search for a new best program.)

The second advantage to this queue-counter scheme is that the best runnable program is the runnable program with the lowest queue counter. Similarly, the worst runnable program is the one with the highest queue counter. Up to this point, the queues have been described as though they were separate lists. This is not necessary, and in fact the allocation of sufficient space to store 12 different queue lists would be quite inefficient. Both space and search time can be saved by assigning each program a fixed location in a queue-counter table. The Swapper then searches the table for the lowest (or highest) queue counter when it is looking for the best (or worst) user.

The time quantum used in this system is 32 milliseconds, the approximate amount of time required for a swap on the swapping drum. A running program, of course, may be interrupted by some outside event, which causes it to be swapped out after only a fraction of this time has passed. Therefore, each program contains a clock register which gives the number of milliseconds that it has used which have not been charged to its queue counter. This register is maintained by the Swapper, and the program's queue counter is decremented only when this clock register shows that a full quantum has been used.

B. Priority

If two programs which contain equal and large amounts of computing time are started at the same time, they will each fall to lower queues at about the same rate and can be expected to com-