MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

Reply to: Project MAC
545 Technology Square
Cambridge, Mass. 02139

Telephone: (617) 864-6900 x6201

February 5, 1968

To: F. J. Corbató, R. L. Rappaport

From: J. H. Saltzer

Subject: The value of selective locks in the traffic controller

For selective (per-process) locks to have a positive effect on the performance of the traffic controller, it is necessary that the extra time spent in setting and resetting several locks on each pass through the traffic controller be less than the expected time that would be spent looping on a global interlock set on entrance to the traffic controller. Assuming that a process tends to run for a time long compared with the traffic controller execution time between calls to the traffic controller an approximate value for the expected time spent waiting on a global interlock would be

$$\frac{(N-1)*T_t}{T_r} \cdot \frac{1}{2} T_t$$

where N is the number of processors

T_t is the execution time of the traffic controller

T_r is the running time of the process between calls to the traffic controller

The first fraction is the probability of finding the lock set; the second is the expected length of time it will continue to remain set; the product is the expected time spent waiting on each call to the traffic controller.

Clearly for a one-processor system, one would never find the lock set, and any extra time spent setting a multiplicity of locks is wasted. Thus, the formula gives zero for this case.

Assume for a moment four processors and a ratio of T_t/T_r of .05 (5% of the system time spent in the irreducible part of the traffic controller) both assumptions probably erring in the direction of high interference. The expected wait on the interlock is then

$$.075*T_t$$

with high-interference assumptions, or in other words, the effective running time of the traffic controller is increased by 7.5%.

Thus for selective interlocks to be beneficial, the total time spent setting and unsetting them, plus the expected values of waits on those interlocks must not increase the running time of the traffic controller more than a corresponding 7.5%.

Implementation of an interlock in EPL requires construction of a pointer argument with the ADDR built-in function and two subroutine calls (the equivalent EPLBSA sequence, assuming master mode, requires 6 instructions.) Two compilations of the traffic controller, one with selective interlocks in place, the other assuming a global interlock set in the EPLBSA interlude procedure through which all calls to traffic control are passed, resulted in lengths of executable program of 1734 and 1006 locations, respectively. Thus the current implementation of selective locks increases the number of instructions (and presumptively the execution time) of the traffic controller by 1734-1006 or 728 locations, about 72%. If the selective locks were coded in EPLBSA in ideal fashion, the extension would be only about 20*6 instructions (for 20 interlock setting and resetting locations), adding about 12% to the EPL-produced remainder.

Thus, one may conclude that for 645 Multics, the selective interlocks will not pay off and should be removed.