BASIC STRATEGY   DO NOT COPY   Appendix 1

OOD/Gordon Bell

Last edit: 11/17/78 -- Latest edit 1/10/79 - Wed

ID#354   COPY # 95   ASSIGNED TO: Original Product Strategy Document

BASIC PRODUCT STRATEGY

Provide a set of homogeneous distributed computing system products so a user can interface, store information and compute, without re-programming or extra work in many styles and the following computer system sizes:

as a single user computer within a terminal;
at a small, local shared computer system; or
via a large central computer or network.

Achieve a single VAX, distributed computing architecture by 1985 (as measured by revenue) through:

focusing on homogeneous distributed computing with varying computing styles including high availability and ease (economy) of use as the DEC advantage;
building new 11 hardware to fill the product space below VAX;
building new 11 software products that also run on VAX; and
developing software for 11-VAX migration and 11 user base protection.

Provide essential standard IBM and international network interfaces.

Define, and make clear statements internally and to our users about programming for DEC compatibility.

Provide general applications-level products that run on 8, 10/20 and 11/VAX-11 above the language-level to minimize user costs, including:

word processing, electronic mail, and profession-based CRT-oriented calculators; "Visicalc"
transaction processing and data base query; / LISA package
general libraries, such as PERT: simulation, etc. aimed at many professions that cross many institutions (industry, government, education, home); and
general management libraries for various sized business.

Provide specific profession (e.g. electrical engineering, actuarial statistician) and industry (e.g. drug distributor, heavy manufacturer) products as needed via the product line groups.

Provide cost-effective 8, 10/20 systems through:

building hardware that runs current operating systems; and
making market support and DEC-standard language enhancements.

This strategy is intended to cover the full range of DEC's future products. Since technology shifts rapidly and market opportunities emerge that we don't now understand, it may be necessary to provide non-compatible, point products. These should be proposed and reviewed accordingly.

Essence and Rationale of the Strategy

The essence of the strategy is simplicity through adopting a single architecture. This simplicity is needed so that we can build the network and distributed processing structures which our customers are now demanding. The strategy is a" evolutionary result of the 1975 choice to extend the 11 architecture and cover its customer base.

Given that the architecture and early customer acceptance are in place, the strategy moves to build our subsequent products on VAX, while continuing to sell 8's, 10/20's and 11's. Focus is imperative in order to avoid the redundant development efforts across base hardware and software, and to move development to fully distributed computing and to applications. The strategy also minimizes manufacturing and field start-up costs and takes advantage of the learning effect by moving to a single architecture.

The motivations for the homogeneous architecture are numerous and include the customer desires for a range of products on which to build products (in the case of OEMs) and applications (in the case of end users). Such a range in size and over time, allows planning and investment of software and it permits computers to be associated with various organizational units (eg. central group, small group, office, the person, or the home) on a" "as needed" basis. Although, superficially it appears to be possible to have numerous architectures that are segmented by size and by market, the user requirements to cross both size and applications boundaries are significant. I" fact, give" that IBM is segmenting its products both by size and application, the main strength of the strategy is to have a single architecture with which a user can be comfortable rather than bounded by a manufacturer segmentation.

The most compelling reason for basing the strategy on the single VAX architecture, besides the technical excellence of the product is the belief that we can not build the truly distributed computing system of the 80's with heterogenous architectures. It is possible to build distributed computing networks as we do today, but the homogeneous architecture approach insures that programs may be assigned to any "ode, where they will give the Same results. There is no need for the organizational and computation overhead signified by different manuals, separate training, recompilation of programs, and translation of data among machines in the network.

This strategy is aimed at beating the competition using our existing highly tuned minicomputer hardware and software to support and grow our existing user base. It provides us with a unique offering in the marketplace of the '80's which is likely to be based on the defacto standard IBM 360/370 architecture and the ensuing defacto architectures coming from the semiconductor companies. Since VAX is fundamentally better than either of these architectures, we must make it the standard architecture via transition from the PDP-11, which has been the standard architecture of the 70's.

The strategy is aimed at high volume through multiple channels of distribution, versus a more stable, low growth through support of an existing multi-system, customer base.

How Can We Win Against IBM?

A competive viewpoint is the most important check on strategy. Both the recently accounced IBM 8100 Distributed Processsing system and the System 38 computers are the first computers from IBM that, on the surface, look worth owning. They may be as significant as the 360 and their Selectric typewriter. The System 38 with a 48-bit virtual address is technically unique and may offer the user some very large benefits.

The 8100 is a radical departure from IBM pricing as 0.5 Megabytes of primary memory and a 60 Megabyte disk are $ 29 K. A comparable DEC product sells for several times this now. The 8100 is exactly in the price range of the systems we sell and where we make most of our revenue. It is the second product in this price range within a year; the Series 1 minicomputer family patterned after the 11/04-11/34 was the first product. On the surface, the product is low priced, with lots of capability, but it also has a new communications structure (versus the one we have used substantially unchanged since 1961). This structure permits easy peripheral and terminal interfacing for both the office and factory environment. There is a" extensive range of peripherals, terminals and communications to the 360/370. Since the product is sold by DPD, the strategy seems to keep account control and to make the money on the numerous locked-i", generally overpriced terminals.

IBM will 'have: a 3601370 line in the $100 K to $10 M pi-ice range with lots of plug compatible competitors, several operating systems to support, a large backlog, a newly announced 8100 for Distributed Processing around the mainframe; a System 32/34/38 for Distributed Processing and as a Mainframe for small organizations; the Systems 3 to 15 for Distributed Processing; the System 1 for the would-be minicomputer buyer; the 5100-series Personal Computers for the scientist, engineer, analyst and small business; and several inevitable products for computing in the terminal. All of these are incompatible, except for a communications link and the fact that they all use the 8-bit EBCDIC byte. Products are relatively segmented to customer clauses and different languages are used to further segment and hinder application mobility. Finally, they've sold via DPD and GSD, with Office Products no doubt looking on and waiting for a" entry via electronic mail and word processing.

• The IBM PC + • IBM Tech. PC • IBM 802 PC in 1 1/2 yrs 1/83 GB

While on the surface, the 8100 stands to be IBM's most significant product, it seems to be a serious mistake as it introduces another incompatible computer system with which customers will have to deal. This means that the making of a compatible, fully distributed processing system will be essentially impossible. However, since IBM feels it can not move very rapidly in any product space because of the installed base, product options are limited. Hence new products seem to be highly targeted at specific, new non-IBM markets in a" incompatible fashion to get incremental revenue and growth.

They have a series of ~~these~~ random, point products which they interconnect in a hodge-podge fashion via SNA. A user has no basis to protect software or share info across systems! 1/83

How Can We Win Against Other Competition?

There are established competitors too, such as DG, HP and Prime. DG and Prime have very simple, single architectures and have been most profitable and have grown most rapidly. HP is converging on a single architecture around the 3000, but it will have to be extended eventually. The NOVA will also be extended. The large manufacturers (Univac, Honeywell and Burroughs) which operate with an established base are less profitable, have grown slowly and have multiple, poor architectures. Honeywell, with a simple, but adequate minicomputer architecture seems to be doing well by selling minis to its old line, mainframe base. There is no evidence that they're developing or pursuing the mainframe business actively.

There are probably more significant threats from the companies that can be easily founded to build systems into disks by using the newly announced zero-processor-cost, 16-bit microprocessors which have 22-bit address spaces and the performance of the 11/34-11/45. All of these architectures need to be extended for multiprogramming and to handle larger virtual memories. High level systems, functionally equivalent to our systems such as RSTS can be built easily and cheaply and can quite possibly target a specific existing, trained user base.

There are also the Japanese and TI which can be lumped together because of their similar behavior. Both believe in targeted, high-volume products with forward pricing. Neither have an adequate architecture. TI is strictly limited to 16-bits with almost no escape, and the Japanese are aimed at copying, using U.S. companies to distribute hardware. It's inevitable that they'll supply IBM compatible 360/370's to the Service Bureaus for distribution. This later channel of distribution is another formidable competitor.

The strategy supports very high volumes for dumb, pre-programmed (smart) and programmable (intelligent) terminals using the 11 until VAX is appropriate in terms of price and functionality. In the mid and high priced minis, the strategy is compatibility and volume, phasing as appropriate from 11 to VAX. For example, since there is not a high priced 11 after the 11/74 and the 11/44, there is a phasing to VAX (through COMET) and lower priced 11's based on 11 microprocessor implementation. The question here will be how fast we can provide high performance microprocessors using HMOS and narrower line VLSI technologies.

PRODUCTS IN 1981-82

HARDWARE COOMPONENTS

HMOS LSI, with first "test" product

Interconnection hierarchy with software compatibility
- 1-10 Mhz and/or 10-100 Mhz inter-computer bus ICCS
- 50+ Khz comm.-compatible multidrop for terminals, peripherals, and small systems;
- 0.3-19.2 Khz comm.-compatible for low cost terminals.

Significant competitive memories
- Solid state modules for software
- Low end floppies and low cost tape
- Removeable and low cost disk RL04
- Hi-volume mid- and hi-end disks in R80/R81 with backup

Terminals for everyone!
- Low cost (dumb) and block mode (VT162)
- Office environment for quality printing, electronic mail, and full-page text
- Professional using graphics (and/or color) with target application software
- Factory environment terminals and interface systems

HARDWARE SUBSYSTEMS

- Remoteable printers, job entry, concentrators, sensor-control
- Communications concentrator - Mercury
- Memory (Hierarchy) Management - HSC50 for R80/R81, RL04, tape and disk cache

KERNEL SYSTEMS based on processor-disk-commlrnications (see family tree figure)

780 replaced by Superstar (const. price >3x performance)
- 780 - Memory Manager - Comm. Concentrator
- 780 - Multiprocessor
- 780 - RP/R80-81 + RL02-04
- 780 - RK/RL04

Comet - RP/R80-81 + RL
Hydra (Including Memory Manager - Comm. Concentrator)

Nebula - R80-81 + RL
Nebula - RL02/RL04 (higher cost, quick to market personal computer)

LSI VAX - RL04 - Graphics Terminal (personal computer)

11/74 with no hi end replacement
- 11/74 - multiprocessor
- 11/74 - RP/R80-81 + RL02-04
- 11/74 - RK/RL04

11/44 replaced by HMOS LSI-11 with >256 Kbytes
11/44 - RP/R80-81 + RL
11/44 - RL

11/23 - Unibus Fonz RL replaced by HMOS >256 Kbyte
11/22 - Q-Fonz RL
11/22 - Q-Fonz - RX (floppy)
PDT Fonz - RX (floppy)
PDT Fonz - TU58

Tiny chips, replaced by HMOS tiny <256 Kbytes

SOFTWARE

Diminish the 11 software investment for mature products (RSTS, IAS, MUMPS) and provide only minor enhancements to recent 11 based products (TRAX, SCS-11, PDT Software) to extend the market life and limit the VAX transition risk. Orient new development on VAX and 20 toward IBM compatibility and explicitly invest in tools designed to permit easy customer movement between VAX and 20. DEC 20 development will be aimed at high level tools and applications support. Shift the bulk of the PDP-11 software investment to VAX, tracking VAX hardware and aggressively moving to round out commercial capability.

Develop a single VMS operating system to span the product range if technically and operationally feasible; "low end" products will mask the VMS capability for the unsophisticated users or, if efficiency demands, new code compatible at all interfaces with compilers and utilities will be developed. VMS will offer full mainframe capabilities allowing concurrent batch transaction, processing, and time-sharing, along with limited real-time.

. Provide superior data-base capabilities in the two - three year time frame.

. Focus on data access and data manipulation tools for the non-programmer, heavily based on graphics terminals.

. Provide word processing and electronic mail as applications on the general purpose VAX systems.

. Data integrity will be a feature available independent of high-availability (non-stop) operation through Hydra.

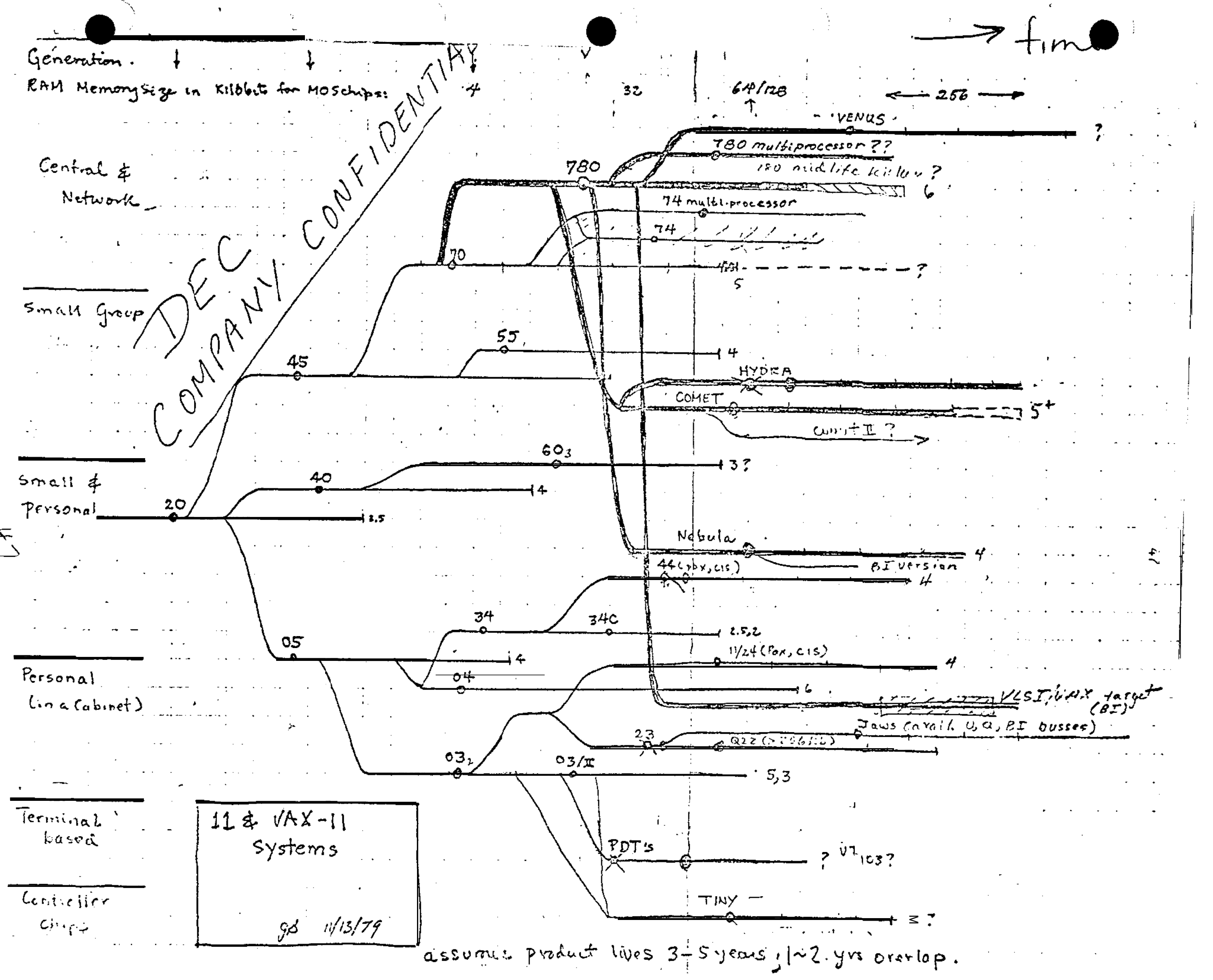
. High-availability (Hydra) will be a standard attribute of VAX systems at the customer option.

. Fire-wall funds to stimulate acquisition of cross-industry applications packages. Provide industry specific applications via internal development or acquisition. Leverage field resources by investing heavily in product quality assurance and self installing systems capacity including remote software update and diagnostic strategies.

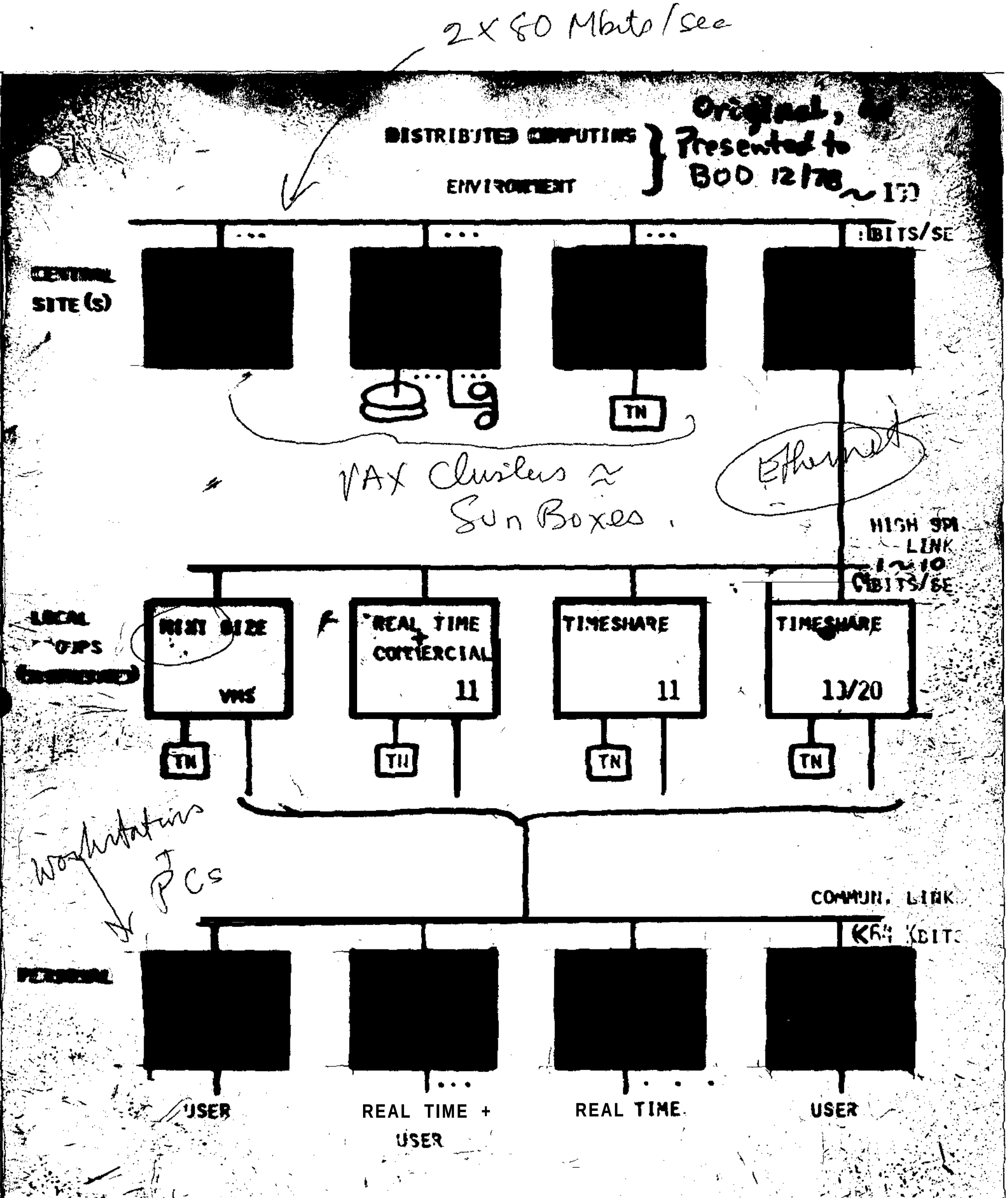
. Move systems-level code for 11 based software (SCS-11, TRAX) to VAX compatibility mode if technically or strategically viable (under investigation now) otherwise provide user-level compatibility via native mode VMS layered products.

. Shift DECNET strategy to strong IBM interconnect and VAX binary image compatibility for distributed processing; constrain PDP-11 DECNET FUNCTIONALITY EXTENSIONS, speedup DEC20 network capabilities.

. Converge on ease of DEC '20 to VAX movement through common language definitions, (common implementations where feasible) common user-level utilities and data conversion routines. For each new DEC 20 or' VAX customer, as time progresses, make the movement between systems more attractive.



→ time
Generation
RAM Memory Size in Kilobits for MOS chips: 4 32 64/128 ← 256 →
Central & Network
Small Group
Small & Personal
Personal (in a Cabinet)
Terminal based
Controller Chips
DEC COMPANY CONFIDENTIAL
20
45
70
780
VENUS
780 multiprocessor ??
780 midlife kicker?
6
74 multi-processor
74
5
55
4
HYDRA
COMET
5+
Comet II ?
60₃
3?
40
4
3.5
Nebula
BI version
44 (ybx, cis)
34
34C
2.5,2
05
11/24 (Fox, cis)
04
6
VLSI VAX target (BI)
Jaws (avail. U, Q, BI busses)
23
Q22
03₂
03/II
5,3
PDT's
VT103?
TINY
11 & VAX-11 Systems
gb 11/13/79
assume product lives 3-5 years, 1~2 yrs overlap.



TN = TERMINAL NETWORK--CONNECTS SIMPLE TERMINALS, MOST PERIPHERALS, AND PERSONAL COMPUTERS, AND PROCESS COMPUTERS

ID#332

Why Change the Current Strategy?

We have arrived at the current strategy by integrating our past customer needs, with the result that nearly every past system we have ever built is being evolved. This evolution creates too many systems with converging functionality. By prolonging the phaseover to VAX, we're unable to invest enough in VAX due to continuing and evolutionary support costs. Also, we're unable to provide applications, or have any slack resources to respond to competitive threats (eg. large micros or focused products such as the 8100).

We are just beginning to get a feel for the expense of putting new software systems in the field, and there are other systems still to come. Since we provide many choices, we find our sales and customers have difficulty deciding what to sell and buy. This makes us difficult to understand and to do business with. Lots of low volume products mean we don't have adequate volume to amortize the start-up manufacturing, sparing and training expenses.

Why Not Aggressively Evolve All Four Base Hardware Architectures?

In reality, our past strategy has been almost a divisional product structure. Customers can choose among the 4 basic hardware computer systems with 2+3+7+1 models and then select the appropriate software system, among 2+2+7+1 software systems for 8, 10/20, 11 and VAX respectively. This gives us several hundred systems. The number of alternatives is too large, resulting in small and decreasing volumes of each of the systems as all architectures are extended to cover a full range that we believe our customers require. We can not afford all the necessary enhancements to support four architectures over the range of size and use that our customers demand.

While any of the architectures can be implemented at any size down to and including LSI chips, there is no significant differential cost of the processor between the 10/20, VAX and an 11 with commercial and scientific instruction-sets.- An evolved 8 to handle the strategic range would even be the same cost. The main differentials are: the cost of the memory to hold the task:: and the size of the operating system software. The 10/20 operating systems have been oriented to generality, and while VMS and TOPS 20 have roughly the same functionality, the 10/20 requires 512K bytes of resident memory, whereas VMS require 256K bytes. This occurs because TOPS 20 has evolved and because of the efficiency of VAX architecture. VMS also has real time capability. Similarly, it is now inappropriate to consider 10/20 based architecture for terminals and personal computers, when compared with VAX, because small problems cannot be encoded to be competitive with modern 8- and 16-bit microprocessors. Furtheremore, extensions to the 10/20 architecture would require basic work in the operating system and languages to build a VAX competitive product.

Why Not Segment Products By Market?

Since the 10/20 has significant commercial software and since it is believed that our customers arc insensitive to architecture, we might simply have a market segmented approach and use 11's at the low eitd and 10/20's in the high end. Lower priced 10/20's would be implemented over time as appropriate.

Our technical users (EDU, ESG and even LDP) do not segment computer purchases into commercial vs scientific. A "control" customer such as DuPont doesn't segment its applications either. Even NASA wants COBOL to off-load their mainframe and to do administrative EDP. Universities likewise want a single machine, and hence the software will be "pulled" into existence. Version 4 of VAX COBOL executes faster than the 20's already.

Since there is basic incompatibility between the 11 and 10 architectures, the migration problem is enormous. Now our large commercial customer base is with 11's. Our users perceive VAX and 11 are of the same family.

The 10/20 still requires basic changes (CIS, 30-bit addressing) to bring it up to VAX performance and capability together with compilers and some basic software (eg. multi-keyed ISAM). TRAX-36 and RSTS 36 will also have to build off our 11 base. In short, while it might bc feasible to build 10/20 software so that our 11 users could meet our strategic goals for distributed processing, we would still fall short of the distributed system we can build with a single architecture as described in a subsequent rationale.

How Do Customers Preceive The Situation?

In mid October, a group at Bell Laboratories, building PBX systems visited us and made the comments:

> "Only you have the basic architecture in VAX to cover the range of products we need for distributed processing. This includes: terminals, offices and large offices.
>
> Give us a truly compatible range of VAX machines, starting with a VAX-on-a-chip and extending through the IBM 3033. (Don't corrupt VAX, since as in the 11, we must preserve our software base, given that the processor is only 4% of the cost.)
>
> The machines must have a reliability and security orientation.
>
> Why don't you do it?
>
> We will help fund the development."

Recent discussions with Stanford, ITT, CERN, NASA(Ames) indicate concurrence even though they are large 10/20 and 360/370 users. MIT is proposing to build a homogeneous VAX-based network. DuPont wants a similar structure, but is less rigid on the need for a homogenous architecture even though they've standardized on the RSTS machine internally for many of their systems. (There's a videotape describing their needs arid ideas.) CERN, and NASA (Ames), for example, feel that the large mainframe may be on the way out as we offer small group-level computing with VAX.. There are probably 10/20 customers who feel strongly that we should base our future on 10/20's. The main reason to focus on the single architecture is that it is part of the 11 family.

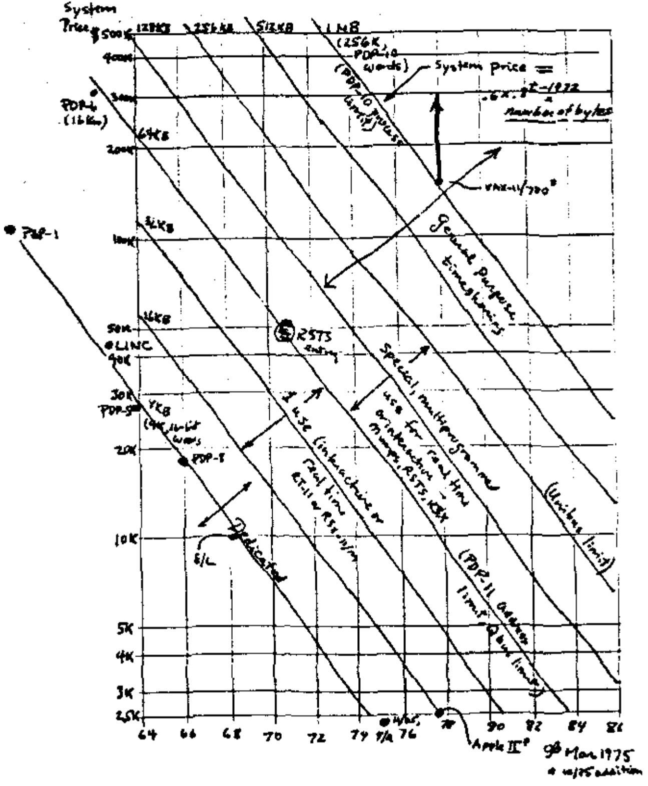
Why Have A Single Architecture? — Homogeneous Computing

There are technical, marketing and economic reasons for choosing a single architecture at this time on which to base a major part of our future. However, this does not mean that we must neglect our 12- and 36-bit user base.

While computer networks can and have been built with heterogenous computers and IBM is betting that it can build distributed computing systems with only similar machines, a single architecture is the most effective for distributed computing systems. The homogeneous (identical) architecture approach insures that software will give the came results no matter where executed and therefore programs may be run anywhere in the network, data stored anywhere and programs moved about in their object form without the overhead of recompiling or translation as data is transferred. This also insures that the human interface to the system remains constant, because identical software is executed in different machines instead of relying on software that is specificed to have identical interfaces (e.g. languages, command languages, file systems, utilities).

From a user viewpoint, the homogeneity is ideal, and the success can be verified by reviewing the history of IBM's decision to build the 360 (and not continue with the 1401 1410, 7070 and 7090 series machines), even though there was an incredible base of these machines. This was also the time that Honeywell established itself with the 200-series and RCA with the 301. The homogeneity provides a simpicity for the entire DEC organization and its customers, and lets us all focus on end use applications rather than choosing a particular operating system and language. Currently, we have too many low level, incomplete choices and the software efforts of us and our users are not focused. An applications base can only be built effectively on a good, stable architecture. 1/43

Economically, a homogeneous architecture is essential because it allows us to concentrate and become a focused, high volume manufacturer and take advantage of learning curves. While 10% learning curves mean a doubling of manufactured quantity causes a 10% decrease in cost, they also imply that having two very similar products at one-half volume causes 10% higher costs in each. There are similar effects of learning in hardware, software and sales training costs, although the learning costs are small in comparison with the logistics and start-up costs associated with our many, different though functionally equivalent, products. We become difficult to do business with in the process.



System Price
$500K
400K
300K
200K
100K
50K
40K
30K
20K
10K
5K
4K
3K
2.5K
128KB
256KB
512KB
1 MB
(256K, PDP-10 words)
(PDP-10 address limit)
64KB
32KB
16KB
4KB
(4K, 16-bit words)
System price = .6 × 2^(t-1972) × number of bytes
VAX-11/780*
General purpose timesharing
RSTS entry
Special, multiprogrammed use for real time and interactive minimaps, RSTS, IDEX
1 user (interactive or real time RT-11 or RSX-11/M
Dedicated
E/L
(Unibus limit)
(PDP-11 address limit)
Q bus limit
PDP-1
PDP-4 (16Kw)
LINC
PDP-5
PDP-8
Wbr, 1/A
Apple II*
64
66
68
70
72
74
76
78
80
82
84
86
9 Mar 1975
* 10/75 addition

Why Base The Architecture on VAX?

Although we went through the arguments in the spring of 1975 when we decided to build VAX instead of building lower cost versions of the 36-bit architecture, we now have a real machine that met it:; development goals and has user acceptance on which to base future products in a natural, evolutionary fashion.

Mostly, the choice of VAX in 1975 was based on having a large, PDP-11 user base. Furthermore, the choice to stay with the 8-bit byte was of convenience because of the IBM and communications worlds.

The VAX architecture was designed to permit the building of a range of machines with sizes that are important to us. Our targeted range of implementation was 1000:1 and this is attainable with an LSI implementation for terminal applications in January 1982. This is why a small page size and simple paging system was chosen, versus a larger page size and more complex scheme that would have been particularly oriented at large systems. However, it would not be wise to build the machine 1000 times as large in 1982, because it would take the system size beyond the suggested $250 K selling price limit and into mainframe price and customer expectations territory. Thus, in January 1982 the LSI VAX could sell for several hundred dollars at a board level. An ECL technology machine might be configured to sell for $ 400 K, giving the 1000:1 in price and a range of 64 Kbytes of RAM and ROM for VMS in the terminal to as much as 32 Mbytes in the large configuration (4000 64 K chips, costing $60K and occupying 20 PC Boards).

VAX was also designed to address the high cost of programming. Already VAX has been acclaimed (by a customer in our ads) as the best machine for implementing software. The large address space eliminates the need for much of the effort we spend encoding large programs into overlays. The architecture has instructions for the important data-types, the addressing is independent of the data-types and the important language constants are built into the hardware. There is clear separation among program and data. The procedure call instruction allows more sub-program sharing than with architectures that are dependent on conventions (e.g. 360 and 10/20) and it eliminates a class of systems programming errors resulting from the multiple assignment of general registers among different programs.

The 32-bit address space of VAX appears adequate for the computing needs in the foreseeable future and there is extension capability given that any special needs arise. The address space and protection modes also give us a capability to run sub-programs written in different languages as a single program. This capability is unique and may turn out to be the single most important attribute of the machine. Since only one other computer has the capability, we don't understand it or how valuable it will be.

Another technical reason is based on the encoding efficiency of the VAX instruction-set. The VAX architecture can encode a Fortran program in about 1/2 to 2/3 the space of a comparable large computer such as a 360 or our 36-bit computer, while providing 32-bit addresses versus 24 or 20 bits of addressing for the 360 or 10/20. Benchmarks in BLISS and FORTRAN show this now, and the Common Family Architecture studies also indicate similar results. While memory cost is decreasing, memory is still a significant fraction of system cost. Three years of cost decline at the historical rate of 20%, is required to get factor of 2 the cost difference back. That is, from a memory cost viewpoint, we have a 3-year cost edge on the market. More importantly, there is a similar effect on performance. By having only 1/2 the bits to move between primary and secondary memories, the performance is higher due to disk-MOS memory swapping bottlenecks.

Finally, we have an 11 user base on which to build that is approximately 7 and 50 times as large as our 36-bit base in terms of installed equipment dollars and installed units.

Why Not Use The 10/20 As The Base?

The software and user base on the 10/20 is the major reason to not arbitrarily reject the architecture. On the other hand, since the 11 user base is larger and has grown more rapidly, its software base is larger and we have to protect and build on it as a higher priority.

Right now, the 10/20 requires incremental investment to make it competitive with VAX and the rest of the mainframe market. Extension to provide a large address space, to extend the floating point range to fulfil customer commitments, and to give a competitive commercial instruction set for COBOL are needed. Making these hardware investments requires comparable software investments and we must again wait to compete because there is a new machine and software to support.

Subsequent implementations for small systems will be expensive both in terms of new software and start-up because TOPS 20 has been oriented toward large mainframe generality. Smaller systems will require contractions. Also it stands to only cloud the market more as alternatives for mid-range systems will include 2 VAX and 1 or 2 11-based systems. As small systems are implemented there is a need for compatibility with the even larger 11 base.

Why Distributed Computing?

Distributed computing keys off our strength in interactive computing through timesharing, small systems, real time computation, terminals, and networks. Furthermore, we believe this is what our customers want. The issue is not distributed computing, but solving the problems that it creates.