To: Multics Administrative Distribution

From: J. H. Saltzer

Subject: Targets for Multics performance

Date: March 18, 1971

This note outlines current targets for Multics performance over the next two years. These targets are worth documenting, both to indicate the magnitude of (and interest in) the job ahead, and also to facilitate planning of table sizes, channel and storage capacities, and supporting equipment inventories.

Figure one illustrates the history and targeted future performance level of Multics measured in "Load Units", or average interactive users. In summary, at the end of the two year period, the one-cpu 256 K-word system should handle 50 load units (50 average users) and the 2 cpu-384K configuration should handle 90 load units. The illustration indicates an approximate linear improvement during 1971 to 50 and 80 load units respectively, followed by a less steep rise during 1972 to 90 load units. Several assumptions underly these targets:

1. It is presumed that development work will proceed at approximately the present pace during 1971, and at a slightly slower pace during 1972. Correspondingly, it is presumed that development machine time on a 1 cpu/128K configuration is available except during prime shift hours.
2. The targets indicate performance of the Multics/CPU/memory combination without regard to potential bottlenecks in supporting equipment. Thus it is assumed that
   a. disk channel capacity,
   b. disk storage capacity,
   c. typewriter line capacity, and
   d. printer/card reader/punch capacity

   are suitably expanded to go with the numbers of users involved. (Figure one may be used as a guide to the appropriate rate of expansion.) The details of supporting equipment expansion are not projected here.

3. It is presumed that no important, but presently unknown, inherent bottlenecks are discovered in the system itself as it expands. Presently understood software limitations have been factored into the targets.
4. The performance is measured in load units, rather than users, and the actual number of users may be larger or smaller, depending on the relative popularity of restricted users and absentee usage, respectively.
5. All evidence indicates that the peak hour load will rise to meet the performance level, whatever that is.

A variety of ways of improving the present performance are known. Among those presently planned for implementation, or under study to evaluate their potential effects, are the following:

1. Adjust "tuning" parameters (SST size, scheduling parameters, tty buffer size, APT size, working set estimate factor, working set estimate addend, eligibility parameters) to maximize performance of the full equipment configuration.
2. Machine-language coding of argument validation. (speeds supervisor calls).
3. Compatible overhaul of the call/save/return sequence (cuts down cost of a call).
4. Take advantage of recent processor rewiring to permit small page tables (frees up wired-down core memory).
5. Gear up to allow missing page faults on some interrupts. (frees up wired-down core memory).
6. Put multiple copies of popular commands on drum; page in first available copy. (lowers waiting time; improves core usage).
7. Remove linkage definitions from pre-linked procedures. (frees up core memory; reduces working sets).
8. Rebind system with new binder. (smaller working sets everywhere).
9. Replace interprocess communication. (smaller working set for small user).

10. Dynamic multilevel file migration. (increases frequency of drum usage compared with disk usage).
11. Recompile system using PL/I optimizer (smaller, faster code everywhere).
12. Version II PL/I compiler. (the most popular command, as measured by cpu time used, becomes a lighter load).
13. Fix cpu associative memory to not store segment descriptor words. (increases cpu speed slightly when large working set is in use).
14. Eliminate all EPL code in system. (PL/I code is generally smaller and faster).
15. Eliminate "notify" on lock resetting if no one is waiting. (speeds file system primitives.)
16. Combine PDF and PDS to remove 1 per-process segment. (small user becomes cheaper).
17. Replace FORTRAN. (should be smaller and faster).
18. Speed up ALM. (makes some users a lighter load).
19. PL/I version of Runoff. (makes some users a lighter load).
20. Modify disk DIM to start reading in the middle of a record, if appropriate. (lowers average waiting time for disk I/O slightly).

Perhaps the single, most important performance improvement, only peripherally mentioned as item 1 above, is the harnessing of the 2-cpu, 384K system to its full capability. The performance of the full configuration is still a major unknown, and experience using it will probably suggest a number of additional appropriate changes.

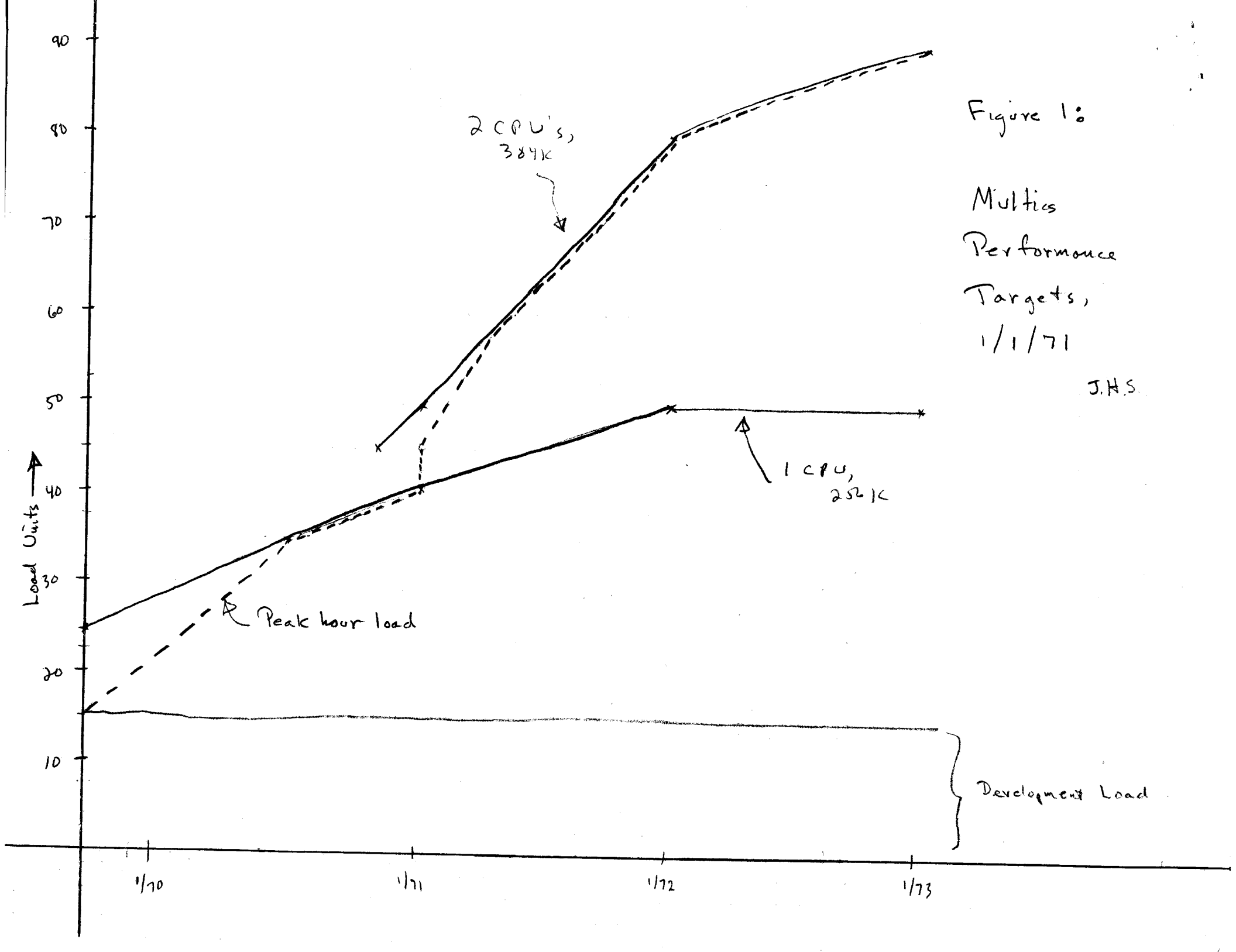

Figure 1:

Multics Performance Targets, 1/1/71

J.H.S.