C O N F I D E N T I A L

PDP-X Technical Memorandum # 6

Title: PDP-X System Architecture

Author(s): H. Burkhardt
L. Seligman

Index Key(s): Architecture
System
Design Decisions

Distribution Key: F

Obsolete: None

Revision: None

Date: July 6, 1967

PDP-X System Architecture

Introduction

The term system architecture is intended to convey the concept of logical structure as opposed to its physical realization or implementation. The PDP-5 architecture, for example, has been re-implemented several times; the last, PDP-8/I, the fastest, least expensive, and smallest of the implementations, executes the same programs and runs the same peripherals as its predecessors. The lifetime of the architecture, with its system and diagnostic software, internal options, peripherals, customer acceptance, and training, is far longer than that of any implementation; thus, a company's stake in a good architectural design is far higher.

The overall system architectural description contained in this document, together with the more detailed descriptions found in associated documents, do not merely define a single processor; PDP-X lends itself to a number of implementations of varying complexity and relative costs. Since the architecture is constant across several possible models, many of the same programs and peripherals are applicable. The first implementation of PDP-X falls in the cost/complexity range expected of the PDP-9 replacement; future implementations are possible, with the smallest very much competitive with PDP-8/I and the largest falling in the current gap between the large and small computers now available from DEC.

PDP-X has grown out of experience with DEC's large and small computer lines, the design strives to achieve performance levels comparable to the largest machines through optional features added to a basically simple structure. In many ways PDP-X is a refinement of PDP-8, sharing the same basic word structure and basic instructions, but also including important design advances that will make it competitive with other products over the next several years. The appendix shows the evolution of basic DEC small computer OP Code structures.

The major characteristics of PDP-X include:

efficient memory utilization
structure amenable to fourth generation implementation
wide range of possible processor performance
software and hardware integration
real-time and multiuser environment
processor module concept
modular implementations
integrated option design

Details of the initial implementation must await experience gained from PDP-8/I, which, it is hoped, will expose IC problem areas and potential packaging traps. PDP-8/I will also expose the new cost relationships so that PDP-X design strategy can focus effectively on achieving minimum manufacturing cost.

Efficient Memory Utilization

One of the more obvious problems with current small-machine designs has been their inefficient use of core memory, a system resource whose cost has been rising relative to the cost of the processor logic. Indeed, a major cost-reduction technique used in PDP-8/S design was the development of a far less-expensive memory as is indicated by the cost breakdown in the appendix. PDP-X architecture makes more efficient use of available core memory through the introduction of a more powerful instruction set and an addressing structure that eliminates the (sometimes hidden) memory waste in sector addressing and single accumulator small-computer designs.

The great majority of instructions written for and executed by small computers, regardless of instruction set, fall into the PDP-8 repertoire. In addition, most address bits contain little information, as they usually reference memory words close to index quantities, close to the program counter, or the lowest words in memory (sector 0). PDP-X allows compressed (16-bit) representation of instructions where possible and permits complete (32-bit) representation when necessary. Hence, although such an instruction set has much of the potential power and scope of a 32-bit processor, only 16-bits are necessary to express most instructions. The ability to directly address all of memory easily, when necessary, simplifies the task facing the programmer by eliminating the need for complex linking structures as found, for example, in the PDP-9. In addition, the more casual customer or application programmer can generate working programs far more quickly. The appendix contains a more detailed analysis of the instruction formats.

The ultimate test of the efficiency of the structure lies in the programming package supplied with the hardware. Since no manufacturer of a PDP-9 class machine has been able to supply a version of FORTRAN for a 4K word-memory system, such a compiler would represent a major competitive advantage as well as testify to the accomplishment of a design goal.

Wide Range of Possible Processor Performance

The architecture is implementable in several processor models whose price and performance span the entire small computer market and include a model small enough to use as part of an IO device controller or selector channel. Smooth evolution and re-implementation should be possible over the next several years as the architecture leads to many new models, each particular model also exhibits a fairly wide range of performance depending on the number and type of internal options and peripherals purchased. To avert proliferation of software systems, however, certain standard configurations will be defined and the software written around them.

The major reasons for the wide range of possible processor performance are:

a. large, partially implemented, OP code set

b. variable number of interrupt levels with associated resister sets

c. use of main core memory to replace hardware registers

d. facility for multi user/multiprocessor configurations without drastic alterations to basic processor

e. use of ROS to create dedicated IO or OEM controllers.

Software and Hardware Integration

PDP-X systems will consist of many diverse configurations ranging from the small, dedicated data gathering system to the large real-time/multiuser installation; useful system software should be available to aid each user in fully realizing the potential power of his particular configuration.

Ease of use, rather than sophistication, is one of the major goals of the software system. Bewildering numbers of conventions, command mnemonics, data formats, and calling sequences are to be avoided. Since most of the users will be relatively inexperienced, error detection and recovery is included in all of the major systems such as the assembler and compiler. All IO data will be parity-checked, check-summed, or both as recorded on the media.

The major systems will be written in modular pieces with clean calling sequences; minimal versions will be available for the smaller configurations.* Programs will not modify themselves; instructions and alterable data will be independently located. All input/output will be done through a common interface that is also accessible to the user. Refer to the appendix for more detailed information on required software components.

The importance of external and internal documentation cannot be overstressed; many customers will want to alter or modify portions of the software system. Software performance and maintenance will become as important in the next few years as it is today for hardware. Many of the problems of documentation, training, employee turnover, and unexpected results can be solved by concurrent design and documentation.

---

* Since, in very small systems, modularity and minimum core usage are often conflicting goals, it may be necessary to sacrifice some modularity in order to make the software [illegible] in minimal systems. [illegible]

Real-time and Multiuser Environment

Real-time usage has become an important factor in computer sales and applications; three very distinct usages stand out. The first is the dedicated on-line system where the cost of the device (e.g., a particle accelerator) interfaced to the computer far outweighs the later's cost. PDP-X hardware provides a good order code, extremely fast interrupt response time, high speed core memory, and a fast IO system; in addition, the software package includes a set of highly optimized arighmetic and utility routines, re-entrant at some level, to allow this sort of real-time usage.

The second is used by an OEM who requires the smallest and simplest processor possible to imbed in his product. Such customers will use much of the same hardware used by DEC in dedicated IO controllers.

The third usage occurs where several real-time operations together occupy only a small fraction of the available processor time. Here the customer (DEC's traditional laboratory user) looks for a multiuser software system. The major hardware feature required for such a real-time, non-dedicated environment, is rapid problem switching. This implies fast interrupt response, low overhead in switching users, and sufficient core memory to allow resident programs to handle the peak service demands. The problems of core usage have already been discussed. User change overhead has always been due to a combination of both software and hardware considerations. The intent of PDP-X design is to reduce problem switching (due to interrupt) to an absolute minimum by providing multiple sets of general registers and a dual memory map. With the addition of this optional hardware, few interrupt cases will require that processor status be explicitly saved and restored.

A complete set of general registers, hardware on the larger models, is provided at each interrupt level. The cost of adding these registers is easily outweighed by the advantages of automatically saving and restoring the program status double-word, accumulators, and index registers. The dual map provides a separate set of mapping registers for the user and the real time or monitor program. Mapping rather than relocation, protection bit per word, or protection bit per block has been selected since it leads to simplified system programming and better core usage, facilitates shared code, and as the most general of the above systems, is most likely to fit a customer's particular needs.

Processor module concept

As advanced engineering/manufacturing methods shrink the cost of computer arithmetic processors the cost of IO controllers grows relative to them. Today, one finds tape systems, displays, etc., almost as complex as the arithmetic processor and certainly more difficult to manufacture. To satisfy user demand for still more powerful IO command structures, including more flexible interfaces, higher bandwidth, and less main (arithmetic) processor interference, these controllers must grow even more complex.

The most common approach to increasing the IO controller capability while reducing system cost has been the imbedding of part of the controller in the arithmetic processor. The success of this approach has been limited by the amount of processor time stolen relative to the costs saved. Extremely simple controllers, relying heavily on main processor assistance, have been very unsuccessful for reasons of efficiency.

A new approach may be termed the processor module concept. Here, the specialized IO controllers are replaced by small, general purpose processors dedicated to IO control. Much of the special purpose hardware normally found in the controllers is replaced by appropriate software and ROS programming. Devices which, by their complexity, lend themselves to this implementation include:

Magtape
DECtape
Display
Multistation teletype control
Line printer control and buffer

It should be noted, however, that the intent at this time is not general purpose multiprocessing. These dedicated processors will be sold only as IO controllers whose implementation just happens to be a standard processor.

Since, however, most of the new software systems are designed to achieve some form of simultaneity of subsystem operation, it would appear that hardware explicitly designed to aid this multiprocessing is the natural extension of

current design. PDP-X architecture permits it to serve as a vehicle for software development of more general multiprocessor systems. Multiprocessors offer:

a. optimization over diverse problem mixes through dynamic restructuring

b. system size scales, by adding identical units, over an extremely wide performance range

c. extreme reliability since malfunction merely lowers system capacity.

A typical system component interconnection is given in the appendix.

Modular Implementations

One of the most obvious, and perhaps expected, facts of digital system manufacture is that the labor cost and time of test rises as a square law rather than linearly with module count. Some statistics are presented in the appendix to justify these conclusions; although they are not as accurate as one might desire, the general trend is clear. Independent subassembly construction and testing seems to be one effective method of minimizing system manufacturing cost and in-process construction time. Indeed, even if test were a small fraction of system cost, the (un)availability of properly skilled manpower strongly influences the production rate. As labor costs rise and component costs drop, the need for modular construction techniques increases. A PDP-X processor (memory, or major option) is partitioned into a number of independent subassemblies which are small enough to be suitable for automated test equipment yet, which reduce the number and cost of interconnections. These subassemblies are considered repairable only at the subassembly construction level and, like most of todays modules, they are replaced, not repaired, by system test and maintenance personnel.