OK as is

Abstract

Digital's five 18-bit computers (the PDP-1, 4, 7, 9 and 15) have evolved over two hardware generations (i.e., transistors and integrated circuits). The PDP-4 instruction-set formed the repetitive basis for the continued evolution. The final 18-bit implementation, PDP-15, evolved to have floating point arithmetic, multiple processors and an attached support processor for i/o and file control.

~~with the exception of PDP-15) was carried out~~

Only one implementation ~~occurred~~ at each technology time (versus a set of machines to span a range). Each new implementation provided a combination of both lower price and increased performance versus either being exclusively lower cost (constant performance) or exclusively constant price (higher performance). This middle-of-the-road strategy ultimately may have contributed to the 18-bit family demise.

The machines and the factors contributing to their evolution are presented.

PDP-1 chapter

THE DEC 18-BIT COMPUTERS

Gordon Bell, Gerald Butler, Bob Gray, Ronald Wilson, and Donald Vonada

Although Digital was formed in 1957 with the explicit goal to build computers, the PDP-1 was not demonstrated until the winter of 1959. The principal backer of DEC, American Research and Development (ARD), was initially skeptical whether a computer company could succeed, but was enthusiastic about the possibilities for digital logic modules for laboratory and system use. Hence, the plan to build computers was conditional on the success of the modules. After one year of operation, DEC had met its profit and sales goals and was permitted to move on to computers. However, Ken Olsen felt that another year's wait would be worthwhile in order to gain additional business experience and to have a larger business base. Ben Gurley came to DEC in the summer of 1959 to design and build the PDP-1. ✓

Many DEC engineers (Ken Olsen, Harlan Anderson, Stan Olsen, Dick Best and Ben Gurley) were Lincoln Laboratory alumni, and the DEC modules were directly patterned after the Lincoln circuits for the TX-0 and TX-2 computers. Hence, the PDP-1 was clearly influenced by the Lincoln Laboratory TX-0 computer, and its predecessor, Whirlwind I.

M.I.T. (WHIRLWIND & MEMORY TEST COMPUTER) & LINCOLN LABORATORY (TX-0) ANCESTRY

The Lincoln Laboratory experimental, transistorized computer, TX-0, was one of the earliest computers to use transistors. TX-0 in turn was related to MIT's Whirlwind, a computer which was operational in 1950. Whirlwind had a 16-bit word length and might be regarded as the forerunner to the modern minicomputer. This ancestry is uniquely Whirlwind's because it operated much faster (random access to the Williams storage tube; later the core memory versus having a drum primary memory) and--uniquely--had a short word length (16-bits) as compared to the 32- to 40-bit word length for scientific machines of the von Neumann era. The core memory was developed at M.I.T. by Jay W. Forrester, head of the computer laboratory. The first core memory was exercised under control of a special computer, Memory Test Computer (MTC). Ken Olsen had responsibility for the design and construction of MTC.

ask Ken about this fact.

TX-0 was designed to test transistor circuitry, to verify that a 256 by 256 (65 Kword) core memory could be built, and as a prelude to building the large scale TX-2 computer. This work was part of the air defense project that Lincoln ran. It featured a 6 microsecond cycle time and resembled Whirlwind because of the large (12 inch) cathode ray tube and light pen console. TX-0 also had paper tape I/O with a Friden Flexowriter typewriter, and two toggle switch registers that the program could access (the first 16 memory words were toggle switches (for variables and/or program)).

Whirlwind was also dismantled in 19~~xx~~59 and moved to Wolf Research and Development where it was reassembled and operated until ~~19xx~~ the 1970s. Whirlwind is now part of the Digital Museum Project, although the first core memory module and other parts have been given to the Smithsonian and other museums.

Because of the experimental nature of TX-0, it was extremely simple, having only two registers which could be accessed by the program: the accumulator, and the live register -- which was used for controlling and buffering transfers to various I/O equipment. Later, at MIT, TX-0 was extended to have an index register, but the initial version had only four instructions encoded in 2 bits, with 16 bits to access memory. Three instructions accessed memory: "store in location"; "add from location", and "transfer if accumulator is negative to location". The fourth instruction, "operate", was for programmed controlled I/O transfers, and it also included commands that could be combined to give a large number of instructions. The encoding of the operate instruction was called microprogramming because bits in the instruction specified particular register transfer operations (e.g., "clear the right half of the accumulator", "cycle the accumulator right one position", "start the paper tape reader") which could occur at one of six possible times during the execution of the instruction. By encoding the bits, a number of useful instructions could be formed (e.g., with one instruction, a point could be displayed on the screen, and a new pseudo-random point generated).

As the TX-0 tests were concluded, the TX-0 was transferred from Lincoln Laboratory to M.I.T. (1958) for laboratory experiment control and for teaching. The memory size was reduced from 65 Kwords to 4 Kwords and eventually added back to 8 Kwords. It remained in service at M.I.T. until 1975; it was eventually purchased by DEC for display in the Digital Distributed Museum Project.

PDP-1

The PDP-1 featured an 18-bit word length like that of its immediate predecessor, the TX-0, and was compatible with the notion of the 6-bit character of the second generation. Unlike the large scale scientific and business computers of the second generation, its short word length and high speed were unique and were particularly suited to the laboratory and scientific control applications that were to emerge later in the second generation. Even the small, lower cost scientific computers from Bendix (the G-15) and Librascope (LGP-30) had long word lengths but were slow because they were serial, using drum for primary memory. They were of limited utility in computation, control and laboratory applications due to their slow speed. The fact that the PDP-1 was unorthodox by being high speed (5 microsecond memory cycle), while having only an 18-bit word with no built in floating point arithmetic created a market credibility problem. For comparison, the IBM 709 had a 12 microsecond cycle, and its successor, the 7090, had a 2.12 microsecond cycle. It was also difficult to market the PDP-1 because potential buyers really doubted whether a company with only 100 employees and less than a million dollars in sales would be a reliable computer supplier.

The first description of the PDP-1 order code by Harlan Anderson, DEC's Vice President, appeared in a company memoranda, M-1060, dated October 27, 1959. Although a few instructions were later added to improve subroutine calling, that two page memo assigned the order codes and their names.

Fig. 2 Fig. 1

MJ ask Jack Brown

The PDP-1 engineering prototype (1/A) was first shown (see photo) in Boston at the Eastern Joint Computer Conference, December? 1959. Note how the CRT was integrated into the console (see photo). This was subsequently dropped for cost reasons, never to return to the DEC main line of computers, except briefly with the LINC and PDP-12. The consulting company, Bolt, Beranek, and Newman (BBN), purchased the first production machine (see ~~photo~~) (1/B) for delivery in November 1960. A third machine was constructed for internal use. After building the first two production machines, it was clear that modifications were needed for improved producibility, cost and reliability. The separate console required many cables, and the unreliable connectors between the console and computer meant the two parts could not communicate. The final, more producible design, called the 1/C, used DEC-produced cabinets, with the operator/maintenance console integrated into the cabinets (see photo). This design had improved air flow using the cabinet as an air plenum with air entering from the base of the cabinet. The cabinet and module mounting scheme was used directly on PDP-4 and -5. The cabinet has remained relatively unchanged. Even today the same basic design houses the smaller metal boxed minicomputers and options for all computers.

Fig. 3.

Fig. ~~2~~ 4

The first few PDP-1s were sold by and large as planned, that is for applications in scientific computation and real time control. Users interacted with the PDP-1 directly via its typewriter, CRT and console. Customers included Lawrence Livermore Laboratory (for high speed support processing ala the IBM 1401 but with graphics I/O that was compatible with most large scale computers); Bolt, Beranek and Newman (for psycho-acoustics and general computer science research); the Atomic Energy of Canada, Limited (for pulse height analysis and van de Graaf generator experiment control). The most important sale (in terms of DEC's future) was to ITT for a message switching system.

Marcie

or "1s" ask

Nearly half of the "1's" were sold to ITT as their ADX 7300 for this message switching application. This application was, in essence, the automation of a torn tape switching center and included interfaces for up to 256 teletype lines. These lines were switched under program control in a store-and-forward fashion on a character-by-character basis using the interrupt facility of the PDP-1. The PDP-1 was uniquely suited for this application because of its high speed and high performance sequence break (interrupt) system which permitted low cost line units (telegraph line interface).

Aside from the training of having to produce computers that could run unattended and without service on a high reliability basis, the most important result of the ITT order was that it allowed DEC to build a number of identical machines without special engineering. The first few machines ordered by other customers were nearly all unique, requiring us to build options that were only sold a few times. In addition to the nearly one-of-a-kind options, many of the machines had special interfaces that were peculiar to the applications.

It should be noted that because the hardware for the "1" was relatively inexpensive, we had a good supply of basic modules for building special interfaces, and it was not difficult to interface, building specialized hardware was relatively easy compared to modern day hardware design. If an engineer made errors in estimating the time required to design and test the option, the project would still be financially successful because the cost of materials was low enough to allow for such errors. Also, design errors were easier to wire around compared to modern day, expensive printed circuit boards with hard to change "read only" memories. Finally, the special interfaces and controllers for PDP-1 were quite simple compared to modern designs.

Physical Structure/Implementation

The PDP 1/C version formed the basis of the machines that were mass produced. It used four cabinets instead of the three in the earlier version, and it preassigned space for the various options: multiply-divide, sequence break (the name given the 16-channel interrupt mechanism), memory extension control, and the high speed channel (what we now call the direct memory access capability). Numerous options were initially offered for PDP-1. These are shown in the Fig. Block1 5 in a fashion that corresponds to the physical structure. Note that cables connect in a radial fashion to the various options, as opposed to being interconnected in a bussed fashion as computers are now. (The reader should note that design is reverting back to radial configuration in the early fourth generation because of the decreased cost of logic, I need the high interconnect cost) and to bound the system.)

A side view of the cabinet in Fig. 6 ~~Sidvue1~~, shows the space for interconnecting to other options. Note that expansion was accommodated by adding bays to the four-bay structure and by interconnecting stand-alone options via cables. Since the I/O structure was fundamentally radial (versus modern day bussed), it was easy to cable to free-standing controllers and their options (e.g., magtape, displays, printer, card~~s~~ equipment, etc.).

As previously mentioned, the PDP-1 was built after DEC had a complete line of general purpose modules. The extra year before PDP-1 was constructed permitted more low speed (500 Khz) modules (for I/O equipment) to be designed using the same circuit techniques, but with less expensive, slower transistors. Of the computers built with modules (versus modern third and fourth generation computers), PDP-1 was built from only 34 module types (including memory modules), and each module type was fully general purpose! Only five module types were added for the (analog) memory circuitry. The module types and their characteristics include:

Circuit Type	High Speed (5 Mhz clock)	Low Speed (500 Khz clock)
. Inverters, gates and decoders	7	5
. Pulse amplifiers and delay lines	4	2
. Flip-flop configurations	2	3
. Special drivers and signal conditioning	4	~~3~~2
. Core memory circuits	5	-
	-----	-----
	22	12

← printer error

Although it was unstated as such, a major design theme for the PDP-1 was easy interfacing and the ability to handle a variety of I/O devices. These goals were incorporated in a physical structure that permitted various I/O devices to be easily interfaced to the computer. One of the first user manuals was the Input-Output Systems Manual written by this author which described the methods now standard to minicomputer and microcomputer interface design. These methods included:

1. Program controlled transfers.

2. Program controlled transfers with sequence break (now called interrupt).

3. Multiple channel interrupt programmed control.

4. High speed channel data transmission (now called direct memory access).

The optional 16-channel sequence break (interrupt) system, based on the Lincoln Laboratory TX-2, provided the PDP-1 with a unique capability. The sequence break system permitted a program to handle much of the processing associated with I/O devices instead of using special hardwired control. Each time an I/O device had information to be transferred to memory, it caused an interrupt to the processor for handling. The PDP-1 sequence break was a real alternative to the large computers which evolved extensive I/O processors (e.g., the IBM 7090 channels). In fact, the PDP-1 was less expensive than a single IBM "channel". When I/O character transmission rates (e.g., magnetic tape, drums) exceeded that which could be handled by a program, the information was transmitted directly to the PDP-1's memory in blocks under the device's control. Inter-block control was then managed using the interrupt. This basic scheme has remained even in today's DEC computers.

Figure Mag1 7 shows a block diagram of Type 51 Magnetic Tape Control Unit that operated under program control. This controller used a minimum of hardware but required 100% of the processor's time to read or write data. Note that various signals were connected directly into the program flags so that the program could operate as fast as possible. Also, note there were no word buffers in the controller; rather, characters were assembled in the processor's I/O Register. While we would not build such a controller today (requiring 100% of a $120,000 computer's attention), this structure is identical to modern day microprocessor-based controllers. (Note each computer generation goes exactly through all stages of evolution ~~that~~ of predecessor generations.)

The PDP-1 Instruction-Set Processor

There is no record of the goals, constraints and the objective function to measure PDP-1 design. However, it is clear that the PDP-1 ISP was a reaction to the TX-0. Rather than being program compatible, PDP-1 was really oriented to being producible and useable by a variety of programmers having more instructions than a prototype like TX-0 and having a simpler I/O structure for ease of interfacing. It is unclear that making PDP-1 compatible to TX-0 was ever considered. As it turned out, TX-0 was continuously extended; hence compatibility would have been a dubious goal.

Figure REG. 1 8 is a diagram of the PDP-1 (taken from the original programming manual) showing its registers and functional units. Most of the registers were named after those of the TX-0, and only the TX-0's live register was renamed to be the input-output register in the PDP-1. The I/O register is in reality a multiplier-quotient register as well, and used as an accumulator extension. An index register was apparently not considered probably because of the increased cost.

Since the memory cost strongly determined the machine's price, a 4 Kword base was selected for the PDP-1. The instruction format was selected to be: 12 bits for addressing the 4 Kword memory; 1 bit for indirect addressing; and leaving 5 bits to select 32 possible instructions. Since the machine was oriented to control and because it was low cost, only word, integer and Boolean vector (logical) data types were included in the basic design. This only required seven data operators (+, -, x, /, and, or, exclusive or) for the one accumulator structure plus control instructions.

Altogether, only 24 instructions were assigned in the initial design, and 28 instructions were eventually used in production machines. The ISP of PDP-1 is given in Table ~~ISP 1 + 4~~ 1 and for comparative purposes the PDP-4 ISP is given side by side.

Role of PDP-1 in Timesharing

A number of computer scientists at M.I.T. and BBN believed that it was necessary to provide interactive access to computers (as opposed to batch). Furthermore, the only way to make this economically viable was to simultaneously share the computer among the users. Two early experiments (with which we were familiar) were carried out to demonstrate the feasibility: the IBM 7090 at M.I.T. which was later turned into the Compatible Time Sharing System, CTSS; and the shared PDP-1 at BBN. We made modifications to BBN's PDP-1 so that users sharing the machine could not issue I/O instructions that would interfere with the operation of the I/O equipment assigned to another user and we protected users from accessing each others' programs residing in primary memory. Since the PDP-1 mainly operated in 4 Kword memory segments, a program swapping drum memory was built that permitted programs in core to be exchanged with those on the drum in only one drum revolution (33 milliseconds). These modifications formed the PDP-1/D which used the multiport memory we had designed for the PDP-6. Timeshared computers were built and operated at BBN, Stanford and M.I.T. This effort contributed to later timesharing by the PDP-6 computer.

PDP-1. These are shown in the Fig. 5 in a fashion that corresponds to the physical structure. Note that cables connect in a radial fashion to the various options, as opposed to being interconnected in a bussed fashion as computers are now. (The reader should note that design is reverting back to radial configuration in the early fourth generation because of the decreased cost of logic, the high interconnect cost, and need to bound the system.)

A side view of the cabinet in Fig. 6, shows the space for interconnecting to other options. Note that expansion was accommodated by adding bays to the four-bay structure and by interconnecting stand-alone options via cables. Since the I/O structure was fundamentally radial (versus modern day bussed), it was easy to cable to free-standing controllers and their options (e.g., magtape, displays, printer, card equipment, etc.).

As previously mentioned, the PDP-1 was built after DEC had a complete line of general purpose modules. The extra year before PDP-1 was constructed permitted more low speed (500 Khz) modules (for I/O equipment) to be designed using the same circuit techniques, but with less expensive, slower transistors. Of the computers built with modules (versus modern third and fourth generation computers), PDP-1 was built from only 34 module types (including memory modules), and each module type was fully general purpose! Only five module types were added for the (analog) memory circuitry. The module types and their characteristics include:

Circuit Type	High Speed (5 Mhz clock)	Low Speed (500 Khz clock)
. Inverters, gates and decoders	7	5
. Pulse amplifiers and delay lines	4	2
. Flip-flop configurations	2	
. Special drivers and signal conditioning	4	3
. Core memory circuits	5	-
	-----	-----
	22	12

Market experience with the PDP-1

~~The PDP-1 featured an 18-bit word length like that of its immediate predecessor, the TX-0, and was compatible with the notion of the 6-bit character of the second generation. Unlike the large scale scientific and business computers of the second generation,~~ Because of its short word length and high speed, ~~were unique and were~~ the PDP-1 was particularly suited to the laboratory and scientific control applications that were to emerge later in the second generation. Even the small, lower cost scientific computers from Bendix (the G-15) and Librascope (LGP-30) had long word lengths but were slow because they were serial, using drum for primary memory. They were of limited utility in computation, control and laboratory applications due to their slow speed. The fact that the PDP-1 was unorthodox by being high speed (5 microsecond memory cycle), while having only an 18-bit word with no built in floating point arithmetic created a market credibility problem. For comparison, the IBM 709 had a 12 microsecond cycle, and its successor, the 7090, had a 2.12 microsecond cycle. It was also difficult to market the PDP-1 because potential buyers really doubted whether a company with only 100 employees and less than a million dollars in sales would be a reliable computer supplier.

The first few PDP-1s were sold by and large as planned, that is for applications in scientific computation and real time control. Users interacted with the PDP-1 directly via its typewriter, CRT and console. Customers included Lawrence Livermore Laboratory (for high speed support processing ala the IBM 1401 but with graphics I/O that was compatible with most large scale computers); Bolt, Beranek and Newman (for psycho-acoustics and general computer science research); the Atomic Energy of Canada, Limited (for pulse height analysis and van de Graaf generator experiment control). The most important sale (in terms of DEC's future) was to ITT for a message switching system.

Nearly half of the "1s" were sold to ITT as their ADX 7300 for this message switching application. This application was, in essence, the automation of a torn tape switching center and included interfaces for up to 256 teletype lines. These lines were switched under program control in a store-and-forward fashion on a character-by-character basis using the interrupt facility of the PDP-1. The PDP-1 was uniquely suited for this application because of its high speed and high performance sequence break (interrupt) system which permitted low cost line units (telegraph line interface).

Aside from the training of having to produce computers that could run unattended and without service on a high reliability basis, the most important result of the ITT order was that it allowed DEC to build a number of identical machines without special engineering. The first few machines ordered by other customers were nearly all unique, requiring ~~us~~ DEC to build options that were only sold a few times. In addition to the nearly one-of-a-kind options, many of the machines had special interfaces that were peculiar to the applications.

It should be noted that because the hardware for the "1" was relatively inexpensive, DEC ~~we~~ had a good supply of basic modules for building special interfaces. ~~and it~~ It was not difficult to interface to and building specialized hardware was relatively easy compared to modern day hardware design. If an engineer made errors in estimating the time required to design and test ~~the~~ an option, the project would still be financially successful because the cost of materials was low enough to allow for such errors. Also, design errors were easier to wire around compared to modern day, expensive printed circuit boards with hard-to-change, read-only memories. Finally, the special interfaces and controllers for PDP-1 were quite simple compared to modern designs.

The Role of PDP-1 in Timesharing

A number of computer scientists at M.I.T., ~~and~~ BBN, and Oxford University believed that it was necessary to provide interactive access to computers (as opposed to batch). Furthermore, the only way to make this economically viable was to simultaneously share the computer among the users.

~~Strachey first suggested the idea of time-sharing in 1959 (Strachey, 1960).~~ Two ~~early~~ experiments were carried out to demonstrate its feasibility: the IBM 7090 system at M.I.T. (Corbato *et al*, 1962) which later became the Compatible Time Sharing System, CTSS; and the shared PDP-1 at ~~Bolt, Beranek, and Newman~~ BBN (McCarthy *et al*, 1963).

¶ Batch ~~M~~multiprogramming was an important part of the design of the Stretch computer (Buchholz, 1962) and the Atlas computer (Kilburn, 1962). It is oriented towards hardware efficiency in the sense of ~~trying~~ trying to attain high utilization of all components. Time-sharing, on the other hand, is concerned with the efficiency of persons trying to use a computer - the efficiency of man-computer interaction (Corbato *et al*, 1962).

A set of ~~The following~~ requirements were identified ~~are needed~~ for a time-sharing system. Unless ~~the~~ a workload ~~was~~ ~~is to~~ is to be restricted to (a) programs that are specially designed to run concurrently and (b) programs which are error-free, one needs:

a. memory protection ~~between user programs and between the supervisor and user programs.~~

b. program and data relocatability

c. a supervisor program

d. a timed return to the supervisor

e. interpretive execution of i/o instructions

The BBN Time-Sharing System on the PDP-1 began operation in September, 1962. Five teletype users shared the upper 4K words of memory; the lower 4K held the supervisor program, called the "channel 17 routine." The modifications to the PDP-1 to effect time-sharing were embodied in a "restricted mode" of operation. They matched the above requirements list ~~given above~~ in the following way.

a. memory protection

~~[G.B: can you supply]~~ [Switching between the two 4K areas required the use of an i/o instruction.]

b. relocatability

Because only one user was resident at one time, this was not needed.

c. a supervisor program

The Channel 17 routine fulfilled this function.

d. a clock interrupt

The Channel 17 clock generated an interrupt every 20 msecs.

e. i/o instructions

Whenever the PDP-1 was in restricted mode, an attempt to obey an i/o instruction caused a channel 16 sequence break.

¶ The TYC Control Language was regarded as a key portion of the software support. This debugging aid was adapted from the DDT language devised for the TX-0 and TX-2 computers.

¶ The "restricted mode"

. ~~These~~ modifications formed the PDP-1/D, which also used the multiport memory ~~we had~~ designed for the PDP-6. Timeshared computers were built and operated at BBN, Stanford and M.I.T. ~~This~~ These efforts ~~contributed to~~ influenced later timesharing by the PDP-6 computer (Chapter 22).

[CM: Read Licklider & Clark SJCC 62
on the benefits of close man-machine interaction
during problem solving, for
(1) references to PDP-1
(2) mention of this essential minicomputer
contribution.]

Clark, Wesley A The Lincoln TX-2 Computer Development, Proc WJCC, 1957, pp 143-145.

Everett, R.R. The Whirlwind I Computer, AIEE-IRE Conf, 1951, pp 70-74 (Reprinted in Bell & Newell Chapter 6 pp. 137-145)

Forrester, J.W. Digital information storage in three dimensions using magnetic cores, J. Applied Physics, <u>22</u>, pp 44-48, ~~1951~~ January 1951.

McCarthy, J., Boilen, S., Fredkin, E., and Licklider, J.C.R. A Time-Sharing debugging System for a Small Computer. Proc SJCC, 1963, pp 51-57

Strachey, C. Time-Sharing in Large Fast Computers. Proceedings of the International Conference on Information Processing, UNESCO, Paris 15-20 June 1959, UNESCO, Paris, 1960, pp 336-341.

Corbató, F.J., Merwin-Daggett, M., and Daley, R.C. An Experimental Time-Sharing System Proc ~~SJCC or WJCC~~ SJCC 1962

Buchholz, W. (ed). Planning a Computer System Wiley, New York, 1962.

Beckman, F.S., Brooks, F.P. Jr, and Lawless, W.J. Jr Developments in the Logical Organization of Computer Arithmetic and Control Units Proc IRE, <u>49</u>, 1, January 1961, pp 53-66.

Rajchman, J.A. Computer Memories: A Survey of the State-of-the-Art Proc IRE, 49, 1, January, 1961, pp. 104-127.

Frankovich, J.M., and Peterson, H.P.
A Functional Description of the
Lincoln TX-2 Computer
Proc WJCC 1957

Forgie, J. W. The Lincoln TX-2
Input-Output System
Proc WJCC, 1957

[Whirlwind Project

[CM: find exact citation for Whirlwind project,
IEEE Spectrum, October (?) 1977]

Mitchell, J.L., and Olsen, K.H. TX-0, A Transistor Computer, Proc EJCC, 1956, pp. 93-100.

18-bit Chapter

edited 12/6/77

(from Craig)

~~Ch 3~~. The PDP-1 and other 18-bit machines

PDP-1

In contrast to the large-scale scientific and business computers of the second generation, the PDP-1 had a much shorter word (18 bits) and a simpler instruction set (28 instructions in all). The first machine was demonstrated in late 1959. DEC's ~~DEC modules, The~~ 5 MHz 1000 Series System Modules were used in the implementation. Memory capacity was ~~4096~~ 4K words, later expanded to 64~~K~~ Kwords. The I/O structure included a sequence break option (the name given the 16-channel interrupt mechanism) and a high-speed channel (now called ~~a~~ direct-memory-access ~~capability~~). The ~~processor and memory occupied four cabinets~~.

¶ Although Digital was formed in 1957 with the explicit goal to build computers, the PDP-1 was not demonstrated until the winter of 1959. The principal backer of DEC, American Research and Development (ARD), was initially skeptical whether a computer company could succeed, but was enthusiastic about the possibilities for digital logic modules for laboratory and system use. Hence, the plan to build computers was conditional on the success of the modules. After one year of operation, DEC had met its profit and sales goals and was permitted to move on to computers. However, Ken Olsen felt that another year's wait would be worthwhile in order to gain additional business experience and to have a larger business base. Ben Gurley came to DEC in the summer of 1959 to design and build the PDP-1.

10

lower case

M.I.T. (WHIRLWIND & MEMORY TEST COMPUTER) & LINCOLN LABORATORY (TX-0) ANCESTRY

Insert F

¶ The Lincoln Laboratory experimental, transistorized computer, TX-0, was one of the earliest computers to use transistors. TX-0 in turn was related to MIT's Whirlwind (Everett, 1951), a computer which was operational in 1950.* Since Whirlwind had a 16-bit word length, it can ~~and might~~ be regarded as the forerunner of ~~to~~ the modern minicomputer. This ancestry is uniquely Whirlwind's because it operated much faster (random access to the Williams storage tube; later the core memory versus having a drum primary memory) and--uniquely--had a short word length (16-bits) as compared to the 32- to 40-bit word length for scientific machines of the von Neumann era. The core memory was developed at M.I.T. (Forrester, 1951). ~~by Jay W. Forrester, head of the computer laboratory.~~ The first core memory was exercised under control of a special computer, Memory Test Computer (MTC). Ken Olsen headed the team for the design and construction of MTC. MTC operated only a short time because the core memory worked as ~~plann~~ planned; hence, the memory was moved to Whirlwind.

TX-0 was designed to test transistor circuitry, to verify that a 256 by 256 (65 Kword) core memory could be built (Mitchell and Olsen, 1956), and as a prelude to building the large scale, 36-bit word, TX-2 computer. This work was part of the SAGE air defense project that Lincoln ran. INSERT G ¶ TX-0 ~~It~~ featured a 6 microsecond cycle time and resembled Whirlwind because of the large (12 inch) cathode ray tube and light pen console. TX-0 ~~also had~~ used paper tape I/O with a high speed (300 char/sec.) Ferranti paper tape reader and a Friden Flexowriter used both as a typewriter and paper tape punch. It also had ~~typewriter, and~~ two toggle switch registers that the program could access and (the first 16 memory words were also toggle switches for variables and/or program).

~~TX-2 multipl(?)~~

*Whirlwind was dismantled in 1959 and moved to Wolf Research and Development where it was reassembled and operated until the 1970s. Whirlwind is now part of the Digital Museum Project, although the first core memory module and other parts have been given to the Smithsonian and other museums.

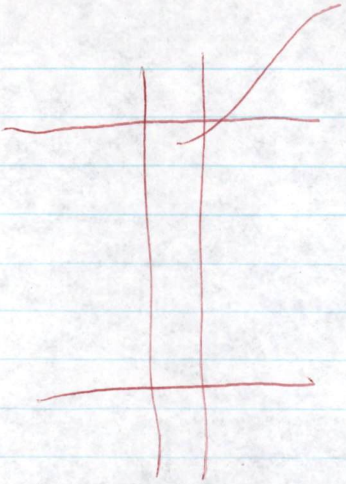
F-insert

For the PDP-1, by far the most influential machines were those constructed at M.I.T. and the M.I.T. Lincoln Laboratories, because many DEC engineers* were alumni. Moreover, the DEC modules were directly patterned ~~at~~ after the circuits of the TX-0 and the TX-2 computers (Chapter 2).

* Ken Olsen, Harlan Anderson, Stan Olsen, Dick Best and Ben Gurley.

[T -insert

The initial computer was called the TX-1, a 32-bit computer patterned after the 32-bit AN/FSQ7 that IBM and (and MIT) had designed and built as part of ~~the~~ the SAGE system. The large core memory work by William Papean, was based on vacuum tubes drivers and the memory design proceeded ~~went on~~ ~~went on~~ ~~after~~ ~~inded~~ ~~was~~ independent of the computer it was to work with. ~~#~~ ~~The~~ The appearance of ~~the~~ Philco's Surface barrier transistor significantly simplified transistor circuit design. ~~required~~ ~~A pro project to~~ Nevertheless, it was felt that a computer was needed in order to test the transistor circuitry ~~that~~ ~~this test vehicl~~ -- this wo ~~The propose~~ ~~test computer~~ was the TX-0.



1/FF each

16 ~~rows~~

~~cols~~

Each col. can be 2 if comb.

Because of the experimental nature of TX-0, it was extremely simple, having only two registers which could be accessed by the program: the accumulator~~,~~ and the live register. ~~-- which~~ The latter was used for controlling and buffering transfers to various I/O equipment. Later, at MIT, TX-0 was extended to have an index register, but the initial version had only four instructions encoded in 2 bits, with 16 bits to access memory. Three instructions accessed memory: "store in location"; "add from location", and "transfer if accumulator is negative to location". The fourth instruction, "operate", was for programmed controlled I/O transfers, and it also included commands that could be combined to give a large number of instructions. The encoding of the operate instruction was called microprogramming because bits in the instruction specified particular register transfer operations (e.g., "clear the right half of the accumulator", "cycle the accumulator right one position", "start the paper tape reader") which could occur at one of six possible times during the execution of the instruction. By encoding the bits, a number of useful instructions could be formed (e.g., with one instruction, a point could be displayed on the screen, and a new pseudo-random point generated).

As the TX-0 tests were concluded, the TX-0 was transferred from Lincoln Laboratory to M.I.T. (1958) for laboratory experiment control and for teaching. The memory size was reduced from 65 Kwords to 4 Kwords and eventually added back to 8 Kwords. It remained in service at M.I.T. until 1975; it was eventually purchased by DEC for display in the Digital Distributed Museum Project.

Following the completion of TX-0, work began on the TX-2; ~~22000 transistors were used~~ it had 22000 transistors, whereas TX-0 used 3600. The TX-2 development is described in (Clark, 1957) A principal design goal was an I/O organization more efficient than existing machines. The designers rejected a separate I/O processor and instead chose ~~to~~ a minimum buffering scheme with direct transfer to memory. Additional program sequences (and associated program counters) were provided for these I/O transfers. Thus the inherent processing facilities of the CPU were used to effect ~~I/O transfer of I~~ I/O transfers. The PDP-1 sequence break is directly patterned after the TX-2 I/O system (Forgie, 1957). Much internal ~~pad~~ parallelism existed in the CPU. ~~Distin~~ separate adders for indexing, PC incrementing, and instruction execution were provided. The 36-bit arithmetic unit could be reconfigured for 9-bit, 18-bit, and 36-bit arithmetic (Frankovich and Peterson, 1957).

By 1959, the year in which the PDP-1 was designed, most of the important ideas of logical organization (covering addressing and address modification, sequencing and control, arithmetic, and i/o control) had been invented.[1] A review of the state of the art of logical organization pertaining at the time is given in (Beckman *et al*, 1961).

The most widely used random-access memories were based on ferrite cores; a ~~st~~ survey of the state-of-the-art is given in (Rajchman, 1961).

~~For the PDP-1, by far the most influential machines were those constructed at M.I.T. and the MIT Lincoln Laboratories, because many DEC engineers[2] were alumni of those laboratories. These machines were Whirlwind, the Memory Test Computer, TX-0, and TX-2. The relationship of the PDP-1 to these antecedents is discussed after the description of the PDP-1. The DEC modules were directly patterned~~

~~[2] Ken Olsen, Harlan Anderson, Stan Olsen, Dick Best, and Ben Gurley.~~

[1] By contrast, the major advances in realization were yet to come. Machines were just in the second technology generation - the integrated circuit gave rise to the third generation. Today, we ~~are~~ are in the fourth generation, that of large-scale integration.

The PDP-1 Instruction-Set Processor

There is no record of the goals, constraints and the objective function to measure PDP-1 design. However, it is clear that the PDP-1 ISP was a reaction to the TX-0. Rather than being program compatible, PDP-1 was really oriented to being producible and useable by a variety of programmers. To this end, it had ~~having~~ more instructions ~~than a prototype like TX-0~~ and had ~~having~~ a simpler I/O structure for ease of interfacing. It is unclear that making PDP-1 compatible to TX-0 was ever considered. As it turned out, TX-0 was continuously extended; hence compatibility would have been a dubious goal.

Figure 8 is a diagram of the PDP-1 (taken from the original programming manual) showing its registers and functional units. Most of the registers were named after those of the TX-0, and only the TX-0's live register was renamed to be the input-output register in the PDP-1. The I/O register is

in reality a multiplier-quotient register as well, and used as an accumulator extension. An index register was apparently not considered probably because of the increased cost.

Since the memory cost strongly determined the machine's price, a 4 Kword base was selected for the PDP-1. The instruction format was selected to be: 12 bits for addressing the 4 Kword memory; 1 bit for indirect addressing; and leaving 5 bits to select 32 possible instructions. Since the machine was oriented to control and because it was low cost, only word, integer and Boolean vector (logical) data types were included in the basic design. This only required seven data operators (+, -, x, /, and, or, exclusive or) for the one accumulator structure plus control instructions.

The first description of the PDP-1 order code by Harlan Anderson, DEC's Vice President, appeared in a company memoranda, M-1060, dated October 27, 1959. Although a few instructions were later added to improve subroutine calling, that two page memo assigned the order codes and their names.

Altogether, only 24 instructions were assigned in the initial design, and 28 instructions were eventually used in production machines. The ISP of PDP-1 is given in Table 1 and for comparative purposes the PDP-4 ISP is given side by side.

3.3 I/O Structure

Although it was unstated as such, a major design theme for the PDP-1 was easy interfacing and the ability to handle a variety of I/O devices. These goals were incorporated in a physical structure that permitted various I/O devices to be easily interfaced to the computer. One of the first user manuals was the Input-Output Systems Manual written by this author which described the methods now standard to minicomputer and microcomputer interface design. These methods included:

1. Program controlled transfers.

2. Program controlled transfers with sequence break (now called interrupt).

3. Multiple channel interrupt programmed control.

4. High speed channel data transmission (now called direct memory access).

The optional 16-channel sequence break (interrupt) system, based on the Lincoln Laboratory TX-2, provided the PDP-1 with a unique capability. The sequence break system permitted a program to handle much of the processing associated with I/O devices instead of using special hardwired control. Each time an I/O device had information to be transferred to memory, it caused an interrupt to the processor for handling. The PDP-1 sequence break was a real alternative to the large computers which evolved extensive I/O processors (e.g., the IBM 7090 channels). In fact, the PDP-1 was less expensive than a single IBM "channel". When I/O character transmission rates (e.g., magnetic tape, drums) exceeded that which could be handled by a program, the information was transmitted directly to the PDP-1's memory in blocks under the device's control. Inter-block control was then managed using the interrupt. This basic scheme has remained even in today's DEC computers.

Figure 7 shows a block diagram of Type 51 Magnetic Tape Control Unit that operated under program control. This controller used a minimum of hardware but required 100% of the processor's time to read or write data. Note that various signals were connected directly into the program flags so that the program could operate as fast as possible. Also, note there were no word buffers in the controller; rather, characters were assembled in the processor's I/O Register. While ~~we would not build~~ such a controller ~~today~~ which would not be designed today, ~~(requiring~~ requires 100% of a $120,000 computer's attention), this structure is identical to modern day microprocessor-based controllers.[1]

Footnote: ~~(Note~~ Each computer generation goes exactly through all stages of evolution of predecessor generations.) [Ref to part of book where this key observation is elaborated].

3-1 Physical Structure / Implementation

The PDP-1 engineering prototype (1/A) was first shown (see Fig. 1) in Boston at the Eastern Joint Computer Conference, December 1959. Note how the CRT was integrated into the console (Fig. 2). This was subsequently dropped for cost reasons, never to return to the DEC main line of computers, except briefly with the LINC and PDP-12. The consulting company, Bolt, Beranek, and Newman (BBN), purchased the first production machine (Fig. 3) (1/B) for delivery in November 1960. A third machine was constructed for internal use. After building the first two production machines, it was clear that modifications were needed for improved producibility, cost and reliability. The separate console required many cables, and the unreliable connectors between the console and computer meant the two parts could not communicate. The final, more producible design, called the 1/C, used DEC-produced cabinets, with the operator/maintenance console integrated into the cabinets (Fig. 4). This design had improved air flow using the cabinet as an air plenum with air entering from the base of the cabinet. The cabinet and module mounting scheme was used directly on PDP-4 and -5. The cabinet has remained relatively unchanged. Even today the same basic design houses the smaller metal boxed minicomputers and options for all computers.

The PDP 1/C version formed the basis of the machines that were mass produced. It used four cabinets instead of the three in the earlier version, and it preassigned space for the various options: multiply-divide, sequence break (the name given the 16-channel interrupt mechanism), memory extension control, and the high speed channel (what we now call the direct memory access capability). Numerous options were initially offered for

Craig

THE PDP-1 AND OTHER 18-BIT COMPUTERS[1] 12/6/77

PDP-1

In contrast to the large-scale scientific and business computers of the second generation, the PDP-1 had a much shorter word (18-bits) and a simpler instruction set (28 instructions in all). The first machine was demonstrated in late 1959. DEC's 5 MHZ 1000 series system modules were used in the implementation. Memory capacity was 4 Kwords, later expanded to 64 Kwords. The I/O structure included a sequence break option (the name given the 16-channel interrupt mechanism) and a high-speed channel (now called DMA for direct~~-~~memory~~-~~access). The processor and memory occupied four cabinets.

Although Digital was formed in 1957 with the explicit goal to build computers, the PDP-1 was not demonstrated until the winter of 1959. The principal backer of DEC, American Research and Development (ARD), was initially skeptical as to whether a computer company could be successful ~~succeed~~, but was enthusiastic about the possibilities for digital~~-~~logic modules for laboratory and system use. ~~Hence,~~ Thus the plan to build computers was conditional on the success of the modules. After one year of operation, DEC had met its profit and sales goals and was permitted to move on to computers. However, Ken Olsen felt that another year's wait would be worthwhile since it would give ~~in order to gain~~ additional business experience and provide ~~to have~~ a larger customer and financial ~~business~~ base. Ben Gurley came to DEC in the summer of 1959 to design and build the PDP-1.

[1] Principal author: Gordon Bell

M.I.T. (Whirlwind & Memory Test Computer) & Lincoln Laboratory (TX-0) Ancestry

For the PDP-1, by far the most influential machines were those constructed at M.I.T. and the M.I.T. Lincoln Laboratories, ~~because~~ since many DEC engineers* were alumni. Moreover, the DEC System modules were directly patterned after the circuits of the TX-0 and the TX-2 computers (Chapter 2). 1

The Lincoln Laboratory experimental~~, transistorized~~ computer, TX-0, was one of the earliest computers to use transistors. TX-0 in turn was related to MIT's Whirlwind (Everett, 1951 ~~and~~ Redmond and Smith, 1977), a computer which was operational in 1950~~*~~. Since Whirlwind had a 16-bit word length, it can be regarded as the forerunner of the modern minicomputer. This ancestry is uniquely Whirlwind's because it operated much faster~~:~~ ~~In~~ instead of ~~(random access to the~~ Williams storage tube~~;~~ and later ~~the~~ a core memory ~~versus~~ having a drum for primary memory~~)~~ it used a random-access ~~and--uniquely--had a short~~ Moreover, its short word length (16-bits) was unique. ~~as compared to the 32- to 40-bit word length for scientific machines of the von Neumann era.~~ The core memory was developed at M.I.T. (Forrester, 1951). The first core memory was exercised under control of a special computer, the Memory Test Computer (MTC). MTC operated ~~only~~ for just a short time because the core memory worked as planned; ~~hence,~~ the memory was then moved to Whirlwind.** Ken Olsen headed the team for the design and construction of MTC. 2

TX-0 was designed to test transistor circuitry, to verify that a 256 by 256 (65 Kword) core memory could be built (Mitchell and Olsen, 1956), and as a prelude to building the large scale, 36-bit word, TX-2 computer. This work was part of the SAGE air defense project ~~that~~ run by Lincoln ~~ran.~~ Labs. The initial computer was called the TX-1, a ~~32-bit~~ long word computer patterned after the 32-bit

2 **Whirlwind was dismantled in 1959 and moved to Wolf Research and Development where it was reassembled and operated until the 1970s. Whirlwind is now part of the Digital Museum Project, although the first core memory module and other parts have been given to the Smithsonian and other museums.

1 *Ken Olsen, Harlan Anderson, Stan Olsen, Dick Best and Ben Gurley.

SAGE AN/FSQ7 ~~that~~ designed & built by IBM (and M.I.T.) ~~had designed and built as part of the SAGE system~~. The large core memory work ~~by William Papian was~~ based on vacuum tube drives, was done by William Papian 2 and Dick Best (Best, 1957) and ~~the memory design~~ proceeded independently of the computer. ~~it was to work with~~. ~~The~~ Although, the appearance of Philco's surface-barrier transistor significantly simplified transistor circuit design, ~~Nevertheless,~~ it was felt that a computer design was needed in order to test the transistor circuitry. ~~-- this was the TX-0.~~

~~TX-0 featured a~~ The 6 microsecond cycle time ~~and~~ TX-0 resembled Whirlwind because of the large (12 inch) cathode ray tube and light pen console. TX-0 used paper tape I/O with a high speed (300 char/sec.) Ferranti paper tape reader and a Friden Flexowriter used both as a typewriter and paper tape punch. ~~It also had~~ T~~two toggle switch registers that~~ enabled the program ~~could~~ to access ~~and the first 16~~ memory words were also toggle switches for variables ~~and/~~or program. use.
The two toggle switch registers were accessible to a program & the first 16

Because of the experimental nature of TX-0, it was extremely simple, having ~~only two~~ just two program-accessible registers: ~~which could be accessed by the program:~~ the accumulator and the live register. The latter was used for controlling and buffering transfers to various I/O equipment. Later, at MIT, TX-0 was extended to have an index register, but the initial version had only four instructions encoded in 2 bits, with 16 bits to access memory. Three instructions accessed memory: "store in location"; "add from location", and "transfer if accumulator is negative to location". The fourth instruction, "operate", was for programmed controlled I/O transfers, and it also included commands that could be combined to give a large number of instructions. The encoding of the operate instruction was called microprogramming because bits in the instruction specified particular register transfer operations, and could be "programmed" (e.g., "clear the right half of the accumulator", "cycle the accumulator right one position", "start the paper tape reader"). These ~~which~~ could occur at one of six possible times during the execution of the instruction. By encoding the bits, a number of useful instructions could be formed, (e.g., with one instruction, a point could be displayed on the screen, and a new pseudo-random point generated).

~~Footnote:~~

~~As the TX-0 tests were concluded,~~ In 1958, the TX-0 was transferred from Lincoln Laboratory to M.I.T. (1958) for laboratory experiment control and for teaching. The memory size was reduced from 6~~5~~4 Kwords to 4 Kwords and eventually ~~added back~~ enlarged to 8 Kwords. It remained in service at M.I.T. until 1975 when it was ~~eventually~~ purchased by DEC for display in the Digital Distributed Museum Project.

Following the complexion of TX-0, work began on the much more extensive TX-2 ~~, it had~~ which used 22000 transistors ~~, whereas TX-0 used 3600.~~ as opposed to the 3600 in TX-0. The TX-2 development is described in (Clark, 1957). A principle design goal was an I/O organization more efficient than existing machines. The designers rejected a separate I/O processor and instead chose a minimum buffering scheme with direct transfer to memory. Additional program sequences (and associated program counters) were provided for these I/O transfers. Thus the inherent processing facilities of the CPU were used to effect I/O transfers. The PDP-1 sequence System break is directly patterned after the TX-2 I/O system (Forgie, 1957). Much internal parallelism existed in the CPU. Separate adders for indexing, PC incrementing, and instruction execution were provided. The 36-bit arithmetic unit could be reconfigured for 9-bit, 18-bit, and 36-bit arithmetic (Frankovich and

footnote

Peterson, 1957).

By 1959, the year in which the PDP-1 was designed, most of the important ideas of logical organization (covering addressing and address modification, sequencing and control, arithmetic, and i/o control) had been invented.[1] A review of the state of the art of logical organization ~~pertaining~~ at the time is given in (Beckman et al, 1961).

The most widely used random-access memories were based on ferrite cores; a survey of the state-of-the-art is given in (Rajchman, 1961).

The PDP-1 Instruction-Set Processor

There is no record of the goals, constraints and ~~the~~ objective function to measure the PDP-1 design. However, it is clear that the PDP-1 ISP was a reaction to the TX-0. Rather than being program compatible, PDP-1 was really oriented to being producible and useable by a variety of programmers. To this end, it had more instructions and had a simpler I/O structure for ease of interfacing. It is unclear that making PDP-1 compatible to TX-0 was ever considered. As it turned out, TX-0 was continuously extended; hence compatibility would have been a dubious goal.

Figure 8 is a diagram of the PDP-1 (taken from the original programming manual) showing its registers and functional units. Most of the registers were named after those of the TX-0; ~~and~~ only the TX-0's live register was renamed to be the input-output register in the PDP-1. The I/O register ~~is in~~

[1] By contrast, the major advances in realization were yet to come. Machines were just in the second technology generation--the integrated circuit gave rise to the third generation. Today, we are in the fourth generation, that of large-scale integration.

~~reality a~~ multiplier-quotient register. ~~as well, and~~ when used as an accumulator ~~[illegible]~~ completes the extension. An index register was ~~apparently not considered~~ rejected (probably because of the increased cost).

Since the ~~memory~~ cost of memory strongly determined the machine's price, a 4 Kword ~~base~~ minimum was selected for the PDP-1. The instruction format was ~~selected to be:~~ as follows 12 bits for addressing the 4 Kword memory; 1 bit for indirect addressing; ~~and~~ the ~~leaving~~ remaining 5 bits ~~to~~ selected 32 possible instructions. Since the machine was oriented to control applications and because ~~it was~~ low cost was a goal, the only ~~, only~~ data types which were included were word, integer and Boolean vector (logical). ~~data types were included in the basic design. This~~ ~~included were~~ Hence, ~~only required~~ just seven data operators (+, -, x, /, and, or, exclusive or) for the one accumulator structure ~~plus~~ and some control instructions were required.

The first description of the PDP-1 order code by Harlan Anderson, DEC's Vice President, appeared in a company memoranda, M-1060, dated October 27, 1959. Although a few instructions were later added to improve subroutine calling, that two page memo assigned the order codes and their names.

Altogether, only 24 instructions were assigned in the initial design, and 28 instructions were eventually used in production machines. The ISP of PDP-1 is given in Table 1 and for comparative purposes the PDP-4 ISP is given side by side.

I/O Structure

Although it was unstated as such, a major design theme for the PDP-1 was easy

[1] By contrast, the major advances in realization were yet to come. Machines were just in the second technology generation--the integrated circuit gave rise to the third generation. Today, we are in the fourth generation, that of large-scale integration.

interfacing and the ability to handle a variety of I/O devices. These goals were incorporated in a physical structure that permitted various I/O devices to be easily interfaced to the computer. One of the first user manuals was the Input-Output Systems Manual, ~~written by this author~~ which described the methods which are now standard ~~to~~ in minicomputer and microcomputer interface design. These ~~methods~~ included:

1. Program controlled transfers;

2. Program controlled transfers ~~with sequence break (now called interrupt)~~ using a program interrupt mechanism (called the Sequence Break System).

3. Multiple channel interrupt programmed control; and

4. High speed channel data transmission (now called DMA for direct memory access).

The optional 16-channel sequence break ~~(interrupt)~~ system~~, based on the Lincoln Laboratory TX-2,~~ provided the PDP-1 with a unique capability. The sequence break system (i.e. program interrupts) permitted a program to handle much of the processing associated with I/O devices instead of using special hardwired control. Each time an I/O device had information to be transferred to memory, it caused an interrupt to the processor for handling. The PDP-1 sequence break was a ~~real~~ marked alternative to the large computers which evolved to using extensive I/O processors, (e.g., the IBM 7090 channels). In ~~fact~~ addition, the PDP-1 was less expensive than a single IBM channel. When I/O character transmission rates, (e.g., those for magnetic tape, drums), exceeded ~~that~~ the rates which could be handled by ~~a~~ program, the information was transmitted directly to the PDP-1's memory in blocks under the

[1] By contrast, the major advances in realization were yet to come. Machines were just in the second technology generation--the integrated circuit gave rise to the third generation. Today, we are in the fourth generation, that of large-scale integration.

device's control. Inter-block control was then managed using the interrupt. This basic scheme is still in use ~~has remained even~~ in today's DEC computers (~~The logic, behind the To a discussion of why I believe this is valid is con~~ see also page 00 for a discussion of the rationale).

Figure 7 shows a block diagram of Type 51 Magnetic Tape Control Unit that operated under program control. This controller used a minimum of hardware but did require~~d~~ 100% of the processor's time to read or write data. ~~Note that~~ For high speed operation, the various signals were connected directly into the program flags. ~~so that the program could operate as fast as possible~~. Also, note there were no word buffers in the controller; instead ~~rather~~, characters were assembled in the processor's I/O Register. While such a controller, which requires 100% of a $120,000 computer's attention, would not be designed today, this structure is identical to modern day microprocessor-based controllers.[1]

Physical Structure/Implementation

The PDP-1 engineering prototype (1/A) was first shown (see Fig. 1) in Boston at the Eastern Joint Computer Conference, December 1959. Note how the CRT was integrated into the console (Fig. 2). This was subsequently dropped for cost reasons, never to return to the DEC main line of computers, except briefly in a few PDP-6s, and with the LINC and PDP-12. ~~Now,~~ In the post fourth generation, we return to having the computer in the CRT. better coupling by The consulting company, Bolt, Beranek, and Newman (BBN), purchased the first production machine (Fig. 3) (1/B) for delivery in November 1960. A third machine was constructed for internal use. After building the first two production machines, it was clear that modifications were needed for improved producibility, cost and reliability. The separate console required many cables, and the unreliable connectors between the

[1] Each computer generation goes exactly through all stages of evolution of predecessor generations. ~~[Refer to part of book where this key observation is elaborated]~~. ~~(See also page~~ (see ~~also~~ page 00)

console and computer ~~meant the two parts could not communicate~~ were unreliable. The final, more producible design, called the 1/C, used DEC-produced cabinets, with the operator/maintenance console integrated into the cabinets (Fig. 4). This design, had improved air flow by using the cabinet as an air plenum with air entering from the base of the cabinet. The cabinet and module mounting scheme was used directly on PDP-4 and -5. The cabinet has remained relatively unchanged. ~~Even toda~~y the same basic cabinet design houses the smaller metal boxed minicomputers and options for all computers. in the ~~later~~ third and fourth generations, too.

The PDP 1/C version formed the basis of the machines that were mass produced. It used four cabinets instead of the three in the earlier version; ~~and it~~ space was preassigned ~~space~~ for the various options: multiply-divide, sequence break (the name given the 16-channel interrupt mechanism), memory extension control, and the high speed channel (what we now call the direct memory access capability). Numerous options were initially offered for PDP-1. These are shown in the Fig. 5. ~~in a fashion that corresponds to the physical structure.~~ A radial-interconnect scheme was used; this contrasts with the bussed interconnect scheme common today ~~Note that cables connect in a radial fashion to the various options, as opposed to being interconnected in a bussed fashion as computers are now.~~ (The reader should note that new design is reverting back to radial configuration in the early fourth generation because of the decreased cost of logic, the high interconnect cost, and need to bound the system.)

A side view of the cabinet in Fig. 6, shows the space for interconnecting to other options. ~~Note that~~ Expansion was accommodated by adding bays to the four-bay structure and by interconnecting stand-alone options via cables. Since the I/O structure was fundamentally radial ~~(versus modern day bussed),~~

~~Each computer generation goes exactly through all stages of evolution of predecessor generations. [Refer to part of book where this key observation is elaborated].~~

it was easy to cable to free-standing controllers and their options (e.g., magtape, displays, printer, card equipment, etc.).

~~As previously mentioned, the PDP-1 was built after DEC had a complete line of general purpose modules.~~ The extra year of module design before PDP-1 was constructed permitted more low speed (500 Khz) modules (for I/O equipment) to be designed using the same circuit techniques, but with less expensive, slower transistors. ~~Of the computers built with modules (versus modern third and fourth generation computers),~~ the PDP-1 was built from only 34 module types (including memory modules), and each module type was fully general purpose! Only five module types were added for the (analog) memory circuitry. The module types and their characteristics include:

Circuit Type	High Speed (5 Mhz clock)	Low Speed (500 Khz clock)
. Inverters, gates and decoders	7	5
. Pulse amplifiers and delay lines	4	2
. Flip-flop configurations	2	
. Special drivers and signal conditioning	4	3
. Core memory circuits	5	-
	-----	-----
	22	12

Market Experience with the PDP-1

Because of its short word length and high speed, the PDP-1 was particularly suited to the laboratory and scientific control applications that were to emerge later in the second generation. ~~Even~~ The small, lower cost scientific computers from Bendix (the G-15) and Librascope (LGP-30) had long word lengths but were slow because they were serial, a consequence of using drum for primary memory. ~~They~~ This slow speed ~~were of~~ limited their utility in computation, control and laboratory applications.

~~due to their slow speed~~. The fact that the PDP-1 was unorthodox ~~by being~~ high speed with a short word length ~~(5 microsecond memory cycle), while having only an 18-bit word with~~ and no built-in floating point arithmetic also created a market credibility problem. For comparison, the IBM 709 had a 12 microsecond cycle, and its successor, the 7090, had a 2.12 microsecond cycle. It was also difficult to market the PDP-1 because potential buyers really doubted whether a company with only 100 employees and less than a million dollars in sales would be a reliable and long-lived computer supplier.

The first few PDP-1s were sold by and large as planned, that is for applications in scientific computation and real time control. Users interacted with the PDP-1 directly via its typewriter, CRT and console. Customers included Lawrence Livermore Laboratory (for high speed support processing ala the IBM 1401 but with graphics I/O that was compatible with most large scale computers); Bolt, Beranek and Newman (for psycho-acoustics and general computer science research); the Atomic Energy of Canada, Limited (for pulse height analysis and van de Graaf generator experiment control). The most important sale (in terms of DEC's future) was to ITT for a message switching system.

Nearly half of the PDP-1s ~~"1s"~~ were used, ~~sold to ITT as their~~ as the ADX 7300, for this message switching application. This application was, in essence, the automation of a torn tape switching center, ~~and included interfaces for up~~ up to 256 teletype lines could be interfaced. These lines were switched under program control in a store-and-forward ~~fashion~~ scheme on a character-by-character basis using the interrupt facility of the PDP-1. The PDP-1 was uniquely suited for this application because of its high speed and high performance sequence break ~~(interrupt)~~ system (interrupt system) which permitted low cost line ~~units (~~telegraph line interfaces~~)~~.

Aside from the ~~training of~~ experience gained from having to produce computers that could run unattended and without service, ~~on a high reliability basis,~~ the most important result of the ITT order was that it allowed DEC to build a number of identical machines without special engineering. The first few machines ordered by other customers were nearly all ~~unique~~ different, requiring DEC to build options that were only sold a few times. In addition, ~~to the nearly one-of-a-kind options,~~ many of the machines had ~~special~~ interfaces that were ~~peculiar~~ unique to the applications.

It should be noted that because the hardware for the "1" was relatively inexpensive, DEC had a good supply of basic modules for building special interfaces. It was ~~not difficult~~ easy to interface ~~to~~ and building specialized hardware was relatively easy compared to modern day hardware design. If an engineer made errors in estimating the time required to design and test an option, the project would still be financially successful because the cost of materials was low enough to allow for such errors. Also, design errors could be corrected by wiring changes, ~~which are more~~ a much easier process than that demanded by ~~were easier to wire around compared to~~ modern day, expensive printed circuit boards with hard-to-change, read-only memories. Finally, the special interfaces and controllers for PDP-1 were quite simple compared to modern designs.

The Role of PDP-1 in Timesharing

A number of computer scientists at M.I.T.~~,~~ and BBN~~, and Oxford University~~ believed that it was necessary to provide interactive access to computers. ~~(as opposed to batch)~~. Furthermore, the only way to make this economically viable was to simultaneously share the computer among the users. Two experiments were carried out to demonstrate its feasibility: the IBM 7090 system at M.I.T. (Corbato et al, 1962) which later became the Compatible Time Sharing System, CTSS; and the shared PDP-1 at BBN (McCarthy et al, 1963).

Batch multiprogramming (see also Strachey, 1959) was an important part of the design of the Stretch computer (Buchholtz, 1962) and the Atlas computer (Kilburn, et al 1962). It is oriented towards hardware efficiency in the sense of trying to attain high utilization of all components. Timesharing, on the other hand, is concerned with the efficiency of persons trying to use a computer--the efficiency of man-computer interaction (Corbato et al, 1962).

A set of requirements were identified for a timesharing system. Unless a workload is restricted to (a) programs that are specially designed to run concurrently; and (b) programs which are error-free, one needs:

a. memory protection

b. program and data relocatability

c. a supervisor program

d. a timed return to the supervisor

e. interpretive execution of i/o instructions

The BBN Timesharing System on the PDP-1 began operation in September, 1962. Five teletype users shared the upper 4 Kwords of memory; the lower 4K held the supervisor program, called the "Channel 17 routine". The modifications to the PDP-1 to effect timesharing were embodied in a "restricted mode" of operation. They matched the above requirements list in the following way.

a. memory protection

Switching between the two 4K areas required the use of an i/o instruction.

b. relocatability

Because only one user was resident at one time, this was not needed.

c. a supervisor program

The Channel 17 clock routine fulfilled this function.

d. a clock interrupt

The Channel 17 clock generated an interrupt every 20 msecs.

e. i/o instructions

Whenever the PDP-1 was in restricted mode, an attempt to obey an i/o instruction caused a Channel 16 sequence break.

The TYC Control Language, ~~was regarded as a key portion of the software support. This~~ a debugging aid ~~was~~ adapted from the DDT language devised for the TX-0 and TX-2 computers, was regarded as important.

The "restricted mode" modifications formed the PDP-1/D, which also used the multiport memory designed for the PDP-6. Timeshared computers were built and operated at BBN, Stanford and M.I.T. These efforts influenced later timesharing by the PDP-6 computer (Chapter 22).

~~[CM: Read Licklider & Clark SJCC62 on the benefits of close man-machine interaction during problem solving, for~~

~~1. references to PDP-1~~

~~2. mention of this essential minicomputer contribution.]~~

Alphabetize these and move clear to end of this text. of fig 4

Best, R. L. Memory Units in the Lincoln TX-2, Proceedings of WJCC, 1957, pp. 160-166.

Clark, Wesley A. The Lincoln TX-2 Computer Development, Proceedings of WJCC, 1957, pp. 143-145.

Everett, R. R. The Whirlwind I Computer, AIEE - IRE Conference, 1951, pp. 70-74 (reprinted in Bell and Newell Chapter 6 pp137-145).

Forrester, J. W. Digital Information Storage in Three Dimensions Using Magnetic Cores, J. applied Physics, 22, pp. 44-48, January 1951.

McCarthy, J., Boilen, S., Fredkin, E., and Licklider, J. C. R. A Timesharing Debugging System for a Small Computer, Proceedings of SJCC, 1963, pp. 51-57.

stet ~~Strac~~k~~hey, C. Timesharing in Large Fast Computers, Proceedings of the International Conference on Information Processing, UNESCO, Paris, 15-20 June 1959, UNESCO, Paris, 1960, pp. 336-341.~~

Corbato, F. J., Mervin-Daggett, M., and Daley, R. C. An Experimental Timesharing System, Proceedings of SJCC, 1962.

Buchholtz, W. (e~~x~~d) Planning a Computer System, Wiley, New York, 1962.

Kilburn, T., Edwards, D. B. G., Lanigan, M. J., and Sumner, F. H. One-Level Storage System, IRE Trans. vol EC-11, no. 2, pp 223-235, April, 1962

Beckman, F. S., Brooks, F. P. J., and Lawless, W. J. Jr
Developments in the Logical Organization of Computer Arithmetic and Control Units, Proceedings of IRE, 49, 1 January 1961, pp. 53-66.

Rajchman, J. A. Computer Memories: A Survey of the State-of-the-Art, Proceedings of IRE, 49, 1 January 1961, pp. 104-127.

Frankovich, J. M., and Peterson, H. P.
A Functional Description of the Lincoln TX-2 Computer, Proceedings of WJCC, 1957.

Forgie, J. W. The Lincoln TX-2 Input-Output System, Proceedings of WJCC, 1957.

~~[CM: Find exact citation for Whirlwind project, IEEE Spectrum, October (?) 1977]~~

Mitchell, J. L. and Olsen, K. H.
TX-0, A Transistor Computer, Proceedings of EJCC, 1956, pp. 93-100.

Redmond, K. C. and Smith, T. M.
Lessons from "Project Whirlwind", IEEE Spectrum, October 1977, Vol 14, 10, pp 50-59.

Get a copy of this for Mario to check.

Table 1. The Instruction-Set Processors of PDP-1 and PDP-4

Basic PDP-1		Basic PDP-4	
Processor state			
AC<0:17>	accumulator	AC<0:17>	
IO<0:17>	io register		
PC<6:17>	program counter	PC<5:17>	
OV	overflow	L	link bit, multi-prec.
PF<1:6>	program flags		
Run	denotes operation	Run	
SB	sequence break mode	PI	program interrupt
Primary memory			
M[0:4095]<0:17>	4096 word memory	M[0:8191]<0:17>	8192 word memory
Console			
TWS<0:17>	test word switches	ACS<0:17>	ac switches
SS<1:6>	sense switches		
Instruction format			
i<0:17>	instruction	i<0:17>	instruction
op<0:4>:=i<0:4>	op code	op<0:3>:=i<0:3>	op code
ib :=<5>	indirect bit	ib :=i<4>	indirect bit
y<6:17>:=i<6:17>	address	y<5:17>:=i<5:17>	address

Instructions

<u>ad and store</u>

lac	load accumulator AC <- M[y]	lac	load accumulator AC <- M[y]
lio	load io register IO <- M[y]		
law	load accumulator with word AC <= op<5:17>	law	load (literal) word AC <= op<5:17>
dac	deposit accumulator M[y] <- AC	dac	deposit accumulator M[y] <- AC
dio	deposit io register M[y] <- IO		
dap	deposit address part M[y]<6:17> <- AC<6:17>		
p	deposit instruction part M[y]<0:5> <- AC<0:5>		
dzm	deposit zero in memory M[y] <- 0	dzm	deposit zero in memory M[y] <- 0

Arithmetic and logical

add	one's complement add {oc} OV@AC <- AC + {oc} M[y]	add	one's complement add L@AC <- AC + {oc} M[y]
		tad	two's complement add L@AC <- L@AC + M[z]
sub	one's complement subtract OV@AC <- AC - {oc} M[y]		
mus	multiply step AC@IO <- f(AC,IO,M[y])		
dis	divide step OV@AC@IO <- f(AC,IO,M[y])		
and	logical and AC <- AC AND M[y]	and	logical and AC <- AC AND M[y]
ior	inclusive or AC <- AC OR M[y]		
xor	exclusive or AC <- AC XOR M[y]	xor	exclusive or AC <- AC XOR M[y]

Program Control

jmp	jump PC <- y	jmp	jump PC <- y
jsp	jump and save program counter AC <- OV@(PC + 1); next PC<- y	jms	jump to subroutine M[y] <- PC; next PC <- y + 1
jda	jump and deposit accumulator M[y]<- AC; next AC<-OV@(PC+1); next PC<- y + 1		
cal	call subroutine M[100] <- OV@(PC + 1); next	cal	call subroutine M[20] <- PC; next PC <- 21

	PC <- 101		
x	index		
	AC <= M[y] <- M[y] + {oc} 1		
isp	index and skip if positive	isz	index and skip if zero
	AC <- M[y] <- M[y] + {oc} 1; next		M[y] <- M[y] + 1; next
	if AC GEQ 0 => PC <- PC + 1		if M[y] EQL 0 => PC <- PC + 1
sad	skip if accumulator & y differ	sad	skip if accumulator & y differ
	if AC NEQ M[y] => PC <- PC + 1		if AC NEQ M[y] => PC <- PC + 1
sas	skip if accumulator & y are the same		
	if AC EQL M[y] => PC <- PC + 1		
xct	execute instruction at y	xct	execute
	instruction <- M[y]		instruction <- M[y]

<u>IO control</u>

t	in-out transfer	iot	in-out transfer
	y selects device		y selects device and is microcoded to clear AC and generate control pulses
esm;lsm	enter; leave sequence breaks	ion;iof	interrupt on; off
	SBS <- 1; SBS <- 0		PI <- 1; PI <- 0

Rotate/shift

[illegible] to nine shifts can be specified per instruction. There are 12 instructions in all combinations of: right or left; rotate or shift; AC or IO or combined AC and IO.

Skip

sma	skip on minus AC if AC<0> EQL 1 => PC <- PC+1
spa	skip on plus AC if AC<0> EQL 0 => PC <- PC+1
sza	skip on zero AC if AC EQL 0 => PC <- PC+1
spi	skip on plus io register if IO EQL 0 => PC <- PC+1
szo	skip on zero overflow if OV EQL 0 => PC <- PC+1
szf	skip on zero flags if PF<s> EQL 0 => PC <- PC+1
szs	skip on zero sense switch if SS<s> EQL 0 => PC <- PC+1

Operate

cla	clear accumulator AC <- 0
cli	clear io register IO <- 0

Rotate instructions are encoded as part of operate class

Operate (Micro-encoded instruction)

First sequential operation

sma	skip on minus AC if AC<0> EQL 1 => PC <- PC+1
spa	skip on plus AC if AC<0> EQL 0 => PC <- PC+1
sza	skip on zero AC if AC EQL 0 => PC <- PC+1
sna	skip if non-zero AC if AC NEQ 0 => PC <- PC+1
szl	skip if zero link if L EQL 0 => PC <- PC+1
snl	skip if non-zero link if L NEQ 0 => PC <- PC+1
skp	skip always PC <- PC + 1

Second sequential operation

cla	clear accumulator AC <- 0
cll	clear link L <- 0

Third sequential operation

cma	complement accumulator	cma	complement accumulator
	AC <- NOT AC		AC <- NOT AC
		cml	complement link
			L <- NOT L
lat	load AC with switches	oas	inclusive or ac switches
	AC <- TWS		AC <- AC OR ACS
clf	clear selected prog. flag	rl & rr	rotate right or left, 1 or 2
	PF<s> <- 0		L@AC <- L@AC rotate left or right, 1 or 2
stf	set selected prog. flag		
	PF<s> <- 1		
hlt	halt	hlt	halt
	Run <- 0		Run <- 0

PDP-4 (Gordon Bell) 12/12/77

The PDP-4 was designed to extend DEC's line of computers by offering 5/8 of the performance of the PDP-1 at about 1/2 the price. It was a lower-cost successor, because we ~~We~~ believed ~~that~~ the demand for computers was completely elastic. The first delivery of the PDP-4 was made in the summer of 1962, about six months after the project was started.~~*~~[1]

Figure 9 shows the basic computer with 4 or 8 Kwords of memory, Teletype 28 KSR, and 300 char/sec. paper tape reader together with the optional 63.3 char/sec Teletype Corp. paper tape punch.

GOALS, CONSTRAINTS AND BASIC DESIGN DECISIONS

The primary goal of the PDP-4 was to have a mini(mal) to implement computer, yet one that was extremely easy to program, to configure, and to interconnect to real time and process control equipment. The Instruction-Set Processor (ISP) was in some respects simpler than that of the 12-bit PDP-5 because memory was directly addressable. The ISP is given in Table 1 and can be compared with that of the PDP-1. It had only 16 basic instructions. The block diagram of the processors, memory and the I/O interface section of the computer is shown in Fig. 10.

The Small Machine Concept

The notion of a much smaller machine started in the fall of 1961 with a discussion of process control applications with Foxboro Corporation and

[1] ~~*Footnote:~~ The PDP-4 was designed and implemented rapidly by Bell, Bill Colburn, who was a technician at the time, and Don White, who designed three ~~additional~~ special modules and enhanced the 500 Khz and 1 Mhz module line.

various customers. A machine, called the ~~DC 12 (for~~ Digital Controller~~)~~, or DC-12', was proposed; it was a 12-bit computer which would be oriented to process~~-~~control~~-~~data~~-~~collection and laboratory~~-~~processing applications. It incorporated ideas that Wesley Clark ~~was developing when he~~ designed in the Laboratory Instrument Computer (LINC) at Lincoln Laboratory. ~~and we studied~~ The CDC 160 was also studied. In the true tradition of the minicomputer, it was the smallest machine I could imagine, although we briefly considered building a copy or version of a computer Wes Clark designed called the L-1 I.

Footnote 1

¶The DC 12 became the PDP-5 (built in the Spring of 1963) for a PDP-4 process control front-end application. We had worked on a daisy chain I/O Bus (also for PDP-6, which was being done concurrently) architecture to reduce the cost of interfacing. The logic design of the PDP-5 was carried out by Edson De Castro, an applications engineer responsible for building the analog front end for a reactor control computer system for the Atomic Energy of Canada, Limited at Chalk River. The control computer was a PDP-4, but an elaborate analog monitoring system was needed for the front end. Hence, after visiting Chalk River in the winter of 1962 we decided on the PDP-5 as the simplest and most reliable method to carry out the task. The PDP-5 ISP was specified in detail by Alan Kotok.

The decision to switch from a 12-bit to an 18-bit design was taken very lightly in December, 1962. In retrospect, it may have been a poor decision. We probably should have examined whether ~~evolved~~ the PDP-1 could have evolved to a lower cost design. Although it is difficult to remember the events surrounding the decision, I now (perhaps poorly) recollect a number of reasons, all reactive to PDP-1 experience.

~~*Footnote: The PDP-4 was designed and implemented rapidly by Bell, Bill Colburn, who was a technician at the time, and Don White, who designed three additional special modules.~~

I L-1 was ~~a 9-bit~~ had a 9-bit word length. ~~and in it originated~~ The link bit, L, was introduced that permeated the LINC and PDP-4/5 and computers.

1. Simple machines with few instructions for a given number of data-types perform nearly as well those with ~~lots of~~ many instructions. Much of my own programming experience was using TX-0. I felt a significantly simpler 18-bit machine could be built than the PDP-1. This turned out to be true as the PDP-4 processor was implemented in less than 1/2 the space of the PDP-1, although it's unclear whether the size reduction was due to a simpler architecture, a much better logical design implementation or increased logic packing density.

2. I could build a better machine the second time around (actually 3rd or 4th if one counted MIT's Whirlwind and TX-0 computers). This belief was coupled with some new ideas about architecture.

   a. I had conceived the idea of auto-incrementing memory registers. This allowed vectors to be accessed easily instead of using index registers (which were more expensive to implement), and performed about as well.

   b. The PDP-1 used one's complement arithmetic ~~and this~~ which was especially poor for ~~doing~~ the fast multiple precision operations and floating point arithmetic which our users needed.

   c. Two control instructions were changed so that they did not ~~e~~affect the AC or ~~and interfere with~~ stet the arithmetic instructions. The jump to subroutine stored the return link in the program area (a convention that would not be used today) and the index and skip instruction ~~only~~ operated on memory only.

d. There were PDP-1 ~~several~~ features ~~of the PDP-1~~ that cost logic but ~~without~~ added little ~~adding much~~ to performance, ~~(~~e.g., program flags, and sense switches. ~~which were equivalent to IBM's sense lights)~~. These were eliminated as was the ~~The~~ wired-in program (read-in mode) to control the automatic reading of paper tape. ~~was also eliminated. Also, in general,~~ the goal was to build a modular design such that the optional equipment cost was associated with the option instead of being wired-in to all machines.

e. ~~Although the PDP-1 had~~ Because the ~~The~~ PDP-1 used a 4 Kword memory ~~before~~ memory bank switching; was common ~~but much of~~ because the useful system software required 8 Kwords of memory. I felt (erroneously) that ~~making a machine that would directly access~~ 8 Kwords of directly addressable memory would be ideal. (Little did I understand that programs expand to fill nearly any physical memory size--another possible corollary of Parkinson's Law.) However it did turn out that most PDP-4's stayed within the 8 Kword constraint although the machine could ~~would~~ operate with up to 32 Kword of memory.

3. In general,

4 ~~3~~. I wanted to use the Teletype Model 28 instead of a modified IBM Model B typewriter on PDP-1 to get better system MTBF and easier ~~easy~~ servicing.

5 ~~4~~. The logical design of the PDP-1, though quite straightforward, was not optimized ~~was not very~~ ~~had some~~ ~~tight.~~ It had redundant terms and the instructions weren't encoded to simplify the implementation. I felt that starting over, with a new ISP, was the only way to get a significantly smaller machine.

6~~5~~. ~~Began~~ The existing ~~We had both~~ peripherals and memories for the PDP-1 and ~~these~~ could be used immediately to assist the implementation of a successor ~~successful~~ computer. This again favored the ~~pushed us to~~ choice of an 18-bit (versus 12-bit) design.

12/29/77

Software

EAE

Another important consideration in the design was FORTRAN and scientific computation. 1 1 Footnote This in turn influenced us to ~~quickly~~ design ~~of~~ the Extended Arithmetic Element quickly. Our first users were adamant that they would write process control applications in a higher level language. We briefly considered ALGOL 60, but quickly decided to provide a FORTRAN II for the PDP-4. It turned out that ~~a few users did use~~ FORTRAN was used for computation, but most early users stayed with assembly language programming.

Intended Market

We had a fairly clear idea of whom the machine would serve. The PDP-4 was to be used predominately for process control, with some use in the laboratory for activities such as pulse height anlysis and data gathering, in much the same way as PDP-1~~s~~, ~~but it would be smaller and easier to use~~. We were talking with Foxboro about applications at Nabisco for baking control (with manual override) and to Corning about the control of a glass tube manufacturing process. We installed machines at both companies. Since Foxboro had been using a 24-bit RCA control computer, it is conceivable that they also ~~so~~ could have influenced us to decide on 18-bits. ~~instead of the 12-bit DC 12 we had proposed in order to speed control calculations and be more in line with conventions and intuition~~.

The PDP-1 4 Instruction Set Processor

[GB: need paragraph to support RT-level diagram]

6A insert

A

I/O Structure

The I/O interconnection was especially important so that users could connect equipment easily. The I/O section, ~~was~~ called the Real Time Control, ~~and~~ included the capability to interface with PDP-1 peripherals. The structure of the PDP-4 with its options is shown in Fig. 11. We did not consider ~~the I/O~~ ~~the~~ an an I/O bus structure (party line or daisy chain), ~~that~~ although one was developed for the PDP-5 one year later, but focused on improving the more conventional radial scheme, ~~that~~ to achieve ~~was used on computers at that time (e.g., PDP-1). This permitted~~ ~~to achieve~~ peripheral [GB:?] compatibility. In effect, the I/O bus was a few inches long, and consisted of an area where cable drivers and input gates could be patched (via taper pins) to special cables and equipment.

A

Process control was really process monitoring since computers were supposedly not reliable enough for control. As a result, we had slightly better input capabilities than output cable drive. The I/O was wired with taper pins to create a very flexible patch panel for ease in wiring controllers together within the same or adjacent racks. In fact, a complete PDP-4 with card reader, magnetic tape, display, and other options required only three bays, and many systems could fit within the two standard bays (see Fig. 12). This was less than half the size of a comparable PDP-1 system.

The PDP-4 was quite small indeed as previously discussed. ~~It can be most directly compared with PDP-1 using by observing the~~ Table 1 gives a direct comparison of ~~it with~~ the PDP-1 and PDP-4 ISPs. Here, ~~we~~ one can observe the slight difference in processor state and ~~num~~ number of instructions. ~~alternatively~~ A second method of comparison is to ~~observe the register~~ compare the ~~organization~~ structure of the two processors at the register transfer levels (~~see~~ figs 00 and 00).

Another consideration in the I/O structure was the ability to count events. This was used in scientific applications[1] (e.g. pulse height analysis) and in control to measure flow (e.g., via flow meters) or to count discrete items. Options such as a 16-channel clock used memory for event counting. This feature was implemented by having the device access a memory cell and then rewrite its contents plus one, thus changing the contents of memory as it is rewritten. Counting could occur at event rates up to 125 Khz.

This basic counting capability enabled the design of a relatively low cost, high performance direct memory access feature (See also PDP-8, page 00) called the three-cycle data break. This feature was first used in the Type 57A Magnetic Tape Controller for PDP-4. A direct memory access transmission via three cycle data break occurred as follows:

1. the word count word (in memory) to control the block length is incremented;

2. the pointer word (in memory) to the memory location being transferred is incremented; and finally

3. the data word pointed to is transferred between memory and the i/o device.

Physical Structure/Implementation

The main basis for building a new machine is a new technology. In this case, DEC had been extending its main business, the sale of logic modules, by extending the lower cost, slower speed, 500 Khz version of the 5 Mhz modules that were used in the PDP-1. The 500 Khz systems were significantly less expensive to build since they used germanium alloy transistors versus the MADT

[1] Resulted in a sale ~~at to~~ to the Columbia University Physics Department.

transistors for a 50% cost savings under the 5 Mhz modules. Some of the modules operated at a 1 Mhz clock. By being significantly slower, they were easier to use and more reliable because of the lower data rate and wider clock pulse. At the lower speed, two techniques further reduced the cost: diode-transistor logic versus one transistor per gate in the original circuits (with the wider clock); and capacitor-diode gates for the AND gates to registers. These techniques permitted: increased densities; lower speed and greater noise immunity; greater reliability through fewer active components; and much lower cost. Furthermore, we could now design the several dedicated modules including the accumulator and Teletype interface. Since the system modules only had 22 pins (18 pins for signals), the increased densities could not be directly applied to logical functions. We added a 10-pin connector on the back of each module for the register transfer gating signals so that bit-slice architecture could be used. In this way, one bit of the accumulator register and all the input gates were packaged on a single module as shown in Fig. 14.

The multi-stable state device was an especially interesting logic circuit devised ~~(and patented)~~ to simplify the control section of PDP-4. This ~~patented~~ circuit was a generalization of a flip-flop to n-stable states using n, n-1 input NAND gates wired in a cross-coupled way. I received a patent for the circuit and it was subsequently used in other computers and in the module line.

Maintenance

Maintenance was especially important although parity memory was not used in the design because the memory represented only a small fraction of the basic machine and perhaps we were overly concerned with cost. At the time, maintainance costs did not represent ~~the~~ such a high fraction of the ~~machine's~~ total cost. ~~they do in the~~ In addition to having a very simple architecture and a straightforward implementation, designers exercised a great deal of care in the logical design structure and the maintenance documentation. The machine instruction-set description only occupied one page, and the logical design flowchart (state diagram) was contained on one D-size drawing. The logical design drawings for the processor occupied seven D-size sheets.

As a further refinement to understanding the machine operation, each signal name source was named with a prefix to match one of the seven drawing names. This convention was carried forward through many DEC machines. The operator's console included several functions to assist maintenance. The console (see Fig. 13) had a speed control to enable the console functions (read, read next, write, write next, start, continue) to be repeated at a variable clock rate. This simplified testing by permitting a scope to be used easily. The most useful console functions were the reading and writing of information from the next location of memory. Thus, one could perform simple checks on memory by observing the contents of memory locations indicated by the console lights.

EVALUATION

PDP-1 Compatibility

At the time of the PDP-4 design, there was not any understanding or concern on DEC's part about the high cost of producing software. Since the PDP-1 was generally used in dedicated applications, there was no concern for system software, hence no notion that a new machine should capitalize on previous software. DEC had invested very little in the PDP-1 software; the users generated their own programs. M.I.T. had contributed a good macro assembler, linking loader, and interactive debugging program - DDT. Bolt, Beranek and Newman had contributed various subprograms. Because of this, there wasn't a knowledge of software cost nor was there an apparent need to know. It was easy to believe that a small part of the savings in the cost of a simpler machine could easily pay to write new software.

Now, I regard ~~the PDP-1~~ compatibility ~~of the "4"~~ to be a constraint in a new design. Furthermore, any new machine should be on an improving (with time) cost/performance line (as discussed in the final section). At that time there were about twenty PDP-1's installed at the time, and fifty were eventually built. A compatible machine ~~could probably~~ might have been built that would have interpreted most of the PDP-1 programs, offer improved cost/performance ratio features, and still not have been very much larger than the original PDP-4.

The Market

PDP-4 had limited success, although it did meet the corporate profit standard. It didn't sell as well as we expected, or obey the characteristics of having a completely elastic demand--relative to demand for the PDP-1. ~~This simplified~~ According to the ~~technolo~~ ~~demand~~ model ~~was to accommodate improvements we expected from technological advances. This model is~~ discussed in the final section of this chapter, a machine with~~. Thus, in order to~~ a lower price should have had the ~~meet the elastic demand, a machine released two years later should have been~~ ~~should have been pric~~ ~~priced less, and featured~~ with the same performance as the PDP-1. Alternatively it ~~could~~ must be priced much lower to account for the relatively poor performance. In summary, the PDP-4 was neither agressive enough in performance nor price. (The same argument is made about the PDP-8/S and 9/L.) Like most other first implementations of a series (i.e., PDP-1, PDP-5, PDP-6, LINC/8, PDP-14, PDP-11/20) the PDP-4 performed the poorest, financially, ~~of the 18-bit series~~. The PDP-7, 9, 9/L and 15 were necessary ~~follow ons~~ successors that utilized the software and hardware option base developed for PDP-4.

PDP-7 ~~(~~, Principal author: Ron Wilson~~)~~ 12/12/77

The original concept of the PDP-7 (or what finally was named the PDP-7) started with the design of the PDP-1/D. The PDP-1/D was a modified design of the PDP-1 implementing a memory bus concept (stemming from PDP-6) and ~~some~~ logic for timesharing. The initial plans were to simply repackage the PDP-1 using some higher density systems modules and to reduce the processor cycle time: a goal of much better performance at a lower price. Of all DEC's 18-bit computers, the PDP-7 had the best gain in performance/price over its predecessors (the PDP-1 and PDP-4).

GOALS AND CONSTRAINTS

Because the PDP-4 had a more extensive software (including a basic operating system and a FORTRAN compiler) and peripheral hardware option base, the PDP-7 was switched to be a follow-on to it rather than the PDP-1. Thus, primary design goals were to maintain program compatibility with the PDP-4 and to provide an increased performance at a lower price. Therefore the I/O interface scheme was constrained by the timing and structure of the past computers.

My personal goal was to sell 120 systems, a number greater than the combined total of the other DEC computer systems sold at that time. These goals, although sounding quite broad, were rather restrictive, especially the requirement for program and peripheral compatibility.

The performance goal was to decrease the cycle time from 8 microseconds to 1.75 microseconds, -- the practical limit of core memories. This was a rather ambitious goal and required designing a new core memory system and a ~~special~~ set of modules called the Blue Line (~~the~~ stet high speed series, of ~~within~~ the new Flip Chip modules) as described in chapter 3. They were based on our 10 megahertz systems modules. The whole module series would consist of various speed ranges and would be as general purpose as possible. ~~The central processor and the memory used the high speed (10 Mhz clock) Blue Line series. The~~ Whereas the CPU and memory used the 10 Mhz circuits, the I/O section of the system, although originally implemented using the Blue and White series, was later redesigned to use a majority of R-series modules (2 Mhz clock).

Time was considered a very important factor in the design[1] of the PDP-7.

[1] The entire logic implementation was undertaken by myself and one assistant, Jack Williams. Later we were joined by a Field Serviceman, Don Zereski, who literally hand-built the first production system to be delivered. also add X from page 5

The project started on April 1, 1964, and the first production system was delivered to Bell Laboratories December 22, 1964 (see Fig. 15).

Program compatibility between the PDP-4 and the PDP-7 (although maintained generally) was slightly modified in the I/O section. The Teletype® KSR28 ~~as used~~ on the PDP-4 ~~was~~ used a 5-level Baudot code. A shift to the ASCII 8-level code had already started in the industry, and we wanted to use the Teletype® KSR33 (Keyboard Send-Receive -- no paper tape) as the system terminal ~~input/output device~~. This necessitated a test at the front end of all programs to see if the system was a PDP-4 or a PDP-7. Other than this, an upward compatibility was maintained even though several additional instructions were added along with a trap feature and an option multi-level interrupt to allow multi-user environments. Also, note the return ~~addition~~ of a program read-in mode to the console (see Fig. 16), ~~Ironically, this feature had originated in the PDP-1 and was removed in the PDP-4 design.~~ a facility offered on the PDP-1.

PDP-7 Instruction-Set Processor

Figure 19 shows the structure of the processors with its registers and the interface to I/O equipment and to core memory. Note the structure and style of design was fundamentally that used in the earlier designs (suitably modified for the higher speed technology). In fact, using the notion of chapter ~~0~~1, PDP-~~1~~7 and -4 have identical architectures, very similar implementations, and radically different realizations. In order to use the slower (PDP-4 compatible) I/O equipment, special pulses were used when a slow cycle of 8 microseconds was requested. The I/O section and new options were designed to operate at the 1.75 microsecond cycle rate.

Physical Structure/Implementation

A system diagram of the PDP-7 processor is given in Fig. 17 showing the options and the general interconnection scheme. It was fundamentally the same structure as its predecessors, as it was designed for use with many of the earlier peripheral controllers.

A semi-automatic wire-wrapping technique was developed to allow a much higher speed production of wire-wrapped back planes. Also a Gardner Denver automatic wire-wrap machine was ordered and the programs to control it were developed. Mechanical block holders were designed to mount the connector blocks in the cabinets.

Physically, the PDP-7 was larger than the PDP-4, ~~its predecessor,~~ because the console was mounted on the side plane instead of the end. This mounting of the console facilitated maintenance. Notice (see Fig. 18) that the number of logic panels for the processor was the same as in the PDP-4, even though the printed circuit board area was slightly greater using the new modules (PDP-4 was 6 x 25 x 5-1/2 x 4 = 3300 sq.in., and the PDP-7 was 6 x 2 x 32 x 2-1/4" x 3-7/8" = 3348 sq.in.). Although it does not show in the diagrams or photos, a significant volume of the machine contained cable connectors to various subassemblies. The PDP-7 improved the cabling by having all the connectors in the backplane so that all wiring could be done as a single operation. The third bay of the PDP-7 housed the console and, I/O Paper Tape equipment (that had been on an extra table in the PDP-1 and -4).

EVALUATION

The Market

The PDP-7 was a successful project[1]; it did sell over 120 systems. [1] Design costs, excluding module and labor costs, were less than $100,000 to get the first prototype. Technically, it was a very sound system which developed an excellent reputation for reliability. Quite a few PDP-7s are still in operation.

Major Development Problems

~~These were:~~

1. A complete line of new modules, the Flip Chip series, ~~had to be~~ were being developed in the company ~~developed to implement the design of the computer~~, although the 10 Mhz circuits had been tested in the PDP-6.

2. New connector blocks had to be obtained to hold the modules. (This design effort was concurrent with PDP-8.)

3. New wire-wrap techniques had to be devised to ease labor requirements and speed up wiring systems. A program was ultimately developed for the PDP-4 for wire routing and controlling the Gardner-Denver automatic wiring machinery.

~~4~~. System layouts had to be developed to facilitate wire-wrapping.

4 ~~5~~. Mechanical packaging and cooling had to be altered to hold systems wire-wrapped panels. The air plenum scheme used in PDP-1, 4 and 5 was completely blocked by the new connector blocks.

5. The memory core operated at 1.75 microseconds, which was faster than the 2 microsecond time for ~~older~~ PDP-6. ~~Also~~ Cost was also an important design consideration.

The memory performance and cost goals were the most difficult part of the design.

X goes in footnote page 2

~~6~~. Memory control and stack were carried out by a memory design engineer, Derrick Chin, who coordinated his design with the processor logic design.

Acknowledgements ← add at page 7

The ambitious design and time goals on the PDP-7 project were met by long hours and much cooperation of a large number of people in the many different departments. The module design engineers designed the lay-out, supervised the manufacturing and specified the testing of hundreds of modules. The mechanical engineering department designed & ~~ordered~~ and ~~ordered~~ obtained connector blocks ~~which were originally purchased from an outside source~~.

purchased

PDP-9 BEGINNINGS

As the design phase of the PDP-7 neared an end and production models were being delivered, two developments occurred that made us consider an improved production model. The R-series Flip Chip modules were lower cost, lower speed and more complete than the B-series forming the processor.

After analyzing the configurations being ordered, we were able to redesign the I/O panel. The new I/O panel was designed using as many R-series modules as possible. The controls for several of the most popular peripherals were prewired into it to reduce the amount of special wiring required to produce a system. We tagged this improvement as the PDP-7A.

With the PDP-7A now completed, our thoughts turned to the next generation system. Many more tools were now in place to allow some significant improvements in system design. The Gardner Denver automatic wire-wrapping system had been installed and was ready for use. We started on our next system which we called the 7X project (it later became the PDP-9). The design criteria called for the following:

1. It would be completely automatic wire-wrappable.

2. A system with 8 Kwords would sell for approximately $35,000.

In order to meet these goals, a new cabinet design was started so that we could mount the wire-wrap door-type frames. The frames ~~both~~ holding the modules opened to allow access to ~~a~~ either ~~both~~ the ~~pin side~~ connector (back panel) ~~and~~ or modules (for replacement or oscilliscope signal tracing) This cabinet also required better air flow characteristics because of the high density of the logic then available. Past cabinets pulled air from the floor which was not a desirable feature in ~~many~~ some installations, so a horizontal air flow system was implemented.

The system costs were becoming a major factor. To meet our price goals, the computer was divided both logically and physically into three divisions: 1 - Memory, 2 - Central Processor, 3 - Input/Output Logic.

This was done to allow us to control and calculate costs more closely.

The cabinet design was completed, and ~~with~~ the logic design was moving ahead smoothly, the project was taken over at this point by Larry Seligman who had designed the Extended Arithmetic Element for the PDP-7.

~~Acknowledgements~~

~~[get from p5]~~

[1] Principal author:

PDP-9 and 9/L[1] (Don Vonada) 12/12/77

The PDP-9 (see Fig. 20) was introduced in August 1966 and a less expensive version, the PDP-9/L was introduced in November, 1968.

Technology ~~at the time~~ was ~~still~~ discrete PNP transistor (-3 volt signals), capacitor-diode gate[1], packaged on single, double and quad height standard double-sided Flip-Chip modules~~[1]~~. These modules, whose size was approximately 2-1/2 in. (5 in. or 10 in.) by 5-1/2 in., were plugged into an assembly of 144-pin connector blocks using 24 gauge wire-wrap pins as the interconnect media. The basic hardware was identical to that used for the PDP-7.

Like its predecessors, the central factor in the technology was the cost and speed of core memory. A new memory design using 30 mil inside diameter cores and a 2-1/2 D driving sense structure ~~known as 2-1/2 D~~ (in contrast to the standard coincident current selection; see PDP-8 Chapter) allowed the memory to cycle in 1 microsecond. This required only 3 wires through each core in the stack, rather than 4 as in earlier designs (see PDP-8, page 00). The memory was oriented to an 8 Kword organization which offered a cost advantage. It must be noted that discrete components logic costs were high compared to memory, but memory limited system performance. The PDP-7 had a 1.75 microsecond memory cycle; the system performance of the PDP-9 with its 1 microsecond memory was improved by a factor of 1.75 over that of the PDP-7.

The PDP-9 Instruction-Set Processor

~~**[CM: need an RT-level block diagram]**~~ The basic PDP-9 implemented the PDP-4 ISP with Extended Arithmetic Element Option using microprogrammed control. Figure 00 shows a register transfer level diagram of the ~~bus~~ processor together with I/O and Memory interface lines.

The I/O Structure

1 ~~Footnote 1:~~ Although some integrated circuits were available, there were no standards set yet and no cost or speed advantage ~~from~~ by using IC's.

The I/O control extended the features of earlier models by implementing an eight level nested automatic priority interrupt facility and a data channel transfer facility. The Automatic Priority Interrupt (API) had four levels of hardware interrupt capability available at the I/O Bus and four levels of software priority. The data channel transfer facility was the same as a direct memory channel except ~~in the location of~~ that ~~the~~ word count and current address registers were located in core memory — called the three cycle data break. The I/O structure ~~was designed to allow these~~ with control registers ~~to be~~ implemented in core memory, further attest~~ing~~ed to the fact that core was cheaper than logic. (This scheme was first used in PDP-4 with the Type 57A magnetic tape control.)

The Direct Memory Access channel (DMA) was the most disappointing part of the I/O Bus concept because ~~an i~~ another ~~set~~ cable set was required beyond the I/O Bus. The speed requirement dictated the use of ~~an~~ the extra set of data and address lines which were carried between the DMA device and the memory bus multiplexer via an extra set of cables. Also, a second port to memory was required. The idea of a "clean" bus cabling scheme for high speed transfer devices was not possible because of the extra radial lines required. (Alternatively, we might have slowed the machine down to handle the transfers.)

1 ~~Footnote 1:~~ Although some integrated circuits were available, there were no standards set yet and no cost or speed advantage from using IC's.

Physical Structure/Implementation

Figure 21 shows how the logic was mounted in three self-cooled frame sections, each capable of holding eight rows of forty modules. Each section had self-contained final power regulations (occupying four module spaces). A system block diagram of the PDP-9 (Fig. 22) shows ~~how~~ the ~~PDPs have evolved~~ evolution to the I/O and memory bus structured computer. This scheme is in contrast to the radial structure of the earlier 18-bit computers and provided greater modularity.

The major cost improvement was the implementation of an I/O Bus, a concept derived from the PDP-5 and 6. The bus was daisy-chained from device to device using twisted pair cables. Compared with the PDP-7 which was custom wired for each option, this technique provided uniformity in I/O backplane wiring; it allowed independent development, manufacturing and test of I/O options and simplified field installation of options. Also, it allowed costs to be associated with each option rather than being initially higher as in the radial scheme where all options had to be planned for in the central processor. The bus structure created the illusion that systems of unlimited size could be built.

EVALUATION

Except for the 300 wire field change on the first ten processor backplanes, the PDP-9 enjoyed a good reputation for performance and up time. It was later accompanied by the cost reduced version PDP-9/L. The cost reduction took the form of a new (and somewhat cumbersome) power supply design, a 4 Kword memory design which used the Teletype 33 ASR (Automatic Send-Receive

- with paper tape) instead of separate paper tape reader and punch. The memory planes were borrowed from the PDP-8 line and adapted to provide half the memory in half the space. The life of the 9/L was comparatively short, for it was soon overtaken by the PDP-15.

Microprogramming

The PDP-9 was the first microprogram controlled processor implemented at DEC. Although in itself a technological advancement, it was also an interesting precursor of things to come. The structure of the processor is shown in Fig. 23. Note its simplicity compared to the earlier designs. It used a more general data path through an adder compared to the ad hoc register structure of the earlier machines. A 64-word, 36-bit, 212-nanosecond cycle read-only, rope, magnetic memory was used as the microprogrammed control store for the processor. The design allowed for easy bench modification in the event the microcode required changing. The original concept of a unary encoded[1] control soon was lost in the complexities of the design. Encoding the bits was the resulting compromise position which eliminated the possiblity of providing special purpose machines (we had hoped for) by a simple ROM change. The size of the control memory further constrained the extendability and use of the microprogram control. It would have been nice to be able to build in floating point arithmetic. There were not enough words, a problem all too familiar whether ~~implementing~~ programming ~~machines all of any type~~ either ~~both~~ macro ~~and~~ or micro machines.

A ~~further~~ testimony to the power of control store was the fact that the extended arithmetic element, a hardware 36-bit multiply/divide option, was implemented with only six single height flip chip modules. By comparison, the processor occupied about 320 module slots or a total printed circuit

[1] also known as horizontal microprogramming. ~~Ho~~ ~~Hw~~, Each bit in the microinstruction denotes an action and can be specified independent of ~~all~~ other microinstructions. The behavior is ~~identi~~ similar to the Operate class of ~~PDP-15~~ the 18-bit and 12-bit computers (see page 80)

board area of 3,100 sq.in. This area also included the optional arithmetic element and much of the I/O control; thus it was about half the size of the earlier machines. (We ~~also~~ found an error in the PDP-1 signed integer divide algorithm.) A discrete carry adder[2] which developed the carry over 18 bits in under 30 nsec. was necessary in order to meet the performance goals.

[2] Design was carried out ~~of~~ by Richard Sogge.

12/30/77

1 ~~written by~~ Principal author:

PDP-15[1] (Gerry Butler) 12/12/77

Unlike its predecessors, the PDP-15 was designed to provide a range of systems with both hardware and software. While the early 18-bit machines evolved to include several configurations, the notion of a planned range was explicit from the start. As it turned out, the PDP-15 evolved over a considerably wider range than was anticipated. Table 2 shows the ~~various~~ range; ~~systems and options which served as bases for new systems, but only those~~ only the models through the 15/40 were ~~planned from the start.~~ in the original plan.

As in the past, the goal was to provide a machine with better cost/performance than ~~the~~ its predecessor, ~~prece~~ ~~PDP-9.~~ The PDP-7 to PDP-9 transition ~~did~~ had not provide a big cost improvement; however, the new ~~The PDP-15~~ semiconductor technology could. ~~provided the impetus for a new implementation.~~ The 7400 and 74H00 series integrated circuits of transistor logic (TTL) permitted clock speeds of 10 to 20 Mhz, lower costs and higher packing densities (for lower packaging costs) than the discrete circuits used in PDP-9. The basic PDP-15/10 turned out to be the smallest (see Fig. 24) of the 18-bit series, while providing a number of options and additional features including an additional instruction set with an index and limit register. Also, the memory size was extended to 131 Kwords of memory. A separate control unit, called the I/O processor, handled the bookkeeping for the I/O channels and I/O bus. Figure 25 shows a ~~fundamentally~~ markedly smaller system (including the core memory). Note the two processors occupy only a third of the cabinet space, yet are faster and have more capability than the PDP-9. Also note that the packaging reverted to the simplicity of the earlier PDP-1 -4 an -7 using ~~no swing-out gates~~ a. fixed mounting structure.

Table 2: The PDP-15 Family of 18-bit Computer Systems

	Hardware	Software
PDP-15/10		
(Basic Paper Tape System)	CPU	Assembler
	4K Memory	Editor
	Teletype	Debugger
		Utilities
PDP-15/20		
(Keyboard Monitor using	CPU	Keyboard Monitor
DECTape File System)	8K Memory	Fortran IV
	EAE	Focal
	Paper Tape	PIP
	DECTape	Utilities
	Teletype	
PDP-15/30		
(Background/Foreground)	CPU	B/F Monitor
	16K Memory	Fortran IV
	EAE	Focal
	API	PIP
	Memory Protect	Utilities
	Clock	
	Paper Tape	

	DECTape	
	2 Teletypes	
PDP-15/35	(15/30 with Disks)	
PDP-15/40		
(Disk Based B/F)	CPU	Disk B/F Monitor
	24K Memory	Fortran IV
	EAE	Focal
	API	PIP
	Memory Protect	Utilities
	Clock	
	Paper Tape	
	DECTape	
	524K Fixed Hd. Disk	
	2 Teletypes	
PDP-15/50	(16K memory based)	
PDP-15/76	15/40 + PDP-11	11-based File and I/O Device Management

MJ
spelling?

~~Unlike its predecessors,~~ T the change from discrete transistors to integrated circuits did cause a perterbation~~s~~ which had not previously occurred in the 18-bit evolution ~~since~~: the ICs used ground and +5 volt logic signals, ~~in contrast~~ whereas the predecessor circuits used ~~compared to the~~ ground and -3 volt signals. ~~of its predecessors.~~ This change caused a great break with the past: ~~in terms of using~~ earlier peripherals could not be used. A block diagram to the final, evolved state of the PDP-15/76 is shown in Fig. 26. The initial design did not have floating point arithmetic; however a design ~~but the project~~ was started and 50% of the ~~basic design~~ base PDP-15 design completed at ~~prior to~~ the first shipment. Table 3 shows the effect of ~~a~~ the floating point processor with built-in data-type. The PDP-11 (and the Unibus) did not exist at PDP-15 introduction time. Hence, the initial structure had only a connection to memory, two processors, and an I/O bus that was somewhat compatible with the PDP-9 (so that equipment could be designed for either PDP-9 or PDP-15). There were well-defined initial goals which are described below. Finally, there is a discussion of the evolution beyond the initial design.

Table 3: **FP15 Floating Point Computation Times**

Program Type	Was Software Time	With FP15	Speed Up*
Matrix Inversion	12 Sec.	5 Sec.	2.4
Fourier Transform	16.9 Sec.	2.9 Sec.	5.8
Least Squares Fit	5.1 Sec.	.7 Sec.	7.3
Test of all F.P. Functions	11.4 Sec.	1.4 Sec.	8.1
A Physics Application Program	37 Sec.	3 Sec.	12.3

GOALS

850 Nanosecond Cycle Time

An 800 nsec. cycle time exceeded our (850 ns) objective, but it was difficult for several reasons:

> The CPU, memory, and I/O were made asynchronous to reduce I/O latency reduction. The synchronizing logic was complicated and resulted in significant circuit delays.
>
> A DC (round trip) interlocked memory bus was used in order to handle speed-independent memories. This caused communications delays.
>
> To minimize cabling, we shared one set of lines for address and data to memory. This caused additional communication delays.

PDP-9 Compatibility

PDP-9 instruction compatibility was achieved, but with three minor exceptions. These caused no customer or DEC software problems, to our knowledge.

Manufacturing Cost

We exceeded our manufacturing cost goal in the first fifty units. In the process, we learned that the majority of costs were in the mechanical packaging. Sights were set on reducing cabling, using a single power harness which could be built and tested on a jig, and using standard H911 mounting panels for logic. The cabling was reduced to:

1 - console cable

1 - teletype cable

1 - I/O bus cable assembly

2 - memory bus cables

In trying to limit console cabling, a time division multiplex communication scheme was designed to get enough signals to the lights. In this fashion, a number of signals were transmitted on the same wires on a timeshared basis. The lamp filament was used as the storage element at the console. This was the machine's only patent. ~~Ultimately~~ the console logic was too complex; when ~~the console~~ it failed, it was harder to fix than the processor.

Improved Priority Interrupt Latency

We were not able to reduce interrupt latency to the two microsecond goal. With parity, memory protect and relocate options implemented, and with adder and synchronizing delays, a four microsecond time was achieved, which was acceptable ~~in~~ for the system.

One Cabinet Package

The CPU, I/O processor, console and 32 Kword memory were packaged in one cabinet, but it was close. There are virtually no spare module slots in the CPU.

I/O Bus Length

The I/O Bus was extended to 75 feet by switching to a terminated bus. The PDP-9 used an unterminated transmission line with diode clamps. The PDP-15 has cleaner signals and no reflections. A new set of bus driver/receiver modules were designed which were faster and offered less bus loading. The penalty paid was higher power consumption and greater power supply cost than in PDP-9.

Better Maintainability

A means to look at more than 400 signal points was provided in the logic. This, coupled with the single step feature allowed troubleshooting of the machine from the console without an oscilliscope. As it turned out, this feature was used infrequently because of the required training.

Other Cost and Manufacturing Goals

Since the PDP-15 was one of the first DEC computers to use ICs extensively, it was important to minimize the number of logic types. The PDP-15 has 21 semiconductor types. This includes ICs, transistors, and diodes. All are available from multiple suppliers.

Having no twisted pairs in the backplane was a tough goal to achieve, It was necessary because there was no inexpensive wiring capability at the time. We achieved this goal, but logic had to be moved before ~~the~~ a reliable machine could be produced ~~and reliably.~~

~~Fixed Configuration Rules with Field Installation of Options~~

The PDP-15 has fixed configuration rules and very easy field installation of options. This required more space (partially full cabinets) and as a result the cost is slightly higher.

Margin testing of the PDP-15 was planned using a combination of varying logic timing and temperature. Special test equipment was provided for the production line to allow rapid heat cycling of CPUs and memories. In addition, a fast program loader system was designed around a PDP-8 with multiple DECtape units. Diagnostic programs could be loaded into the memory of a unit being tested by merely pressing a button. This saved considerable time in computer check-out.

Subassembly replacement was planned at the final assembly and test group. The concept was that if a processor, memory, power supply or whatever failed to work when it was integrated into a system, the whole subassembly would be replaced and sent back to its appropriate test line rather than doing the repair in the area. This process proved impractical, as the production line was never filled with enough material to allow this to take place.[1]

Early First Customer Ship

[1] The same ~~planning~~ situation was planned for with ~~in~~ the PDP-9 but was never executed for the same reason.

The first PDP-15 was to be shipped eighteen months after commencement of the project.[1] We ran into a number of difficulties during the project, including personnel turnover, which caused a two-month slip for the first computers.

footnote

Budget

The project was finished at about 92% of budget.

New Peripheral Designs (except Direct Memory Access Controllers) Were to Work on Both PDP-9 and PDP-15

This goal was met, newer devices were used on the PDP-9 by changing the receiver/driver modules. The PDP-15 was the first 18-bit machine to use IC positive logic (versus older negative transistor series logic), bus conversion signals were required in order to be able to use the PDP-9 peripheral controllers.

PDP-15 Instruction-Set Processor

Figure 27 shows the register transfer structure of the processor. It is based on elements and features used in earlier designs. It has a basic data path structure which permits the results from any of the eleven registers to be read into the arithmetic unit and back into the registers. On the other hand, in order to operate at high speeds, a number of separate registers such as the Step Counter, Program Counter, and Multiplier-Quotient registers must operate in parallel with the basic data path. In this way, significant overlap occurs, which is necessary in order to achieve the 800 nanosecond memory cycle time. The contrast with the PDP-4 is noteworthy: it had only four registers in the basic machine. Because the PDP-15 used integrated circuits, the goal of minimizing registers was less important.

The first major extension to the PDP-15 was the addition of a floating point processor (Fig. 28) to enable it to perform well in the scientific/computation marketplace using FORTRAN and other algorithmic languages. With the addition of the FP15 floating point processor the time for a programmed floating point operations was reduced from 100-200 microseconds to 10-15 microseconds, giving about a factor of 10 increase in FORTRAN performance as previously indicated in Table 3. The additional floating point unit required a number of additional instructions. The irony of this extension is that the PDP-11, and nearly all microcomputer ISP extensions exactly follow this evolution!

The least expensive multi-user protection scheme was provided in the form of a relocation register and a boundary register. Because it was marketed as an option, it degraded the machine more than necessary. However, the minimum machine cost was kept within the target.

Physical Structure/Implementation

Since ~~[Most of~~ the physical packaging has been discussed ~~above.]~~ in the goals and constraints, ~~we~~ we will describe the core memory aspects. Although PDP-15 did not (improve (significantly) performance, it did cost less. The IC logic and ~~lower~~ core memory ~~costs permitted the~~ provided the impetus for these cost ~~reductions~~ improvements. The PDP-15 memory is constrasted with the PDP-1 memory in the following table.

	PDP-1	PDP-15	PDP-15 (ME15)
Year	1960	19xx	19yy
Stack size	4 Kwords	4 Kwords	24 Kwords
Cycle time (microseconds)	5	0.8	0.8
Density (words/cabinet)	12 Kwords	48 Kwords	96 Kwords

		(32 Kwords with CPU included)	
Size electronics	1/3 cabinet	1/12 cabinet	1/24 cabinet
Configuration	3D stack	5 planes, 4 bits/plane; planar stack	
Core size	30 mil	18 mil	18 mil
Wires/Core	4	3	3

new page

TWO CENTRAL PROCESSOR PDP-15[1] ~~(Bob Gray)~~

[1]Principal Author: Bob Gray

The product line had sold a system that was a dual processor. From the sale came the MX15, a dual port memory, which eventually was transferred to the PDP-15 standard product line. The MX15 also expanded memory to the full 128 Kwords built into the PDP-15 addressing structure. The unit occupied a single rack and used the M series logic cards. Since there was space to add a third "port" within the rack unit, the MX15 turned out to be a three port device. At the time, the lab breadboard was, indeed, an impressive array of three cabinets containing 128 Kwords of memory and two processors.

The logic included what went unrecognized as a "synchronizer"~~*~~[1] problem for some two months despite reviews by some senior engineers. Once the problem was recognized the design went to a quick completion. Since the multiport memory ha~~s~~d three inputs (2 ports and the memory clock) to synchronize, it ~~can~~ could become unstable.

[1] ~~*~~A classical logic design problem that has turned out to be "theoretically" unsolvable. When synchronizing (detecting) the presence of an event occurring at a random time with a fixed clock event, a small amount of energy is available to set a flip-flop. When a flip-flop is triggered with a small signal, it can go into an undecided state undecided for a relatively long (even undecided) of time.

UNICHANNEL 15/76 (PDP-15 + PDP-11) - AND PDP-15/XVM

The Unichannel began as a problem: find the most cost-effective way to attach the RK05 disk to the PDP-15. After a review of the problem, it became clear that the correct[1] way to put the RK05 on the PDP-15 was to use an -11 processor, together with the RK05 controller on PDP-11. The key reason was not the cost of modifying the RK05 disk controller, rather, it was the cost of doing new diagnostics that would have to be executed in PDP-15 code. ~~Another reason was the ease of adding other PDP-11 family peripherals.~~[1] Our project was to run somewhat over six months and had a budget for the hardware of about $75K.

footnote

As the system design progressed, it became clear that the PDP-11 could run other PDP-11 family peripherals. Since most development and production resources were directed to the 16-bit, PDP-11 series, all factors favored using existing peripherals rather than inventorying a new set for PDP-15. The list of peripherals quickly included communications lines, plotter, printer, and card equipment. It was also conceived as a PDP-15. Figure 29 shows the options of the PDP-15/76 and Fig. 30 shows the physical cabinets together with powers, etc.

and

The project had a very small, but excellent staff.[1] [2] Al Helenius, did much of the logic design of the memory multiplexer device using M series logic modules. the prototype ~~It~~ was first operational in early November 1972. ~~In general~~ the hardware part of the program went very smoothly. The complexity and size of the software task was clearly underestimated.[3] Rick Hully proposed an

use this sentence as a 3rd footnote see next page operating system structure that, for the era and initial application, was both elegant, advanced and yet straightforward. However, the successful system operation depended on more software.[3] The reality was that we had evolved to a true "multiprocessor" system. Today the industry has begun to use the term "back-end processor" or "file-processor" for what was accomplished on the PDP-15 in the early seventies. (Also, this structure was used by IBM in their coupled 7090/7044 system and in their 360 Attached Support Processor.)

It is interesting that the dual processor structure is again receiving attention as microcomputers make the CPU no longer a ~~restricted~~ constrained resource, but one to be used generously throughout a system. The UNICHANNEL and 15/76 programs hold much valuable technical experience that has yet to be exploited or understood, even within DEC. One of the major reasons for the slow diffusion of this innovation has been the ongoing controversy of using "symmetrical multiprocessing" or "specialized task oriented multiprocessing". The current activity, in my opinion, vindicates the specialized approach, which the 18-bit computer line understood years ago.

The Market

From a market and user viewpoint, the 15/76 was successful. Users were able to offload I/O and file processing and at, the same time, obtain much lower priced peripherals. About 100 15/76s were delivered, and 300 were ~~retrofit to~~ added to upgrade existing PDP-15 installations.

PDP-4 (Gordon Bell)

The PDP-4 was designed to extend DEC's line of computers by offering 5/8 of the performance of the PDP-1 at about 1/2 the price. We believed that the demand for computers was completely elastic. The first delivery of the PDP-4 was made in the summer of 1962, about six months after the project was started. It was designed and implemented rapidly by myself, Bill Colburn, who was a technician at the time, and Don White, who designed ~~some~~ three additional special modules. ~~Photo 4~~ Figure 9 shows the basic computer with 4 or 8 Kwords of memory, Teletype 28 KSR, and 300 char/sec. paper tape reader together with the optional paper tape punch.

Goals, Constraints and The Design

← all Caps

The primary goal of the PDP-4 was to have a mini(mal) to implement computer, yet one that was extremely easy to program, to configure, and to interconnect to real time and process control equipment. The Instruction-Set Processor (ISP) was in some respects simpler than that of the PDP-5 because memory was directly addressable. The ISP is given in ~~Fig. ISP1/4~~ Table 1 and can be compared with that of the PDP-1. It had only 16 basic instructions. The block diagram of the processors, memory and the I/O interface section of the computer is shown in Fig. ~~Reg4.~~ 10.

The notion of a much smaller machine started in the fall of 1961 with a discussion of process control applications with Foxboro Corporation and various customers. This machine, called the DC 12 (for Digital Controller), was a 12-bit computer which would be oriented to process control data collection and laboratory processing applications. It incorporated ideas that Wesley Clark was developing when he designed the Laboratory Instrument Computer (LINC) at Lincoln Laboratory and we studied the CDC 160. In the true tradition of the minicomputer, it was the smallest machine I could imagine. Note, the DC 12 became the PDP-5 which was built in the Spring of 1963 for a PDP-4 process control front-end application. The PDP-5 ISP was specified in detail by Alan Kotok. We had worked on a daisy chain I/O Bus (also for PDP-6, which was being done concurrently) architecture to reduce the cost of interfacing. ~~In retrospect, the PDP-5 and PDP-6 I/O interfaces should have been made~~ The logic design of the PDP-5 was carried out by Edson De Castro, an applications engineer responsible for building the analog front end for a reactor control computer system for the Atomic Energy of Canada, Limited at Chalk River. The control computer was a PDP-4, but an elaborate analog monitoring system was needed for the front end. Hence, after visiting Chalk River in the winter of 1962 we decided on the PDP-5 as the simplest and most reliable method to carry out the task.

The decision to switch from a 12-bit to an 18-bit design was taken very lightly in December 1962. In retrospect, it may have been a poor decision. We probably should have evolved the PDP-1 to a lower cost design. Although it is difficult to remember the events surrounding the decision, I now (perhaps poorly) recollect a number of reasons. These were reactive to PDP-1 experience. These included:

1. Simple machines with few instructions for a given number of data-types perform nearly as well those with lots of instructions. Much of my own programming experience was using TX-0. I felt a significantly simpler 18-bit machine could be built than the PDP-1. This turned out to be true as the PDP-4 processor was implemented in less than 1/2 the space of the PDP-1, although it's unclear whether the size reduction was due to a simpler architecture, a much better logical design implementation or increased logic packing density.

2. I could build a better machine the second time around (actually 3rd or 4th if one counted MIT's Whirlwind and TX-0 computers). This belief was coupled with some new ideas about architecture.

   a. I had conceived the idea of auto-incrementing memory registers. This allowed vectors to be accessed easily instead of using index registers (which were more expensive to implement), and performed about as well.

   b. The PDP-1 used one's complement arithmetic and this was especially poor for doing fast multiple precision and floating point arithmetic which our users needed.

   c. Two control instructions were changed so that they did not effect the AC and interfere with the arithmetic instructions. The jump to subroutine stored the return in the program (a convention that would not be used today) and the index and skip instruction only operated on memory.

d. There were several features of the PDP-1 that cost logic without adding much to performance (e.g., program flags, and sense switches which were equivalent to IBM's sense lights). The wired-in program (read-in mode) to control the automatic reading of paper tape was also eliminated. Also, in general, the goal was to build a modular design such that the optional equipment cost was associated with the option instead of being wired-in to all machines.

e. The PDP-1 used a 4 Kword memory before memory bank switching; but much of the useful system software required 8 Kwords of memory. I felt (erroneously) that making a machine that would directly access 8 Kwords of memory would be ideal. (Little did I understand that programs expand to fill nearly any physical memory size--another possible corollary ~~controlling~~ of Parkinson's Law.) However it did turn out that most PDP-4's stayed within the 8 Kword constraint although the machine would operate with up to 32 Kword of memory.

3. I wanted to use the Teletype Model 28 instead of a modified IBM Model B typewriter on PDP-1 to get better system MTBF and easy servicing.

4. The logical design of the PDP-1 though quite straightforward was not very tight. It had redundant terms and the instructions weren't encoded to simplify the implementation. I felt that starting over, with a new ISP, was the only way to get a significantly smaller machine.

5. We had both peripherals and memories for the PDP-1 and these could be used immediately to assist the implementation of a computer. This pushed us to an 18-bit (versus 12-bit) design.

The previous section focussed on how the PDP-1 was used as a starting point. Furthermore, there still was not any understanding or concern on our part about the high cost of producing software. Since the PDP-1 was generally used in dedicated applications, there was no concern for system software, hence no notion that a new machine should capitalize on previous software. DEC had invested very little in the PDP-1 software; the users generated their own programs, ~~(~~M.I.T. had contributed a good macro assembler, linking loader, and interactive debugging program - DDT~~.~~ Bolt, Beranek and Newman ~~(BBN)~~ had contributed various subprograms~~)~~. Because of this, there wasn't a knowledge of software cost nor was there an apparent need to know. It was easy to believe that a small part of the savings in the cost of a simpler machine could easily pay to write new software.

Now, I regard the PDP-1 compatibility of the "4" to be a constraint in a new design. Furthermore, any new machine should be on an improving (with time) cost/performance line (~~to be~~ as discussed in the final section). At that time there were about twenty PDP-1's installed at the time, and fifty were eventually built. A compatible machine could probably have been built that would have interpretted most of the PDP-1 programs, offer improved cost/performance ratio features, and still not have been very much larger than the original PDP-4.

We had a fairly clear idea of whom the machine would serve. The PDP-4 was to be used predominately for process control, with some use in the laboratory for activities such as pulse height anlysis and data gathering, in much the same way as PDP-1, but it would be smaller and easier to use. We were talking with Foxboro about applications at Nabisco for baking control (with manual override) and to Corning about the control of a glass tube manufacturing process. We installed machines at both companies. ~~The Nabisco machine operated from 1962 till ? with only X failures for an MTBF of ? hours~~. Since Foxboro had been using a 24-bit RCA control computer, it is conceivable that they also influenced us to decide on 18-bits instead of the 12-bit DC 12 we had proposed in order to speed control calculations and be more in line with conventions and intuition.

Caps

The I/O Section

The I/O interconnection was especially important so that users could connect equipment easily. The I/O section was called the Real Time Control and included the capability to interface with PDP-1 peripherals. The structure of the PDP-4 with its options is shown in Fig. ~~Block4.~~ 11 We did not consider the I/O bus structure (party line or daisy chain) that was developed for the PDP-5 one year later but focused on improving the more conventional radial scheme that was used on computers at that time (e.g., PDP-1). This permitted peripheral compatibility. In effect, the I/O bus was a few inches long, and consisted of an area where cable drivers and input gates could be patched (via taper pins) to special cables and equipment.

Process control was really process monitoring since computers were supposedly not reliable enough for control. As a result, we had slightly better input capabilities than output cable drive. The I/O was wired with taper pins to create a very flexible patch panel for ease in wiring controllers together within the same or adjacent racks. In fact, a complete PDP-4 with card reader, magnetic tape, display, and other options required only three bays, and many systems could fit within the two standard bays (see ~~Sidvue4~~ Fig. 12). This was less than half the size of a comparable PDP-1 system.

Another consideration in the I/O structure was the ability to count events. This was used in scientific applications and in control to measure flow (e.g., via flow meters) or to count discrete items. Options such as a 16-channel clock used memory for event counting. This feature was implemented by having the device access a memory cell and then rewrite its contents plus one, thus changing the contents of memory ~~before~~ as it ~~was~~ is rewritten. Counting could occur at event rates up to 125 Khz.

This basic counting capability enabled the design of a relatively low cost, high performance direct memory access feature called the three cycle data break. This feature was first used in the Type 57A Magnetic Tape Controller for PDP-4.

A direct memory access transmission via three cycle data break occurred as follows:

1. the word count word (in memory) to control the block length is incremented;

2. the pointer word (in memory) to the memory location being transferred is incremented; and finally

3. the data word pointed to is transferred between memory and the i/o device.

Cap

Maintainence

Maintenance was especially important although parity memory was not used in the design. In addition to having a very simple architecture and a straightforward implementation, designers exercised a great deal of care in the logical design structure and the maintenance documentation. The machine instruction-set description only occupied one page, and the logical design flowchart (state diagram) was contained on one D-size drawing. The logical design drawings for the processor occupied seven D-size sheets.

As a further refinement to understanding the machine operation, each signal name source was named with a prefix to match one of the seven drawing names. This convention was carried forward through many DEC machines. The operator's console included several functions to assist maintenance. The console (see Fig. 13 ~~photo~~) had a speed control to enable the console functions (read, read next, write, write next, start, continue) to be repeated at a variable clock rate. This simplified testing by permitting a scope to be used easily. The most useful console functions were the reading and writing of information from the next location of memory. Thus, one could perform simple checks on memory by observing the contents of memory locations indicated by the console lights.

Caps

Physical Structure / Implementation

The main basis for building a new machine is a new technology. In this case, DEC had been extending its main business, the sale of logic modules, by extending the lower cost, slower speed, 500 Khz version of the 5 Mhz modules that were used in the PDP-1. The 500 Khz systems were significantly less expensive to build since they used germanium alloy transistors versus the MADT transistors for a 50% cost savings under the 5 Mhz modules. Some of the modules operated at a 1 Mhz clock. By being significantly slower, they were easier to use and more reliable because of the lower data rate and wider clock pulse. At the lower speed, two techniques further reduced the cost: diode-transistor logic versus one transistor per gate in the original circuits (with the wider clock); and capacitor-diode gates for the AND gates to registers. These techniques permitted: increased densities; lower speed and greater noise immunity; greater reliability through fewer active components; and much lower cost. Furthermore, we could now design the several dedicated modules including the accumulator and Teletype interface. Since the system modules only had 22 pins (18 pins for signals), the increased densities could not be directly applied to logical functions. We added a 10-pin connector on the back of each module for the register transfer gating signals so that bit-slice architecture could be used. In this way, one bit of the accumulator register and all the input gates were packaged on a single module as shown in Fig. ~~AC4.~~ 14.

The multi-stable state device was an especially interesting logic circuit devised (and patented) to simplify the control section of PDP-4. This circuit was a generalization of a flip-flop to n-stable states using n, n-1 input NAND gates wired in a cross-coupled way. ~~The author~~ I received a patent for the circuit and it was subsequently used in other computers and in the module line.

SOFTWARE

Another important consideration in the design was FORTRAN and scientific computation. This in turn influenced us to quickly ~~the~~ design of the Extended Arithmetic Element. Our first users were adamant that they would write process control applications in a higher level language. We briefly considered ALGOL 60, but quickly decided to provide a FORTRAN II for the PDP-4. It turned out that a few users did use FORTRAN, but most early users stayed with assembly language programming.

Caps

The MARKET

PDP-4 had limited success, although it did meet the corporate profit standard. It didn't sell as well as we expected, or obey the characteristics of having a completely elastic demand--relative to demand for the PDP-1. This simplified demand model was to accommodate improvements we expected from technological advances. This model is discussed in the final section. ~~when we discuss the evolution.~~ Thus, *a machine released two years later should have been priced less, and featured the same performance as the PDP-1. (Alternatively it could be priced much lower to account for the relatively poor performance.) In summary, the PDP-4 was neither agressive enough in performance nor price. (The same argument is ~~can be~~ made about ~~for~~ the PDP-8/S.) Like most other first implementations of a series (i.e., PDP-1, PDP-5, PDP-6, LINC/8, PDP-14, PDP-11/20) the PDP-4 performed the poorest, financially, of the 18-bit series. The PDP-7, 9, 9/L and 15 were necessary follow ons that utilized the software and hardware option base developed for PDP-4.

*In order to meet the elastic demand,

PDP-7 (Ron Wilson)

— Marie is Flip chip hyphenated?

The original concept of the PDP-7 (or what finally was named the PDP-7) started with the design of the PDP-1/D. The PDP-1/D was a modified design of the PDP-1 implementing a memory bus concept (stemming from PDP-6) and some logic to facilitate timeshared systems. The initial plans were to simply repackage the PDP-1 utilizing some newer designed systems modules of a higher density and to reduce the cycle time in order to speed up operations. This provided much better performance at a lower price. In fact, PDP-7 had the best gain in performance/price over its predecessors, PDP-1 and PDP-4, of all 18-bit computers.

Because of a more extensive software and peripheral hardware option base, the PDP-7 was switched to be a follow-on to the PDP-4. Thus, ~~T~~he primary design goals were ~~thus~~ to maintain program compatibility with the PDP-4 and to provide an increased performance at a lower price. ~~Also, it was desirable to utilize as many 18-bit peripherals as possible from the PDP-1 and -4.~~ Therefore the I/O interface scheme was constrained by the timing and structure of the past computers. My personal goal was to sell 120 systems, a number greater than the combined total of the other DEC computer systems sold at that time. These goals, although sounding quite broad, were rather restrictive, especially the requirement for program and peripheral compatibility.

The performance goal was to decrease the cycle time from 8 microseconds to 1.75 microseconds. This was a rather ambitious goal and required designing a special set of modules called the Blue Line (the high speed series, within the new Flip~~-~~Chip modules). They were based on our 10 megahertz systems modules. The whole module series would consist of various speed ranges and would be as general purpose as possible. The central processor and the memory used the high speed (10 Mhz clock) Blue Line series. The I/O section of the system, although originally implemented using the Blue and White series, was later redesigned to use a majority of R-series modules (1 Mhz clock).

1.75 Microsecond cycle time -- derived from a set of delay lines which interacted with memory to complete one cycle. (timing chain. not clock)

Don Vonada 2422

ask Dick Best (should be 2 ?)

Time was considered a very important factor in the design of the PDP-7. The entire logic implementation was undertaken by myself and one assistant, Jack Williams. Later we were joined by a Field Serviceman, Don Zereski, who literally hand-built the first production system to be delivered. The project started on April 1, 1964, and the first production system was delivered December 22, 1964 (see ~~photo 7~~ Fig. 15).

Program compatibility between the PDP-4 and the PDP-7 (although maintained ~~in the general sense~~ generally) was slightly modified in the I/O section. The KSR28 as used on the PDP-4 was a 5-level Baudot code. A shift to the ASCII 8-level code had already started in the industry, and we wanted to use the KSR33 (Keyboard Send-Receive -- no paper tape) as the system input/output device. This necessitated a test at the front end of all programs to see if the system was a PDP-4 or a PDP-7. Other than this, an upward compatibility was maintained even though several additional instructions were added along with a trap feature and option multi-level interrupt to allow multi-user environments. Also, note the addition of a program read-in mode to the console (see ~~photo Console 7~~ Fig. 16).

The major development problems were:

1. A complete line of new modules, the Flip-Chip series, had to be developed to implement the design of the computer, although the circuits had been tested in the PDP-6.

2. New connector blocks had to be obtained to hold the modules. (This design was concurrent with PDP-8.)

3. New wire-wrap techniques had to be devised to ease labor requirements and speed up wiring systems. A program was ultimately developed for the PDP-4 for wire routing and controlling the Gardner-Denver automatic wiring machinery.

4. System layouts had to be developed to facilitate wire-wrapping.

5. Mechanical packaging and cooling completely had to be altered to hold systems wire-wrapped panels. The air plenum scheme used in PDP-1½, 4 and 5 was blocked, ~~by the~~ by the new connector blocks.

6. Memory control and stack were ~~assigned to~~ carried out by a memory design engineer, Derrick Chin, who coordinated his design with the processor logic design.

The ambitious design and time goals on the PDP-7 project were met by long hours and much cooperation of a large number of people in the many different departments. The module design engineers designed the lay-out, supervised the manufacturing and specified the testing of hundreds of modules. The mechanical engineering department designed and ordered connector blocks which were originally purchased from an outside source.

A semi-automatic wire-wrapping technique was developed to allow a much higher speed production of wire-wrapped back planes. Also a Gardner Denver automatic wire-wrap machine was ordered and the programs to control it were developed. Mechanical block holders were designed to mount the connector blocks in the cabinets.

A system ~~System~~ ~~register transfer~~ ~~block~~ diagram of the PDP-7 processor is given in Fig. 17 ~~Block7~~ showing the options and the general interconnection scheme. It was fundamentally the same structure as its predecessors, as it was designed for use with many of the earlier peripheral controllers. Physically, the PDP-7 was larger than the PDP-4, its predecessor, because the console was mounted on the side plane instead of the end. This mounting of the console facilitated maintenance. Notice (see Fig. 18 ~~Sidevue7~~) that the number of logic panels for the processor was the same as in the PDP-4, even though the printed circuit board area was slightly greater using the new ~~FF.~~ modules ~~flip-chips~~ (PDP-4 was 6 x 25 x 5-1/2 x 4 = 3300 sq.in., and the PDP-7 was 6 x 2 x 32 x 2-1/4" x 3-7/8" = 3348 sq.in.). Although it does not show in the diagrams or photos, a significant volume of the machine contained cable connectors to various ~~This further reduced cost and r~~ subassemblies. The PDP-7 improved the cabling by having all the connectors in the backplane so that all wiring could be done as a single operation. The third bay of the PDP-7 housed the console equipment (that had been on an extra table in the PDP-1 and -4).

Figure ~~Reg7~~ 19 shows the structure of the processors with its registers and the interface to I/O equipment and to core memory. Note the structure and style of design was fundamentally that used in the earlier designs (suitably modified for the higher speed technology). In fact, using the notion of chapter ~~1~~ 0, PDP-1 and -4 have identical architectures, very similar implementations, and radically different realizations. In order to use the slower (PDP-4 compatible) I/O equipment, special pulses were used when a slow cycle of 8 microseconds was requested. The I/O section and new options were designed to operate at the 1.75 microsecond cycle rate.

The PDP-7 was a successful project; it did sell over 120 systems. Design costs, excluding module and labor costs, were less than $100,000. Technically, it was a very sound system which developed an excellent reputation for reliability. Quite a few PDP-7s are still in operation.

PDP-9 Beginning

Caps.

#B-series forming the processor

As the design phase of the PDP-7 neared an end and production models were being delivered, two developments occurred that made us consider an improved production model. The ~~lower cost, lower speed~~ R-series ~~flip-chip~~ Flip Chip modules were lower cost, and lower speed and more complete than the ~~previous PDP-7 circuits~~ ~~and~~, After ~~an analysis of the types of~~ analyzing the configurations being ordered, ~~allowed us to~~ we were able to redesign the I/O panel. The new I/O panel was designed using as many R-series modules as possible. The controls for several of the most popular peripherals were prewired into it to reduce the amount of special wiring required to produce a system. We tagged this improvement as the PDP-7A.

With the PDP-7A now completed, our thoughts turned to the next generation system. Many more tools were now in place to allow some significant improvements in system design. The Gardner Denver automatic wire-wrapping system had been installed and was ready for use. We started on our next system which we called the 7X project (it later became the PDP-9). The design criteria called for the following:

1. It would be completely automatic wire-wrappable.

2. A system with 8 Kwords would sell for approximately $35,000.

In order to meet these goals, a new cabinet design was started so that we could mount the wire-wrap frames. This cabinet also required better air flow characteristics because of the high density of the logic then available. Past cabinets pulled air from the floor which was not a desirable feature in many installations, so a horizontal air flow system was implemented.

The system costs were becoming a major factor. To meet our price goals, the computer was divided both logically and physically into three divisions: 1 - Memory, 2 - Central Processor, 3 - Input/Output Logic. This was done to allow us to control and calculate costs more closely.

The cabinet design was completed, and with the logic design moving ahead smoothly, the project was taken over by Larry Seligman who had designed the Extended Arithmetic Element for the PDP-7.

PDP-9 and 9/L (Don Vonada)

The PDP-9 (see ~~photo 9~~ Fig. 20) was introduced in August 1966 and a less expensive version, the PDP-9/L was introduced in November, 1968. ~~The Fig. Sidevue9L~~ Figure 21 shows how the logic was mounted in three self-cooled frame sections, each capable of holding eight rows of forty modules. Each section had self-contained final power regulations (occupying four module spaces). A system block diagram of the PDP-9 (Fig. ~~Block9~~ 22) shows how the PDPs have evolved to the I/O and memory bus structured computer. This scheme is in contrast to the radial structure of the earlier ~~radially distributed~~ 18-bit computers and provided greater modularity.

Technology at the time of the PDP-9 was still discrete PNP transistor (-3 volt signals), capacitor-diode gate, packaged, on single, double and quad height standard double-sided Flip-Chip modules. Although some integrated circuits were available, there were no standards set yet and no cost or speed advantage from using IC's. These modules, whose size was approximately 2-1/2 in. (5 in. or 10 in.) by 5-1/2 in., were plugged into an assembly of 144-pin connector blocks using 24 gauge wire-wrap pins as the interconnect media. The basic hardware was identical to that used for the PDP-7.

The central factor in the technology was the speed of core memory. A new memory design using 30 mil inside diameter cores and a driving sense structure known as 2-1/2 D (in contrast to the standard coincident current selection; see PDP-8 Chapter ~~X~~) allowed the memory to cycle ~~at a rate of~~ in 1 microsecond. The memory was oriented to an 8K 8 Kword organization which offered a cost advantage. It must be noted that discrete components logic costs were high compared to memory, but memory limited system performance. The PDP-7 had a 1.75 microsecond memory cycle; the system performance of the PDP-9 was improved by a factor of 1.75 over that of the PDP-7.

Other improvements which reduced cost and improved maintainability and reliability were also implemented. The major cost improvement was the implementation of an I/O Bus, a concept derived from the PDP-5 and 6. The bus was daisy-chained from device to device using twisted pair cables. Compared with the PDP-7 which was custom wired for each option, this technique provided uniformity in I/O backplane wiring; it allowed independent development, manufacturing and test of I/O options and simplified field installation of options. Also, it allowed costs to be associated with each option rather than being initially higher as in the radial scheme when all options had to be planned for in the central processor. The bus structure created the illusion that systems of unlimited size could be built.

The I/O control extended the features of earlier models by implementing an eight level nested automatic priority interrupt facility and a data channel transfer facility. The Automatic Priority Interrupt (API) had four levels of hardware interrupt capability available at the I/O Bus and four levels of software priority. The data channel transfer facility was the same as a direct memory channel except in the location of word count and current address registers. The I/O structure was designed to allow these control registers to be implemented in core memory, further attesting to the fact that core was cheaper than logic. (This scheme was first used in PDP-4 with the Type 57A magnetic tape control.)

The Direct Memory Access channel (DMA) ~~has been~~ was the most disappointing part of the I/O Bus concept. The speed requirement dictated the use of an extra set of data and address lines which were carried between the DMA device and the memory bus multiplexer via an extra set of cables. Also, a second port to memory was required. The idea of a "clean" bus cabling scheme for high speed transfer devices was not possible because of the extra radial lines required. (Alternatively, we might have slowed the machine down to handle the transfers.)

The PDP-9 was the first microprogram controlled processor implemented at DEC. Although in itself a technological advancement, it was also an interesting precursor of things to come. The structure of the processor is shown in Fig. ~~Reg9~~ 23. Note its simplicity compared to the earlier designs. It used a more general data path through an adder compared to the ad hoc register structure of the earlier machines. A 64-word, 36-bit, 212-nanosecond cycle read-only, rope, magnetic memory was used as the microprogrammed control store for the processor. The design allowed for easy bench modification in the event the microcode required changing. The original concept of a unary encoded control soon was lost in the complexities of the design. Encoding the bits was the resulting compromise position which eliminated the possiblity of providing special purpose machines (we had hoped for) by a simple ROM change. The size of the control memory further constrained the extendability and use of the microprogram control. It would have been nice to be able to build in floating point arithmetic. There were not enough words.

A further testimony to the power of control store was the fact that the extended arithmetic element, a hardware 36-bit multiply/divide option, was implemented with only six single height flip chip modules. By comparison, the processor occupied about 320 module slots or a total printed circuit board area of 3,100 sq.in. This area also included the optional arithmetic element and much of the I/O control; thus it was about half the size of the earlier machines. (We also found an error in the PDP-1 signed integer divide algorithm.) A discrete carry adder which developed the carry over 18 bits in under 30 nsec. was necessary in order to meet the performance goals.

Except for the 300 wire field change on the first ten processor backplanes, the PDP-9 enjoyed a good reputation for performance and up time. It was later accompanied by the cost reduced version PDP-9/L. The cost reduction took the form of a new (and somewhat cumbersome) power supply design, a 4 Kword memory design which used the Teletype 33 ASR (Automatic Send-Receive - with paper tape) instead of separate paper tape reader and punch. The memory planes were borrowed from the PDP-8 line and adapted to provide half the memory in half the space. The life of the 9/L was comparatively short, for it was soon overtaken by the PDP-15.

PDP-15 (Gerry Butler)

Unlike its predecessors, the PDP-15 was designed to provide a range of systems with both hardware and software. While the early 18-bit machines evolved to include several configurations, the notion of a planned range was explicit from the start. As it turned out, the PDP-15 evolved over a considerably wider range than was anticipated. Table ~~15~~ 2 shows the various systems and options which served as bases for new systems, but only those models through the 15/40 were planned from the start.

As in the past, the goal was to provide a better cost/performance than the PDP-9. The PDP-7 to PDP-9 transition did not provide a big cost improvement. The PDP-15 semiconductor technology provided the impetus for a new implementation. The 7400 and 74H00 series integrated circuits of transistor logic (TTL) permitted clock speeds of 10 to 20 Mhz, lower costs and higher packing densities (for lower packaging costs) than the discrete circuits used in PDP-9. The basic PDP-15/10 turned out to be the smallest (see ~~photo 15~~ Fig. 24) of the 18-bit series, while providing a number of options and additional features including an additional instruction set with an index and limit register. Also, the memory size was extended to 131 Kwords of memory. A separate control unit, called the I/O processor, handled the bookkeeping for the I/O channels and I/O bus. Figure ~~Sidevue15~~ 25 shows a fundamentally smaller system (including the core memory). Note the two processors occupy only a third of the cabinet space, yet are faster and have more capability than the PDP-9. Also note that the packaging reverted to the simplicity of the earlier PDP-1 using no swing-out gates.

Table ~~15~~ 2: The PDP-15 family of 18-bit computer systems

	Hardware	Software
PDP-15/10		
(Basic Paper Tape System)	CPU	Assembler
	4K Memory	Editor
	Teletype	Debugger
		Utilities
PDP-15/20		
(Keyboard Monitor using	CPU	Keyboard Monitor
DECTape File System)	8K Memory	Fortran IV
	EAE	Focal
	Paper Tape	PIP
	DECTape	Utilities
	Teletype	
PDP-15/30		
(Background/Foreground)	CPU	B/F Monitor
	16K Memory	Fortran IV
	EAE	Focal
	API	PIP
	Memory Protect	Utilities
	Clock	
	Paper Tape	
	DECTape	

	2 Teletypes	
PDP-15/35		
(15/30 with Disks)		
PDP-15/40		
(Disk Based B/F)	CPU	Disk B/F Monitor
	24K Memory	Fortran IV
	EAE	Focal
	API	PIP
	Memory Protect	Utilities
	Clock	
	Paper Tape	
	DECTape	
	524K Fixed Hd. Disk	
	2 Teletypes	
PDP-15/50		
(16K memory based)		
PDP-15/76	15/40 + PDP-11	11-based File and
		I/O Device ~~Managing~~ Management

Unlike its predecessors, the change from discrete transistors to integrated circuits did cause perterbations in the 18-bit evolution since the ICs used ground and +5 volt logic signals, compared to the ground and -3 volt signals of its predecessors. This change caused a great break with the past in terms of using earlier peripherals. A block diagram to the final, evolved state of the PDP-15/76 is shown in Fig. ~~Block15S~~ 26. The initial design did not have floating point arithmetic, but the project was started and 50% completed prior to the first shipment. Table ~~Float~~ 3 shows the effect of a built-in data-type. The PDP-11 (and the Unibus) did not exist at PDP-15 introduction time. Hence, the initial structure had only a connection to memory, two processors, and an I/O bus that was somewhat compatible with the PDP-9 (so that equipment could be designed for either PDP-9 or PDP-15). There were well-defined initial goals which are described below. Finally, there is a discussion of the evolution beyond the initial design.

Table ~~Float~~ 3.: FP15 Floating Point Computation ~~Speed Up~~ Times

Program Type	Was Software Time	With FP15	Speed Up*
Matrix Inversion	12 Sec.	5 Sec.	2.4
Fourier Transform	16.9 Sec.	2.9 Sec.	5.8
Least Squares Fit	5.1 Sec.	.7 Sec.	7 3
Test of all F.P. Functions	11.4 Sec.	1.4 Sec.	8.1
A Physics Application Program	37 Sec.	3 Sec.	12.3

OBJECTIVES AND RESULTS

Most objectives were achieved or exceeded. The following is a summary of various explicit goals, together with the decisions and results of the decisions.

850 Nanosecond Cycle Time

An 800 nsec. cycle time exceeded our (850 ns) objective, but it was difficult for several reasons:

> The CPU, memory, and I/O were made asynchronous to reduce I/O latency reduction. The synchronizing logic was complicated and resulted in significant circuit delays.
>
> A DC (round trip) interlocked memory bus was used in order to handle speed-independent memories. This caused communications delays.
>
> To minimize cabling, we shared one set of lines for address and data to memory. This caused additional communication delays.

Manufacturing Cost

We exceeded our manufacturing cost goal in the first fifty units. In the process, we learned that the majority of costs were in the mechanical packaging. Sights were set on reducing cabling, using a single power harness which could be built and tested on a jig, and using standard H911 mounting panels for logic. The cabling was reduced to:

1 - console cable

1 - teletype cable

1 - I/O bus cable assembly (? conductors)

2 - memory bus cables (? conductors)

} Call Gerry Butler.

In trying to limit console cabling, a time division multiplex communication scheme was designed to get enough signals to the lights. The lamp filament was used as the storage element at the console. This was the machine's only patent. Ultimately the logic was too complex: when the console failed, it was harder to fix than the processor.

In this fashion, a number of signals ~~shared the~~ were transmitted on the same wires on a timeshared basis.

Improved Priority Interrupt Latency

We were not able to reduce interrupt latency to the two microsecond goal. With parity, memory protect and relocate options implemented, and with adder and synchronizing delays, a four microsecond time was achieved, which was acceptable in the system.

One Cabinet Package

The CPU, I/O processor, console and 32 Kword memory were packaged in one cabinet, but it was close. There are virtually no spare module slots in the CPU.

I/O Bus Length

The I/O Bus was extended to 75 feet by switching to a terminated bus. The PDP-9 used an unterminated transmission line with diode clamps. The PDP-15 has cleaner signals and no reflections. A new set of bus driver/receiver modules were designed which were faster and offered less bus loading. The penalty paid was higher power consumption and greater power supply cost than in PDP-9.

Minimum Multi-User Protection Scheme

The least expensive multi-user protection scheme was provided in the form of a relocation register and a boundary register. Because it was marketed as an option, it degraded the machine more than necessary. However, the minimum machine cost was kept within the target.

Better Maintainability

A means to look at more than 400 signal points was provided in the logic. This, coupled with the single step feature allowed troubleshooting of the machine from the console without an oscilliscope. As it turned out, this feature was used infrequently because of the required training.

PDP-9 Compatibility

PDP-9 instruction compatibility was achieved, but with three minor exceptions. These caused no customer or DEC software problems, to our knowledge.

Minimum Semiconductor Types

Since the PDP-15 was one of the first DEC computers to use ICs extensively, it was important to minimize the number of logic types. The PDP-15 has 21 semiconductor types. This includes ICs, transistors, and diodes. All are ~~multiple sourced~~. available from multiple suppliers.

No Twisted Pairs

Having no twisted pairs in the backplane was a tough goal to achieve because there was no inexpensive wiring capability at the time. We achieved this goal, but logic had to be moved before the machine could be produced ~~made easily~~ and reliably.

Fixed Configuration Rules with Field Installation of Options

The PDP-15 has fixed configuration rules and very easy field installation of options. This required more space (partially full cabinets) and as a result the cost is slightly higher.

Improve Producibility

Margin testing of the PDP-15 was planned using a combination of varying logic timing and temperature. Special test equipment was provided for the production line to allow rapid heat cycling of CPUs and memories. In addition, a fast program loader system was designed around a PDP-8 with multiple DECtape units. Diagnostic programs could be loaded into the memory of a unit being tested by merely pressing a button. This saved considerable time in computer check-out.

Subassembly replacement was planned at the final assembly and test group. The concept was that if a processor, memory, power supply or whatever failed to work when it was integrated into a system, the whole subassembly would be replaced and sent back to its appropriate test line rather than doing the repair in the area. This process proved impractical, as the production line was never filled with enough material to allow this to take place.

Early First Customer Ship

The first PDP-15 was to be shipped eighteen months after commencement of the project. We ran into a number of difficulties during the project, including personnel turnover, which caused a two-month slip.

Budget

The project was finished at about 92% of budget.

New Peripheral Designs (except Direct Memory Access Controllers) Were to Work on Both PDP-9 and PDP-15

This goal was met, newer devices were used on the PDP-9 by changing the receiver/driver modules. The PDP-15 was the first 18-bit machine to use IC positive logic (versus older negative transistor series logic), bus conversion signals were required in order to be able to use the PDP-9 peripheral controllers.

Caps

Memory Evolution

While we have not commented on the memory packaging per se, it is worth noting the evolution that transpired between the PDP-1 and PDP-15.

Table 4 Memory Technology Evolution ~~In 19~~ PDP-1 ~~&~~ to 15.

	PDP-1	PDP-15	PDP-15 (ME15)	PDP-15 (MF15)
Year	1960	~~19xx~~ 1969	19~~yy~~73	1975
Stack size	4 Kwords	? ~~4~~ 8 Kwords	? 24 Kwords	? call John Elsbree x6939 MR
Cycle time (microseconds)	5	0.8	~~0.8~~ 0.980	?
Density (words/cabinet)	12 Kwords	48 Kwords (32 Kwords with CPU included)	96 Kwords	?
Size electronics	1/3 cabinet	1/12 cabinet	1/24 cabinet	?
Configuration	3D stack	5 planes, 4 bits/plane; planar stack	?	
Core size	30 mil	18 mil	18 mil	?
Wires/Core	4	3	3	?
Price /16 Kwords ($)	120,000	28,000	12,000	8,000
Price /word ($)	7.32	1.70	.73	.49

Caps

The Processor and Its Evolution

Figure ~~Reg15~~ 27 shows the ~~diagram~~ register transfer structure of the processor. It is based on elements and features used in earlier designs. It has a basic data path structure which permits the results from any of the eleven registers to be read into the arithmetic unit and back into the registers. On the other hand, in order to operate at high speeds, a number of separate registers such as the Step Counter, Program Counter, and Multiplier-Quotient registers must operate in parallel with the basic data path. In this way, significant overlap occurs, which is necessary in order to achieve the 800 nanosecond memory cycle time. Here, the reader should note the change by design philosophy from the PDP-4, which contained only four registers in the basic machine. With integrated circuits, the goal of minimizing registers is less important. The first major extension to the PDP-15 was the addition of a floating point processor (Fig. ~~Reg15FP~~ 28) to enable it to perform well in the scientific/computation marketplace using FORTRAN and other algorithmic languages. With ~~FPP (~~ the addition of the FP15 floating point processor the time for a programmed floating point operations was reduced from 100-200 microseconds to 10-15 microseconds, giving about a factor of 10 increase in FORTRAN performance. The additional floating point unit required a number of additional instructions. The irony of this extension is that the PDP-11, ~~microprocessor,~~ and nearly all ~~machine~~ microcomputer ISP extensions ~~have~~ exactly follow~~ed~~ this evolution!

Caps Two Central Processor PDP-15 (Bob Gray)

The product line had sold a system that was a dual processor. From the sale came the MX15, a dual port memory, which eventually was transferred to the PDP-15 standard product line. The MX15 also expanded memory to the full 128 Kwords built into the PDP-15 addressing structure. The unit occupied a single rack and used the M series logic cards. Since there was space to add a third "port" within the rack unit, the MX15 turned out to be a three port device. At the time, the lab breadboard was, indeed, an impressive array of three cabinets containing 128 Kwords of memory and two processors.

~~The MX15 project had its ups and downs.~~ The logic included what went unrecognized as a "synchronizer"* problem for some two months despite reviews by some senior engineers. Once the problem was recognized the design went to a quick completion. Since the multiport memory has three inputs to synchronize, it can become unstable.

Caps → Unichannel 15/76 (PDP-15 + PDP-11) – also PDP-15/XVM

The Unichannel began as a problem: find the most cost-effective way to attach the RK05 disk to the PDP-15. After a review of the problem, it became clear that the correct way to put the RK05 on the PDP-15 was to use an -11 processor, together with the RK05 controller on PDP-11. The key reason was not the cost of modifying the RK05 disk controller, rather, it was the cost of doing new diagnostics that would have to be executed in PDP-15 code. Another reason was the ease of adding other PDP-11 family peripherals. Our project was to run somewhat over six months and had a budget for the hardware of about $75K.

As the system design progressed, it became clear that the PDP-11 could run other PDP-11 family peripherals. Since most development and production resources were directed to the 16-bit, PDP-11 series, all factors favored using existing peripherals rather than inventorying a new set for PDP-15. The list of peripherals quickly included communications lines, plotter, printer, and card equipment. It was also conceived as a PDP-15. Figure 29 ~~(Block 15/7~~6) shows the options of the PDP-15/76 and Fig. 30 ~~(Frontvue15/76)~~ shows the physical cabinets together with powers, etc.

*A classical logic design problem that has turned out to be "theoretically" unsolvable. When synchronizing (detecting) the presence of an event occurring at a random time with a fixed clock event, a small amount of energy is available to set a flip-flop. When a flip-flop is triggered with a small signal, it can go into an undecided state undecided for a relatively long of (even undecided) time.

The project had a very small, but excellent staff. Al Helenius, did much of the logic design of the memory multiplexer device using M series logic modules. It was first operational in early November 1972. In general the hardware part of the program went very smoothly. The complexity and size of the software task was clearly underestimated. Rick Hully proposed an operating system structure that, for the era and initial application, was both elegant, advanced and yet straightforward. However, the successful system operation depended on more software. The reality was that we had evolved to a true "multiprocessor" system. Today the industry has begun to use the term "back-end processor" or "file-processor" for what was accomplished on the PDP-15 in the early seventies. (Also, this structure was used by IBM in their coupled 7090/7044 system and in their 360 Attached Support Processor.)

It is interesting that the dual processor structure is again receiving attention as microcomputers make the CPU no longer a restricted resource, but one to be used generously throughout a system. The UNICHANNEL and 15/76 programs hold much valuable technical experience that has yet to be exploited or ~~even~~ understood, even within DEC. One of the major reasons for the slow diffusion of this innovation has been the ongoing controversy of using "symmetrical multiprocessing" or "specialized task oriented multiprocessing". The current activity, in my opinion, vindicates the specialized approach, which the 18-bit computer line understood years ago.

*A classical logic design problem that has turned out to be "theoretically" unsolvable. When synchronizing (detecting) the presence of an event occurring at a random time with a fixed clock event, a small amount of energy is available to set a flip-flop. When a flip-flop is triggered with a small signal, it can go into an undecided state undecided for a relatively long of time.

Unichannel Results

From a market and user viewpoint, the 15/76 was successful. Users were able to offload I/O and file processing and at, the same time, obtain much lower priced peripherals. About 100 were delivered, and 300 were retrofit to existing PDP-15 installations.

edited
12/12/77

PDP-4 (Gordon Bell)

The PDP-4 was designed to extend DEC's line of computers by offering 5/8 of the performance of the PDP-1 at about 1/2 the price. We believed that the demand for computers was completely elastic. The first delivery of the PDP-4 was made in the summer of 1962, about six months after the project was started.* *Footnote: The PDP-4 ~~It~~ was designed and implemented rapidly by Bell ~~myself~~, Bill Colburn, who was a technician at the time, and Don White, who designed three additional special modules. Figure 9 shows the basic computer with 4 or 8 Kwords of memory, Teletype 28 KSR, and 300 char/sec. paper tape reader together with the optional paper tape punch.

GOALS, CONSTRAINTS AND ~~THE~~ BASIC DESIGN DECISIONS

The primary goal of the PDP-4 was to have a mini(mal) to implement computer, yet one that was extremely easy to program, to configure, and to interconnect to real time and process control equipment. The Instruction-Set Processor (ISP) was in some respects simpler than that of the 12-bit PDP-5 because memory was directly addressable. The ISP is given in Table 1 and can be compared with that of the PDP-1. It had only 16 basic instructions. The block diagram of the processors, memory and the I/O interface section of the computer is shown in Fig. 10.

The small machine concept

The notion of a much smaller machine started in the fall of 1961 with a discussion of process control applications with Foxboro Corporation and ~~others~~ stet ~~various~~ customers. A ~~This~~ machine, called the DC 12 (for Digital Controller) was proposed; it, was a 12-bit computer which would be oriented to process-control-data-collection and laboratory-processing applications. It incorporated ideas that Wesley Clark was developing when he designed the Laboratory Instrument Computer (LINC) at Lincoln Laboratory and we studied the CDC 160. In the true tradition of the minicomputer, it was the smallest machine I could imagine. ~~Note,~~ The DC 12 became the PDP-5 ~~which was~~ (built in the Spring of 1963) for a PDP-4 process control front-end application. The PDP-5 ISP was specified in detail by Alan Kotok. We had worked on a daisy chain I/O Bus (also for PDP-6, which was being done concurrently) architecture to reduce the cost of interfacing. The logic design of the PDP-5 was carried out by Edson De Castro, an applications engineer responsible for building the analog front end for a reactor control computer system for the Atomic Energy of Canada, Limited at Chalk River. The control computer was a PDP-4, but an elaborate analog monitoring system was needed for the front end. Hence, after visiting Chalk River in the winter of 1962 we decided on the PDP-5 as the simplest and most reliable method to carry out the task.

The decision to switch from a 12-bit to an 18-bit design was taken very lightly in December, 1962. In retrospect, it may have been a poor decision. We probably should have evolved the PDP-1 to a lower cost design. Although it is difficult to remember the events surrounding the decision, I now (perhaps poorly) recollect a number of reasons, ~~These were~~ all reactive to PDP-1 experience. ~~These included:~~.

1. Simple machines with few instructions for a given number of data-types perform nearly as well those with lots of instructions. Much of my own programming experience was using TX-0. I felt a significantly simpler 18-bit machine could be built than the PDP-1. This turned out to be true as the PDP-4 processor was implemented in less than 1/2 the space of the PDP-1, although it's unclear whether the size reduction was due to a simpler architecture, a much better logical design implementation or increased logic packing density.

2. I could build a better machine the second time around (actually 3rd or 4th if one counted MIT's Whirlwind and TX-0 computers). This belief was coupled with some new ideas about architecture.

   a. I had conceived the idea of auto-incrementing memory registers. This allowed vectors to be accessed easily instead of using index registers (which were more expensive to implement), and performed about as well.

   b. The PDP-1 used one's complement arithmetic and this was especially poor for doing fast multiple precision and floating point arithmetic which our users needed.

   c. Two control instructions were changed so that they did not effect the AC and interfere with the arithmetic instructions. The jump to subroutine stored the return in the program (a convention that would not be used today) and the index and skip instruction only operated on memory.

d. There were several features of the PDP-1 that cost logic without adding much to performance (e.g., program flags, and sense switches which were equivalent to IBM's sense lights). The wired-in program (read-in mode) to control the automatic reading of paper tape was also eliminated. Also, in general, the goal was to build a modular design such that the optional equipment cost was associated with the option instead of being wired-in to all machines.

e. The PDP-1 used a 4 Kword memory before memory bank switching; but much of the useful system software required 8 Kwords of memory. I felt (erroneously) that making a machine that would directly access 8 Kwords of memory would be ideal. (Little did I understand that programs expand to fill nearly any physical memory size--another possible corollary of Parkinson's Law.) However it did turn out that most PDP-4's stayed within the 8 Kword constraint although the machine would operate with up to 32 Kword of memory.

3. I wanted to use the Teletype Model 28 instead of a modified IBM Model B typewriter on PDP-1 to get better system MTBF and easy servicing.

4. The logical design of the PDP-1 though quite straightforward was not very tight. It had redundant terms and the instructions weren't encoded to simplify the implementation. I felt that starting over, with a new ISP, was the only way to get a significantly smaller machine.

5. We had both peripherals and memories for the PDP-1 and these could be used immediately to assist the implementation of a computer. This pushed us to an 18-bit (versus 12-bit) design.

Software

Insert Section B from page 10

Section A

~~The previous section focussed on how the PDP-1 was used as a starting point.~~ ~~Furthermore~~ At the time of the PDP-4 design, there ~~still~~ was not any understanding or concern on ~~our~~ DEC's part about the high cost of producing software. Since the PDP-1 was generally used in dedicated applications, there was no concern for system software, hence no notion that a new machine should capitalize on previous software. DEC had invested very little in the PDP-1 software; the users generated their own programs. M.I.T. had contributed a good macro assembler, linking loader, and interactive debugging program - DDT. Bolt, Beranek and Newman had contributed various subprograms. Because of this, there wasn't a knowledge of software cost nor was there an apparent need to know. It was easy to believe that a small part of the savings in the cost of a simpler machine could easily pay to write new software.

Now, I regard the PDP-1 compatibility of the "4" to be a constraint in a new design. Furthermore, any new machine should be on an improving (with time) cost/performance line (as discussed in the final section). At that time there were about twenty PDP-1's installed at the time, and fifty were eventually built. A compatible machine could probably have been built that would have interpret~~t~~ed most of the PDP-1 programs, offer improved cost/performance ratio features, and still not have been very much larger than the original PDP-4.

Intended Market

We had a fairly clear idea of whom the machine would serve. The PDP-4 was to be used predominately for process control, with some use in the laboratory for activities such as pulse height anlysis and data gathering, in much the same way as PDP-1, but it would be smaller and easier to use. We were talking with Foxboro about applications at Nabisco for baking control (with manual override) and to Corning about the control of a glass tube manufacturing process. We installed machines at both companies. Since Foxboro had been using a 24-bit RCA control computer, it is conceivable that they also influenced us to decide on 18-bits instead of the 12-bit DC 12 we had proposed in order to speed control calculations and be more in line with conventions and intuition.

The PDP-1 Instruction Set Processor

[GB: need paragraph to support RT-level diagram]

~~THE I/O SECTION~~ → I/O Structure

The I/O interconnection was especially important so that users could connect equipment easily. The I/O section was called the Real Time Control and included the capability to interface with PDP-1 peripherals. The structure of the PDP-4 with its options is shown in Fig. 11. We did not consider the I/O bus structure (party line or daisy chain) that was developed for the PDP-5 one year later but focused on improving the more conventional radial scheme that was used on computers at that time (e.g., PDP-1). This permitted peripheral compatibility. In effect, the I/O bus was a few inches long, and consisted of an area where cable drivers and input gates could be patched (via taper pins) to special cables and equipment.

Process control was really process monitoring since computers were supposedly not reliable enough for control. As a result, we had slightly better input capabilities than output cable drive. The I/O was wired with taper pins to create a very flexible patch panel for ease in wiring controllers together within the same or adjacent racks. In fact, a complete PDP-4 with card reader, magnetic tape, display, and other options required only three bays, and many systems could fit within the two standard bays (see Fig. 12). This was less than half the size of a comparable PDP-1 system.

Another consideration in the I/O structure was the ability to count events. This was used in scientific applications and in control to measure flow (e.g., via flow meters) or to count discrete items. Options such as a 16-channel clock used memory for event counting. This feature was implemented by having the device access a memory cell and then rewrite its contents plus one, thus changing the contents of memory as it is rewritten. Counting could occur at event rates up to 125 Khz.

This basic counting capability enabled the design of a relatively low cost, high performance direct memory access feature called the three cycle data break. This feature was first used in the Type 57A Magnetic Tape Controller for PDP-4. A direct memory access transmission via three cycle data break occurred as follows:

1. the word count word (in memory) to control the block length is incremented;

2. the pointer word (in memory) to the memory location being transferred is incremented; and finally

3. the data word pointed to is transferred between memory and the i/o device.

Move to page 10

Section C

MAINTENANCE ← lower case

Maintenance was especially important although parity memory was not used in the design. In addition to having a very simple architecture and a straightforward implementation, designers exercised a great deal of care in the logical design structure and the maintenance documentation. The machine instruction-set description only occupied one page, and the logical design flowchart (state diagram) was contained on one D-size drawing. The logical design drawings for the processor occupied seven D-size sheets.

As a further refinement to understanding the machine operation, each signal name source was named with a prefix to match one of the seven drawing names. This convention was carried forward through many DEC machines. The operator's console included several functions to assist maintenance. The console (see Fig. 13) had a speed control to enable the console functions (read, read next, write, write next, start, continue) to be repeated at a variable clock rate. This simplified testing by permitting a scope to be used easily. The most useful console functions were the reading and writing of information from the next location of memory. Thus, one could perform simple checks on memory by observing the contents of memory locations indicated by the console lights.

PHYSICAL STRUCTURE/IMPLEMENTATION — lower case

The main basis for building a new machine is a new technology. In this case, DEC had been extending its main business, the sale of logic modules, by extending the lower cost, slower speed, 500 Khz version of the 5 Mhz modules that were used in the PDP-1. The 500 Khz systems were significantly less expensive to build since they used germanium alloy transistors versus the MADT transistors for a 50% cost savings under the 5 Mhz modules. Some of the modules operated at a 1 Mhz clock. By being significantly slower, they were easier to use and more reliable because of the lower data rate and wider clock pulse. At the lower speed, two techniques further reduced the cost: diode-transistor logic versus one transistor per gate in the original circuits (with the wider clock); and capacitor-diode gates for the AND gates to registers. These techniques permitted: increased densities; lower speed and greater noise immunity; greater reliability through fewer active components; and much lower cost. Furthermore, we could now design the several dedicated modules including the accumulator and Teletype interface. Since the system modules only had 22 pins (18 pins for signals), the increased densities could not be directly applied to logical functions. We added a 10-pin connector on the back of each module for the register transfer gating signals so that bit-slice architecture could be used. In this way, one bit of the accumulator register and all the input gates were packaged on a single module as shown in Fig. 14.

The multi-stable state device was an especially interesting logic circuit devised (and patented) to simplify the control section of PDP-4. This circuit was a generalization of a flip-flop to n-stable states using n, n-1 input NAND gates wired in a cross-coupled way. I received a patent for the circuit and it was subsequently used in other computers and in the module line.

Maintenance

Insert Section C

SOFTWARE

Section B

Another important consideration in the design was FORTRAN and scientific computation. This in turn influenced us to quickly design of the Extended Arithmetic Element. Our first users were adamant that they would write process control applications in a higher level language. We briefly considered ALGOL 60, but quickly decided to provide a FORTRAN II for the PDP-4. It turned out that a few users did use FORTRAN, but most early users stayed with assembly language programming.

EVALUATION

PDP-1 Compatibility ← insert A from page 5

~~THE MARKET~~ ← lower case

PDP-4 had limited success, although it did meet the corporate profit standard. It didn't sell as well as we expected, or obey the characteristics of having a completely elastic demand--relative to demand for the PDP-1. This simplified demand model was to accommodate improvements we expected from technological advances. This model is discussed in the final section. Thus, in order to meet the elastic demand, a machine released two years later should have been priced less, and featured the same performance as the PDP-1. (Alternatively it could be priced much lower to account for the relatively poor performance.) In summary, the PDP-4 was neither agressive enough in performance nor price. (The same argument is made about the PDP-8/S.) Like most other first implementations of a series (i.e., PDP-1, PDP-5, PDP-6, LINC/8, PDP-14, PDP-11/20) the PDP-4 performed the poorest, financially, of the 18-bit series. The PDP-7, 9, 9/L and 15 were necessary follow ons that utilized the software and hardware option base developed for PDP-4.

PDP-7 (Ron Wilson)

The original concept of the PDP-7 (or what finally was named the PDP-7) started with the design of the PDP-1/D. The PDP-1/D was a modified design of the PDP-1 implementing a memory bus concept (stemming from PDP-6) and some logic for time sharing ~~to facilitate timeshared systems.~~ The initial plans were to simply repackage the PDP-1 using some higher density ~~utilizing some newer designed~~ systems modules ~~of a higher density~~ and to reduce the processor cycle time: ~~in order to speed up operations. This provided~~ a goal of much better performance at a lower price. ~~In~~ Of all 18-bit computers, the ~~fact~~, PDP-7 had the best gain in performance/price over its predecessors (the PDP-1 and PDP-4). ~~of all 18-bit computers.~~

GOALS AND CONSTRAINTS

Because the PDP-4 had ~~of~~ a more extensive software and peripheral hardware option base, the PDP-7 was switched to be a follow-on to it rather than the PDP-1 ~~the PDP-4~~. Thus, primary design goals were to maintain program compatibility with the PDP-4 and to provide an increased performance at a lower price. Therefore the I/O interface scheme was constrained by the timing and structure of the past computers. ¶ My personal goal was to sell 120 systems, a number greater than the combined total of the other DEC computer systems sold at that time. These goals, although sounding quite broad, were rather restrictive, especially the requirement for program and peripheral compatibility.

The performance goal was to decrease the cycle time from 8 microseconds to 1.75 microseconds. This was a rather ambitious goal and required designing a special set of modules called the Blue Line (the high speed series, within the new Flip Chip modules). They were based on our 10 megahertz systems modules. The whole module series would consist of various speed ranges and would be as general purpose as possible. The central processor and the memory used the high speed (10 Mhz clock) Blue Line series. The I/O section of the system, although originally implemented using the Blue and White series, was later redesigned to use a majority of R-series modules (2 Mhz clock).

Time was considered a very important factor in the design of the PDP-7. The entire logic implementation was undertaken by myself and one assistant, Jack Williams. Later we were joined by a Field Serviceman, Don Zereski, who literally hand-built the first production system to be delivered. The project started on April 1, 1964, and the first production system was delivered December 22, 1964 (see Fig. 15).

Program compatibility between the PDP-4 and the PDP-7 (although maintained generally) was slightly modified in the I/O section. The KSR28 as used on the PDP-4 was a 5-level Baudot code. A shift to the ASCII 8-level code had already started in the industry, and we wanted to use the KSR33 (Keyboard Send-Receive -- no paper tape) as the system input/output device. This necessitated a test at the front end of all programs to see if the system was a PDP-4 or a PDP-7. Other than this, an upward compatibility was maintained even though several additional instructions were added along with a trap feature and option multi-level interrupt to allow multi-user environments. Also, note the addition of a program read-in mode to the console (see Fig. 16).

~~The major development problems~~ These were:

Section D

1. A complete line of new modules, the Flip Chip series, had to be developed to implement the design of the computer, although the circuits had been tested in the PDP-6.

2. New connector blocks had to be obtained to hold the modules. (This design was concurrent with PDP-8.)

3. New wire-wrap techniques had to be devised to ease labor requirements and speed up wiring systems. A program was ultimately developed for the PDP-4 for wire routing and controlling the Gardner-Denver automatic wiring machinery.

4. System layouts had to be developed to facilitate wire-wrapping.

5. Mechanical packaging and cooling had to be altered to hold systems wire-wrapped panels. The air plenum scheme used in PDP-1, 4 and 5 was completely blocked by the new connector blocks.

6. Memory control and stack were carried out by a memory design engineer, Derrick Chin, who coordinated his design with the processor logic design.

The ambitious design and time goals on the PDP-7 project were met by long hours and much cooperation of a large number of people in the many different departments. The module design engineers designed the lay-out, supervised the manufacturing and specified the testing of hundreds of modules. The mechanical engineering department designed and ordered connector blocks which were originally purchased from an outside source.

Insert Section E from page 5

Physical Structure / Implementation

¶A semi-automatic wire-wrapping technique was developed to allow a much higher speed production of wire-wrapped back planes. Also a Gardner Denver automatic wire-wrap machine was ordered and the programs to control it were developed. Mechanical block holders were designed to mount the connector blocks in the cabinets.

A system diagram of the PDP-7 processor is given in Fig. 17 showing the options and the general interconnection scheme. It was fundamentally the same structure as its predecessors, as it was designed for use with many of the earlier peripheral controllers. ¶ Physically, the PDP-7 was larger than the PDP-4, its predecessor, because the console was mounted on the side plane instead of the end. This mounting of the console facilitated maintenance. Notice (see Fig. 18) that the number of logic panels for the processor was the same as in the PDP-4, even though the printed circuit board area was slightly greater using the new modules (PDP-4 was 6 x 25 x 5-1/2 x 4 = 3300 sq.in., and the PDP-7 was 6 x 2 x 32 x 2-1/4" x 3-7/8" = 3348 sq.in.). Although it does not show in the diagrams or photos, a significant volume of the machine contained cable connectors to various subassemblies. The PDP-7 improved the cabling by having all the connectors in the backplane so that all wiring could be done as a single operation. The third bay of the PDP-7 housed the console equipment (that had been on an extra table in the PDP-1 and -4).

Section E

PDP-7 Instruction-Set Processor

Figure 19 shows the structure of the processors with its registers and the interface to I/O equipment and to core memory. Note the structure and style of design was fundamentally that used in the earlier designs (suitably modified for the higher speed technology). In fact, using the notion of chapter 0, PDP-1 and -4 have identical architectures, very similar implementations, and radically different realizations. In order to use the slower (PDP-4 compatible) I/O equipment, special pulses were used when a slow cycle of 8 microseconds was requested. The I/O section and new options were designed to operate at the 1.75 microsecond cycle rate.

EVALUATION

The Market

The PDP-7 was a successful project; it did sell over 120 systems. Design costs, excluding module and labor costs, were less than $100,000. Technically, it was a very sound system which developed an excellent reputation for reliability. Quite a few PDP-7s are still in operation.

Major development problems

insert section D on p. 3&4

PDP-9 BEGINNINGS

As the design phase of the PDP-7 neared an end and production models were being delivered, two developments occurred that made us consider an improved production model. The R-series Flip Chip modules were lower cost, lower speed and more complete than the B-series forming the processor. After analyzing the configurations being ordered, we were able to redesign the I/O panel. The new I/O panel was designed using as many R-series modules as possible. The controls for several of the most popular peripherals were prewired into it to reduce the amount of special wiring required to produce a system. We tagged this improvement as the PDP-7A.

With the PDP-7A now completed, our thoughts turned to the next generation system. Many more tools were now in place to allow some significant improvements in system design. The Gardner Denver automatic wire-wrapping system had been installed and was ready for use. We started on our next system which we called the 7X project (it later became the PDP-9). The design criteria called for the following:

1. It would be completely automatic wire-wrappable.

2. A system with 8 Kwords would sell for approximately $35,000.

In order to meet these goals, a new cabinet design was started so that we could mount the wire-wrap frames. This cabinet also required better air flow characteristics because of the high density of the logic then available. Past cabinets pulled air from the floor which was not a desirable feature in many installations, so a horizontal air flow system was implemented.

The system costs were becoming a major factor. To meet our price goals, the computer was divided both logically and physically into three divisions: 1 - Memory, 2 - Central Processor, 3 - Input/Output Logic. This was done to allow us to control and calculate costs more closely.

The cabinet design was completed, and with the logic design moving ahead smoothly, the project was taken over by Larry Seligman who had designed the Extended Arithmetic Element for the PDP-7.

PDP-9 and 9/L (Don Vonada)

The PDP-9 (see Fig. 20) was introduced in August 1966 and a less expensive version, the PDP-9/L was introduced in November, 1968. Figure 21 shows how the logic was mounted in three self-cooled frame sections, each capable of holding eight rows of forty modules. Each section had self-contained final power regulations (occupying four module spaces). A system block diagram of the PDP-9 (Fig. 22) shows how the PDPs have evolved to the I/O and memory bus structured computer. This scheme is in contrast to the radial structure of the earlier 18-bit computers and provided greater modularity.

Section G

Technology at the time ~~of the PDP-9~~ was still discrete PNP transistor (-3 volt signals), capacitor-diode gate, packaged, on single, double and quad height standard double-sided Flip-Chip modules.[1] Footnote 1 Although some integrated circuits were available, there were no standards set yet and no cost or speed advantage from using IC's. These modules, whose size was approximately 2-1/2 in. (5 in. or 10 in.) by 5-1/2 in., were plugged into an assembly of 144-pin connector blocks using 24 gauge wire-wrap pins as the interconnect media. The basic hardware was identical to that used for the PDP-7.

The central factor in the technology was the speed of core memory. A new memory design using 30 mil inside diameter cores and a driving sense structure known as 2-1/2 D (in contrast to the standard coincident current selection; see PDP-8 Chapter) allowed the memory to cycle in 1 microsecond. The memory was oriented to an 8 Kword organization which offered a cost advantage. It must be noted that discrete components logic costs were high compared to memory, but memory limited system performance. The PDP-7 had a 1.75 microsecond memory cycle; the system performance of the PDP-9 was improved by a factor of 1.75 over that of the PDP-7.

The PDP-9 Instruction-Set Processor

[CM: need an RT-level block diagram]

~~Other improvements which reduced cost and improved maintainability and reliability were also implemented.~~ The major cost improvement was the implementation of an I/O Bus, a concept derived from the PDP-5 and 6. The bus was daisy-chained from device to device using twisted pair cables. Compared with the PDP-7 which was custom wired for each option, this technique provided uniformity in I/O backplane wiring; it allowed independent development, manufacturing and test of I/O options and simplified field installation of options. Also, it allowed costs to be associated with each option rather than being initially higher as in the radial scheme where all options had to be planned for in the central processor. The bus structure created the illusion that systems of unlimited size could be built. Section H

I/O Structure

The I/O control extended the features of earlier models by implementing an eight level nested automatic priority interrupt facility and a data channel transfer facility. The Automatic Priority Interrupt (API) had four levels of hardware interrupt capability available at the I/O Bus and four levels of software priority. The data channel transfer facility was the same as a direct memory channel except in the location of word count and current address registers. The I/O structure was designed to allow these control registers to be implemented in core memory, further attesting to the fact that core was cheaper than logic. (This scheme was first used in PDP-4 with the Type 57A magnetic tape control.)

The Direct Memory Access channel (DMA) was the most disappointing part of the I/O Bus concept. The speed requirement dictated the use of an extra set of data and address lines which were carried between the DMA device and the memory bus multiplexer via an extra set of cables. Also, a second port to memory was required. The idea of a "clean" bus cabling scheme for high speed transfer devices was not possible because of the extra radial lines required. (Alternatively, we might have slowed the machine down to handle the transfers.)

Physical Structure / Implementation

Insert Section G (from page 1)
Insert Section H (from page 2)

Section F

The PDP-9 was the first microprogram controlled processor implemented at DEC. Although in itself a technological advancement, it was also an interesting precursor of things to come. The structure of the processor is shown in Fig. 23. Note its simplicity compared to the earlier designs. It used a more general data path through an adder compared to the ad hoc register structure of the earlier machines. A 64-word, 36-bit, 212-nanosecond cycle read-only, rope, magnetic memory was used as the microprogrammed control store for the processor. The design allowed for easy bench modification in the event the microcode required changing. The original concept of a unary encoded control soon was lost in the complexities of the design. Encoding the bits was the resulting compromise position which eliminated the possiblity of providing special purpose machines (we had hoped for) by a simple ROM change. The size of the control memory further constrained the extendability and use of the microprogram control. It would have been nice to be able to build in floating point arithmetic. There were not enough words.

A further testimony to the power of control store was the fact that the extended arithmetic element, a hardware 36-bit multiply/divide option, was implemented with only six single height flip chip modules. By comparison, the processor occupied about 320 module slots or a total printed circuit board area of 3,100 sq.in. This area also included the optional arithmetic element and much of the I/O control; thus it was about half the size of the earlier machines. (We also found an error in the PDP-1 signed integer divide algorithm.) A discrete carry adder which developed the carry over 18 bits in under 30 nsec. was necessary in order to meet the performance goals.

EVALUATION

Except for the 300 wire field change on the first ten processor backplanes, the PDP-9 enjoyed a good reputation for performance and up time. It was later accompanied by the cost reduced version PDP-9/L. The cost reduction took the form of a new (and somewhat cumbersome) power supply design, a 4 Kword memory design which used the Teletype 33 ASR (Automatic Send-Receive - with paper tape) instead of separate paper tape reader and punch. The memory planes were borrowed from the PDP-8 line and adapted to provide half the memory in half the space. The life of the 9/L was comparatively short, for it was soon overtaken by the PDP-15.

Microprogramming

Insert Section F (page 3)

PDP-15 (Gerry Butler)

Unlike its predecessors, the PDP-15 was designed to provide a range of systems with both hardware and software. While the early 18-bit machines evolved to include several configurations, the notion of a planned range was explicit from the start. As it turned out, the PDP-15 evolved over a considerably wider range than was anticipated. Table 2 shows the various systems and options which served as bases for new systems, but only those models through the 15/40 were planned from the start.

As in the past, the goal was to provide a better cost/performance than the PDP-9. The PDP-7 to PDP-9 transition did not provide a big cost improvement. The PDP-15 semiconductor technology provided the impetus for a new implementation. The 7400 and 74H00 series integrated circuits of transistor logic (TTL) permitted clock speeds of 10 to 20 Mhz, lower costs and higher packing densities (for lower packaging costs) than the discrete circuits used in PDP-9. The basic PDP-15/10 turned out to be the smallest (see Fig. 24) of the 18-bit series, while providing a number of options and additional features including an additional instruction set with an index and limit register. Also, the memory size was extended to 131 Kwords of memory. A separate control unit, called the I/O processor, handled the bookkeeping for the I/O channels and I/O bus. Figure 25 shows a fundamentally smaller system (including the core memory). Note the two processors occupy only a third of the cabinet space, yet are faster and have more capability than the PDP-9. Also note that the packaging reverted to the simplicity of the earlier PDP-1 using no swing-out gates.

Table 2: The PDP-15 Family of 18-bit Computer Systems

	Hardware	Software
PDP-15/10		
(Basic Paper Tape System)	CPU	Assembler
	4K Memory	Editor
	Teletype	Debugger
		Utilities
PDP-15/20		
(Keyboard Monitor using	CPU	Keyboard Monitor
DECTape File System)	8K Memory	Fortran IV
	EAE	Focal
	Paper Tape	PIP
	DECTape	Utilities
	Teletype	
PDP-15/30		
(Background/Foreground)	CPU	B/F Monitor
	16K Memory	Fortran IV
	EAE	Focal
	API	PIP
	Memory Protect	Utilities
	Clock	
	Paper Tape	
	DECTape	

	2 Teletypes	
PDP-15/35	(15/30 with Disks)	
PDP-15/40		
(Disk Based B/F)	CPU	Disk B/F Monitor
	24K Memory	Fortran IV
	EAE	Focal
	API	PIP
	Memory Protect	Utilities
	Clock	
	Paper Tape	
	DECTape	
	524K Fixed Hd. Disk	
	2 Teletypes	
PDP-15/50	(16K memory based)	
PDP-15/76	15/40 + PDP-11	11-based File and I/O Device Management

Unlike its predecessors, the change from discrete transistors to integrated circuits did cause perterbations in the 18-bit evolution since the ICs used ground and +5 volt logic signals, compared to the ground and -3 volt signals of its predecessors. This change caused a great break with the past in terms of using earlier peripherals. A block diagram to the final, evolved state of the PDP-15/76 is shown in Fig. 26. The initial design did not have floating point arithmetic, but the project was started and 50% completed prior to the first shipment. Table 3 shows the effect of a built-in data-type. The PDP-11 (and the Unibus) did not exist at PDP-15 introduction time. Hence, the initial structure had only a connection to memory, two processors, and an I/O bus that was somewhat compatible with the PDP-9 (so that equipment could be designed for either PDP-9 or PDP-15). There were well-defined initial goals which are described below. Finally, there is a discussion of the evolution beyond the initial design.

Table 3: FP15 Floating Point Computation Times

	Was	**With**	
Program Type	**Software Time**	**FP15**	**Speed Up***
Matrix Inversion	12 Sec.	5 Sec.	2.4
Fourier Transform	16.9 Sec.	2.9 Sec.	5.8
Least Squares Fit	5.1 Sec.	.7 Sec.	7.3
Test of all F.P. Functions	11.4 Sec.	1.4 Sec.	8.1
A Physics Application Program	37 Sec.	3 Sec.	12.3

~~OBJECTIVES AND RESULTS~~

GOALS

~~Most objectives were achieved or exceeded. The following is a summary of various explicit goals, together with the decisions and results of the decisions.~~

~~850 Nanosecond Cycle Time~~

850 Nanosecond Cycle Time

An 800 nsec. cycle time exceeded our (850 ns) objective, but it was difficult for several reasons:

The CPU, memory, and I/O were made asynchronous to reduce I/O latency reduction. The synchronizing logic was complicated and resulted in significant circuit delays.

A DC (round trip) interlocked memory bus was used in order to handle speed-independent memories. This caused communications delays.

To minimize cabling, we shared one set of lines for address and data to memory. This caused additional communication delays.

Insert Section J from page 8

Manufacturing Cost

We exceeded our manufacturing cost goal in the first fifty units. In the process, we learned that the majority of costs were in the mechanical packaging. Sights were set on reducing cabling, using a single power harness which could be built and tested on a jig, and using standard H911 mounting panels for logic. The cabling was reduced to:

1 - console cable

1 - teletype cable

1 - I/O bus cable assembly

2 - memory bus cables

In trying to limit console cabling, a time division multiplex communication scheme was designed to get enough signals to the lights. In this fashion, a number of signals were transmitted on the same wires on a timeshared basis. The lamp filament was used as the storage element at the console. This was the machine's only patent. Ultimately the logic was too complex: when the console failed, it was harder to fix than the processor.

Improved Priority Interrupt Latency

We were not able to reduce interrupt latency to the two microsecond goal. With parity, memory protect and relocate options implemented, and with adder and synchronizing delays, a four microsecond time was achieved, which was acceptable in the system.

One Cabinet Package

The CPU, I/O processor, console and 32 Kword memory were packaged in one cabinet, but it was close. There are virtually no spare module slots in the CPU.

I/O Bus Length

The I/O Bus was extended to 75 feet by switching to a terminated bus. The PDP-9 used an unterminated transmission line with diode clamps. The PDP-15 has cleaner signals and no reflections. A new set of bus driver/receiver modules were designed which were faster and offered less bus loading. The penalty paid was higher power consumption and greater power supply cost than in PDP-9.

~~Minimum Multi-User Protection Scheme~~

Section H

The least expensive multi-user protection scheme was provided in the form of a relocation register and a boundary register. Because it was marketed as an option, it degraded the machine more than necessary. However, the minimum machine cost was kept within the target.

Better Maintainability

A means to look at more than 400 signal points was provided in the logic. This, coupled with the single step feature allowed troubleshooting of the machine from the console without an oscilliscope. As it turned out, this feature was used infrequently because of the required training.

Section J

PDP-9 Compatibility

PDP-9 instruction compatibility was achieved, but with three minor exceptions. These caused no customer or DEC software problems, to our knowledge.

Other Cost & Manufacturing goals

~~Minimum Semiconductor Types~~

Since the PDP-15 was one of the first DEC computers to use ICs extensively, it was important to minimize the number of logic types. The PDP-15 has 21 semiconductor types. This includes ICs, transistors, and diodes. All are available from multiple suppliers.

~~No Twisted Pairs~~

Having no twisted pairs in the backplane was a tough goal to achieve because there was no inexpensive wiring capability at the time. We achieved this goal, but logic had to be moved before the machine could be produced and reliably.

~~Fixed Configuration Rules with Field Installation of Options~~

The PDP-15 has fixed configuration rules and very easy field installation of options. This required more space (partially full cabinets) and as a result the cost is slightly higher.

~~Improve Producibility~~

Margin testing of the PDP-15 was planned using a combination of varying logic timing and temperature. Special test equipment was provided for the production line to allow rapid heat cycling of CPUs and memories. In addition, a fast program loader system was designed around a PDP-8 with multiple DECtape units. Diagnostic programs could be loaded into the memory of a unit being tested by merely pressing a button. This saved considerable time in computer check-out.

Subassembly replacement was planned at the final assembly and test group. The concept was that if a processor, memory, power supply or whatever failed to work when it was integrated into a system, the whole subassembly would be replaced and sent back to its appropriate test line rather than doing the repair in the area. This process proved impractical, as the production line was never filled with enough material to allow this to take place.

Early First Customer Ship

The first PDP-15 was to be shipped eighteen months after commencement of the project. We ran into a number of difficulties during the project, including personnel turnover, which caused a two-month slip.

Budget

The project was finished at about 92% of budget.

New Peripheral Designs (except Direct Memory Access Controllers) Were to Work on Both PDP-9 and PDP-15

This goal was met, newer devices were used on the PDP-9 by changing the receiver/driver modules. The PDP-15 was the first 18-bit machine to use IC positive logic (versus older negative transistor series logic), bus conversion signals were required in order to be able to use the PDP-9 peripheral controllers.

~~MEMORY EVOLUTION~~

[~~the physical~~ Most of the physical packaging has been discussed above.]

~~While we have not commented on the memory packaging per se, it is worth noting the evolution that transpired between the PDP-1 and PDP-15.~~

Section K

¶ The PDP-15 memory is contrasted with the PDP-1 memory in the following table

	PDP-1	PDP-15	PDP-15 (ME15)
Year	1960	19xx	19yy

Section K cont'd

Stack size	4 Kwords	4 Kwords	24 Kwords
Cycle time (microseconds)	5	0.8	0.8
Density (words/cabinet)	12 Kwords	48 Kwords (32 Kwords with CPU included)	96 Kwords
Size electronics	1/3 cabinet	1/12 cabinet	1/24 cabinet
Configuration	3D stack	5 planes, 4 bits/plane; planar stack	
Core size	30 mil	18 mil	18 mil
Wires/Core	4	3	3

~~THE PROCESSOR AND ITS EVOLUTION~~ PDP-15 Instruction-Set Processor

Figure 27 shows the register transfer structure of the processor. It is based on elements and features used in earlier designs. It has a basic data path structure which permits the results from any of the eleven registers to be read into the arithmetic unit and back into the registers. On the other hand, in order to operate at high speeds, a number of separate registers such as the Step Counter, Program Counter, and Multiplier-Quotient registers must operate in parallel with the basic data path. In this way, significant overlap occurs, which is necessary in order to achieve the 800 nanosecond memory cycle time. The contrast with the PDP-4 is noteworthy: it had ~~time. Here, the reader should note the change by design philosophy from the PDP-4, which contained~~ only four registers in the basic machine. ~~With~~ Because the PDP-15 used integrated circuits, the goal of minimizing registers ~~is~~ was less important.

The first major extension to the PDP-15 was the addition of a floating point processor (Fig. 28) to enable it to perform well in the scientific/computation marketplace using FORTRAN and other algorithmic languages. With the addition of the FP15 floating point processor the time for a programmed floating point operations was reduced from 100-200 microseconds to 10-15 microseconds, giving about a factor of 10 increase in FORTRAN performance. The additional floating point unit required a number of additional instructions. The irony of this extension is that the PDP-11, and nearly all microcomputer ISP extensions exactly follow this evolution!

~~Insert page 12A~~

Insert Section H (from page 7) as a paragraph.
Physical Structure/Implementation
Section K from page 10

TWO CENTRAL PROCESSOR PDP-15 (Bob Gray)

The product line had sold a system that was a dual processor. From the sale came the MX15, a dual port memory, which eventually was transferred to the PDP-15 standard product line. The MX15 also expanded memory to the full 128 Kwords built into the PDP-15 addressing structure. The unit occupied a single rack and used the M series logic cards. Since there was space to add a third "port" within the rack unit, the MX15 turned out to be a three port device. At the time, the lab breadboard was, indeed, an impressive array of three cabinets containing 128 Kwords of memory and two processors.

The logic included what went unrecognized as a "synchronizer"* problem for some two months despite reviews by some senior engineers. Once the problem was recognized the design went to a quick completion. Since the multiport memory has three inputs to synchronize, it can become unstable.

UNICHANNEL 15/76 (PDP-15 + PDP-11) ~~- ALSO~~ AND PDP-15/XVM

*A classical logic design problem that has turned out to be "theoretically" unsolvable. When synchronizing (detecting) the presence of an event occurring at a random time with a fixed clock event, a small amount of energy is available to set a flip-flop. When a flip-flop is triggered with a small signal, it can go into an undecided state undecided for a relatively long (even undecided) of time.

The Unichannel began as a problem: find the most cost-effective way to attach the RK05 disk to the PDP-15. After a review of the problem, it became clear that the correct way to put the RK05 on the PDP-15 was to use an -11 processor, together with the RK05 controller on PDP-11. The key reason was not the cost of modifying the RK05 disk controller, rather, it was the cost of doing new diagnostics that would have to be executed in PDP-15 code. Another reason was the ease of adding other PDP-11 family peripherals. Our project was to run somewhat over six months and had a budget for the hardware of about $75K.

As the system design progressed, it became clear that the PDP-11 could run other PDP-11 family peripherals. Since most development and production resources were directed to the 16-bit, PDP-11 series, all factors favored using existing peripherals rather than inventorying a new set for PDP-15. The list of peripherals quickly included communications lines, plotter, printer, and card equipment. It was also conceived as a PDP-15. Figure 29 shows the options of the PDP-15/76 and Fig. 30 shows the physical cabinets together with powers, etc.

The project had a very small, but excellent staff. Al Helenius, did much of the logic design of the memory multiplexer device using M series logic modules. It was first operational in early November 1972. In general the hardware part of the program went very smoothly. The complexity and size of the software task was clearly underestimated. Rick Hully proposed an operating system structure that, for the era and initial application, was both elegant, advanced and yet straightforward. However, the successful system operation depended on more software. The reality was that we had evolved to a

*A classical logic design problem that has turned out to be "theoretically" unsolvable. When synchronizing (detecting) the presence of an event occurring at a random time with a fixed clock event, a small amount of energy is available to set a flip-flop. When a flip-flop is triggered with a small signal, it can go into an undecided state undecided for a relatively long (even undecided) of time.

true "multiprocessor" system. Today the industry has begun to use the term "back-end processor" or "file-processor" for what was accomplished on the PDP-15 in the early seventies. (Also, this structure was used by IBM in their coupled 7090/7044 system and in their 360 Attached Support Processor.)

It is interesting that the dual processor structure is again receiving attention as microcomputers make the CPU no longer a restricted resource, but one to be used generously throughout a system. The UNICHANNEL and 15/76 programs hold much valuable technical experience that has yet to be exploited or understood, even within DEC. One of the major reasons for the slow diffusion of this innovation has been the ongoing controversy of using "symmetrical multiprocessing" or "specialized task oriented multiprocessing". The current activity, in my opinion, vindicates the specialized approach, which the 18-bit computer line understood years ago.

~~Unichannel Results~~

The Market

From a market and user viewpoint, the 15/76 was successful. Users were able to offload I/O and file processing and at, the same time, obtain much lower priced peripherals. About 100 were delivered, and 300 were retrofit to existing PDP-15 installations.

Chapter 7

Structural Levels of the PDP-8
Bell, Newell, and Siewiorek

~~The PDP-8 will be described in a top-down manner to illustrate the hierarchical~~ nature of ~~computer systems design~~. Insert (A)

A map of the PDP-8 design hierarchy, based on the Structural Levels View of Chapter 1, is given in Fig. 1, starting from ~~PMS~~ processor-memory-switch (PMS) structure, to instruction set processor (ISP) and down through logic design to circuit electronics. These description levels are subdivided to provide more organizational details such as registers, data operators, and functional units at the register transfer level.

~~It should be apparent that~~ The relationship of the various description levels constitutes a tree structure, where the organizationally complex computer is the top node and each descending description level represents increasing detail (or smaller component size) until the final circuit element level is reached. For simplicity, only a few of the many possible paths through the structural description tree are illustrated. For example, the path showing mechanical parts is missing. The path shown proceeds from the PDP-8 computer to the processor and from there to the arithmetic unit or, more specifically, to the Accumulator register (AC) of the arithmetic unit. Next, the logic implementing the register transfer operations and functions for the jth bit of the ~~AC~~ accumulator is given, followed by the flip-flops and gates needed for this particular implementation.

Finally, on the last segment of the path, come the electronic circuits and components of which flip-flops and gates are constructed.

Abstract Representations

Figure 1 also lists some of the methods used to represent the physical computer abstractly at the different description levels. As mentioned previously, only a small part of the PDP-8 description tree is represented here. The many documents, which constitute the complete representation of even this small computer include logic diagrams, wiring lists, circuit schematics, printed-circuit board photo etching masks, production description diagrams, production parts lists, testing specifications, programs for testing and diagnosing faults, and manuals for modification, production, maintenance, and use. As the discussion continues down the abstract description tree, the reader will observe that the tree conveniently represents the constituent objects of each level and their interconnection at the next highest level.

Each level in the abstract description tree will be described in order.

Stet
~~Processor Memory Switch~~

The ~~PMS~~ Level

The PDP-8 computer in PMS notation is:

capital i

```
C('PDP-8; technology:transistors; 12 b/w;
  descendants:'PDP-8/S, 'PDP-8/I, 'PDP-8/L,
  '8/E, '8/F, '8/M, '8/A, 'CMOS8;
  antecedents: 'PDP-5;
  Mp(core; #0:7; 4096 w; tc:1.5 us/w);
  Pc(Mps(2 4 w);
    instruction length:1|2 w
    address/instruction:1;
    operations on data:( , +, Not, And, Minus
    (negate), Srr1(/2), Slr 1 (*2), +1)
    optional operations:(*,/,normalize);
    data-types:word,integer,Boolean vector;
    operations for data access:4);
  P(display; '338);
  P(c; 'LINC);
  S('I/O Bus; 1 Pc; 64 K);
  Ms(disk, 'DECtape, magnetic tape);
  T(paper tape, card, analog, cathode-ray tube))
```

Note: the abbreviations used will be explained in the discussion which follows.

As an example of PMS structure, ~~The PMS structure~~ of the LINC-8-338 consisting of three processors (designated by the letter P); Pc('LINC), Pc('PDP-8), and P.display('338) is shown in Fig. 2. ~~The PDP-8 is just a single processor.~~ The LINC processor described in Chapter 6 is a very capable processor with more instructions than PDP-8~~.~~ and ~~It~~ is available in the structure to interpret programs written for the LINC. Because of the rather limited ~~ISP~~ instruction set being interpreted, one would hardly expect to find all the components present in Fig. 2 in an actual configuration.

The switches (S) between the memory and the processor allow~~s~~ eight primary memories (designated Mp) to be connected. This switch, in PMS called S('Memory Bus; 8 Mp; 1 Pc; time-multiplexed; 1.5 us/w) is actually a bus with a transfer rate of 1.5 microseconds per word. ~~Thus,~~ the switch makes the eight memory modules logically equivalent to a single 32768 word~~s~~ memory module. There are two other connections (a "switch" and a "link") to the processor excluding the console. They are the S('I/O Bus) and L('Data Break; Direct Memory Access) for interconnection with peripheral devices. Associated with each device is a switch, and the I/O Bus links all the devices. A simplified PMS diagram (Fig. 3) shows the structure and the logical-physical transformation for the I/O Bus, Memory Bus, and Direct Memory Access link. Thus, the I/O Bus is:

S('I/O Bus duplex; time-multiplexed, 1 Pc; 64 K; Pc controlled, K requests; t:4.5 us/w)

The I/O Bus is nearly the same for the PDP-5, 8, 8/S, 8/I, and 8/L. Hence, any controller can be used on any of the above computers provided there is an appropriate logic level converter (PDP-5, 8, and 8/S use negative polarity logic, while PDP-8/I and 8/L use positive logic.) The I/O Bus is the link to the controllers for processor-controlled data transfers. Each word transferred is designated by a processor in-out transfer (iot) instruction. Due to the high cost of hardware circa 1965, the PDP-8 I/O Bus protocol was designed to minimize the amount of hardware to interface a peripheral device. As a result, only a minimal number of control signals were defined with the largest portion of I/O control performed by software.

A detailed structure of the processor and memory (Fig. 4) shows the I/O Bus and Data Break connections to the registers and control in the notation used in the initial PDP-8 reference manual. This diagram is essentially a functional block diagram. The corresponding logic for a controller is given in Fig. 3 in terms of logic design elements (ANDs and ORs). The operation of the I/O Bus starts when the processor sends a control signal and sets the IO.SELECT<0:5> lines to

six I/O selection

specify a particular controller. Each controller is hardwired to respond to its unique 6-bit code. The local control, K[k], select signal is then used to form three local commands when ANDed with the three iot command lines from the processor, ~~called~~ These command lines are called: IO.PULSE.1, IO.PULSE.2, and IO.PULSE.4. Twelve data bits are transmitted either to or from the processor, indirectly under the controller's control. This is accomplished by using the AND-OR gates in the controller for data input to the processor, and the AND gate for data input to the controller. A single skip input is used so that the processor can test a status bit in the controller. A controller communicates back to the processor via the interrupt reguest line. Any controller wanting attention simply ORs its request signal into the interrupt request signal. ~~Software polling then determines the specific interrupter.~~ Normally, the controller signal causing an interrupt is also connected to the skip input, and ~~a~~ skip instructions are used in the software polling that determines the specific interrupting device.

The Data Break input for Direct Memory Access provides a direct access path for a processor or a controller to memory via the processor. The number of access ports to memory can be expanded to eight by using the DM01 Data Multiplexer, a switch. The DM01 port is requested from a processor (e.g., LINC or 338) or a controller (e.g., magnetic tape). A processor or controller supplies a memory address, a read or write access request, and then accepts or supplies data for the accessed word. In the configuration (Fig. 1), Pc('LINC) and P('338) are connected to the multiplexor and make requests to memory for both their instructions and data in the same way as the PDP-8 processor. The global control of these processor programs is via the processor over the I/O Bus. The ~~Pc~~ processor issues start and stop commands, initializes their state, and examines their final state when a program in the other processor halts or requires assistance.

When a controller is connected to the Data Break or to the DM01 Data Multiplexer, it only accesses memory for data. The most complex function these controllers carry out is the transfer of a complete block of data between the memory and a high speed transducer or a secondary memory for example, DECtape or disk. A special mode, the three cycle data break (described in Chapter 5), allows a controller to request the next word from a block in memory. ~~In this mode the next word is taken from the block, and a counter (in memory) is reduced each time the controller makes a request. With this scheme, a word transfer takes three memory cycles: one to add one to the block count, one to add one to the address pointer, and one to transmit the word.~~

~~An interesting aspect of DEC~~

The DECtape was derived from M.I.T.'s Lincoln Laboratory LINCtape unit, as indicated in Chapter 6. Data ~~are~~ were explicitly addressed by blocks (variable but by convention 128 w~~)~~ (words). Thus, information in a block ~~can~~ could be replaced or rewritten at random. This operation ~~is~~ was unlike the early and standard IBM format magnetic tape in which data ~~can~~ could be appended only to the end of a file.

Programming Level (ISP)

The ISP of the PDP-8 processor is probably the simplest for a general purpose stored program computer. It operates on 12-bit words, 12-bit integers, and 12-bit Boolean vectors. It has only a few data operators, namely, =, +, minus (negative of), Not, And, slr 1(rotate bits left), srr 1(*2 rotate bits right), (optional) *, /, and normalize. However, there are microcoded instructions, which allow compound instructions to be formed in a single instruction.

The ISP is presented in Appendix 1 . The 2^{12}-word memory (declared M[0:4095]<0.11>) is divided into 32 fixed-length pages of 128 words each (not shown in the ISP description). Address calculation is based on references to the first page, Page.Zero, or to the current page of the program counter ~~for Program Counter (PC)~~ ~~the~~ (PC\Program.Counter). The effective address calculation procedure, called eadd, provides for both direct and indirect reference to either the current page or the first page. This scheme allows a 7-bit address to specify a local page address.

STET but in parentheses

A 2^{15}-word memory is available on the PDP-8, but addressing greater than 2^{12} words is comparatively inefficient. In the extended range, two 3-bit registers, the Program Field and Data Field Registers select which of the eight 2^{12}-word blocks are being actively addressed as program and data. These are not given in the ISP description.

There is an array of eight 12-bit registers, called the Auto.Index registers, which resides in Page.Zero. This array (Auto.Index[0:7]<0:11>:=M[#10:#17]<0:11> possesses the useful property that whenever an indirect reference is made to it, a 1 is first added to its contents. (That is, ← delete this space →

there is a side effect to referencing.) Thus, address integers in the register can select the next member of a vector or string for accessing.

The processor state is minimal, consisting of a 12-bit accumulator (AC\Accumulator<0:11>), an accumulator extension bit called the Link (L\Link), the 12-bit Program Counter, the RUN flip-flop, and the INTERRUPT.ENABLE bit. The external processor state is composed of console switches and an interrupt request.

The instruction format can also be presented as a decoding diagram or tree (Fig. 5). Here, each block represents an encoding of bits in the instruction word. A decoding diagram allows one more descriptive dimension than the conventional, linear ISP description, revealing the assignment of bits to the instruction. Figure 5 still requires ISP descriptions for the memory, the processor state, the effective-address calculation, the instruction interpreter, and finally execution for each instruction. Diagrams such as Fig. 5 are useful in the ISP design to determine which instruction op codes are to be assigned to names and operations, and instructions which are free to be assigned (or encoded).

There are eight basic instructions encoded by 3 opcode bits of the instruction, that is, op<0:2> := i<0:2>. Each of the first six instructions, where the opcode is greater or equal to zero and less than or equal to 5 (where OP Geq 0 and OP Leq 5), have the four addressing modes (direct Page.Zero, direct Current.Page, indirect Page.Zero, indirect Current.Page), thus yielding essentially 24 instructions. The first six instructions are:

data transmission:	deposit and clear-accumulator/dca (Note that the add instruction, tad, is used for both data transmission and arithmetic.)
binary arithmetic:	two's complement add to the accumulator/tad
binary Boolean:	and to the accumulator/and
program control:	jump/set program counter/jmp jump to subroutine/jms index memory and skip if results are zero/isz

The subroutine-calling instruction, jms, provides a method for transferring a link to the beginning (or head) of the subroutine. In this way arguments can be accessed indirectly, and a return is executed by a jump indirect instruction to the location storing the returned address. This straightforward subroutine-call mechanism, although inexpensive to implement, requires reentrant and recursive subroutine calls to be interpreted by software rather than by hardware. A stack for subroutine linkage, as in the PDP-11, would allow the use of read-only memory program segments consisting of pure code. This scheme was adopted in the CMOS-8.

The input-output instruction, opcode 6 iot (op Eqv #6), uses the remaining nine bits of the instruction to specify instructions to input/output devices. The six IO.SELECT bits select 1 of 64 devices. Three conditional pulse commands to the selected device, IO.PULSE.1, IO.PULSE.2, and IO.PULSE.4, are controlled by the iot, io.control<0:2> operation code bits. The instructions to a typical I/O device are:

<u>testing a Boolean Condition of an IO Device</u>

```
    If IO.PULSE.1 => (If IO.SKIP.Flag[IO.SELECT] => PC = PC + 1)
```

<u>output data to a device from Accumulator</u>

```
    If IO.PULSE.4 => (OUTPUT.REGISTER[IO.SELECT] = AC)
```

<u>input data from a device to Accumulator</u>

```
    If IO.PULSE.2 => (AC = INPUT.REGISTER[IO.SELECT])
```

There are three microcoded instruction groups selected by (op<0:2> Eqv #7), called the operate instructions. The instruction decoding diagram (Fig. 5) and the ISP description show the microinstructions which can be combined in a single instruction. These instructions are: operate group 1 ((op<0:2> Eqv #7) And Not ib) for operating on the processor state; operate group 2 ((op<0:2> Eqv #7) And ib<3> And i<11>) for testing the processor state; and the extended arithmetic element group (op<0:2> Eqv #7 And i<3> And i<11>) for multiply, divide, etc. Within each instruction the remaining bits, <4:10> or <4:11>, are extended instruction (or opcode) bits; that is, the bits are microcoded to select additional instructions. In this way, an instruction is actually programmed (or microcoded, as it was originally named before "microprogramming" was used extensively). For example, the instruction, set link to 1, is formed by coding the two microinstructions, clear link, followed by complement link.

```
    If ((op <0:2> Eqv #7) And (group Eqv 0)) => (
                   If i<5> => L = 0; Next
                       If i<7> => L = Not L  )
```

Thus, in operate group 1, the instructions clear link, complement link, and set link are formed by coding i<5,7> = 10,01, and 11, respectively. The operate group 2 instructions are used for testing the condition of the processor state. These instructions use bits 5, 6, and 8 to code tests for the Accumulator. The AC skip conditions are coded as never, always, AC Eql 0, AC Neg 0, AC Lss 0, AC Leq 0, AC Geq 0 and AC Gtr 0. If all the nonredundant and useful variations in the two operate groups were available as separate instructions in the manner of the first seven (dca, tad, etc.), there would be approximately 7 + 12(group 1) + 10(group 2) + 6(eae) = 35 instructions in the PDP-8.

The optional Extended Arithmetic Element (EAE) includes additional ~~MQ~~ Multiplier.Quotient (MQ) and ~~SC~~ Shift.Counter (SC) registers and provides the hardwired operations multiply, divide, logical shift left, arithmetic shift, and normalize.

The Interrupt Scheme

External conditions in the input/output devices can request that processor be interrupted. Interrupts are allowed if the processor's interrupt enable flip-flop is set (If INTERRUPT.ENABLE Eqv 1). A request to interrupt (i.e., INTERRUPT.REQUEST=1) clears the interrupt enable bit (INTERRUPT.ENABLE = 0), and the processor behaves as though a jump to subroutine 0 instruction (jms 0) had been executed. A special iot instruction (i<0:11> Eql #6001) followed by a jump to subroutine indirect to 0, and instruction (i<0:11> Eql #5220) returns the processor to the interruptable state with INTERRUPT.ENABLE a 1. The program time to save the processor state is 6 memory accesses (9 microseconds), and the time to restore the state is 9 memory accesses (13.5 microseconds).

Only one interrupt level is provided in the hardware. If multiple priority levels are desired, programmed polling is required. Most I/O devices have to interrupt because they do not have a program-controlled device interrupt enable switch. For multiple devices, approximately 3 cycles (4.5 microseconds) are required to poll each interrupter.

Register-Transfer Level

More detail is required than is provided by either the PMS or ISP levels to describe the internal structure and behavior of the processor and memory. Figure 4 shows the registers and controllers at a block diagram level, and Figure 6 gives a more detailed version using PMS notation. Table 1 gives the permissible register transfer operations that the processor's sequential control circuit can give to the PDP-8 registers.

Table 1. PDP-8 Register Transfer Control Signals and Data Break Interface

```
AC\Accumulator, L\Link and combined L, AC LAC
    AC = 0; AC = #7777; AC = not AC; LAC = LAC + 1
    L = 0; L = 1; L = Not L;
    LAC = LAC Srr 1; LAC = LAC Srr 2; !rotates right
    LAC = LAC Slr 1; LAC = LAC Slr 2; !rotates left
    AC = AC Or Switches; AC = AC And MB; AC = IO.BUS
    AC = AC Xor MB; LAC = Carry (AC,MB);
    (note that previous two commands form: LAC = AC + MB).

MB\Memory.Buffer
    MB = 0; MB = MB + 1;
    MB = PC; MB = AC; MB = M[MA]; MB = DB.DATA.

MA\Memory.Address
    MA<0:4> = 0; MA = PC; MA = MB; MA<5:11> = MA<5:11>;
    MA = DB.ADDRESS.

PC\Program.Counter
    PC = 0; PC = PC + 1; PC<0:4> = 0;
    PC = MB; PC<5:11> = MB<5:11>.

IR\Instruction.Register
    IR = 0; IR = M[MA]<0:2>

M\Memory[0:4095]<0:11>
    M[MA] = MB !write
    MB = M[MA] !read

DB\DATA.BREAK    interface
    DB.DATA<0:11>             ! Input to MB
    DB.ADDRESS<0:11>        ! Input to MA
    MB<0:11>
    DB.REQUEST              ! Control inputs to Pc
    DB.DIRECTION
    DB.CYCLE.SELECT<0:11>
    ADDRESS.ACCEPTED        ! Control outputs from Pc
    WORD.COUNT.OK
    BREAK.STATE
```

Although electrical pulse voltages and polarities are not shown in Table 1, the operations are presented in considerably more detail than shown in Fig. 4.

As Fig. 6 shows, the registers in the processor cannot be uniquely assigned to a single function. In a minimal machine such as the PDP-8, functional separation is not economical. Thus, there are not completely distinct registers and transfer paths for memory, arithmetic, program, and instruction flow. (This sharing complicates understanding of the machine.) However, Fig. 6 clarifies the structure considerably by defining all the registers in the processor (including temporaries and controls). For example, the memory buffer (MB\Memory.Buffer<0:11>) is used to hold the word being read from or written to memory. ~~MB~~ the memory buffer also holds one of the operands for binary operations (for example, AC = AC And MB). ~~MB~~ The memory buffer is also used as an extension of the Instruction.Register during the instruction interpretation. The additional physical registers, not part of the ISP, are:

MB\Memory.Buffer<0:11>	holds memory data, instruction, and operands
MA\Memory.Address<0:11>	holds address of word in M being

accessed

IR\Instruction.Register<0:2> holds the value of current instruction being performed

State.Register<0:1> a ternary state register holding the major state of memory cycle being performed - declared as 2-bits

F\Fetch:=(If State.Register Eqv 0) memory cycle to fetch instruction

D\Deferred:=(If State.Register Eqv 1) memory cycle to get address of operand

E\Execute:=(If State.Register Eqv 2) memory cycle to fetch (store) operand and execute the instruction

Emphasis in Figure 6 is on the static definition (or declaration) of the information paths, the operations, and state. The ISP interpretation (Appendix 1) is the specification for the machine's behavior as seen by a program. As the temporary hardware registers are added, a more detailed ISP definition must be given in terms of time and in terms of temporary and control registers. Instead, ~~we give~~ a state diagram (Fig. 7) is given to define the actual processor which is constrained by both the ISP registers, the temporary registers implied by the implementation, and time. The relationship among the state diagram, the ISP description, and the logic is shown in the hierarchy of Fig. 1. In the relationships of the figures, ~~we~~ one can observe that the ISP definition does not have all the necessary detail for fully defining a physical processor. The physical processor is constrained by actual hardware logic and lower-level details even at the circuit level. For example, a core memory is read by a destructive process and requires a temporary register (MB) to hold the value being rewritten. This is not represented within a single ISP language statement because ~~we~~ ISP defines only the nondestructive transfer; however, it can be considered as the two parallel operations MB = M[MA]; M[MA] = 0. The explanation of the physical machine including the rewriting of core using ISP, is somewhat more tedious than the highest-level description shown in Appendix 1. For this reason, the state diagram is used (see Figure 7), and the description of the physical machine (in ISP) is left as an exercise to the reader.

The state diagram (Fig. 7) is fundamentally driven by minor clock cycles as seen from both the state diagram and the times when the four clock signals are generated. Thus, there are three (State.Register Eqv #0,#1,#2) X 4 (clock) or 12 major states in the implementation. The IR is used to obtain two more states, F2b and F3b, for the description. The State.Register values 0, 1, and 2 correspond to Fetching, Deferred or Indirect addressing (i.e., fetching an operand address), and Executing. The state diagram does not describe the Extended Arithmetic Element operation, the interrupt state, and the data break states (which add 12 more states). The initialization procedure, including the console state diagram, is also not given. One should observe that when to occurs at the beginning of the memory cycle, a new State.Register value is selected. The State.Register value is always held for the remainder of the cycle; i.e., only the sequences (F0, F1, F2, F3, or D0, D1, D2, D3 or E0, E1, E2, E3) are permitted.

Logic Design Level (Registers and Data Operations)

Proceeding from the register transfer and ISP descriptions, the next level of detail is the logic module. Typical of the level is the 1-bit logic module for an accumulator bit, AC<j>, illustrated in Fig. 8. The horizontal data inputs in the figure are to the logic module from AC<j>, MB<j>, AC<j> input from the IO.Bus.In, and SWITCHES<j>. The control signal inputs whose names are identified using the vertical bar (e.g., ¦AC = 0¦) command the register operations (i.e., the transfers). They are labeled by their respective ISP operations (for example, AC = AC And MB, AC = AC Slr 1, for rotate once left). The sequential state machine, for the processor Pc(K), generates these control signal inputs using a combinational circuit like that shown in Figure 9.

Logic Design Level (Pc Control, PC(K) Sequential State Machine Network)

The output signals from the processor sequential machine (Fig. 9) can be generated in a straightforward fashion by formulating the Boolean expressions directly from the state diagram in Fig. 7. For example, the AC = 0 control signal is expressed algebraically and with a combinational network in Fig. 9. Obviously, these Boolean output control signals are functions which include the clock, the State.Register, and the states of the arithmetic registers (for example, AC = 0, L = 0, etc.). The expressions should be factored and minimized so as to reduce the hardware cost of the control for the interpreter. Although the sequential controller for the processor is mentioned here only briefly, it constitutes about half the logic within the processor.

Circuit Level

The final level of description is the circuits which form the logic functions of storage (flip-flops) and gating (NAND gates). Figures 10 and 11 illustrate some of these logic devices in detail.

In Fig. 10 a direct set/direct clear flip-flop (a sequential-logic element) is described in terms of circuit implementation, combinational logic equivalent, a state table, and its algebraic behavior. Note that this is not a conventional textbook circuit because it has no output delay and responds directly and immediately to an input. Some conventional sequential logic elements are used in the PDP-8 (but not illustrated), including RS (Reset-Set), T(Trigger), D(Delay), and JK. A delay in the flip-flops makes them behave in the same way as the "textbook" primitives in sequential circuit theory. The outputs require a series delay, $\hat{-}$ t, such that, if the inputs change at time t, the outputs will not change until t + $\hat{-}$ t. In actuality, the PDP-8 uses capacitor-diode gates at the flip-flop inputs so that input changes will not be noticed until after the clock passes. This achieves the same effect.

Figure 11 illustrates the combinational logic elements used in the PDP-8. The circuit selection is limited to the inverter circuit with single or multiple inputs. These are more familiarly called NAND gates or NOR gates, depending on whether one uses positive and/or negative logic-level definitions (described in Chapter 4).

The core memory structure is given in Figure 6. A more detailed block which diagram showing the core stack with its twelve 64 x 64 1-bit core planes is needed. Such a diagram, though still a functional block diagram, takes on some of the aspects of a circuit diagram because a core memory is largely circuit-level details. The memory (Figure 12) consists of the component units: the two address decoders (which select 1 each of 64 outputs in the X and Y axis directions of the coincident current memory); selection switches (which transform a coincident logic address into a high-current path to switch the magnetic cores); the 12 inhibit drivers (which switch a high current or no current into a plane when either a 1 or 0 is rewritten); 12 sense amplifiers (which take the induced low sense voltage from a selected core from a plane being switched or not switched and transform it into a 1 or 0); and the core stack, an array M[#0:#7777]<0:11>. Figure 12 also includes the associated circuit-level hardware needed in the core-memory operation (e.g., power supplies, timing, and logic signal level conversion amplifiers).

The timing signals are generated within the control portion of the processor and are shown together with processor clock in Figure 13.

The process of reading a word from memory is:

1. A 12-bit selection address is established on the MA<0:11>

address lines, which is 1 of #10000 (or 4096) unique numbers. The upper 6 bits <0:5> select 1 of 64 groups of Y addresses, and the lower 6 bits <6:11> select 1 of 64 groups of X addresses.

2. The read logic signal is made a 1 at time t2.

3. A high-current path flows via the X and Y selection switches. In each of the X and Y directions, 64 X 12 cores have selection current (Ix and Iy). Only one core in each plane is selected since Ix = Iy = Iswitching/2, and the current at the selected intersection = Ix + Iy = Iswitching.

4. If a core is switched to 0 (by having Iswitching amperes through it), then a 1 is present and is read at the output of the plane bit sense amplifiers. A sense amplifier receives an input from a winding that threads every core of every bit within a core plane [#0:#7777]. All 12 cores of the selected word are reset to 0. The time at which the sense amplifier is observed is tms (the memory strobe), which also causes the transfer MB = M[MA].

5. The read current is turned off by timing in the memory module.

6. The inhibit and write (slightly delayed) logic signals are turned on at time t1. The bit inhibit signal is present or not, depending on whether a 0 or 1, respectively, is written

into a bit.

7. A high-current path flows via the X and Y selection switches, but in an opposite direction to the read case (see item 2). If a 1 is written, no inhibit current is present and the net current in the selected core is -Iswitching. If a 0 is written, the current is -Iswitching +(Iswitching/2) and the core remains reset.

8. The inhibit and write logic signals are turned off at time tmd specified by timing in the memory module, and the memory cycle is completed.

~~We could continue to discuss~~ The discussion could be continued to include the behavior of the transistor as it is used in these switching-circuit primitives but ~~will, instead, leave~~ that is covered in better detail in ~~that to~~ books on semiconductor electronics and physics. It is hoped that the reader has gained a sense of how to think about the hierarchical decomposition of computers into particular levels of analysis (and synthesis) and that the hierarchical approach will prove useful while reading the chapters on the PDP-11.

Chapter 7 Figure Legends

Figure 1. PDP-8 hierarchy of descriptions.

2. LINC-8-338 PMS diagram.

3. PDP-8 S('I/O Bus) logic and PMS diagrams.

4. PDP-8 processor block diagram.

5. PDP-8 instruction decoding diagram.

6. PDP-8 register transfer level PMS diagram.

7. PDP-8 Pc state diagram.

8. PDP-8 AC<j> bit logic diagram.

9. PDP-8 Pc(K) AC = 0 signal logic equation and diagram.

10. PDP-8 direct-coupled flip-flop and logic diagram.

11. PDP-8 combinational circuit and logic diagram.

12. PDP-8 four-wire coincident current (three dimensions) core memory logic diagram.

13. PDP-8 clock and memory timing diagram.