The Intersil and ~~CMOS~~ Harris CMOS 6100, 12-bit Processors

~~These machines were begun~~

In 197x Intersil designed the first ~~CMOS~~ PDP-8 processor to ~~it~~ occupy a single silicon chip using complementary mos (CMOS) technology. We verified that it was a PDP-8 in 197x and began to apply it to a product. In the meantime, Harris Semiconductor became a second source to Intersil to ~~utiliz~~ supply CMOS-8 ~~semicond~~ chips. ~~In the fall of 1976,~~ in the fall of 1976.

As an interesting sidelisht, we began ~~to carry out~~ the design of a single chip ~~PMOS~~ P-~~ch~~ channel MOS processor-on-a-chip PDP-8, called the 8/B in the fall of ~~1973~~ 1972, with the basic definition occurring in the spring of 73 ~~and~~ to obtain ~~the first chips in October 73~~ ~~with final~~ production chips in the spring of 74, ~~and~~ Thus production starting in the fall of 74 seemed possible (after about one year) This project was stopped in the summer of 73 ~~,~~ and the Western Disital project was began, ~~in which~~ with the goal ~~was~~ to put ~~several~~ a PDP-11 on one or more LSI chips. Since the PDP-8 ~~dessingefss~~ designers had progressed ~~went through~~ the design ~~study of~~ tradeoffs in partitioning a ~~n~~ 8 ~~design~~ for a single 40- pin chip in the 8/B project, this was of value in the ~~C/B~~ CMOS-8 design and its partitioning. The LSI-11 design ~~by~~ started ~~Western Disital occurred~~ right at the ~~transition between P-channel and~~ beginning of N-channel, hence, ~~an~~ more complex design yielding a PDP-11 was possible, versus the more straightforward, and less power ful 8/B design we terminated.

The CMOS-8 processor block digaram is given in Figure CMOS8 IMP. ~~02~~ The reader should note, not surprisingly , it looks very much like a conventional PDP-8 ~~the~~ processor design ~~of earlier~~ using integrated circuits, ~~designs in that~~ It has a common data path for ~~operating on~~ manipulating the ~~various resisters~~ registers. PC, MA, MQ AC and Temp ~~(to resolve races)~~.

The ~~IR~~ Instrucution Resister (IR) does not share the common arithmetic logic unit.

A ~~2~~ two-way multiplesxor is used to transfer ~~instructins from memroy and addre~~

The reason to stop the design included the P- to N-channel transition, and the fact that the 8/E bus did not lend itself to cost reduction with LSI technology. The Omnibus, though ideal for MSI, is not as cost-effective and ease of interfacing as time-multiplexed, shorter busses of microcomputer use.

addresses to memory and to recieve and transmit data and instrcutions from memory. This is the basis of the a microprogrammed control processor.

The Programmed Logic Array (PLA) is a form of read only memory providing two levels oflogic, instead of just ~~address look that a ROM~~ data in response to an address that a conventional provides. PLA ~~also~~ enables sparse memories to be built more economically (memories with little data spread over many addresses). The output of PLA, ~~is used to generate control lines~~ controls, via the dotted lines in the figure, to control the various resister transfers ~~and transmit data to and from the ouside world.~~ among the including those to the outside world.

While the ~~Process~~ CMOS-8 is the first DEC processor on a single chip, we are interested here in ~~what~~ system that can be built ~~from it~~ using it. ~~While~~ The CMOS integrated circuits ~~absor~~ also require very little power, however, the main property of interest is that small systems can be built conveniently with the basic 6100 chips. Figure PMSCMOS8 shows ~~the basis~~ a system built with 6100-type components. It should be noted thatseveral other comonents are actually more important than the processor, when it comes to reducing the number of chips and therefore the ~~total~~ total cost of a system. The other chips of interest in the diagram;

1. random access read write memory (ram);
2. read only memory (ROM);
3. A parallel interface which enables interrupt signals to be sent back to the processor and also detects IIOT etc select commands for controlling data transfers; and
4. specific i/o devices like the universal asynchronous reciever transmitter (UART) for detecting and bufferins ~~Te si~~ serial signals from ~~Telegraph~~ communications lines.

The VT78

The VT78 is ~~made~~ possible because of ~~the basic~~ low power consumption and availability of the CMOS-8 chips together with high density random access memory chips. Its goals ~~were~~ are quite simple: to drastically reduce the cost of a complete computer system including keyboard, CRT & Mass Storage by integrating the processor into one of the existing cabinets, ~~and elimin~~ thereby eliminating ~~one~~ much of the ~~packages.~~ packaging and space. Since, the VT 52 CRT had been designed with the notion that more boards would be housed inside it, the VT78 ~~was based on a packaging the~~ is packaged consists of the ~~basic~~ housed CPU system ~~and interfaces~~ within the CRT. The optional mass storage — the RX01 flexible disk drives ( ) that are needed to form a complete stand alone system are housed ~~in an additional metal box~~ separately. The VT78, without mass storage forms the basis for an intelligent terminal, ie. We consider an intelligent terminal to include ~~A terminal with~~ a computer ~~in it that~~ whose program can be loaded ~~with various program~~ (usually via the communications lines) such that it can take on ~~any number of~~ a variety of characteristics --ie it can learn, hence it ~~is intelligent~~, can be thought of (loosely) as being intelligent.

The applications for ~~the~~ an intelligent terminal (with computer) ~~of the VT78~~ are governed by the network to which it is attached, but as a stand alone computer complete with ~~the~~ mass storage ~~this~~, it is designed to function as a desktop computer and run the various ~~commercial~~ software that is built for PDP-8. (especially commercial applications) Since a particular generic application, word processing, ~~using~~ also uses the PDP-8, ~~the~~ it is believed that ~~the~~ this will be dominant application ~~will be word processing~~ based on VT78 on a stand alone basis. The WP78 ~~sut~~ system is reduced in price from ? to ?, by this repackaging; the size is reduced by a factor of 2, two while the ~~and~~ weight ~~by~~ is reduced from ? to ? ~~reduced by a factor of~~. In this wasy the the terminal is easily movable, versus being fixed within a desk on a ~~more or less~~ semi- permanent basis.

Figure PMS78 shows a block diagram of the complete computer system. Note that the structure is precisely the antithesis of most of the systems we have previously considered. It is not modular, but is rather bounded and has a relatively fixed configuration, with open-ended busses. There are While it arrays The VT52 CRT is via a serial line unit #1. The memory is constrained to have 16 Kwords. Two serial line units can be programmed to respond to line rates of 50 to 9600 baud. There is a parallel interface which permits 12 bits to be transferred under program control at up to 15 Kwords/sec., and while the interface is nominally for a printer, and any device using the protocol can attach to it. An especially interesting port is the MR78 which permits serial data to be loaded into the memory each time the start button is pushed. In this a A special serial, read only memory is attached to the port for program loading such that a computer can be provided. In this way with a basic set of programs that are oriented to the particular application, a computer is given a singly by changing the port connection.

The panel memory is a special read only memory that is used for s as to provide the basisc functionality for the computer and to handle the diagnostics prior to initia program loading.

t turning hte computer on via M the MR78 Program Injection mechanism.

15 Kw/sec

called the Program Injector

~~PDP-12~~

draft

~~PDP-8~~ PDP-8 TECHNOLOGY, PRICE AND PERFORMANCE OF THE FAMILY

but these metrics are outside the scope of our current analysis

New technology can be utilized in three basic forms to build computers:

1. at reduced prices while providing roughly constant performance and constant functionality -- the mini(mum) computer;
2. at increased performance and functionality with roughly a constant price; and
3. of new basic structures.

The PDP-8 has been re-implemented ~~ten~~ nine times (in ~~six~~ a few basic structures) with new technology over a period of fifteen years so as to provide roughly constant performance (see Fig. Perf) at a decreasing price (see Figs. Price and Linprice). ~~Overall, the price has~~

This figure shows the effect of the constituent parts ~~by plotting the~~ in three configurations of processor with 4 kwords of primary memory, and with console terminal, and with a ~~minimum price~~ secondary memory for ~~program loading and/or~~ file storage. Note the basic computer consisting of processor, memory, package and power has declined more rapidly (at ~~X%~~ per year) while the terminal and secondary ~~memory~~ parts have declined ~~much~~ less rapidly. ~~Figures PoMpprice, and Tprice Mspriice show the price of the individual components that make up the previous basic systems~~. The price and performance trajectory can be seen in Fig. Priceperf. Note that the earliest implementations had significantly less performance than the classical PDP-8. This increased performance came about through higher speed primary memory technology. The performance (in instructions per second) is completely correlated to memory cycle time. While performance has been relatively easy to obtain, the design emphasis here is on continually lowering cost. By spending more, a somewhat better cost/performance ratio is possible and these machines can be observed in the case of additional floating point hardware for the 8/E and 8/A. Similarly a prototype PDP-8 with cache memory (Cassasent, Bell and Bell) was constructed which had substantially greater performance at only a doubling of processor price. Figure Price also shows the oscillatory stairstep effect that occurs with new designs:

1. The 5 was predicated on minimum price, whereas the classical PDP-8 had additions so as to be more cost effective while not increasing price significantly. It had a performance of 6.7 times the PDP-5.
2. The PDP-8/S was a response to achieve truly minimum price by serial implementation technology. Since the performance of the 8/S was so terribly slow, the pressure induced a constant price, integrated circuit version, the 8/I.
3. Since the 8/I was relatively expensive, the 8/L was introduced as a cost reduced version.
4. The 8/E was then introduced as a high performance, large system machine enabling larger (than 8/I) systems to be built. The ~~8/F and~~ 8/M were cost reduced, smaller cabinet versions.

The performance is constant as determined by the minimum amount spent on a core memory.

with lines of constant Price/performance separated at factors of two.

Primary Memory has declined in price at roughly 20% per year as seen in Fig. Mprice.

(DECtapes or floppies)

22% and more 15% recently 15%

-Insert B-pg of this

Compare multi-chip team

Over a slower speed design

in all machines except PDP-5 and 8/S because these used primary memory for holding the Program Counter, and were implemented using serial techniques, respectively.

approaches (factor of five)

went to

and

crosses

The PDP-8

6

may have helped

and a minimum price memory design

to bring it in line with market needs and expectations

both to PDP-8 and the 8/S

needed for the OEM market.

quickly

three lines of constant price/performance.

5. The 8/A was introduced as a further cost reduction. Recent, higher priced, larger versions of the 8/A have been built for the configurations usually served by the 8/E. Also since the 8/A does not entirely replace the 8/A or E as a systems machine, the 8/E has necessarily remained in production.
6. In 1976, Intersil and Harris introduced one chip processors as true mini(mal) computers at minimal costs. CMOS PDP-8

Figure Linprice gives another view of the computer price versus time by presenting a linear plot of the price. Here we see the switch in price on a fractional basis from the electronic to the electromechanical and package aspects of the computer. A similar plot of just the processor and memory price is given in Fig. Lincomp but separating the power and package costs from the Pc board prices. Note, that since the processor and its components have not been sold separately, these prices have been imputed from the cost. A similar plot relating to board price is given in Fig. Spares. Here we note that the cost of spares has risen sharply with integrated circuits, while the number of Pc boards has decreased rapidly. this has occurred both because integrated circuits with greater functionality are more expensive than the small and medium scale integrated circuits and because there are few, if any, common replaceable printed circuit boards. Thus, the smallest replaceable unit is the computer, complete with power supply. This trend to have only one printed circuit board can be temporarily turned around by using sockets for integrated circuits. However, with very large scale integrated (ULSI) circuits, we would soon expect the complete processor and 4 kwords (50,000 bits) on a single integrated circuit.

The physical attributes of volume, weight, power and basic computer printed circuit board area are given in Fig. Physical. From these graphs it is clear the price declines at a rate roughly proportional to its constituent materials. Figure Physprice displays the per unit cost of each of the constituent material indicators.

Figure Physperf gives the per unit performance metrics. The two most important metrics are the performance (in million instructions per second) per unit space (in ft3) and per unit power (in watts). Note the CMOS 8 on a board has incredible performance/power because of the very low power CMOS technology.

Figure Interconnect shows three measures which are roughly correlated with power: the amount of power required, the number of conductors, and the price to interface to the computers busses. These measures are especially meaningful to minicomputer users as they measure complexity (simplicity), price and operational cost of an interface.

Figure Opprice gives other operational price data including the hourly and particular configuration maintenance prices and the cost/watt of power. Note that the inflation rate is also plotted on the same figure and both the hourly maintenance price and power cost have the same slope. The maintenance and power costs actually decline even though these per unit costs increase with inflation. These occur because of the increased reliability and the rapid decline in the use of power.

Figure MTBF shows how the mean time between failures both on a calculated and observed basis for the basic machine has increased with time. We can make a similar per unit metric using MTBF showing the number of failures per unit of

power and board area (see Fig. Physfail). These measures are roughly constant with time.

Several market aspects of PDP-8 are plotted in Figs. Quan, Tquan, Demand and Memsize. From Fig. Quan we see that the life expectancy of a given model is X years and has increased somewhat these last years. The increased life expectancy has come about through better designs, more stable fabrication technology, the higher cost to produce computers in greater volume and the increased competitive internal pressure fom 16-bit computers. Since the recent 8/E, F, M, A series all use a common bus, individual parts can be made more competitive (evolved) without the necessity for a complete redesign. Figure Demand attempts to show that the demand for machines is completely elastic. Note that each time the price is reduced by X% the quantity produced is doubled. We note that the demand originally was elastic, but then subsequently became inelastic. This occurred because a number of DEC Market Product Lines evolved to use only the PDP-11.

Figure Memsize gives the average memory size of the PDP-8's. The curve shows that a significant amount of memory is added on to existing computers by third party vendors.

the 12-bit machines

Figure PWV, which presents the power, weight and volume of ~~PWV~~ also shows the oscillating design styles. ~~that have~~ In general, the power has remained relatively constant, although the CMOS 8 design drastically cuts the power needs for the VT78 (~~also~~ the power is also less ~~than~~ because ~~the~~. there are limited options). Constant power is needed in the post 1970 designs because the ~~Omni~~ package must ~~so~~ house a fixed number of devices and each device ~~has~~ has a relatively high overhead power cost associated with driving the Omnibus.

likewise the volume is relatively constant for the Omnibus designs. The weight and volume have declined ~~has come down~~ significantly with time ~~do a great deal over the time~~ as the design PDP 8 has moved from 2 cabinets, to 1/2 cabinet and finally to being embedded in a CRT terminal.

~~The~~ figure Bitsprice gives the ~~price~~ performance/price metric and it can be directly compared with other machines in this book. Recall that the 18-bit machines have improved at 52% to 69% per year — over a short time span. Since we have ~~neglected the PDP thrown~~ ignored the PDP-5 design point, the improvement for the ~~PD~~ 12-bit machines has been much less radical with only 22% yearly improvement. For comparison, the range of 18-bit designs are given in the graph.

However, the evolution ~~in~~ of the PDP-5, ~~PDP Card~~ 8, 8/I and 8/L falls with the more rapid, 18-bit evolution

Rather than try to ~~fit~~ a fit a single exponential to the data points, it is ~~instructive~~ worthwhile to consider that ~~if~~ we are observing the transition between two generations, ~~and hence may~~ Two, ~~curves~~ independent exponentials may better serve to model the cost decline. PDP-5 was a mid-second generation (transistor) while the PDP-8 represents a late second generation

(over)

and the 8/I (and 8/L), are beginning third generation designs. These 3 or 4 machines evolved rapidly from 1963 to 1968. With the 8/L, a slower evolution is experienced from 1968 to 1977 as medium scale integrated circuits are used as the implementation technology. In actual practice, the bus width constrains the designs to be basically evolutionary.

The CMOS8 avoids the wide bus problem by moving the bus to a single second of section for lines on the computer printed circuit board. A complete 4 kword PDP-8 only occupies X sq. inches of board area. Note that the reduced VT78 has ~~with~~ less cost-effective than an 8/A, ~~even though~~ because of the factor of three slower performance

draft

TECHNOLOGY, PRICE AND PERFORMANCE OF THE FAMILY

New technology can be utilized in three basic forms to build computers:

1. at reduced prices while providing roughly constant performance and constant functionality -- the mini(mum) computer with constant functionality;

2. at increased performance and functionality with roughly a constant price; and

3. of new basic structures.

The PDP-8 has been re-implemented ten times (in a few basic structures) with new technology over a period of fifteen years so as to provide roughly constant performance (see Fig. Perf) at a decreasing price (see Figs. Price and Linprice). The performance is constant as determined by the minimum amount spent on a core memory.

The figures show the effect of the constituent parts in three configurations of processor with 4 kwords of primary memory; and with console terminal; and with a secondary memory (DECtapes or floppies) for file storage. Note the basic computer consisting of processor, memory, package and power has declined more rapidly (at 22% and more recently 15% per year) while the terminal and secondary memory parts have declined less rapidly. Primary memory has declined in price at roughly 20% per year as seen in Fig. Mp.price. The price and performance trajectory can be seen in Fig. Priceperf with lines of constant price/performance separated at factors of two. Note that the earliest implementations had significantly less performance than the classical PDP-8. This increased performance came about through higher speed primary memory technology. The performance (in instructions per second) is completely correlated to memory cycle time in all machines except PDP-5 and 8/S because these used primary memory for holding the Program Counter, and were implemented using serial techniques respectively. While performance would have been relatively easy to obtain, the design emphasis has been on continually lowering cost. By spending more, better cost/performance ratios are possible and these machines can be observed by additional floating point hardware for the 8/E and 8/A -- but these metrics are outside the scope of our current analysis. Similarly a prototype PDP-8 with cache memory (Cassasent, Bell and Bell) was constructed which had greater performance (factor of five) while only doubling the processor price. Figure Bitsprice gives the performance/price metric and it can be directly compared with other machines in this book. Recall that the 18-bit machines have improved at 52% to 69% per year -- over a short time span. For comparison, the range of 18-bit designs are given in the graph. Since we have ignored the PDP-5 design point, the improvement for the 12-bit machines has been much less radical with only 22% yearly improvement. However, the evolution of the PDP-5, 8, 8/I and 8/L falls with the more rapid, 18-bit evolution.

Rather than try to fit a single exponential to the datapoints, it is worthwhile to consider that we are observing the transition between two generations. Two, independent exponentials may better serve to model the cost decline. PDP-5 was a mid-second generation (transistor) while the PDP-8

each time the start button is pushed. A special serial, read only memory called the Program Injector is attached to the port for program loading. In this way a computer can be given a program that is oriented to a particular application, simply by changing the port connection.

This operates at a speed of x bits/sec, requiring y sec. for a 16 Kw program to load.

1. random access read write memory (RAM);

2. read only memory (ROM);

3. a parallel interface which enables interrupt signals to be sent back to the processor and detects IOT select commands for controlling data transfers; and four specific i/o devices like the universal asynchronous receiver transmitter (UART) for detecting and buffering serial signals from communications lines.

what can be connected to

also the system is not expandable beyond the 5 ports that form the single, integrated system.

The VT78

The VT78 is possible because of low power consumption and availability of the CMOS-8 chips together with high density random access memory chips. Its goals are quite simple: to drastically reduce the cost of a complete computer system including keyboard, CRT and Mass Storage by integrating the processor into one of the existing cabinets, thereby eliminating much of the packaging and space. Since, the VT52 CRT had been designed with the notion that more boards would be housed inside it; the VT78 consists of the CPU system housed within the CRT. The optional mass storage -- (the RX01 flexible disk drives) that are needed to form a complete stand alone system are housed separately. The VT78, without mass storage forms the basis for an intelligent terminal. We consider an intelligent terminal to include a computer whose program can be loaded (usually via the communications lines) such that it can take on a variety of characteristics -- i.e., it can learn, hence it can be thought of (loosely) as being intelligent.

designed to operate on the

The applications for an intelligent terminal (with computer) are governed by the network to which it is attached, but as a stand alone computer complete with mass storage, it is designed to function as a desktop computer and run the various software (especially commercial applications) that is built for PDP-8. Since a particular generic application, word processing, also uses the PDP-8, it is believed that this will be dominant application on a stand alone basis. The WP78 system based on VT78 is reduced in price from ? to ? by this repackaging; the size is reduced by a factor of two while the weight is reduced from ? to ? In this way the terminal is easily movable, versus being fixed within a desk on a semi-permanent basis.

21

Figure PMS78 shows a block diagram of the complete computer system. Note that the structure is precisely the antithesis of most of the systems we have previously considered. It is not modular with open-ended busses, but is rather bounded and has a relatively fixed configuration. The coupling to the VT52 CRT is via a serial line unit #1. The memory is constrained to have 16 Kwords. The serial line units can be programmed to respond to line rates of 50 to 9600 baud. There is a parallel interface which permits 12 bits to be transferred under program control at up to 15 Kwords/sec., and while the interface is nominally for a printer, any device using the protocol can attach to it. The panel memory is a special read only memory that is used to provide the basic functionality for the computer and to handle the diagnostics prior to program loading. An especially interesting port is the MR78 which permits serial data to be loaded into the memory

(180,000 baud)

draft

THE INTERSIL AND HARRIS CMOS 6100, 12-BIT PROCESSORS

In 197x Intersil designed the first PDP-8 processor to occupy a single silicon chip using Complementary MOS (CMOS) technology. We verified that it was a PDP-8 in 197x and began to apply it to a product in the fall of 1976. In the meantime, Harris Semiconductor became a second source to Intersil to supply CMOS-8 chips.

As an interesting sidelight, we began the design of a single chip P-channel MOS processor-on-a-chip PDP-8, called the 8/B in the fall of 1972, with the basic definition occurring in the spring of 1973 so as to obtain production chips in the spring of 1974. Thus production starting in the fall of 1974 seemed possible. Since the PDP-8 designers had progressed through the design tradeoffs in partitioning an 8 for a single 40-pin chip in the 8/B project, this was of value in the CMOS-8 design and its partitioning. This project was stopped in the summer of 1973 (after about one year) and the Western Digital project was begun, with the goal to put a PDP-11 on one or more LSI chips. The key reason to stop the design included the P- to N-channel transition, and the fact that the 8/E bus did not lend itself to cost reduction with LSI technology. The Omnibus, though ideal for MSI, and ease of interfacing is not as cost-effective as time-multiplexed, shorter busses that microcomputers use. The LSI-11 design ~~started right~~ was predicated at the beginning of N-channel, hence, a more complex design yielding a PDP-11 was possible versus the more straightforward, and less powerful 8/B design we terminated.

an 8-on-a-chip would then not have been cost-effective at the system level due to the large number of additional memory and peripheral circuitry and costs.

19

The CMOS-8 processor block diagram is given in Figure IMPCMOS8. The reader should note, not surprisingly, it looks very much like a conventional PDP-8 processor design using integrated circuits. It has a common data path for manipulating the PC, MA, MQ, AC and Temp (registers).

The Instruction Register (IR) does not share the common arithmetic logic unit. A two-way multiplexor is used to transfer addresses to memory and to receive and transmit data and instructions from memory. The Programmed Logic Array (PLA) is a form of read only memory providing two levels of logic, instead of just data in response to an address that a conventional ROM provides. This is the basis of the microprogrammed control processor. PLA enables sparse memories (memories with little data spread over many addresses) to be built more economically. The output of the PLA controls, via dotted lines in the figure, the various register transfers including those to the outside world.

20

While the CMOS-8 is the first DEC processor on a single chip, we are interested here in a system that can be built using it. The CMOS integrated circuit also requires very little power; however, the main property of interest is that small systems can be built conveniently with the basic 6100 chips. Figure PMSCMOS8 shows a system built with 6100-type components. It should be noted that several other components are actually more important than the processor, when it comes to reducing the number of chips and therefore the total cost of a system. The chips of interest in the diagram:

More Omnibus 8 computers have been constructed than any of the previous models. The improvement (gain) in volume appears to be due to a basic simplicity in design, together with the ability for a user to easily build rather arbitrary system configurations.

Whereas the gain from PDP-5 to PDP-8, automated the backpanel wiring process, the Omnibus 8 series eliminates backpanel wiring and replaces it by a printed circuit. The next step to the VT78 eliminates the backpanel by mounting the entire computer on a single printed circuit board

The FPP8/A

~~The greatest~~ a single accumulator floating point processor, with its own ISP was added to the 8/A in 1976 ~~as a~~ to improve the speed of floating point. It is compatible with the FPP12/A, a floating point processor for PDP-12 and occupies two hex sized modules. It supports ~~either~~ 3 or 6 word floating point arithmetic (12-bit exponent and 24- or 60-bit fraction) and 2 word double precision 24-bit arithmetic ~~and has an~~ as a completely independent processor with its own ISP, it has ~~an 8-word~~ its own Program Counter, and 8-index registers. The performance ~~with FP~~ is approximately equal to a 360 Model 40 ~~with flua~~. This ~~gives such a machine somewhere~~ processor provides the (probably) the greatest performance/cost of any ~~machine~~ computer

Recommendation

Complete CAD 3 order immediately.

Deliver a TU70 for CAD 1.

Deliver the RS04's.

Allow the balance of the order to drift into FY78.

mg

the earlier PDP-8 implementations. Since there are no pass-through signals on the bus, any module can occupy any slot -- simplifying configuration, design and testing. A by-product (or perhaps goal) of the Omnibus Structure is that there are a fixed number of slots and well defined electrical transmission characteristics since there is no cabling between options.

The 8/E implementation is mostly determined by the availability of Integrated Circuits (see IMP8E). Here, multiplexors, register files and basic arithmetic logic units to perform PDP-8's basic operations are all that's needed. Read only memories in the early 1970's were only about 64 bits, thus a microprogrammed control is not feasible. The 8/E processor occupies three printed circuit boards (or about 240 square inches), contrast with the original PDP-5 which occupied about 100 boards for a total of 2100 square inches.

The problem of partitioning the logic and assigning it to one of three modules arises in the 8/E. Figure IMP8E shows the clock, timing and interrupt on one module, the data path on a second and the control the the last module. By this partitioning, more pins are required between the data and control modules. This is accomplished by placing additional connectors on the back edge of the modules.

8/A

In 1975 the 8/A was introduced to extend the 8/E family to a lower price threshold and provide additional memory capability particularly by utilizing semiconductor memory chips. The module size was extended to the DEC hex size (8" by 15"), enabling somewhat larger options to fit on a single module. Thus, the 8/A processor could occupy a single module.

In the first implementation, using 1 Kword semiconductor chips, only 48 chips were needed to form the 4 Kword memory. While this size memory easily fit on a single quad sized module (or less) the 8 Kword and 16 Kword core memories are also able to fit. In 1977, using 16 Kword chips a memory of 128 Kwords only requires 12 x 8 or 96 chips. Figure IMP8A shows the implementation of the processor. The register structure is virtually the same as the 8/M, while the control is a form of microprogramming using read only memories. This reduces the processor board area from 240 sq. inch to 120 sq. inch.

With greater use of semiconductor memory, especially read only memories, a scheme was devised and added to the 8/A so that programs written for read-write memory could be run in read-only memory. The scheme adds a 13th bit to the read-only memory and signifies that a particular cell is actually a cell that is both read and written. When the processor detects the 13th bit, the other 12 bits point to a location in some read-write memory which acts to hold the variable information. This effectively provides an indirect memory reference.

draft

THE 8/E, 8/M AND 8/A

The 8/E was introduced in 1971, after the PDP-11, as a lower cost replacement to the PDP-8/L. Since the 8/L had been compromised for cost reasons over the 8/I, the 8/E in many respects was larger and faster in response to a market pressure for more capabilities. Although the evolved PDP-8 I/O Bus was straightforward, the mechanical packaging and cabling was not. The DMA was not bussed, hence any new bus scheme had to address this problem and be more straightforward. While the Omnibus extends the idea of the PDP-8 I/O Bus, it has the basic topology of the DEC unibus, yet is extended to intraprocessor dialogues and smaller (and simpler) with respect to device control. A block diagram of a complete PDP-8/E computer system is shown in Fig. PMS8E showing all of its available options. Notice in the lower half of the diagram, an adapter exists for interconnecting 8/I and 8/L DMA I/O devices. Physically, an 8/E can hold up to 38 8" by 10" printed circuit cards (modules) while the 8/M holds only 20 cards. The processor is mounted on three modules; with a fourth terminator module. The console requires one module and a 4 Kword core memory takes up three slots, giving a basic computer on eight 8" x 10" modules (or 640 sq. in.). Contrast this size to the 18-bit computers which were implemented in about 5000 sq. inches.

The Omnibus has 144 pins, of which 96 are defined as Omnibus signals. The large number of signals enable much of the communication links that exist within a computer to take place. The reader should contrast this scheme with the Unibus. Basically the signals can be grouped as follows:

1. Master timing to all components.

2. Processor state to console -- A console can continuously interrogate and update the processor.

3. Processor request to memory for instructions and data.

4. Processor to I/O device commands - the processor issues I/O Transfer (IOT) instructions to move data between the AC and an I/O Device.

5. I/O Device to processor signalling completion (Program Interrupts).

6. I/O Direct Memory Access control -- The processor takes on the control and update of memory for 3 cycle memory transfers. Unlike the Unibus, the processor handles much more of the control when DMA and interrupt requests are made. If we are only concerned with the signals between the computer and I/O Devices for programmed control, then approximately 30 signals are involved. Adding DMA capability requires 50 more signals. These are the same signals used in

processor, 4 Kwords, serial line interface).

A processor-on-a-chip PDP-8 was introduced by Intersil Corporation (1976) and Harris Semiconductor (1977) using Complementary Metal On Semiconductor (CMOS). The CMOS-8 structure is shown in Fig. PMSCMOS8 in terms of the semiconductors that form the set of chips that constitute the structure. While this processor-on-a-chip structure is interesting, we will explore its application to form an intelligent terminal with computer and its own mass storage in the form of the VT78. Figure PMS78 gives the structure of the terminal.

The ISP of the PDP-8 ~~Pc~~ processor is about the most trivial in the book and anywhere. It has only a few data operators, namely, <--, +, - (negative of), ¬ (not), ∧, /2, x 2, (optional) x, /, and normalize. It operates on words, integers, and boolean vectors. However, there are microcoded instructions, which allow compound instructions to be formed in a single instruction.

The PDP-8 computer is straightforward and illustrates the levels of a computer structure. We can easily look at it from the "top down." The C in PMS notation is ...

-- Use Bell and Newell p120-136 ~~unchanged~~ as follows attached and corrected.

The 8 classical PDP-8 is described in a "top down" fashion in quite some detail. The evolution to the integrated circuits a Omnibus family is described followed by the Large Scale ~~processor~~ Integrated circuit version, CMOS-8 ~~formula~~ which is a complete processor-on-a-chip. Next the LINC family is ~~mentioned~~ mentioned quite briefly as it relates to the ~~8 to 12~~ family. The last section describes the family evolution in terms of its ~~various~~ parameters physical.

draft

Chapter 5

The DEC PDP-8 Family

Introduction

The PDP-8 is a single-address, 12-bit-word computer of the second generation. It is designed for task environments with minimum arithmetic computing and small Mp requirements. For example, it can be used to control laboratory instrument, such as a pulse height analyzer or a spectrum analyzer. These applications are typical of the laboratory and process control requirements for which the machine was designed. As another example, it can serve as a message concentrator by controlling telephone lines to which typewriters and Teletypes are attached. The computer occasionally stands alone as a small-scale general-purpose computer. It is most often used in this role as the laboratory computer for the scientist, and more recently as a word processing console, and as a small business computer. It was introduced as a small-scale general-purpose time-sharing system, called TSS/8, based on work at Carnegie-Mellon University and DEC. It is used as a KT(display) when it has a P(display; '338). The PDP-8 Family has achieved a production status formerly reserved for IBM computers; about 50,000 have been constructed.

The PDP-8 is typical of several 12-bit computers: the early CDC-160 series (1960), CDC-6600 Peripheral and Control Processor, the SDS-92, M.I.T. Lincoln Laboratory's Laboratory Instrument Computer LINC (1963), the DEC LINC and PDP-12, Washington University's Programmed Console (1967), and the SCC 650 (1966). The Family (transistor, 1963), PDP-8 (1965), PDP-8/S (serial, 1966), and PDP-8/I (integrated circuit, 1968), PDP-8/L (integrated circuit, 1968), constitute a series of computers based on evolving technology. All of these have identical ISP's. Their PMS structures are nearly identical, and all components other than Pc and Mp are compatible throughout the series. The LINC-8-338 PMS structure is presented in Fig. 1. A cost performance tradeoff took place in the PDP-8 (parallel-by-word arithmetic) and PDP-8/S (serial-by-bit arithmetic) implementations. A PDP-8/S is one-fifteenth of a PDP-8 at one-half the cost. The performance factors can be attributed to 8/16 or 5.0 for Mp speed and a factor of about 3 for logical organization, even though the same 2-megahertz logic clock is used in both cases. The PDP-8 is about 6.3 times a PDP-5.

In 1971 the physical implementation structure was changed to the Omnibus, along the lines of the PDP-11 Unibus structure. The Omnibus has 96 signals that provide interconnection among all computer components (e.g., processor, memory, Teletype, disk). The PDP-8/E is the first implementation using the Omnibus; it was followed by size and cost reduced versions called the 8/F and 8/M in 1972. The basic 8/E processor has a memory of 1.5 microseconds. The computer components occupy one or more 8" x 10" printed circuit boards (modules). The 8/E and 8/M hold 38 and 20 modules respectively; 8 slots are required for the basic computer and 4 Kword memory.

In 1975 the 8/A, a further cost reduced version of the Omnibus structure, was introduced. The module size was increased to DEC hex size (8" x 15") and the 8/A holds up to ___ modules (___ slots are required for console,

PDP-8 Family
~~12 Bit~~ Computers

Figure Legend (Handout) from: Bell and Newell

1-15 (go through and copy all titles)

16. PDP-8/E Sy system block diagram

17 PDP-8/E ~~Pr~~ processor, console and core memory register transfer diagram

18 PDP-8/A. processor and console register transfer diagram

19 CMOS-8 ~~regi~~ processor register transfer diagram

20 CMOS-8 system block diagram

21 VT 78 ~~microprocs~~ ~~micro co~~ microcomputer [system / terminal] block diagram

22 ~~PD~~ Performance of ~~DEC's 12-bit~~ PDP-8 Family computers versus time ~~(log plot)~~

23 Price of ~~DEC's 12 bit~~ PDP-8 Family computers ~~versus~~ time

24 Price ~~of DEC's 12-bit~~ PDP-8 Family computers versus time (linear plot)

25 Price per word of ~~DEC's 12-b.~~ memory versus time for ~~DEC's 12-bit~~ PDP-8 Family computers

26 Price versus performance of ~~DEC's 12-bit~~ PDP-8 Family Computers

27 Bits accessed by the processor/second/$ versus time for ~~DEC's 12-bit~~ PDP-8 Family computers.

29 Power, weight and volume for ~~DEC's~~ ~~12 bit~~ PDP-8 Family computer versus time

28 Evolution of PDP-8 Family PMS structures

represents a late second generation designs and the 8/I (and 8/L) are beginning third generation designs. These 3 or 4 machines evolved rapidly from 1963 to 1968. With the 8/L, a slower evolution is experienced from 1968 to 1977 as medium scale integrated circuits are used as the implementation technology. In actual practice, the bus width constrains the designs to be basically evolutionary.

Insert Attached X

The CMOS-8 avoids the wide bus problem by moving the bus to lines on the computer printed circuit board. A complete 4 Kword PDP-8 only occupies ~~8~~ sq. inches of board area. Note, that the VT78 is less cost-effective than an 8/A, because of the factor of three slower performance.

Figures 23 and 24 also show the oscillatory stairstep effect that occurs with new design approaches:

1. The 5 was predicated on minimum price, whereas the classical PDP-8 ~~went to~~ had additions to be more cost effective while not increasing price significantly over a slower speed design. The PDP-8 had a performance of 6 times the PDP-5 -- and crosses three lines of constant price/performance.

2. The PDP-8/S was ~~a response~~ an attempt to achieve truly minimum price by serial implementation technology and a minimum price memory design. Since the performance of the 8/S was so terribly slow, the pressure may have helped induce a constant price, integrated circuit version, the 8/I.

3. Since the 8/I was relatively expensive, both to PDP-8 and the 8/S the 8/L was introduced quickly as a cost reduced version to bring it in line with market needs and expectations.

4. The 8/E was then introduced as a high performance, large system machine enabling larger (than 8/L) systems to be built. The 8/M were cost reduced, smaller cabinet versions needed for the OEM market.

5. The 8/A was introduced as a further cost reduction to the 8/E and 8/M. Recent, higher priced, larger versions of the 8/A have been built for the configurations usually served by the 8/E. Also since the 8/A doesn't entirely replace the 8/E as a systems machine, the 8/E has necessarily remained in production.

6. In 1976, Intersil and Harris introduced one chip, CMOS PDP-8 processors as true mini(mal) computers at minimal costs.

Figure PWV which presents the power, weight and volume of the 12-bit machines ~~also~~ shows the oscillating design styles. In general, the power has remained relatively constant, although the CMOS-8 design drastically cuts the power needs for the VT78 (the power is also less because there are limited options). Constant power is needed in the post 1970 designs because the package must house a fixed number of devices and each device has a relatively high overhead power cost associated with driving the Omnibus.

Likewise the volume is relatively constant for the Omnibus designs. The weight and volume have declined significantly with time as the design has moved from 2 cabinets, to 1/2 cabinet and finally to being embedded in a CRT terminal.

– X insert –

The computers have changed ~~quite a lot~~ significantly in physical strucutre ~~thand~~ and this can be seen from PMS disrams their (Fig, 28.).

The ~~first orisinal as family of nesative (PDP-5 to logic~~ nesative and positive logic machines (through the 8/L) have had a relatively centralized strucutre with three busses to ~~distribute information for cont~~ interfacing to memory, prosrammed controlled I/O devices; and to direct memory access ~~devto~~ I/O and mass storage devices. and a seperate cable is connected to the console.

The 8 Omnibus family bundles ~~all~~ these connections tosether into a smaller common set of signals ~~connections~~ by ~~and uses a vast ssi sisnficant amount of time~~ multiplexing ~~of~~ data transmission.

The VT78 reverst back to a pre-PDP-8, non-modular ~~much earlier type~~ structure ~~than even the PDP-8 and~~ ~~(pre-PD pre-PDP-8~~ by bounding the permissible structures, ~~by having~~ only 5 well-defined ports for interconnection ~~to~~ & leave the ~~co~~ & CRT-housed computer

The CMOS-8 processor-on-a-chip based strucutre is quite similar to the Omnibus and Unibus ~~strucutres except that a significant few number of signals are used for the intrconnect~~. (i.e. the number is reduced to about 25

The number of interconnection signals on the bus is reduced to about 25, or ~~about~~ roughly by roughly a factor of four, the same number of pins ~~that are~~ available on the chip, ~~for interconnection.~~

The 8/E, 8/M and 8/A

The ~~P~~ 8/E was introduced in 1971, ~~as a means to drastically lower the cost of building systems and to by making the interface to a number all devices including Direct Memory Access devices to be straightforward.~~ While extending the idea of the I/O Bus of PDP-8, it ~~utilized the ideas within the PB that appeared in~~ has the basic topology of the Omnibus and extended them by making it possible for a ~~on the other hand it has some limitations that are not presnet in the Unibus.~~ (and simpler) is extended to intra processor dialogues and small with respect to device control. A block diagram of a complete PDP-8/E computer system is shown in Fig. PMS8E ~~in~~ showing all ~~the options ava its~~ of its available options. Notice in the lower half of the diagrams, an adapter exists for interconnecting ~~8~~ 8/I and 8/L DMA I/O devices.

Phsically, ~~the~~ an 8/~~E~~ can hold up to ~~cards?~~ 38 printed circuit cards (modules) ~~in~~ while the ~~8/M~~ 8/M holds only 20 cards. ~~The~~ ten processor ~~takses occupes~~ is mounted on ~~5~~ three modules, with a ~~4the~~ fourth terminator module. The console requires ~~1 mod~~ one module and a 4 Kword core memory ~~take~~ takes up ~~3~~ three slots, ~~Thus,~~ giving an eight 8 x 10 modules (or 640 sq. in.). Contast this size ~~of about~~ to the ~~PDP~~ 18 bit computers which were ~~rarely~~ implemented in about 5000 sq. inches. ~~requires 8 slots.~~ ~~The Omnibus~~

~~includes 48~~, of which ~~48~~ 96 are defined as ~~a~~ Omnibus signals.

& The Omnibus ~~takes up~~ has 144 pins ~~has 72 signals on the printed circuit~~ connector which is used to thread bussed to all modules. ~~M~~

These ~~a~~ large number of signals enable ~~all the~~ much ~~E~~ of the communication links that exist ~~in hte Unibus~~ within a computer to take place. The reader should contrast ~~the two~~ this scheme with the Unibus scheme~~s~~. Basically the signals can be grouped as follows:

1. Master timing to all ~~devices~~ components ~~Internal Processor to Memory & Console~~
2. Processor state to Console ~~a~~ ~~Here a~~ console can continuously interrogate ~~the site~~ and update of the processor
3. Processsor request to memory for instructions and data.
4. Processor to I/O Device commands - the processor issue IO Transfer ~~(transmit)~~ (IOT) instructions to move data between the AC and ~~Proces~~ an I/O device ~~Direct~~
5. I/O device to processor signallings ~~for~~ completion (Program Interrupts)
6. I/O Direct Memory Access control, ~~Here~~ the processor takes on the ~~funtion~~ of controlling and the update of memory for 3 cycle memory transferes ~~are effected.~~

Unlike the Unibus, the processor ~~one~~ handles much more of the control when DMA and ~~in~~ interrupts ~~are given.....~~ requests are made. If we only are concerned with the signals betwen the computer and I/O devices, ~~then only nb~~ for programmed control, then ~~onlys bout~~ approximately 30 signals are involved. Adding DMA capability requires ~~15 address lines, and 24 data laines, ine additon to several more control signals.~~ 50 more signals. These are the same signals used in the earlier PDP-8 implementations

~~sdalja~~

~~The motivation for the structure of the 8/M was simply to~~

Since there are no pass-through ~~option~~ signals on the bus, any module can occupy any slot — simplifying ~~testing~~ configuration, design and testing.

for cost reasons over the 8/I, ~~after the PDP-11, as a PDP-8 with~~ a lower cost PDP-8/L. ~~Since the 8/L had been comprimised in cost,~~ the 8/E in many respects was larger and faster. in response to a market pressure for more capabilities.

replacement to the

Although the ~~a~~ evolved PDP-8 I/O Bus was straightforward, the mechanical packaging and cabling was not. The DMA was not bussed, hence any new bus scheme had to address this problem and be more straightforward.

A by-product (or perhaps goal) of the Omnibus structure is that there are a fixed number of slots and ~~x~~ well defined ~~ch~~ electrical transmission characteristics, ~~(the environment) for eq~~ since there is no cabling between options.

The 8/E implementation is ~~reasonably well structured~~ mostly determined (see IMPBE) by the availability of ~~entegr~~ Integrated circuits. Here, multiplexors, ~~and~~ register files and a basic arithmetic logic unit ~~chip operate~~ the early PDP-8's perform Read only memories in 1970's were only about 64 bits, basic operations is all that's needed. ~~At the fist implo~~ The 8/E processor occupies ~~four~~ three ~~4~~ 3 printed circuit boards (or about 240 ~~320~~ ~~sq in~~ square inches) ~~.~~ Contrast with the original PDP-5 which occupied about 100 boards for a total of 2100 ~~sq~~ square inches.

~~No~~

thus a microprogram control is not feasible.

The problem of partitioning the logic and assigning it to one of three modules arises in the 8/E. Figure IMP8E shows that the clock, timing and interrupt on one module, the data path on a second and the control on the last module. By this partitioning, more pins are required between the data and control modules. This is accomplished by placing additional connectors on the back edge of the modules.

Pierre Lamond

566-4412

NAT.

May 25, 77

M

Dir. of IC Ops @ Fairchild

→ V.P. / Gen. Mgr. of Semis.

67-74 (Mfg. + Eng. of Semis)

- Calculator
- Supermarket tend

Finance, Marketing

Jan 74. — Coherent (Lasers / Optics) ←

10% - 15%

Fall 76, Early 76 Sprague / Advent

→ Audio — breakeven, turned it around, 7% pretax.

→ L.S.T.V. — Launch a new product is out of proportion.

Disagree

- Mgmt posit., NOT STAFF.
- Line Mgmt: Experience

~~We have a s~~

. Timedeadline

By Jan 15 Some ideas

Av. Tomm.

. Friday

In 1975 the 8/A w was introduced to extend the 8/E family ~~down~~ to a lower price threshold and ~~did~~ provide ~~new~~ additional memory capability particularly by utilizing semiconductor memory chips. The module size was extended to the DEC hex size (8" by 15"), enabling somewhat larger options to fit on a single module.

In the first implemntation, using 1 Kword semiconductor chips, ~~the~~ only 48 chips were needed to form the 4 Kword memory . While this size memory ~~would~~ easily fit ~~within a single~~ on a single quad sized module (or less) the 8 Kword and 16 Kword core memories are ~~contained in~~ also able to fit. In 1977, using 16 Kword chips a ~~regular~~ memory of 128 Kwords ~~Several hundred~~ only requires 12 x 8 or 96 chips

Thus,
~~The most significant reason for~~ the 8/A ~~was that the~~ processor ~~on a~~ could occupy a single module. Figure IMP 8A shows the implemntation of ~~the~~ the processor. ~~Note that other than~~ The register strucutre is virtually ~~an~~ the same as ~~the 8?~~ 8/m, while the control is a form of microprogramminns using read only memories. This reduces the processor board area from 240 sq. inch to 120 sq. inch.

With greater use of semiconductor memory, especially read only memories, a scheme was devised and added to the 8/A so that ~~read~~ programs written for read-write memory could be run in read-only memory. The scheme adds a 13th bit to the read only memory and signifies that a particular cell is actually a cell that is both read and written. When the processor detects the 13th bit, the other 12 bits point to a location in some read write memory which acts to hold the variable information. This provides ~~an effectively~~ an indirect memory reference.

Price Price

Price

Chapter 5

The Twelve Bit Machines

The Linc

The Laboratory INstrument Computer (LINC) computer was one of the machines that had a great influence on the design of the PDP-4 and the PDP-5, as indicated in Chapter 4. Thus a discussion of the DEC 12-bit machines must start with the LINC.

The LINC (Clark and Molnar, 1964) was designed at the M.I.T. Lincoln Laboratory in collaboration with the Communication Biophysics group of the M.I.T. Research Laboratory for Electronics. While its contribution to DEC history is primarily that of a PDP-4/PDP-5 forerunner, a number of other ideas came from that machine. The LINC tape unit and system ideas that permitted a user to have personal files were later carried over directly into the DECtape design and programs. This device made possible at a reasonable price the first complete personal computer for the user, in this case the researcher. The original LINC had been made from DEC Systems Modules, which was convenient when DEC went on to manufactured LINC machines directly from the Lincoln Laboratory design (Figure 2c). Subsequently, Wes Clark with Dick Clayton designed the LINC-8

~~(see Fig. LINC-8)~~, a two-processor machine (LINC + PDP-8) which executes both ISP's in parallel. Later, Dick Clayton designed the PDP-12 ~~(see Fig. PDP-12)~~, a single physical processor that executed either PDP-8 or LINC instructions sequentially by switching modes.

Some of the characteristics of the LINC Family machines are given in Table ~~00~~ 1. Note that the size remained essentially constant at one cabinet over the computer's life.

Table 1 **LINC Family Characteristics**

	LINC	LINC-8	PDP-12
Project start	1961	Summer 1965	6/67
First ship	3/62 (demo)	8/66	6/69
Withdrawn	12/69	12/69	6/75
No. Produced	50(21 by DEC)	143	1000
Price (min.)	$43,600	$38,500	$28,100
Goals/features	complete system for laboratory user (including file system and CRT ~~s~~)	cost, speed and PDP-8 software/hardware compatibility, ~~use of PDP-8 components~~	larger scope, bus compatible with 8/I

Size	69 x 32 x 30 plus separate tape, keyboard, and console	69 x 32 x 33	76 x 35 x 33
Mp-Pc accesses per sec, (Mega)	0.125 K 6	0.67 667 K PDP-8 memory	0.67 667 K 8/I memory
Weight	?	?	700
Power (watts)	1000	2000	< 2000
CRT	2 - scopes 1 - Oscilloscope, (oscilloscopes) originally 2 (originally)	2 - scopes 1 - Oscilloscope (oscilloscopes) originally 2 oscilloscopes; later only 1.	1 - scope 7"x 9" CRT (7" x 9")

The systems aspect of LINC (circa 1962) were really quite significantly ahead of their time and ahead of other systems including the timesharing work in terms of providing a clean, simple, yet powerful console (two oscilloscope-size CRTs) and file system interface (Fig. 00). The DECtape library system came two years later on PDP-5. On machines prior to the LINC, DEC had been stressing design flexibility and modularity, and had been providing many ways for interconnecting computer components in order to have any type of structure. This detracted from having a base system configuration complete with software. In contrast, the LINC was quite constrained (only 1 Kword or 2 Kwords of primary memory were available) with two LINC tapes and two CRTs. Almost all of the other characteristics including the I/O interface were also quite constrained. By bounding the system in this way, it was possible to provide a complete environment including software and provide for convenient interchange of user software.

The PDP-5

As indicated in Chapter 5, discussions with Foxboro Corporation in the fall of 1961 led to the design, using many LINC ideas, of a 12-bit Digital Controller called the DC-12. Instead of building the DC-12, DEC built the PDP-4 and sold one to Atomic Energy Limited of Canada. As they used it for a reactor control computer system at Chalk River, an elaborate analog monitoring system was needed as a front end for the PDP-4. To reduce the complexity of the analog system, a special front end computer was needed. The 10-bit L-1 was briefly considered but was soon rejected because the encoded analog values required longer words and because the program size and program complexity seemed too great for such a small computer. After visiting Chalk River in the winter of 1962, DEC engineers decided that a 12-bit design based on the DC-12 would be excellent for such a PDP-4 process control front end application. The ISP for the new machine, the PDP-5, was specified in detail by Alan Kotok and Gordon Bell, and the logic design was carried out by Edson DeCastro, the applications engineer responsible for building the analog front end at Chalk River.

The intent of the design was simplicity so that it would take no longer to design the PDP-5 than it had taken to design the analog front end that it would be replacing. The machine used

the standard modules that were developed for the PDP-4, including the concept of bit-slice construction for the accumulator, memory address, and memory buffer registers. The analog nature of the initial application was addressed by building an analog to digital converter into the accumulator, thus providing this capability at extremely low cost. The other part of the design that addressed cost was the use of an I/O bus instead of the radial structure that had been used in the 18-bit designs. The I/O bus permitted equipment options to be added incrementally from a zero base instead of having the pre-allocated space, wiring, and cable drivers that characterized the radial structure. This lowered the entry cost of the system and simplified the later reconfiguring of machines in the field.

Although the design was optimized around the 4 Kword memory, the PDP-5s ultimately evolved to 32 Kword configurations using a memory extension unit. Similarly, although the base machine design did not include built in multiply and divide functions, these were added later in the form of an Extended Arithmetic Element. ~~although~~ While the PDP-5 was designed for control, the aspirations for it to be used generally in a system can be clearly seen in the early photographs (Fig ___).

The PDP-8

While the PDP-5 had been a reasonably successful computer, it

soon became evident that a new machine with far greater performance was required. A new series of modules, the Flip Chip series, was being developed for the PDP-7 and for the new version of the PDP-5. The new logic promised a substantial speed improvement, and new memory technology was becoming available that would permit the memory cycle time to be shortened from the 6 microseconds of the PDP-5 to 1.6 microseconds for the new machine. In addition, the cost of logic was now low enough so that the program counter could be moved from memory to a separate register, substantially reducing instruction execution times. The new machine was called the PDP-8 (see fig ——)

In a fashion similar to the technical developments that marked the eighteen bit family, the new twelve bit machine was phsyically smaller than its predecessor. This time, however, the change was more than simply a change from three cabinets to two or from two cabinets to one. It was a change from one cabinet to a half cabinet. The new small size meant that the PDP-8 was the first true minicomputer. It could be placed on top of a lab bench or built into equipment. It was this latter property that was the most important, as it laid the groundwork for the formation of the OEM (Original Equipment Manufacturer) purchase of computers to be integrated into total systems sold by the OEM.

The improvements in logic density permitted by the new Flip Chip modules also influenced the packaging and the manufacturing

methods. The PDP-8 logic modules were mounted in connector blocks, which were in turn mounted in frames. The two frames were each the maximum size that could be accommodated in the new Gardner-Denver automatic wire-wrapping machine. Automatic wire-wrapping was very important to the mass production success of the PDP-8, as it was fast and accurate. The two wire-wrapped frames hung vertically and were hinged about a vertical axis at the rear of the computer cabinet. In some ways they resembled the pages of a book, with the wire-wrap pins on the surfaces that faced each other. The use of this swinging gate backplane, similar to that on the PDP-9, permitted access by maintenance personnel to both the connection pins and the modules.

Like its predecessor, the PDP-5, the PDP-8 was a single-address 12-bit computer designed for task environments with minimum arithmetic computing and small primary memory requirements. Typical of these environments were process control applications and laboratory applications such as controlling pulse height analyzers and spectrum analyzers.

In addition to the envisioned applications, the PDP-8 was used for innumerable other applications. One of the most interesting was in message switching. The PDP-8 message switching hardware assembled characters by bit sampling, checking the status of teleprinter lines at five times the anticipated bit rate in order to accurately recover the data. Another interesting application was the TSS/8 small-scale general purpose

time-sharing system developed by Carnegie-Mellon University and DEC (reference van de Goor, et al, 1969). [Insert 8A]

The PDP-8 was the first of the "8 Family", a designation which will be used in this book to include all machines containing the letters PDP and the digit 8. A subset, called "Omnibus 8" machines, will be introduced later when the PDP-8/E, PDP-8/M, and PDP-8/A machines are discussed.

The PDP-8, which was first shipped in April 1965, and the other 8 Family machines that followed it, achieved a production status formerly reserved for IBM computers, as over 50,000 machines have been produced, excluding the CMOS-8 based computers. 40,000 During the fifteen years that these machines have been produced, logic cost per function decreased by orders of magnitude, permitting the cost of entire systems to be reduced by a factor of ten. Thus the 8 Family offers a rare opportunity of study the effect of technology on implementations of the same ISP.

The PDP-8 was followed in late 1966 by the PDP-8/S, (see Fig.) a cost reduced version. The 8/S was truly small in size, being scarcely larger than a file cabinet drawer. It achieved its low cost by implementing the 8 ISP in serial fashion. This did reduce the cost, but it so radically reduced the performance that the machine was not a good seller.

In 1968, the PDP-8/I (see Fig —) was produced, using MSI integrated circuits

8A

~~TSS/8[1], the timesharing system for a PDP-8 was an interesting and influencial product (Bell/van der Goor 19xx).~~ While only a hundred or so systems were sold, TSS/8 is ~~significant because it established the notion that multiprogramming even applies to minicomputers, a notion recently~~ the lowest cost (per system and per user) and highest performance/cost timesharing system ~~DEC provides. The reason for the lack of commercial success, may be simply the fact that, as 11 came along, the development and marketing efforts were redirected. On the other hand, TSS/8 required some understanding, and the time required to sell such a system may be greater than that allowed (i.e., it may be underpriced in terms of software complexity).~~ A major side benefit of TSS/8 was the training of ~~one of~~ the implementors, ~~Nate Teichholtz,~~ who went on to ~~propose and first~~ implement the RSTS timesharing system for PDP-11 based on the BASIC language.

[1]TSS/8 was designed at Carnegie-Mellon University, with graduate student Adrian van der Goor, ~~(Reference?)~~ in reaction to the cost, performance, reliability, and complexity of IBM's TSS/360 (Model 67). Although TSS/360 eventually worked, it was not marketed but contributed some ideas and trained thousands of programmers for IBM. At CMU, a TSS/8 operated until 1974 when the special swapping disk expired. The cost per user or per job tended to be about 1/20 of the TSS/360 system they ~~run.~~ ran.

Computer aided design of digital systems at the register transfer (RT) level has been a subject of research for some time. Major work in this area has been done by Barbacci (3) at Carnegie-Mellon University. Figure 1 gives a block diagram of a design process at the RT level as conceived by Barbacci (3). This paper addresses one aspect (or one block) in that design process; namely, the problem of the physical allocation in RT level design. The problem is addressed such that the solutions are easily implementable as an algorithm on a computer and may form a sub-part of the complete design process shown in Figure 1.

A MODEL FOR THE DESIGN TASK

The digital system design task usually starts with a functional specification of some computational activity. The task of the designer is to create a structure - a digital control and a data part - that carries out the computational activity. The design of the control and the data part is a complex activity and may be performed in various ways.

In the data part design task there are several subtasks, which themselves should be studied first in isolation. In this paper one such subtask is considered; namely, the allocation of data operations to data operators. To enable us to study this subtask in isolation, we have assumed that the other aspects of the system design are already bound. This binding is conveyed to the data part designer by way of a control program.

A control program (CP) is defined to be a flowchart specification of the computational activity with the following properties:

1. The variables that are manipulated in the computational activity have already been assigned to registers in the CP, but there has been no binding of data operations to data operators.
2. The sequence of operations is fixed by the CP. In this paper we limit ourselves to CP's that have no concurrently active paths, i.e., serial control programs. Methods for generating and evaluating a variety of serial and concurrent CP's for a computational activity, given an initial CP, are reported by Barbacci (3).

The data operator allocation task is now defined as follows: Given the registers and the data operator types designated by the control program, postulate for the data part of the system a number of physical data opeartors of each type and allocate specific operations from the CP to specific operators of the given

to implement the PDP-8 ISP with better performance than the 8, and at 2/3 the price. For those customers wishing a package with less option mounting space, but still with good performance, the PDP-8/L (see Fig. —) was introduced later in that same year.

The PDP-8/S, PDP-8/I, and PDP-8/L are mentioned only briefly here because their characteristics were basically dictated by the cost and performance improvements made possible by the emerging integrated circuit technology. The cost and performance figures for these machines are examined in greater detail in the charts at the end of this chapter.

The 8/E, 8/M, and 8/A

Shortly after the introduction of the PDP-8/L, it became evident that customers wanted a faster and more expandable machine. The continuing technological trend toward higher density logic and some new concepts in packaging made it possible to satisfy both of these requirements but to still produce a new machine that would be cheaper than its predecessor. The new machine was the PDP-8/E (see fig. —)

A block diagram of a complete PDP-8/E computer system is shown in Figure 16. Note that the lower half of the drawing shows an

adapter for interconnecting the positive ~~logic~~ bus family (8/I and 8/L) I/O devices. In addition, signal converters were available to convert a step further to the older negative ~~logic~~ bus family (5,8,and 8/s) I/O devices. In this way, the new machine could capitalize on the existing hardware option base. It would not be necessary to design a complete new set of options at the time of machine introduction and existing customers could upgrade to the new computer without needing to buy new peripherals.

The reason for using an adapter to connect to existing I/O devices was that the PDP-8/E featured a new unified-bus I/O bus implementation derived from the Unibus that was being designed for the PDP-11. The electrical design of the I/O bus for both the previous negative logic and positive ~~logic~~ bus machines had been straightforward, but the mechanical packaging and cabling had not. A new implementation was needed which would simplify the packaging and cabling and solve the problems created by the direct memory access channel, which had not been bussed in previous designs. Don White, who was leading the design team, conducted a contest to name the new bus. After discarding such entries as "Blunderbus", the name Omnibus was chosen.

The Omnibus, which is still in use in the PDP-8/A, has 144 pins, of which 96 are defined as Omnibus signals. The remainder are power and ground. The large number of signals permit a great number of the intraprocessor communications links as well as the I/O signals to be accommodated. The Omnibus signals can be

grouped as follows:

a) Master timing to all components

b) Processor state information to the console

c) Processor request to memory for instructions and data

d) Processor to I/O device commands and data transfer

e) I/O device to processor signalling completion (interrupts)

f) I/O direct memory access control for both direct and three cycle data break transfers

The approximately thirty signals in groups d) and e) above provide programmed I/O capability. There are about fifty signals in group f) to provide the DMA capability. These eighty signals are nearly equivalent in quantity and function to the preceding PDP-8 I/O bus design making the conversion from Omnibus structure to PDP-8/I and PDP-8/L I/O equipment very simple.

The complement of signals is quite different from the PDP-11 Unibus, which is more strictly an I/O bus, and the PDP-8/E processor handled many more of the DMA and interrupt control functions than does the PDP-11 processor. One of the specific signalling structures that differs between the two machines is

the interrupt system, which in a PDP-11 Unibus passes a bus grant signal through the I/O options to be propagated further or absorbed by the option. There are no such pass-through signals on the Omnibus, hence any option can occupy any slot and intervening slots between installed options can be left vacant. A by-product (or perhaps goal) of the Omnibus structure is that there are a fixed number of slots and the lack of cabling between options causes there to be well defined electrical transmission characteristics.

The processor for the PDP-8/E occupied three 8 inch by 10 inch boards; 4 Kwords of core memory took up three more boards; a terminator board and the console board completed the minimum system configuration. Thus a total of eight 8 by 10 boards formed a complete system. The three board PDP-8/E processor, occupying 240 square inches, was a striking contrast to the 100 board PDP-5 processor, which occupied 2100 square inches.

The PDP-8/E implementation was determined by the availability of integrated circuits. As shown in Figure 17-9, multiplexors, register files, and basic arithmetic logic units performed the basic operations in a straightforward fashion using a simple sequential controller. Microprogrammed control was not feasible because suitable read only memories were not available. The read only rope magnetic memory of the PDP-9 was too expensive and was unsuitable for PDP-8/E packaging. Integrated circuit read only memories available at that time were too small, only

holding about 64 bits.

There was some problem partioning the processor logic amongst the three modules. Figure 17 shows the final arrangement, which was to place timing and interrupt on one module, the data path on a second, and the control on the third. Even with this partioning, more pins were required between the data and control modules than were available through the Omnibus. To provide the necessary connections, additional connectors were installed on each module on the edge opposite the Omnibus connections.

The PDP-8/E was mounted in a chassis which had space and power to accommodate two blocks of Omnibus slots. Thirty eight modules could be mounted in the slots, allowing space for the processor and almost thirty peripheral option controllers. Many customers wanted to build the PDP-8/E into small cabinets and have it control only a few things. They found the large chassis and its associated price to be more than they wanted. To reach this market, the PDP-8/M was designed.

The PDP-8/M was essentially a PDP-8/E that had been cut in half. The cabinet had half the depth of a PDP-8/E and the power supply was half as big. There were 18 slots available, enough for the processor and about ten options. The processor was the same as that for a PDP-8/E.

By 1975, DEC had been building "hex" size printed circuit boards

for the PDP-11/05 and PDP-11/40 for at least two years. The "hex" boards were 8 inches by 15 inches, half again as big as the "quad" boards used in the PDP-8/E and PDP-8/M, which were 8 inches by 10 inches. The dimensional difference was along the contact side of the board. A "hex" board had six sets of 36 contacts while the "quad" board had only four sets. Semiconductor memory chips had also become available, so a new machine was designed to utilize the larger boards and new memories to extend the PDP-8/E, PDP-8/M family to a new lower price range. The new machine was the PDP-8/A. The PDP-8/A processor and register transfer diagram is shown in Figure 18. (see Fig 8/A.1)

The hex modules permitted some of the peripheral controller options that had occupied several quad boards in the PDP-8/E to fit on a single board in the PDP-8/A. (see Fig 8/A.2) The availability of hex boards and the availability of larger semiconductor read-only memories permitted the PDP-8/A processor to use microprogrammed control and fit onto a single board. It should be noted here that when a logic system occupies more than one board, a lot of space on each board is used by etch runs going to the connectors. This was particularly true with the PDP-8/E and PDP-8/M processor boards due to the contacts on two edges of the boards. When an option is condensed to a single board, more space becomes available than square inch comparisons would at first indicate because many of the etch lines to the contacts are no longer required.

The first PDP-8/A semiconductor memory offering took only 48

chips (1 kbits each) to implement 4 Kwords of memory. Memories of 8 Kwords and 16 Kwords were also offered. In 1977, only 96 16Kbit chips were needed to offer a 128 Kword memory. With greater use of semiconductor memory, especially read-only memories, a scheme was devised and added to the 8/A which would permit programs written for read/write memory to be run in read-only memory. The scheme adds a thirteenth bit to the read-only memory to signify that a particular location is actually a location that is both read and written. When the processor detects the assertion of the thirteenth bit, the processor uses the other twelve bits to address a location in some read/write memory which holds the variable information. This effectively provides an indirect memory reference.

In 1976, an option to improve the speed of floating point computation was added to the PDP-8/A. This option, the FPP8/A, is a single accumulator floating point processor occuping two hex boards and compatible with the floating point processor for the PDP-12, the FPP12/A. It supports 3 or 6 word floating point arithmetic (12-bit exponent and 24 or 60-bit fraction) and 2 word double precision 24-bit arithmetic. As a completely independent processor with its own ISP, it has its own program counter and eight index registers. The performance, approxiamately equal to an IBM 360 Model 40, provides what is probably the highest performance/cost ratio of any computer.

More "Omnibus 8" computers (PDP-8/E, PDP-8/M, PDP-8/A) computers

have been constructed than any of the previous models. The high volume appears to be due to the basic simplicity of the design, together with the ability of the user to easily build rather arbitrary system configurations.

In the Fall of 1972, DEC began the design of a single chip P-channel MOS processor to execute the PDP-8 instruction set. This processor would be called the PDP-8/B and it was hoped that production chips could be obtained by the Spring of 1974 for production computer systems to be shipped in the Fall of 1974. The designers had progressed through the design tradeoffs in partitioning an 8 for a single 40-pin chip when the project was stopped in the Summer of 1973. The key reasons for stopping the design included the industry trend from P-channel to N-channel and the fact that the Omnibus did not lend itself to cost reductions with LSI technology. While the Omnibus was ideal for MSI and ease of interfacing, it was not as cost effective as the time division multiplexed (address and data on the same leads at different times), shorter busses that microcomputers used. The percentage of system cost and complexity represented by the processor in an Omnibus 8 system was too low to make the move to as LSI processor attractive at that time. For these reasons, it was decided to apply the newer N-channel LSI process to a system where the processor was a more complex and costly part of the system - the PDP-11 family. Thus, in the Summer of 1973, a project was started in cooperation with Western Digital Corporation to build a PDP-11 on one or more N-channel LSI

chips.

In 197x, Intersil designed the first PDP-8 processor to occupy a single chip, using Complementary Metal Oxide on Silicon (CMOS) technology. DEC verified that it was a PDP-8 in 197x and began to apply it to a product in the Fall of 1976. In the meantime, Harris Semiconductor became a second source to Intersil to supply the chips. The two manufacturers each have their own designation for these chips, but in the discussion below they will be called "CMOS-8" chips.

The CMOS-8 processor block diagram is given in Figure 19. Not surprisingly, it looks very much like a conventional PDP-8/E processor design using integrated circuits. It has a common data path for manipulating the PC (Program Counter), MA (Memory Address), MQ (Multiplier/Quotient), AC (Accumulator) and Temp (Temporary) registers. The IR (Instruction Register), however, does not share the common arithmetic logic unit. Register transfers, including those to the "outside world", are controlled by a PLA (Programmed Logic Array), as indicated by the dotted lines in the figure. The PLA is a form of read only memory, but differs from a conventional ROM in that the outputs are based on boolean logical (combinatorial logic) relationships amongst the inputs rather than being simply particular output data in response to particular addresses as inputs. They are implementations of sparse memories (memories with little data spread over many addresses), but can be built more economically

than ROM implementations of the same combinatorial logic function. A microphotograph of the CMOS-8 is given in Fig. —.

While the CMOS-8 is the first DEC processor to be built on a single chip, the most interesting thing about it is the systems configurations that it makes possible. It is not only small in size (a single 40 pin chip), but it also has miniscule power requirements due to its CMOS construction. Thus, some very compact systems can be built using it. Figure 20 shows a system built with a CMOS-8 and CMOS-8 compatible components as it is packaged in the base of the VT52 CRT, forming the VT78 intelligent terminal. In contrast to past systems, some of the other components in the system now represent more dollar cost and more physical space than the processor. Among these are the RAM (random access read/write memory), the ROM (read only memory), and the Parallel Interface Elements associated with the I/O devices. The Parallel Interface Elements enable interrupt signals to be sent back to the processor and detect the IOT (Input/Output Transfer) commands that control data transfers. Also shown in the figure are some specific I/O devices such as the UART (Universal Asynchronous Receiver Transmitter) chips that do serial/parallel conversions and formatting for communication lines.

An excellent example of the use of a CMOS-8 as part of a packaged system is the VT78. The goals for the VT78 are to drastically reduce costs by including keyboard, CRT, and processor in a single package the size of an ordinary terminal. The CMOS-8 chip and high density RAM chips make this possible.

To form a complete stand alone computer system that supports five terminals, mass storage is added. Because the mass storage is floppy disks, the mass storage is not in the terminal, but rather is in a small cabinet. Even without the mass storage, however, the VT78 forms an "intelligent terminal". An intelligent terminal is usually difined to include a computer whose program can be loaded (usually via a communication line) such that it can take on a variety of characteristics - i.e. it can learn.

An intelligent terminal can either be used as part of a network or as a stand alone computer system. In the former case, the application is determined by the network to which the terminal is attached, but in the latter case, the terminal functions as a desktop computer running various PDP-8 software. An example of the latter use is the WP78 word processing system which permits a user to do editing, do justification, and recall a store of standard forms in addition to other word processing functions.

Figure 21 shows a block diagram of a VT78 computer system. Note that the structure is precisely the antithesis of most of the systems previously described because it is not modular with open ended busses, but rather is tightly bounded. A single VT52 terminal is provided and the memory is limited to 16K words. There is an interface provided that permits 15,000 12-bit words per second to be transferred to a parallel device such as a printer. A special read only memory is used to provide the basic

functionality for the computer and to carry out self diagnostics prior to program loading. Program loading is done by a serial ROM-controlled port called the Program Injector. This permits the VT78 to be loaded with a program oriented to a particular application.

The PDP-8 has seen the new technology over a period of fifteen years. The characteristics of these implementations are given in Figure 23. New technology can be utilized in the computer industry in three ways. These ways are a) lower cost implementations at constant performance and functionality, b) higher performance implementations at constant cost, and c) implementation of new basic structures. Of these three ways, the PDP-8 family has primarily used lower cost implementations of constant performance and functionality.

The points in Figures 23 & 24 are arranged to show the cost trends of three configurations. The first configuration is merely a CPU with 4 Kwords of primary memory. The second configuration adds a console terminal, and the third configuration adds DECtapes or floppy disks for file storage. Note that the basic system represented in the first configuration has declined in price most rapidly. It declined 22% per year in the early days and 15% per year in more recent years. Primary memory, on the other

hand, has declined at a 20% per year rate, as seen in Figure 00.

The price and performance trajectories for PDP-8 family machines are plotted in Figure 26, with lines of constant price performance separated at factors of two. Note that the early implementations had significantly lower performance than the original PDP-8. Memory performance and instruction execution performance were directly related in all of these machines except the PDP-5 (which kept the PC in primary memory) and the PDP-8/S (which was a serial machine). Thus, with the design emphasis on lowering the cost with each new machine, performance continued to lag the PDP-8 until higher speed primary memory was available without paying a cost penalty. Other performance improvements, such as the addition of hardware floating point or the addition of a cache, are outside the scope of this comparative analysis.

Figure 27 gives the performance/price metric for the PDP-8 family machines, and it can be directly compared with other machines described in this book. The 18-bit machines improved at a rate of 52% to 69% per year over a short time span (indicated on the graph). Ignoring the PDP-5 design point, the improvement for the 12-bit machines was similar during the same time span, but has since been less radical with only a 22% per year improvement.

Rather than try to fit a single exponential to the performance /

price data points in Figure ~~xx~~ 21, it might be better to try two independent exponentials. The reason for this is that the data points really mark the transition between two generations. The PDP-5 was a mid second (transistor) generation machine, and the PDP-8 represents a late second generation machine. The 8/I and 8/L were beginning third (integrated circuit) generation designs. These four machines represent a relatively rapid evolution from 1963 to 1968. After the 8/L, the evolution slows somewhat during the period 1968 through 1977 as medium scale integrated circuits continued to be used as the implementation technology, and the packaging and connectivity costs continued to be controlled by the relatively wide bus structure.

During the evolution, the DEC 12-bit computers have significantly changed in physical structure, as can be seen from their PMS diagrams (Figure 28). The negative logic machines and the positive logic machines up through the 8/L had a relatively centralized structure with three busses to interface to memory, program controlled I/O devices, and to direct memory access I/O and mass storage devices. The Omnibus 8 machines bundled these connections together into simpler physical structure. The CMOS-8 avoids the wide bus problem by moving the bus to lines on a printed circuit board. The number of interconnection signals on the bus is reduced by roughly a factor of four, to about 25, so that the signals can be brought into and out of the chip within the number of pins available.

Figures 23 and 24 and Table 1 show the stairstep effect that occurs as a result of the design evolution summarized below:

a) (Improve the Performance) While the PDP-5 was predicated on mimimum price, the PDP-8 had additions to improve the performance while not increasing price significantly over a slower speed design. The cost per word was modestly higher with the PDP-8 than the PDP-5, but the PDP-8 had six times the perfomance of a PDP-5. Thus, the PDP-8 crosses three lines of constant price/performance in Figure 26.

b) (Lower the Price) The PDP-8/S was an attempt to achieve a truly minimum price by using serial logic and a minimum price memory design. However, the performance of the PDP-8/S was terribly slow.

c) (Improve the Performance) The market pressures created by 8/S performance probably caused the return to the PDP-8 design, but in an integrated circuit implementation, the PDP-8/I.

d) (Lower the Price) The PDP-8/I was relatively expensive, so the PDP-8/L was introduced quickly as a cost reduced version to bring the design into line with market needs and expectations.

e) (Improve the Performance) The PDP-8/E was introduced as a high performance machine that would permit systems larger than those possible with the 8/L to be built.

f) (Lower the Price) The PDP-8/M was a lower cost, smaller cabinet version of the PDP-8/E and was intended to meet the needs of the OEM market.

The stairstep process essentially ends at the PDP-8/M, as the subsequent machines have also been primarily price reductions. The PDP-8/A was introduced as a further cost reduction to the 8/E and 8/M, although some large system configurations are still built with PDP-8/E machines. In 1976 Intersil and Harris introduced the CMOS-8 chips as true mini(mal) computers (now called microcomputers). These represent a substantial cost reduction, but also a substantial performance reduction.
The CMOS-8 performance is 1/3 that of a PDP-8/A, so a stand alone system using a CMOS-8, such as the VT78, is less cost effective than an 8/A. when using cpu as the only performance criterion. How The main reason for using LSI is for ↑ less cost and the smaller package & the design permits
Obviously, the next step is increased performance. Another possibile next step is more memory on the same chip, or else both more performance and more memory.

In addition to showing the "design stairsteps", Figure 29 and Table 1 present the power, weight, and volume of the 12-bit machines. In general, the power requirements have remained relatively constant. This is both because each package must house a fixed number of devices and because each device has a relatively high overhead power cost associated with driving the

Omnibus. The limited configuration, lack of an Omnibus, and low power requirements of CMOS make the VT78 an exception to the constant power rule. The weight and volume have declined significantly with time as the design moved from two cabinets to 1/2 cabinet, and then from 1/2 cabinet to being imbedded in a CRT terminal.

Special PDP-8 Based Devices

The PDP-8/A and the products which incorporate the CMOS-8 chip, such as the VT78, are the current 12-bit product offerings, so the discussion of the development of DEC's 12-bit computers in chronological order must stop here. However, during the development of the main line of 12-bit computers, some interesting systems based on DEC 12-bit processors were developed, both by DEC and by others. Among these are the DEC 338 Display Computer, the Carnegie Mellon Cache-Based-8, and the PDP14 Programmable Controller (a 1-bit machine for handling Boolean equations, but with some ISP similarities to the PDP-8 and using Omnibus packaging concepts).

DEC 338 Display Computer

The 338 (shown in Fig. 0, page 00), a variant of the PDP-8, is included for its historical importance (see Chapter 25, Bell and Newell, 1971). It was one of the very earliest display processor based computers--if not possibly the first. The problem of displaying data on a CRT clearly shows how the application need drives a complete change in hardware in order to interpret the necessary data type (a graphic picture).

The 338 idea was extended and applied to PDP-9 (in the form of the 339), extended for the PDP-15 VT15 display processor, and also applied to the PDP-11 series of display processors. Although the 338 had the right general capabilities, it did not have the refinements that later display processors for PDP-15 and PDP-11 had (GT40 and GT60). Because the display processor is relatively expensive, the cost of the host processor should be adjusted accordingly: it should not be a performance bottleneck, nor need it be significantly cheaper than the display part.

An observation that display processors and other specialized processors evolve in a fashion called the wheel of reincarnation (Myer and Sutherland, 1968) is diagrammed in Figure 3. The figure shows how one starts with a very simple basic design (here to have graphics picture output for a computer). The trajectory around the wheel follows:

Position 1: Point-plotting. The computer includes a single

instruction display controller which can plot a picture on a point-by-point basis under command of the central processor. For most displays, except storage scopes, the processor can barely calculate the next point fast enough to keep the scope refreshed. Hence, the system is processor bound, and the display may be idle. The original PDP-1 Type 30 scope is typical of this position. A scope of this type is offered on most DEC minis.

Position 2: Vector-plotting. By adding the ability to plot lines (i.e., vectors), a single instruction to the display processor will free some of the processor and begin to keep all but the fastest scope busy.

Position 3: Character-plotting and alphanumeric. With the realization that characters are a major part of what is displayed, commands to display a character are added, further freeing the processor. Many of the point plotting displays were extended to have character generation plotting. The alphanumeric scopes have this capability and usually do not plot vectors; such scopes (e.g., VT50) are not integrated with the computer, but are connected via a serial communication link. Special scopes such as the VT8E and VT14 have been added to PDP-8 to carry out this function in a tightly coupled fashion.

Position 4: General figure and character display. In reality, a picture doesn't consist of just characters and vectors; each

element of the picture is actually a string of characters to be displayed at a particular point and a set of closed or open polygons to be displayed starting at a particular point. By providing the control display with a Direct Memory Access channel, the display can fetch each string of text and generate polygons with no central processor interaction.

Position 5: Display processors. Now with the ability to put up sub-pictures with no processor intervention, it is easy for the whole picture to be displayed by linking the elements together in some fashion. This merely requires jump and subroutine call instructions so that common picture elements don't have to be redefined. The 338 and other display processors have roughly this capability.

Position 6: Integrated display and central processor. Now, all the data paths and states are present for a fully general purpose processor so that the central processor need never be called on again. This requires a slightly more general purpose interpreter. There is full generality by minor perturbations, and the processor design can be refined in such a way as to execute the same ISP as the original host because the cost of incompatibility is too great. Two processors require two compilers, diagnostics, manuals, and support for use. This state provides the same capability as the initial starting point, Position 1. The original processor is completely free, and there is a display processor with the capability of

executing both the original ISP plus the display ISP.

Position 7: Two computer structures. Alternatively, the processor can be isolated as a separate computer and reconnected in some fashion to the original's central processor-primary memory pair. Such a structure is just a basic computer plus state 4.

THE CACHE-BASED PDP-8

This structure uses a small fast memory to hold the results of recent references to primary memory. The structure has been subsequently used in the KL10, 11/70 and 11/60 (see chapters 0, 0 and 0).

A PDP-8 with cache was designed and constructed by Professor David Casasent at CMU (Bell, Casasent and Bell, 1974; Bell and Casasent and Bell, 1971). Initially, the project was proposed as exploratory advanced development on the desirability and feasibility of using Emitter Coupled Logic (ECL) logic for designing fast computers (including PDP-10s). As the investigation proceeded, the need for a large, fast memory emerged. Such a memory turned out to be so costly that a computer so equipped could not be feasibly marketed. The easiest way to build a cost-effective, fast minicomputer turned out to be using a cache structure in order to reduce the cost of primary memory. Also, instead of

designing a very fast PDP-8, which ECL logic would have provided, the goal was modified to be less fast, much less costly, and potentially marketable. This caused TTL Schottky to be used in the design even though the logic family was just beginning to evolve.

In order to make the prototype more marketable without completely redesigning it, the project was constrained to use the 8/E Omnibus backplane and parts. The prototype did not become operational as quickly and cleanly as possible and was therefore not used to stimulate a market. Throughout the development of the design there was little interest in its marketability. The design was carried forward in order to test the cache applicability, circuitry, ease and techniques for building ~~a~~ faster computers. The difficulty in stimulating interest in marketing was typical of products that are substantially different from those already existing. Marketing people are often driven by the last sale that was lost. If a sale was most recently lost to a competitor that offered a purple whiz-bang, marketing people want a better purple whiz-bang developed as soon as possible. This attitude leads to the conclusion that if **no sales are being lost to very fast, non-existent minicomputers, therefore there is no market for very fast minicomputers.** Furthermore existing customers are of little or no help because, unless faced with the widescale availability of significantly faster machines, the applications can't be defined and emerge.

A number of things were learned from the research on the cache based 8. It was decided that a block size of 1 was adequate, given the PDP-8/E 1 word memory width constant. The design had 512 words with no separation of the cache between data and instructions. A 100 nanosecond PDP-8 processor with 512 cache had the characteristics shown in ~~the following table~~ Table 2.

Note that this width was also selected for the 11/60 as described in chapter 0.

and standard 8/E core memory

Table: Performance/Cost Comparison

Configuration	Costs			Performance Factor	Performance/Cost Ratio Compared to 8/E		
	Minimum	Average	Large		Minimum	Average	Large
8/E	$ 5K	$10K	$35K	1.0	1	1	1
8/E with cache	$10K	$15K	$40K	5.0	2.5	3.3	4.3

Note that the performance/cost ratio asymptotically approaches 5 as the system price increases. This argues for always incorporating ~~does~~ incremental performance improvements in the most expensive machines.

The cached-8 did influence the 11/45, although the designers rejected the cache notion (Chapter 00) because they believed that the cache concept did not apply to PDP-11 program behavior. Eventually Bill Strecker persevered and convinced other PDP-11 designers to build a cache-based larger scale minicomputer, and this became the 11/70. The work on the cached-8 illustrates the

use of the 8 as an experimental vehicle for understanding a design in terms of program behavior. It also allows one to observe the flow of ideas from research, through advanced development, and finally through to the production of machines. Finally, it shows how, by rather minor perturbations, a design can be kept compatible with its predecessors while providing rather dramatic performance and performance/cost ratio improvements as shown in Table 0.

The PDP-14

The PDP-14 was designed expressly as a controller for complex electromechanical machinery such as transfer machines, conveyors, and simple milling machines. The need for such a controller was first recognized when General Motors expressed their need to control a large machine which handled engine blocks by a sequence of operations (transfers). The design of a controller evolved from the use of standard DEC K-series industrial modules (see Chapter 00) for sequential circuits. The K-series modules provided increased reliability and replaced electromechanical control components such as relays by using solid state sensing and switching. The new controller, designated the PDP-14, represented a cost reduction over ~~and~~ for ad hoc designs controllers composed strictly of K-series modules. It did this by using time-multiplexing so that one control structure (in

memory), could serve as the interconnection (and processing) structure versus physically interconnecting the modules together. Such a system is shown in Fig. 3A.

The design requirements that the PDP-14 had to meet were as follows:

a) be lower priced (with lower life cycle costs) and easier to operate than existing control alternatives

b) solve the control problem and be programmable by users who have solved problems using a different representation (e.g., relay ladder diagrams)

c) operate in a high electrical noise environment

d) operate in the poor physical environment characteristic of the machine it controlled

e) have the appropriate I/O interfaces to sense contacts and to control power relays

Although a PDP-8/I might have been programmed to carry out the task, it was either not considered or else rejected because the cost was perceived to be too great, and there was some perception that a conventional stored program computer could not solve the problem. In addition, the PDP-8/I circuits did not

appear to have adequate noise margins to operate in the anticpated environment, and there was inadequate I/O capability.

As a result, the PDP-14 (see Fig. 34) was proposed and designed expressly to solve the problem and cost less than the 8/I which was just going into production. The PDP-14 had no data operations except on a single boolean value using a 1-bit accumulator called TEST. Even with so little arithmetic capability, the machine's structure and processor state was roughly equivalent to a PDP-8 design. Ultimately, the processor state was extended beyond that of a PDP-8, as the problem changed (e.g., when communication was required with host processors), but these extensions will not be discussed.

In order to solve the boolean equations that a conventional relay controller solves in parallel, the 14 had to solve equations sequentially at a rate of approximately 30 hz - fast enough to give the illusion that the equations are solved in parallel.

To operate in the high electrical noise environment, the circuit logic was slowed down in order to have improved noise margins. It was felt that core memory did not have adequate noise immunity, so a braided read only memory was used. To battle the effects of the poor physical environment, the unit was housed in a dust proof enclosure. To sense contacts and control power relays, appropriate I/O interfaces were designed.

The ISP of the PDP-14 is given in Fig. 35. It is the smallest, most trivial ISP that can be found. Technically, The PDP-14 was called a computer because it could perform computation in the same way a Turing machine can--without an arithmetic unit. However, it encoded the boolean data operators for which it was designed more efficiently than nearly any other computer, provided the equations were simple enough.

There were four instructions to take values from input switches or relay outputs and to compute new output values (the right hand side of a boolean equation). Therefore, the 14 also could simulate a Sequential Machine (state diagram or flow chart). Two additional instructions sensed the value of intermediate results (stored in TEST) and were used to eliminate the need to completely evaluate an equation each time. To direct program flow, there were four more instructions: Jump, Skip, Jump to Subroutine (a single level) and Return from Subroutine. To handle the "accessories box", there was special I/O rather than having this carried out internally to a program. This I/O included up to 16 boolean variables consisting of external control multivibrators and memory bits.

A good way to understand the 14's operation is to start with the application. Figure 36a shows a combinational relay logic network that evaluates a boolean expression (in parallel). When this network is implemented with the PDP-14, the inputs and

outputs are simply connected, and the program forms the interconnection which constitutes the solution of the equation (Fig. 36b). Figure 36c gives the boolean expression for the network in Fig. 36a. To evlauate this equation using a PDP-14 requires a sequential program (see Fig. 36d). This program requires between 120 microseconds and 200 microseconds to compute the output value, y8, since each instruction requires 20 microseconds. The speed of a computerized controller compared to relay operations is phenomenal. Heavy duty industrial control relays typically operate at a 30hz rate (33 milliseconds). If the PDP-14 can solve each equation with 4 terms in approximately 150 microseconds, the PDP-14 can solve 222 such equations in the time necessary to operate the relay. The memory requirements to solve the 222 equations are not large either. This equation required twelve locations; hence, 222 such equations requires about 2.5 Kwords. Here, it is instructive for the reader to compare this memory need with that implied using Amdahl's constant (see chapter 00).

A number of PDP-14s were built and installed for the intended applications over the period 1970-1972. Programming was carried out in languages supported by compilers that operated on PDP-8. The languages allowed users to:

a) write ordinary assembly programs (resembling PDP-8 programs)

b) express a problem directly as a set of boolean equations

c) express ladder diagrams (in effect, these are a set of boolean equations)

d) write a program as a flow chart, i.e., as a sequential machine that goes state by state and tests and branches on various input values to create output state, permitting both combinational (boolean equations) and sequential circuits to be implemented

e) simulate the behavior

As the 14 and its competitive machines were used, a demand for a second generation controller was created. By 1972, the additional requirements included lower cost, higher speed, an easily changed read only memory, and the ability to load programs via a communications line or connected console. In addition, the controllers were required to connect in a network fashion and report back status and results to a supervisory computer at the next level of a hierarchy. The second generation controller should also be capable of recording events including counting the number of parts processed. It also needed timers which could be used as part of the control equations. The new unit should operate over an even wider environmental range than existing PDP-14 and have a more complete set of I/O interfaces.

From these requirements, the PDP-14/30 evolved (see Figs. 37 and 37a). The initial read only memory was replaced by an 8 Kword core memory. In this way, the programs could be easily changed rather than having to be returned to DEC for manufacturing. Because the original 14 was so slow compared to the capability of the logic from which it was made, the instruction time was reduced from 20 microseconds to 2.5 microseconds to have better frequency response and to handle a larger number of equations. Additionally, because a large number of special registers had been added to hold numeric values (the shift registers, timers and counters), an arithmetic unit was added to the 14/30 in an ad hoc fashion. All these additions forced the ISP to change. The 14/30 extensions were unable to be made in such a way as to have binary compatibility; thus, software changes were also required.

An interesting offshoot of the PDP-14 development was the creation of the VT14 terminal for a programming, program load and observation console. This consisted of a CRT and PDP-8 mounted in a portable housing. Since the 14/30 could report the status of its input and output variables, the VT14 also had the ability to display the status of ladder diagrams (i.e., relay and contact position). A typical VT14 screen display is shown in Fig. 38.

At the time when the 14/30 was proposed, there were some who felt that it should not be built because a standard 8 Family

computer was cheaper to build, and more production volume and lower costs could be obtained by not constructing a special unit. In addtion, the 8 Family machine could be extended trivially to have the original 14 ISP, and the 8 ISP would be available to do the tasks that would inevitably evolve, such as doing self-diagnosis, doing more extensive counting and timing functions, and dealing with non-boolean data such as time or non-discrete events including angular position. The more powerful 8 ISP would also be useful for handling general control in both the analog and the digital domain, communicating with computer networks requiring protocol control for intelligent and error-free communication, and using algorithms to encode the control function instead the using of relatively large program state methods with no ability to perform computation.

Many of the previous arguments against using PDP-8s had now lost their merit. Since the 14/30 was proposed to be built using the same circuit family as that of the PDP-8s, the electrical noise margins arguments no longer held. Furthermore, the 8 could be packaged in a proper cabinet for the physical environment, and there could be adequate interfaces built. Besides, the proposed 14/30 would incorporate an 8 anyway, and two computers were obviously more expensive than one. In addition, adding the necessary cabinet and interface enhancements to the 8 would have greatly improved the marketability of PDP-8 for all industrial applications. Although the design group did not buy the arguments that the 14/30 should become an 8 with appropriate

extensions and packaging, some of the 8 parts were used in the 14/30 design.

The product was eventually ~~downgraded~~ removed as a mainstream product primarily because it required high, specialized support. The control problem had evolved to require a computer, ~~even though~~ just as the 14 ultimately evolved to be a complete, general purpose machine.

While it is easy to dismiss the 14 as a trivial special purpose controller that should never have existed (especially if one feels that the 14/30 architecture was poorly evolved), this is to say that the problem being solved should not have existed. Such a conjecture flies in the face of the adage "that the customer is always right". Rather, the problem was worthwhile and should have been solved with the full generality of a computer. The ultimate state (several next generations) of the controller is clear: a computer is needed to handle the full generality of the problem as it was, as it slowly evolved (when the 14/30 was designed) and as it will clearly continue to evolve. The solution was, and still is, to adapt and evolve a general purpose computer by a combination of hardware instructions and software to be the best controller. Such evolution surpasses the fixed control structure that evolved and was continuing to evolve. Not evolving the 8 or the 11 (and other microcomputers or minicomputers) to the application by just adding a few special instructions given in Fig. 35 was the

should have occured

error.

References:

Bell, C.g. and Casasent, D. Implementation of a Buffer Memory In Minicomputers, Computer Design, November 1971, p 83-89.

Bell, J., Casasent, D. and Bell, C.G., An Investigation of Alternative Cache Organizations, IEEETC, Volume C-23, Number 4, April 1974, pp. 346-351.

Clark, W.A. and Molnar, C.E., The LINC: A Description of the Laboratory Instrument Computer, Annals of the New York Academy of Sciences, Volume 115, July 1964, pp. 653-668.

Myer, T.H. and Sutherland, I., On the Design of Display Processors, CACM, Volume 11, Number 6, June 1968, pp. 410-414.

~~Clark and W~~

Clark, W.A. and Molnar, C.E. A Description of the LINC Computers in Biomedical Research, edited by Bruce Waxman, Vol II, chapter ~~II~~, 2 Academic Press, NY, 1965.

Van de Goor, A., Bell, C.g., Witcraft, D.A. Design and Behavior of TSS/8: A PDP-8 Based Time-Sharing System, IEEE Trans. on Computers, Vol. C-18, no. 11, Nov. 1969, p 1038-1043

THE 12-BIT FAMILIES

draft

The Families

This part on the First Minicomputer includes the descriptions of the 12-bit computers that DEC has built and been associated with that effected the PDP-8 Family design. The title is slightly presumptuous because the main contribution of the 8 Family is the number produced (in use), the notion that significantly cheaper machines can always be built and applied to new applications--hence the name minicomputer for mini(mal) computer. The 8 Family also clearly either directly or indirectly contributed to the name, Minicomputer--not an insignificant contribution in a world that often must have a simple way (name) to identify a concept. The early history of the Minicimputer, is more accurately represented in Fig. 1 which shows the inter-relationship of early computers.

Figure 2 shows both the family tree of the 12-bit machines with significant events outside DEC and the corresponding internal response events that made up the evolution (e.g., technology use, first shipment of a compiler). The family tree position shows that there are five nuclear groupings (or sub-families) within the complete family. There are two ~~distant~~ relative families, the LINC family--which contributed some base ideas to the original PDP-5 and the PDP-14 Industrial Controller Family, which also had a 12-bit word and bears some relationship to the 8 Family. ~~Later~~ Members of the 14 used parts from the Omnibus family, including the Omnibus and memory components.

Later members of the LINC family were made up of major physical and software subassemblies that were part of the 8 Family, and the original LINC was built from DEC Systems Modules.

It is clear that the CDC 160 was the first commercial 12-bit computer and as such contributed ideas and organization to both the LINC and 8 Families, as did the DEC 18-bit Family (as we describe in Chapter ?).

The processor-on-a-chip Family is built totally outside of DEC by two other organizations, Intersil and Harris Semiconductors. Our efforts to build the 1 chip computer only lasted for about a year, before the efforts were turned to the PDP-11 Family. The existence of this family is particularly significant as it forms the basis for the one-board-computers and the computer-in-a-terminal.

Packaging

The time line part of the figure is divided into basic semiconductor and module packaging technology, computer languages and computer operating systems. On the packaging and semiconductor line we note there is a one to five year delay between when an idea is available and when it is applied to a machine (note the application of Silicon transistors and TTL logic). For example, it is possible that the 8 could have used TTL logic and been even further out on the state of the art line--negating need for 8/F and 8/L). Since the packaging was different for the 8, and done with automatic wirewrap machinery, this change was enough for a single machine. Also, TTL logic, as initially marketed, was significantly more expensive than the discrete logic circuits that the 8 was built from. It took five years to adopt the automatic wirewrap machinery that was first used in the IBM 7090.

This was partially due to the fact that it represented both a major packaging change and a major capital expenditure to DEC (sales were about $10M then). The various read only and read write memories have not been essential to the designs ~~and~~ since the ISP is so simple. A large register file and low cost ROM for microprogramming makes the floating point processor and its control possible in the 8 price range (note FPP versus semiconductor memory availability).

~~Software~~ Languages ~~and Appl~~

FORTRAN, in 4 Kwords, is probably the most significant accomplishment in terms of languages for the 8. It was designed for and operated on the PDP-5. Eventually both BASIC and FOCAL interpreters were written to provide better interactive languages. These are used extensively as a calculator, and for real time control including laboratory experiment and industrial control. The languages were both extended for graphics display. The 8 is also used extensively for teaching about computers and computer aided instruction.

In 1970 DIBOL, a business language interpreter, was introduced. Its roots include COBOL, ~~but as an interpreter,~~ it was designed to provide faster response to the writing and debugging of programs and to ~~make more efficient~~ use of memory space. These are excellent tradeoffs for the single user system whose problems are usually disk performance bound vis a vis accessing records. An especially powerful editor and language was introduced in 1976 in the form of WPS8 for the word processing market. It also is the forcing function to have very low cost standalone computers...in this case the VT78 resulted.

The Operating Systems

The operating system line shows that the influence for software came from two areas: the original LINC, which was designed to provide a single user with a private, apparent computer with his own filing and processing system. A recent product, DECnet, ~~a~~ our method of interconnecting computers over greater physical distances, came from the minicomputer population growth, which in turn forces the computers to communicate with one another in some form. Much of the basic work of DECnet was derived from the ARPAnet research project. It is interesting to note, that DECnet 8 is the first application of DECnet within DEC. This has quite possibly occurred because the basis operating system is clean; as such, the complexity of DECnet can be understood and integrated more easily.

The various single user and real time operating systems are quite unequalled, even though comparatively smaller efforts have been devoted to their design and implementation. Another by-product of the single user designs has been the understanding developed for the creation of the single user operating system, RT-11, for the PDP-11.

A system that one of us (Bell) has a special attachment for is TSS/8, the timesharing system for the PDP-8. Only a hundred or so systems were sold, while technically it is the lowest cost (per system and per user) and highest performance/cost timesharing system we provide. The reason for the lack of commercial success, I suspect, is simply attributed to the fact

that the 11 came along and the development and marketing efforts were redirected. On the other hand, TSS/8 requires some understanding and the time to sell such a system may be greater than we have allowed for (i.e., it may be underpriced in terms of software complexity). A major side benefit of TSS/8 was the training of one of the implementors, Nate Teichholtz, who went on to propose and first implement the RSTS timesharing system based on the language, BASIC. TSS/8 was designed while I was at Carnegie-Mellon University, by a graduate student, ?Ned van der Goor, (Reference?) as a reaction to the incredibly high cost, poor performance, and poor reliability that IBM was providing the TSS/360 product (Model 67 design). Although TSS/360 eventually worked, it wasn't marketed, and at most contributed some ideas and trained thousands of programmers for IBM. At CMU, a TSS/8 operated until 1974 when the special swapping disk we had used gave out. The cost per user or per job tended to be about 1/20 of the TSS/360 system they run.

The Chapters

The following section describes the motivations for including the chapters, how the machines relate to the family and their contributions to minicomputers. The final section describes the PDP-14, how it came to be, and my own version of what it could (should) have been.

The PDP-8

This chapter is a substantial extension to the original first appearing in Bell and Newell, (1971). The extensions include the Omnibus and CMOS-8 families which came out after the chapter was written. Now enough time has passed to gain a historical perspective. In this regard we look at how technology has shaped the evolution and resulting structures from a price, performance, and packaging characteristics over the last 15 years.

The main part of the original chapter remains in tact and is present to show how a machine can be understood by decomposing it into a number of levels which correspond to its physical structure.

The LINC

Since the LINC was designed outside DEC, it doesn't precisely fit the constraints the other chapters meet as being about DEC's computers. However, since the LINC was initially built from DEC Systems Modules, we feel somewhat justified. A stronger tie is perhaps that DEC went on to manufacture the LINC directly from the LINC design. Subsequently Dick Clayton (in conjunction with Wes Clark) designed both the LINC-8, a two-processor machine (LINC + PDP-8) which executes both ISP's in parallel, and the PDP-12, a single physical processor that executes either PDP-8 or LINC instructions sequentially (by switching modes).

Some of the characteristics of the LINC Family machines is given in the following table. Note the size has remained essentially constant at one cabinet over the computer's life.

Table **LINC Family Characteristics**

	LINC	LINC-8	PDP-12
Project start	1961	summer 1965	6/67
First ship	3/62 (demo)	8/66	6/69
Withdrawn	12/69	12/69	6/75
No. Produced	50(21 by DEC)	143	1000
Price (min.)	$43,600	$38,500	$28,100
Key designers	Wes Clark & Charles Molnan (Lincoln Lab)	Wes Clark, Dick Clayton (DEC)	Dick Clayton
Goals, features	complete system for laboratory user...including file system and crt's	cost, + PDP-8 software compatibility, use 8 components	larger scope, bus compatible with 8/I
Size	69 x 32 x 30 + tape, keyboard, and console	69 x 32 x 33	76 x 35 x 33
Mp-Pc accesses (Mega)	0.125	0.67 (PDP-8 memory)	0.67 8/I memory
Weight	?	?	700
Power (watts)	1000	2000	< 2000
CRT	2 - oscilloscopes	2 - scopes	1 - 7" x 9" (10")

There are other reasons to include the LINC as members of the DEC 12-bit family. The LINC strongly influenced other PDPs (including PDP-4 and -5). The LINC tape unit and system ideas permitting a user to have personal files has been carried over directly in the DECtape design and programs. This device made the personal computer for the researcher possible at a reasonable economic level. We have also included the LINC because the paper is especially well written, even though it does stress applications and minimally discusses the construction, structure and other engineering aspects of the design.

The systems aspect of LINC (circa 1962) were really quite significantly ahead of their time at DEC, and of other systems including the timesharing work in terms of providing a clean, simple, yet powerful console (two CRTs) and file system interface (~~note~~ Fig. timeline). Note the DECtape library system came two years later on PDP-5. We had been stressing design flexibility and modularity and provided large ways for interconnecting computer components together in order to have any type of structure. LINC is quite constrained (only 1 Kword or 2 Kwords were available) with two

LINC tapes and two CRTs. Most all other characteristics including the I/O interface were quite constrained. By bounding the system in this way, it is possible to provide a complete environment including software.

DEC 338 Display Computer

The 338 is included mainly for historical reasons and because it is a variant of the PDP-8 Family. It clearly shows how the application need drives a complete change in hardware so as to interpret the necessary data type (a graphic picture). We believe it is one of the very earliest display processor based computers--if not the first. The question of being first is clouded by the state of the evolution as we discuss below.

The 338 was extended and applied to PDP-9 (in the form of the 339) and then greatly extended in the PDP-15's VT15 display processor. Although the 338 has the right general capabilities, it doesn't have the refinements that later display processors for PDP-15 and PDP-11 have (GT40 and GT60). Also there is a question of having a balanced computer. Since the display processor is so relatively expensive, then the host processor should not be a bottleneck, nor must it be significantly cheaper than the display part. The DEC display processors have tended to extend the 338, whereas there have been departures from this structure (CMU thing?).

An observation that display processors (Evans and Myers, 19??) and perhaps other specialized processors evolve in a fashion akin to the wheel of reincarnation. Figure Rein shows how one starts with a very simple basic design to have graphics picture output for a computer.

1. The computer includes a single instruction display controller which can plot a point (on a point-by-point basis) under command of the central processor. For most all displays, except storage scopes, the processor can barely calculate the next point fast enough to keep the scope refreshed. Hence, the system is processor bound, and the display may be idle. The original PDP-1, Type 30 scope is typical of this position.

2. By adding the ability to plot lines (vectors) a single instruction to the display processor will free some of the processor and begin to keep all but the fastest scope busy.

3. Having realized that characters are mainly displayed, then commands to display a character are added, further freeing the processor. Many of the point plotting displays were extended to have character generation plotting. The alphanumeric scopes have this capability (less vector plotting).

4. In reality, a picture doesn't consist of just characters and vectors, but each element of the picture is actually a string of characters to be displayed at a particular point, and a set of closed or open polygons to be displayed starting at a particular point. By providing the control display with a Direct Memory Access channel, it can fetch each string of text and generate polygons with no central processor interaction.

5. Now with the ability to put up sub-pictures with no processor intervention, it is easy for the whole picture to be displayed by linking the elements together in some fashion. This merely requires jump and subroutine call instructions (so common picture elements don't have to be redefined). This is roughly the capability of the 338 and most other display processors.

6. Now, we realize that all the paths and all the state is present for a fully general purpose processor, such that the central processor need never be called on again. This requires a slightly more general purpose interpreter. There is full generality by minor perturbations, and the processor design can be refined in such a way as to execute the same ISP as the original host because the cost of incompatibility is too great. Two processors require two compilers, diagnostics, manuals, and support for use. Finally, we have arrived back to the same capability as we started. The original processor is completely free and there is a display processor with the capability of executing both the original ISP plus the display ISP.

7. Alternatively, we can take this processor and isolate it as a separate computer which we reconnect in some fashion back to the original's central processor-primary memory pair.

The Cache-Based PDP-8

Chapter 7 describes a design for a PDP-8 constructed by Professor David Casasent at CMU. Initially the project was proposed for exploratory advanced development on the desirability and feasibility of using Emitter Coupled Logic (ECL) for designing very fast, marketable minicomputers. As the investigation proceeded, a large, very fast memory need emerged. Such a memory turned out to be so costly that such a computer would not be feasible from a market standpoint. The easiest way to build a cost-effective, fast minicomputer turned out to be using a cache structure in order to reduce the cost of primary memory. Also, instead of designing a very fast PDP-8, which ECL would have provided, the goal was modified to be more modest and potentially marketable. TTL Schottky was used in the design, even though the logic family was just beginning to evolve.

In order to make the prototype more nearly marketable, without a complete redesign, the design was also constrained to be compatible with most of the 8/E Omnibus backplane and parts. The prototype did not become operational as quickly and cleanly as possible, hence the use of the prototype to stimulate a market did not occur. All through the design there was little interest in the marketability of the design. It was carried forward in order to test the cache applicability, circuitry, ease and techniques for building a fast 8. As is true in most new products, the fact a market is not there is clear. We were not losing sales to very fast minicomputers, hence there was (and is) no market for very fast computers. Furthermore our customers are of little or no help because, unless faced with the widescale availability of such a machine, the applications generally don't emerge.

A number of things were learned from doing the work. It did influence the 11/45 although the designers rejected the cache notion...since they believed that the cache concept clearly didn't apply to PDP-11 program behavior. Eventually Strecker persevered and convinced other 11 designers to build a cache-based larger scale minicomputer...and this became the 11/70, ~~as described in Chapter X~~. Fortunately, the processor was designed to operate with all bipolar, fast memory. The paper illustrates the use of the 8 as an experimental vehicle for understanding a design in terms of program behavior. It also allows one to measure the time flow of ideas from research, advanced development, and finally into production machines. It also shows how, by rather minor perturbations, a design can be kept compatible with its predecessors while providing rather dramatic performance and performance/cost improvements.

PDP-14

The PDP-14 was designed expressly as a controller for complex electromechanical machinery (such as transfer machines, conveyors, ~~non-numerical~~ milling). The problem first originated at General Motors as a controller for a large, transfer machine which made engine blocks by a sequence of operations. It often replaced or supplemented large relay sequential circuits.

The Basic Design Problem

Such a controller should:

1. be lower priced (with lower life cycle costs) and easy to operate;
2. solve the control problem;
3. operate in the high electrical noise environment;
4. operate in the poor physical industrial environment characteristic of the machine it controls;
5. have the appropriate I/O interfaces to sense contacts and output to control power relays.

Although a PDP-8 might have been programmed to carry out the task, it was either not considered or else rejected because:

1. the cost was too great;
2. there was some perception that a conventional stored program computer could not solve the problem;
3. the circuits did not have adequate noise margins to operate in the environment;
4. the physical environment constraints couldn't be met;
5. there was inadequate I/O.

The notion of a ~~sequential~~ controller ~~came out of the~~ evolved from ~~of~~ the use of standard DEC K-series industrial modules, see Chapter 00. The K-series modules provided increased reliability ~~through~~ over the electromechanical control (relay) components by using solid state sensing and switching. ~~versus being electromechanical.~~ The 14 ~~was~~ represented a further cost reduction over the K-series modules by using time-multiplexing so that one control structure (in memory) could serve as the interconnection ~~and processing~~ set structure versus physically interconnecting the modules together.

Additions to PDP-14

Jump to Sub. + Sub. regs

Arith units regs addrs ptrs

3 address.

Retentive mem.
Ctrs events
Timers

The Design

As a result, the PDP-14 was proposed and designed expressly to solve the problem and cost less than the ~~PDP-8 (or rather the~~ 8/I~~--~~which was just going into production~~)~~. The 14 has no data operations except on a single ~~bit (~~boolean~~)~~. ~~In fact the 14 has only a Program Counter, and a single 1-bit Accumulator. The most complex instruction is Jump to Subroutine.~~

1. In order to solve the boolean equations that a conventional relay controller solves in parallel, the 14 must solve equations sequentially at a rate of approximately 30 hz, giving the illusion (which is quite adequate) that the equations are solved in parallel. The 14 has instructions for taking data from input sensors, solving the equation and outputting the result to a relay. The ISP of the PDP-14 is given in Fig. x.

2. The circuit logic was slowed in order to have improved noise margins. It was felt that core memory did not have adequate noise susceptibility, hence couldn't be used. The braided, read only memory was used.

3. The unit was housed in a dust proof enclosure.

4. Appropriate I/O interfaces were designed.

A number of 14s were built and installed in the intended applications over the period 1970-1972. Programming was carried out in languages supported by compilers that operated on PDP-8. The languages allowed users to write:

1. ordinary assembly programs (resembling PDP-8 programs);

2. express ladder diagrams (in effect, these are a set of boolean equations);

3. program as a flow chart; i.e., as a sequential machine.

The Evolution

As the 14 and its competitive, contemporary machines were used, a demand for a second generation controller was created. By 1972, the additional requirements included:

1. better cost, due to competitive similar products, including the companies whose relay products the 14 displaced.

2. There were many more functional demands including having an easy to change read only memory with the ability to load programs via a communications line or connected console. In addition the controllers should connect in a network fashion and report back status and results to a supervisory compputer at the next level of a hierarchy. The controller should be capable of recording events (e.g., counting the number of parts it processed). It needs timers which can be set as part of the control equations.

3-5. The unit should operate over an even wider environmental range, with a more complete set of I/O.

From the ISP description, we note that it is the smallest, most trivial ISP that one ~~would ever~~ could find. Technically, it must be called a computer because it can ~~be used to~~ perform computation in the same way a Turing machine can. ~~It~~ It encodes the Boolean data operators for which it was designed more efficiently ~~better~~ than nearly any other computer ~~in the books~~ - provided the equations are ~~to~~ simple.

There are 4 instructions to take ~~input~~ Boolean values from input switches (2) or the current state of the outputs (relays), ~~the outputs~~ and to compute new ~~output~~ boolean values (the right hand side of a boolean equation). Therefore, it ~~is a~~ also can simulate a Sequential Machine (~~Two~~ state diagram or flow chart). Logically, two instructions set output values either to 0 or 1, and two instructions sense the value of intermediate results (stored in TEST) and are used so as to eliminate the need to completely evaluate an equation each time.

There is one instruction, JUMP, to direct program flow.

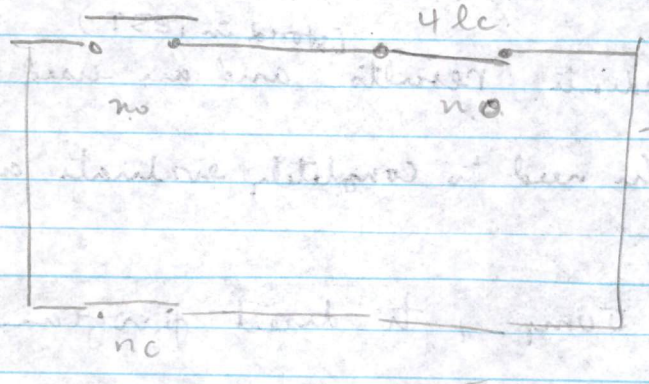
$Y6 = X6 \wedge X14 \quad \vee \neg X7 \wedge \neg X15$

a Boolean equation

Location

```
40   →TXF  6      } Turn TEST ON if either
41    TXF  14     } 6 or 14 is OFF
42    JFF  46        TEST = ¬X6 or ¬X14
43    TXN  7      } Turn Test on if either
44    TXN  15     } 7 or 15 is ON
45    JFN  51        TEST = X7 ∨ X15
46    SYN  6 ←    turn on
47   ←{JMP
50     40
51    SYF  6 ←    turn off
52    {JMP
53     40
```

assume TEST = 0 at start



1

7 5

another
Figure The ~~best~~ way to understand the ~~extra~~ 14's operation, ~~and~~ is to start with the application. Figures Xa, ~~Xb~~, ~~and Xc~~ shows how ~~a~~ we start with what is a combinatorial, relay logic network that singly evaluates ~~computes~~ a boolean expression (in parallel) When this is ~~applied~~ implemented with ~~connected to~~ the 14, the ~~inputs~~ inputs and outputs are singly connected and the program forms the interconnection. (Fig. X6)

Figure Xc ~~shows~~ gives the boolean expression for the network in Fig. Xa. ~~To impl~~ To implement ~~this~~ (evaluate) this equation, ~~on the~~ using a 14 requires a sequential program (See Fig Xd). This program requires a between ~~minimum of~~ 120 microseconds (~~to x z 0~~) and ~~to~~ 180 microseconds to compute an output value, ~~&~~ y6, ~~and~~ since each instruction requires 20 microseconds. EV

It should be clear to the reader that as the number of equations to solve, increases, the time to solve the set decreases. Since

D

30 hz rate

relays typically operate at a (133 ~~50~~ milliseconds) rate, and each equation with 4 terms takes approximately 150 microseconds, then 222 ~~333~~ such equations can be evaluated by exist in such a system. Also note that this equation required 12 locations, and hence a memory full of such equations requires about 2.5 Kwords.

~~Recall~~

~~If we compare this with Amdahl's constant, a machine operating at 20 20 mic 50 K~~ This requires much more

Here, it is instructive ~~to~~ for the reader to ~~compute the~~ ~~lower~~ memory ~~requirements~~ compare this memory need with ~~that~~ that implied using Amdahl's constant.

Thus, the PDP-14/30 evolved (see Photo Y), Insert E ~~as can be seen in Fig. Y.~~ Another critical part of the 14 includes a programming, program load and observation console, the VT14, which consists of a CRT and PDP-8 mounted in a portable housing. Since the 14/30 can also report back status of its input and output variables, the VT14 also has the ability to display the status of the ladder diagram (i.e., whether a relay is on or off and the contact position). ~~A diagram~~ is shown in Fig. ?. At the time when the 14/30 was proposed, I urged that it not built because:

1. A standard PDP-8 computer is cheaper to build and we would get more production volume and lower costs by not building a special unit.

2. The PDP-8 could be extended trivially to have the ~~complete~~ original 14 ISP. Thus the PDP-8 ISP would be available to do the tasks that would inevitably evolve to:

   a. do self-diagnosis;
   b. do more extensive counting and timing functions;
   c. deal with non-boolean data (e.g., time, non-discrete events including angular position);
   d. handle general control in both the analog and digital domain;
   e. communicate with computer networks requiring much better protocol control for intelligent and error-free communication.
   f. use algorithms to encode the control function instead of a relatively large program state with no ability to perform computation

3. The PDP-8 circuits had adequate electrical noise margins.

4. The 8 could be packaged in the proper cabinet for the physical environment.

5. There could be adequate interface parts.

Although I was unsuccessful in converting the 14/30 to become an 8 with appropriate extensions and packaging, some of the 8 parts were used in the 14/30 design. At this time, some of the initial conjecture about the 14 was correct. The product was ~~discontinued~~ primarily because it required high, specialized ~~intensive~~ support ~~that was outside the mainstream resources~~, hence dropping represented a strengthening of purpose.

While it is easy to dismiss the 14 as a trivial special purpose controller that should never have existed, this is to say that the problem being solved should not exist. Such a conjecture flies in the face of the adage "that the customer is always right".

The problem is worthwhile and should have been solved with the full generality of a computer. The ultimate state (several next generations) of the controller is clear: a computer is needed to handle the full generality of the problem as it was, as it slowly evolved (when the 14/30 was designed) and as it will clearly evolve to. The solution was, and still is, to adapt and evolve a general purpose computer by a combination of hardware and software to be the best controller. Such evolution can easily surpass any fixed control structure that would have evolved...not evolving the 8 to the application was the error.

Can't
@C'
Closed
@N
Wait...'
"/'n'z}

—Insert E —

E

the initial core memory was replaced by an 8K word core memory. In this way, the programs could be easily changed rather than having to be returned to DEC for manufacture. ~~The memory was size was also increased to 8 Kwords, and~~ Since the original 14 was so slow compared to its inate, circuit capability the instruction time was reduced from to 20 microseconds to 2.5 microseconds, to ~~This~~ have ~~enabled both~~ better frequencey response and to handle a larger number of equations ~~to be solved.~~ ~~Since it wass there~~ A ~~subrou very s~~ primative subroutine calling instruction ~~mechanism~~ was added ~~to the 14~~ giving it 1 level of subroutines by using a single register to hold the caller. ~~A An arithe~~ Since a large number of special registers were added to hold ~~numb~~ numeric values ~~such as~~ the shift registers, timers and counters) ~~a set of~~ an arithmetic unit was added to it, in an ~~a very~~ ad hoc fashion.

(print date 11/21/77)

add here to this.

- Myer and Sutherland

THE 12-BIT FAMILIES

~~The Beginning of the Minicomputer~~

This part on the ~~First~~ Early Minicomputer includes the descriptions of the 12-bit computers that DEC has built and been associated with that affected the PDP-8 Family design. ~~The title is perhaps slightly presumptuous because the main contribution of the 8 Family is the number produced (in use); the notion that~~ significantly cheaper & smaller machines can always be built and applied to new applications--hence the name minicomputer for mini(mal) computer. The 8 Family also clearly ~~either directly or indirectly~~ contributed to the name, Minicomputer--not an insignificant contribution in a world that often must have a simple way (name) to identify a concept. The early history of the Minicomputer, is ~~more accurately~~ represented in Fig. 1 which shows the inter-relationship of early computers.

The Machine Families

Figure 2 shows both the family tree of the 12-bit machines with significant events outside DEC and the corresponding internal response events that made up the evolution (e.g., technology use, first shipment of a compiler). The family tree position shows that there are five nuclear groupings (or sub-families) within the complete family. There are two close relative families, the LINC Family which contributed some base ideas to the original PDP-5; and the PDP-14 Industrial Controller Family, which also had a 12-bit word ~~and bears some relationship to the 8 Family~~. Later members of the LINC Family were made up of major physical and software subassemblies that were part of the 8 Family, and the original LINC was built from DEC Systems Modules. Members of the ~~14~~ family used parts from the

Page 2A here

PDP-14 Family

Omnibus Family, including the Omnibus and memory components.

A move on page 1

It is clear that the CDC 160 was the first commercial 12-bit computer, and as such, contributed ideas and organization to both the LINC and 8 Families, as did the DEC 18-bit Family (as we describe in Chapter 00).

The Processor-on-a-chip Family is built totally outside of DEC by two other organizations, Intersil and Harris Semiconductors. DEC's efforts to build the one chip computer only lasted for about a year before the efforts were turned to the PDP-11 Family. The existence of this family is particularly significant as it forms the basis for the one-board-computers and the computer-in-a-terminal.

(see Chapter 00.)

Packaging

The time line part of the figure is divided into basic semiconductor and module packaging technology, computer languages, and computer operating systems. On the packaging and semiconductor line, we note there is a one-to-five-year delay between when an idea is available and when it is applied to a machine (note the application of Silicon transistors and TTL logic). For example, it is possible that the 8 could have used TTL logic and been even further out on the state-of-the-art line--negating the need for 8/F and 8/L. Since the packaging was different for the 8, and done with automatic wirewrap machinery, this change was enough for a single machine. Also, TTL logic, as initially marketed, was significantly more expensive than the discrete logic circuits that the 8 was built from. It took five years to adopt the automatic wirewrap machinery that was first used in the IBM 7090.

B from page 3 also gets moved

This was partially due to the fact that it represented both a major packaging change and a major capital expenditure to DEC (sales were about $10M then). The various read-only and read-write memories have not been essential to the designs since the ISP is so simple. However, a large register file and low cost ROM for microprogramming makes the floating point processor and its control possible in the 8 price range (note the FPP introduction versus semiconductor memory availability). Such an option significantly improves performance by at least a factor of 10 (see Chapter 00), and in the case of the 8, more like a factor of 20-50.

Languages

FORTRAN, in 4 Kwords, is probably the most significant accomplishment in terms of languages for the 8. It was designed for and operated on the PDP-5. Eventually both BASIC and FOCAL interpreters were written to provide better interactive languages. These are used extensively as a calculator and for real time control including laboratory experiment and industrial control. The languages were both extended for graphics display. The 8 is also used extensively for teaching about computers and computer-aided instruction.

In 1970 DIBOL, a business language interpreter, was introduced. Its roots include COBOL. It was designed as an interpreter to provide faster response to the writing and debugging of programs and to use memory space efficiently. These are excellent tradeoffs for the single-user system whose problems are usually disk performance bound vis à vis accessing records. An especially powerful editor and language was introduced in 1976

in the form of WPS8 for the word processing market. It also ~~is the forcing function to have~~ provided the impetus for the construction of very low cost standalone computers, in this case, the VT78 resulted.

Operating Systems

~~Two areas: LINC + DECnet?~~

The operating system line shows that the influence for software came from ~~two areas~~: the original LINC and a recent product DECnet. The LINC ~~which~~ was designed to provide a single user with an interactive, personal computer with his own filing and processing system to assist in the laboratory. A recent product, DECnet, our method of interconnecting computers over greater physical distances, came from the minicomputer population growth, which ~~in turn~~ forced the computers to communicate with one another in some form. Much of the basic work of DECnet was derived from the ARPAnet research project (reference?). It is interesting to note, that DECnet 8 is the first application of DECnet within DEC. This has ~~quite possibly~~ possibly occurred because the basic operating system is clean; as such, the complexity of DECnet can be understood and integrated more easily ~~(Comparative, than what?)~~ than with larger systems.

The various single user and real-time operating systems are quite unequalled, even though comparatively smaller efforts have been devoted to their design and implementation than for larger computers. Another by-product of the single user designs has been the understanding developed for the creation of the single user operating system, RT-11, for the PDP-11.

~~A system that one of us (Bell) has a special attachment for is TSS/8, the timesharing system for the PDP-8.~~ (1) The timesharing system for a PDP-8 was TSS/8 an ~~especially~~ interesting and ~~important~~ influential product (Bell/van De Goor 19xx). While only a hundred or so systems were sold, ~~while technically it is~~ TSS/8 is the lowest cost (per system and per user) and

See next page for a footnote

MJ add this reference and put ref with figs so it can be book in master bibliog

highest performance/cost timesharing system we provided. The reason for the lack of commercial success, I suspect, is simply attributed to the fact that, the 11 came along, and the development and marketing efforts were redirected. On the other hand, TSS/8 required some understanding, and the time to sell such a system may be greater than allowed for (i.e., it may be underpriced in terms of software complexity). A major side benefit of TSS/8 was the training of one of the implementors, Nate Teichholtz, who went on to propose and first implement the RSTS timesharing system for PDP-11 based on the BASIC language. TSS/8 was designed at Carnegie-Mellon University, with a graduate student, Adrian van der Goor, (Reference?) as a reaction to the cost, performance, reliability, and complexity of IBM's TSS/360 (Model 67). Although TSS/360 eventually worked, it wasn't marketed, but contributed some ideas and trained thousands of programmers for IBM. At CMU, a TSS/8 operated until 1974 when the special swapping disk expired. The cost per user or per job tended to be about 1/20 of the TSS/360 system they run.

THE CHAPTERS

The following section describes the motivations for including the chapters, how the machines relate to the family and their contributions to minicomputers. The final section describes the PDP-14, how it came to be, and my (Bell) own version of what it could (should) have been.

Insert C

PDP-8

This chapter is a substantial extension to the original chapter which first

appeared in Bell and Newell (1971). The extensions include the Omnibus and CMOS-8 families which came out after that chapter was written. Now enough time has passed to gain more historical perspective. In this regard, we look at how technology has shaped the evolution and resulting structures from a price, performance, and packaging characteristics perspective these last 15 years.

The main part of the original chapter remains intact. We include it to show how a machine can be understood by decomposing it into a number of levels which correspond to its physical structure.

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.

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 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.

C Insert

PDP-1 and PDP-4 Based 18-Bity Computers

This can chapter was written expressly for the book to ~~recall and~~ reconstruct ~~as much of the~~ history as ~~poso possible about how the computers~~ of the 18-bit computers ~~came into existence~~. It ws was co-authored by ~~r~~ the designers or key project team members of the machines where possible. ~~As such, it the~~ The ~~lonsest~~ descriptions ~~ase~~ of the ~~original~~ first DEC computer, the PDP-1, includes the ~~which established an architecture based on the~~ earlier MIT influences; ~~Since the I~~ although we however we can ~~ot~~ only speculate about some of the goals because they ~~aait well recorded~~ weren't very explicit.

Since the PDP-~~5~~4 departed from the PDP-1 architercture, there is a ~~long~~ self-justiification that gives the thoughts surrounding the architect change. This section should be especially valuable to computer designers because it is (or at least familiar) and all of the system has a set of argumennts that recusr throughtout many ~~computers~~ designs to initiate ~~many~~ needless, new architectures.

~~The thoughts can also be applied to many other design objects that are needlessly chansed and redesigned.~~ The authors have heard the arguments many times and expect to hear them ~~a~~ again many times ~~int he future.~~

There are brief descriptions about the PDP-7 and PDP-9.

The ~~last~~ ~~last~~ PDP-15 ~~also~~ represented a major departure from the PDP-4 architecture and is described, although the ~~ca~~ architectural changes aren't described in the ~~paper,~~ chapter.

~~The 15 c~~ The most ~~important~~ interesting part of the 15 ~~discussion is~~ description by Jerry Butler ~~is an attempt to be~~ is retrospective and even ~~somewhat~~ critical of ~~how~~ the ~~various~~ design ~~des~~ decisions ~~were made.~~ Thus, ~~it~~ This section provides a good model of how ~~we wa wa we believe~~ design should be carried out (hopefully ~~on an~~ a priori basis) and reviewed — goals and ~~a~~ of the series.

The 15 is the most significant in terms of longevitiy, ~~x~~ number ~~x~~ in use and the ~~ranse of~~ product ~~it covered.~~

The final section ~~of on of the 18-bti computers~~ compares all the machines ~~from a~~ in terms of ~~various~~ cost, performance and physcial metrics, ~~and could conv conceivsbly read~~ the section can be read ~~before ands of the ot~~ independently of the machines design descriptions.

Here, it is important for designers to realize there is a continuity to design and subsequent designs have to be better along one or more of the important ~~design~~ evaluation dimensions. It shows how by ignoring or not understanding ~~some of the~~ a the dimensions can lead ~~les dl lead~~ to failure in the ~~marktplace.~~ marketplace.

~~There~~ is a complete ISDP desciption of both the PDP-1 and PDP-4 computers is presented on a comparitive basis so that the reader can see ~~how~~ the ~~diff~~ small ~~the~~ differences ~~are.~~ It is from the ISP, the cost factors can be determined for a particular implementation and realization.

In the ~~12-bit~~ chapter on the PDP-8 there is a ~~description of the~~ similar side by side companion ISPs ~~ISP descriptio~~ of ~~both~~ the PDP-4 and PDP-5 ~~pc computers~~ and how they influenced one another.
so ~~that~~ the ~~pr~~ reader can make ~~this de~~ a direct comparison of the two machines. We expect

that the reader can ~~easily so thoru thsourh through and~~ understand all these ISP descriptions

~~and~~ quite intuitively ~~and if need be reade teh the ISPS~~ primer in appendix 1. is given

with no other reference, but ~~preferably~~ for the unitiated ~~ISP~~ who does not read the ISPS language, a ~~pr~~