1546

Bell Laboratories Cover Sheet for Technical Memorandum

The information contained herein is for the use of employees of Bell Laboratories and is not for publication (see GEI 13.9-3)

Title: **Program Profiling under MERT using a DEC KW11-P Clock**

Date: **February 20, 1980**

TM: **80-3121-3**

Other Keywords: **MERT**
Profiling

Author(s)	Location	Extension	Charging Case: 49622
M. A. Isman	**HO 3B-514**	**4388**	Filing Case: **49622**

ABSTRACT

A set of programs have been written which permit the profiling of a program or a set of programs at a resolution completely independent of the system clock. This profiling system can be used to gather statistics about the performance of a user level program, a supervisory process running on behalf of a user program, and the operating system itself.

Pages Text: 3	Other: 3	Total: 6
No. Figures: 2	No. Tables: 0	No. Refs.: 2

Bell Laboratories

subject: **Program Profiling under MERT using a DEC KW11-P Clock**
Case: **49622**
File: **49622**

date: **February 20, 1980**

from: **M. A. Isman**
HO 3121
3B-514 x4388

TM: **80-3121-3**

MEMORANDUM FOR FILE

1. INTRODUCTION

The MERT operating system provides a system structured in four layers[1] (See Figure 1). The lowest layer, the kernel, provides certain basic services. These services include interrupt and trap handling, interprocess communication, process dispatching, memory management, and scheduling. The second layer, kernel processes, consists of the file manager and I/O device handlers. The third layer consists of supervisory processes which provide the programming environment for the user level. The fourth layer is the user level where application programs are run.

The PDP 11/70 hardware supports three distinct modes of operation[2], kernel, supervisor and user modes. Each mode has its own processor stack and memory management registers. The four layers of the MERT system are distributed among the three modes that the PDP 11/70 hardware implements. The kernel and all kernel processes (layers 1 and 2), are run in the kernel mode. The supervisory processes (layer 3) run in the supervisor mode, while user processes (layer 4) run in user mode.

1.1 System Profiling

Standard MERT profiling consists of sampling the program counter at each line clock interrupt (sixty times per second). For user programs, the output produced includes a list of routines and the time spent in each one. For kernel programs the output produced is a table listing the eleven major areas of the operating system and the number of seconds of processing time spent in each one. These eleven areas include the scheduler, dispatcher, memory manager, system idle, etc. MERT provides inadequate measures to pinpoint time spent within each kernel subroutine and there is no capability of profiling a supervisory process. In order to provide complete system profiling including the kernel, kernel processes, supervisory processes, and user processes, we implemented additional profiling tools. The tools use an extra piece of hardware, the DEC KW11-P programmable clock to profile kernel, supervisor and user level processes at a rate independent of the line frequency clock. The implementation is discussed in the remainder of this paper.

2. IMPLEMENTATION DISCUSSION

The DEC KW11-P programmable clock is used to generate interrupts for program counter sampling. This clock can generate interrupts at a rate independent of the line frequency clock, yielding a better sampling of program counters.

2.1 User Interface

To use this profiling system, a user simply enters the process id of the process to be profiled and the duration of the profiling, using the profiling command. Various options are then available to the user, including the ability to profile the kernel (which has no process id), a kernel process, a user process, and to profile as many supervisors as the user desires. The user can also profile just one supervisory process, or any single process which can be identified by its process id. A kernel process can be profiled, since each kernel process has a unique process id. After gathering statistics about a process, a command compares these program counters with the process' symbol table to generate meaningful output.

The profiling system allows users to simultaneously profile different processes. The same process may even be profiled simultaneously by separate users. The only limitation to the number of users concurrently accessing the profiling system is the size of some tables within the programs. These limits are arbitrarily set, and can easily be changed by the system administrator.

2.2 System Implementation

The profiling command communicates with another user process, which provides central control for the profiling system (See Figure 2). If the user control program is not running (i.e. no one else is using the profiling system), then the profiling command starts the control program. This in turn starts a kernel process which actually drives the KW11-P clock and is entered each time the KW11-P clock generates an interrupt. The kernel process and the control program running on the user level communicate via events and shared memory.

The kernel process accesses the program counter and the process status word values which existed before the interrupt, and then figures out which process was running by accessing the operating system's tables. The process id, along with the program counter, is placed into a buffer in shared memory. When the buffer is full, an event is sent to the control program, which then assigns the program counters to the appropriate profiled process. The kernel process does as little as possible in order to prevent it from using more of the system's real time. All the bookkeeping is done in the user level control program which runs only on the order of every few seconds. The frequency of the control program's runs is determined by the clock frequency and the buffer size. In our system, the control program runs approximately once for every thousand program counter samplings.

All of the programs for this profiling system are run only when profiling is requested. Since they are not part of the kernel of the operating system, when they are not requested they do not use any system resources. The KW11-P clock only generates interrupts when the kernel process is running, so that it, too, does not require any system time while no one is profiling. Once the interrupt vector for the KW11-P clock is added to the system, the machine does not have to be rebooted for any changes associated with profiling. Thus all work on the profiling system can be done without interfering with the other users of the system. This profiling system allows each kernel process to be profiled separately; and it also allows a user to profile all UNIX* supervisors at the same time while excluding other supervisory processes from corrupting the statistics. A single UNIX supervisory process can also be profiled. Another feature of this profiling scheme is that it is invoked at run time and it is not necessary to recompile any programs to use it.

This profiling system is not part of the kernel of the operating system. Because of this separation, the time requirements of this profiling system are greater than those of a system where the sampling is done within the operating system. The increase in resource utilization does not seem to be significant enough to affect the rest of the computer system.

---

* UNIX is a trademark of Bell Laboratories

3. FUTURE EXTENSIONS

One extension presently envisioned for this profiling system is the capability of letting the user specify the sampling rate. Presently, the administrator of the profiling system must change the clock frequency and then recompile the kernel process. Other envisioned additions include the counting of characters and blocks of data being input from and output to various peripheral devices. This would involve very minor additions to the appropriate software device drivers. With these additional tools, character throughputs for disks, terminals, tapes and the line printer could be easily measured.

4. SUMMARY

The KW11-P clock profiling system adheres to the concepts of modularity of large systems by separating the profiling service and the operating system. This is done without substantially increasing the running time of profiling. The advantages of having a separable profiling system which offers added features clearly outweighs any increase in run time that may occur.

M. A. Isman

HO-3121-MAI-troff

M. A. Isman

Atts.
References
Figure 1
Figure 2

Copy to
Members of Department 3121
D. L. Bayer
T. M. Raleigh

REFERENCES

[1] Lycklama, H., and Bayer, D. L., "The MERT Operating System", *The Bell System Technical Journal,* July-August, 1978, Vol. 57, No. 6, Part 2, pp. 2049-2086.

[2] Digital Equipment Corporation, *PDP 11/70 Processor Handbook,* 1976

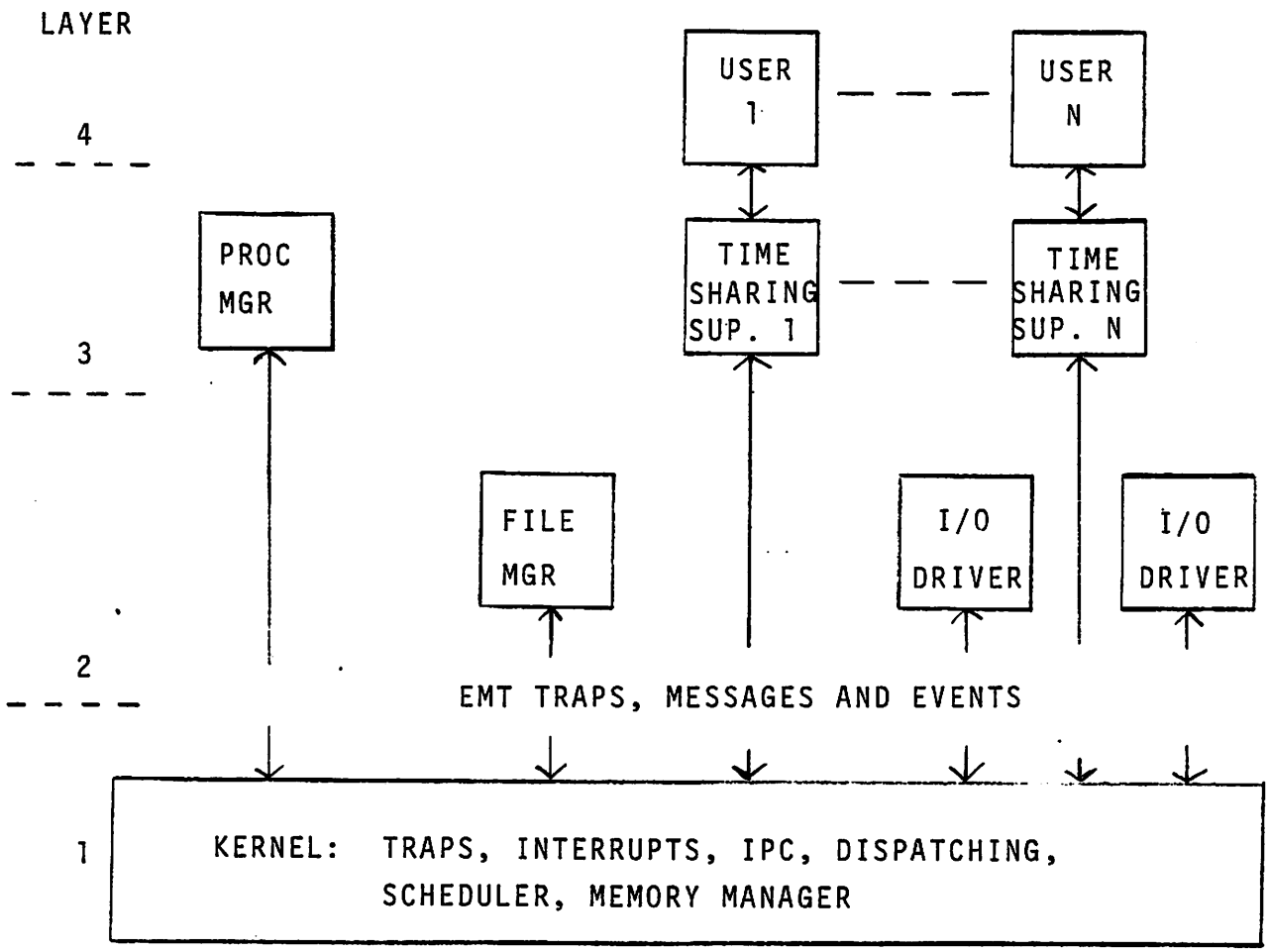

FIGURE 1 - MERT SYSTEM STRUCTURE

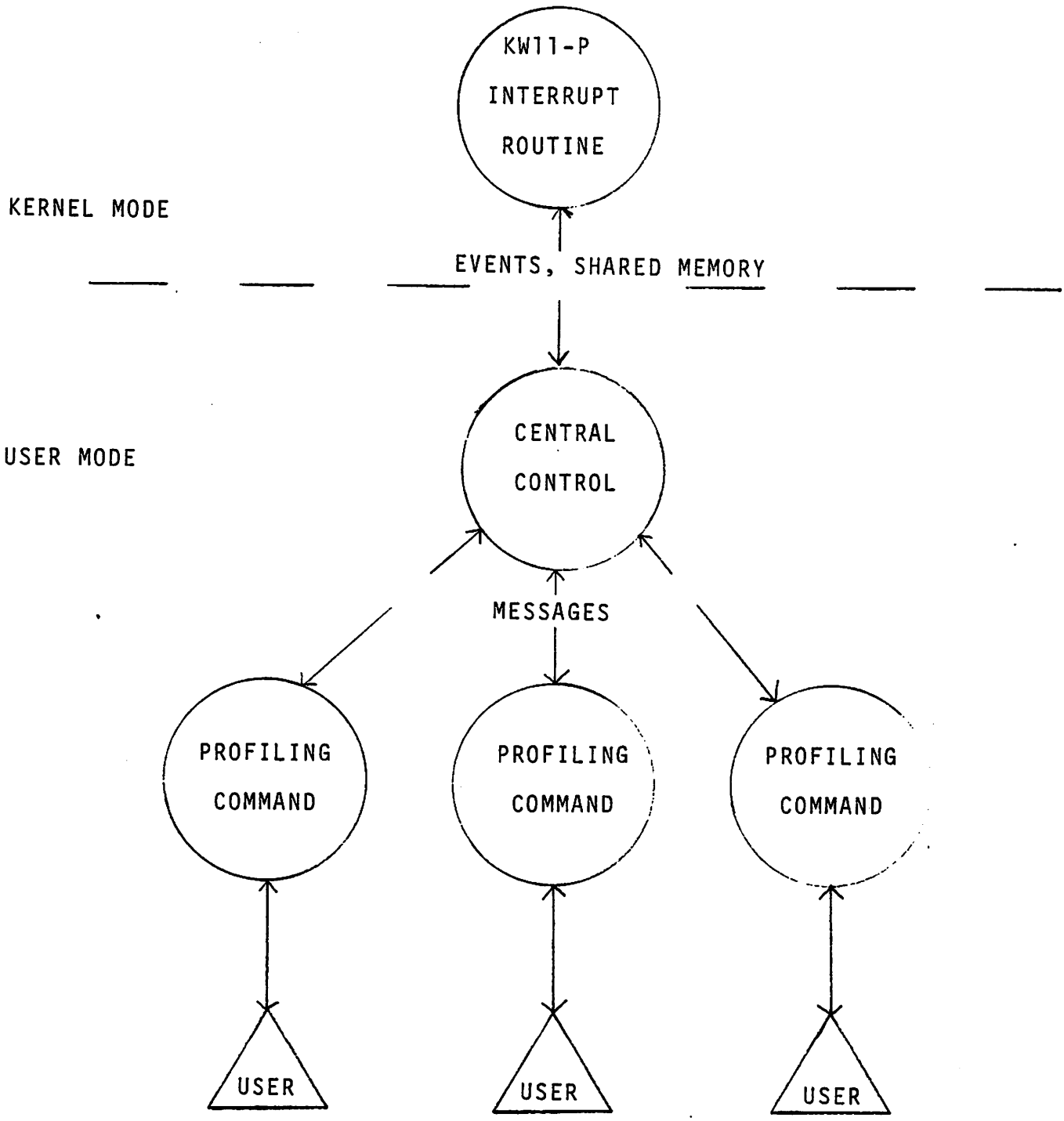

FIGURE 2 - PROFILING SYSTEM STRUCTURE