CONFIDENTIAL

PDP-X Technical Memorandum # 3

Title:	PDP-X Design Goals
Author(s):	E. deCastro L. Seligman
Index Keys:	Design Goals Goals
Distribution Key:	A, B, C
Obsolete:	None
Revision:	None
Date:	April 25, 1967

PDP-X design goals

This memo outlines the goals of the PDP-X project. While the objectives are reasonably firm, the techniques to be used in implementation are still very much under study.

1. Modern processor design

a. Current DEC small computer designs are limited by their architecture to make very inefficient use of the available technology. For example, both the PDP-8I and the PDP-9 have all the necessary hardware to implement an index register but their Op Code structure prohibits this. PDP-X architecture should make far better use of the available technology, yielding an improved price performance ratio. Twice PDP-9 performance at half its price is a reasonable goal.

b. Over the past few years the relative costs of memory and digital logic has shifted. PDP-X architecture should restore the balance thus minimizing total system costs. The principal method will be to reduce program core storage requirements without unduly increasing central processor complexity. The Op Code set should, for example, feature more powerful instructions and a more flexible addressing structure.

c. PDP-X basic configurations must expand neatly over a far wider range than current products. Use of read only storage (ROS) for Op Code expansion and special perhipheral controllers should permit increasing system capability at moderate cost. The addition of integrated, active memory arrays will speed interrupt processing as well as the more complex instructions. If the design sells well, one can reasonable expect to eventually offer 3 processors, all of which use the same architecture. These would replace the present PDP-8, PDP-9, and PDP-24.

d. The architecture should be implemented around hardware readily available at the begining of processor life but should also be amenable to the use of internal scratch pad memories, gate arrays, and other forms of large scale integration (LSI).

e. Our current experience in the hardware necessary for complex software systems (eg: background/foreground monitor) must be imbedded into the basic design rather than tacked on as an afterthought. The hardware necessary for dynamic memory allocation/protection, privileged instruction traps, etc., should be easily, but optionally, implemented.

2. Fourth generation design

a. The basic processor, if it is to be the basic building block of the next several years products, must easily expand into the next generation designs. Multiprocessing capability in its most general (and effective) form still lies beyond the reach of current techniques; however, it should be possible to implement the larger perhipheral devices (eg: Magtape) using the basic processor and suitable ROS. Such "mini-multiprocessor" design should also help break the production bottleneck on perhipherals that we face today.

b. An ultimate goal is the facility for horizontal system expansion; the ability to increase system performance by adding additional, identical processors. The attainment of this goal lies far ahead.

3. Standardized IO

a. To minimize the cost of the simpler perhipheral devices, identical hardware must be used with all versions of the processor. A standard byte interface is the most likely choice to accomplish this.

b. While the software goals still lack much detail, the following three stages have been culled from Larry Portner's group and anticipated competition:

stage 1 - basic assembler
editor
basic utility routines

stage 2 - macro assembler
compiler
IO system

stage 3 - keyboard monitor
extended monitor (as required)

Stage 1 software should be oriented around a minimum configuration, stages 2,3 around a PDP-9 class system with requisite options.

c. The software should be modular so that it can be eventually run in a monitor environment and require as little as possible to be simultaneously core resident. It should also be re-entrant whenever possible to facilitate use during interrupts and in a multi-user environment.

6. Anticipated schedule

a. hardware	
system architecture	Sept 1 '67
prints for basic system	Feb 1 '68
prototype constructed	Apr 1 '68
prototype running	July 1 '68
b. software	
simulator	Dec 1 '67
stage 1 basic	Mar 1 '68
basic diagnostic	Feb 1 '68
final processor diagnostic	Sept 1 '68

c. documentation

architecture	Sept 1 '67
specification*	Sept 1 '68
user handbook	Jan 1 '69

* documentation will be concurrent with development. Specifications refers to detailed IO timing, etc., sufficiently firm that they may be used in contracts.