2  processor_required fixed bin(17),
```

```
        2  dsbr_value bit(36),
        2  pstep bit(18),
        2  class fixed bin(17),
        2  event_thread bit(18),
        2  filler              ;
```

thread — an APT entry is always threaded into some list; the threads are pointers relative to the base of <tc_data>. Unused pointers are reset to zero.

level — this number specifies the ready-list queue into which this process belongs (see ready-list below.)

state — is the process' execution state and can assume one of the following values:

0 = empty entry

1 = process running

2 = process ready

3 = process waiting

4 = process blocked

5 = process quit

timer_residue — whenever the processor is given away, the timer register gets stored in this item, to be restored when the process is run again.

time_last_run — is a clock reading taken whenever the process gives its processor away.

process_id — the process ID

load_state — this variable reflects the process' loading state as follows:

0 = empty entry

1 = process is unloaded

2 = intermediate state, process being loaded/unloaded

3 = process is loaded, may be unloaded

	4 and higher = process is loaded, must not be unloaded
wakeup_waiting	is the process' wakeup-waiting switch (see BJ.3)
quit_pending	is the process' quit-pending switch (see BJ.4)
processor_required	certain processes can execute on specific processors only; if this item is non-zero, the process will be run only on the processor specified.
dsbr_value	this is the value loaded into the DBR by subroutine swap_dbr (see BJ.7)
pstep	relative pointer to the process PST entry, needed for process loading/unloading.
class	is the process' class as follows: 0 = empty entry 1 = regular user's process 2 = regular system process 3 = regular system daemon 4 = hardcore process 5 = idle process
event_thread	relative pointer to head of event-queue (see BJ.3)
filler	to make the entry an even 16 words long

The APT lists

As mentioned above, the traffic controller maintains a number of lists threaded through the APT, and every APT entry (except when the associated process is currently running) is always threaded into one of these lists. The APT lists (except event lists) must each satisfy one or more of the following requirements:

1. It must be possible to thread an entry into either head or tail of the list.
2. It must be possible to thread an entry into either head or tail of a subset of the list (queue)

3. A queue within a list must be directly accessible (without having to follow the list's thread)

In order to implement these requirements, the APT contains a number of dummy entries, named "sentinels", which consist of only two items as declared

```
declare   1  sentinel based(p),
          2  thread,
             3  forward bit(18),
             3  backward bit(18),
          2  dummy_level fixed ;          /*level= -1 */
```

and which recognize the APT primitives by their negative level number. These sentinels are threaded into the lists as if they were normal APT entries, yet may be directly (symbolically) accessed.

The empty list

This is a threaded list of all unused APT entries. It is initially set up by subroutine tc_data_init. It is flanked by two sentinels which constitute its first and last entries and which can be accessed by referencing <tc_data>|[empty_q] and <tc_data>|[empty_q]+2 respectively.

The blocked list

This is the list of all blocked and quit processes. It is flanked by two sentinels which have the symbolic names <tc_data>|[block_q] and <tc_data>|[block_q]+4. A third sentinel, <tc_data>|[block_q]+2, divides the blocked-list into two sub-lists, the upper "blocked-loaded" list and the lower "blocked-unloaded" list. Whenever a process has to be selected for unloading, a process is taken off the tail of the blocked-loaded list, gets unloaded, and is then rethreaded into the head of the blocked-unloaded list. A process which calls block threads itself into the head of the blocked-loaded list, a process that calls quit threads itself into the tail of the blocked-

loaded list, thus making itself a prime candidate for unloading.

The ready list

The ready-list is the list of all processes which would be running, had a processor been available to them. Whenever a process gives its processor away, it selects the next process to run off the top of the ready list. The ready list is broken up into a number of sublists (queues) which are separated from one another by sentinels. These queue are used by the scheduler (see BJ.5) to thread different processes into different, pre-determined, relative locations within the ready list. There is a sentinel in front of each queue (and therefore in the back of each queue as well), and in addition there is one at the end of the ready list (for n queues there are n+1 sentinels.) Sentinel i could be accessed by referencing <tc_data>|[ready_q]+2*(i-1). Putting an entry onto the ready list at the head of the third queue means threading it into the list directly following the third sentinel. Similiarly, putting it at the tail of the third queue means putting it in front of the fourth sentinel. Searching the ready list for the highest-priority process means finding the first non-sentinel entry on the list.

Three variables are maintained in conjunction with the ready list: a count of the total number of ready processes, a count of the total number of loaded processes and a pointer to the lowest-priority queue (i.e. the sentinel for the queue) on which a loaded process exists.

primitives are provided in procedure <pxu> to do all the threading and unthreading of the above-mentioned lists. (see BJ.7)

The event lists

Waiting processes are threaded into event lists, the heads of which are in the AET. Each event list is associated with a certain event name; all the processes waiting for event 'A' thread themselves into the list associated with event 'A'. A process that detects the occurance of event 'A' notifies all the processes which are threaded on event list 'A'. The association between event-name and event-list is made in the AET. (see BJ.2)