A Historical Look at the VAX: The Economics of Microprocessors [Part I]

By: John Mashey (jmashey@realworldtech.com)

Updated: 01-24-2006

Editor's Note: This series of articles first appeared last summer in the comp.arch newsgroup, in a series of posts by John Mashey. Since then, it has been updated, edited and enriched with additional material and graphs by David Kanter, all with permission of the author.

Introduction

The VAX was an orthogonal, 32-bit CISC instruction set architecture and an associated family of CPUs and systems, produced by the Digital Equipment Corporation. The VAX is considered to be the quintessential CISC architecture and VAX systems (often called VAXen) typically used an operating system known as VMS and were shipped from 1977 till 1992, when the line was discontinued in favor the Alpha. In the wake of the death of the Alpha architecture, there have been many discussions regarding the feasibility of cheap, fast and timely VAXen in the early 1990's. Many have asserted that if DEC had continued with the VAX, rather than transitioning to the Alpha that the company and its products would be in better shape today. Unfortunately, most of this discussion is filled with fantasies about what could have been; the issue is not just what was technically feasible, but what was cost-competitive and time-to-market competitive.

This series of articles will examine the issues confronting DEC and the VAX architecture in the early 1990's. The first part covers some of the economics of systems and microprocessors. The second part in this series will address the competitive challenges that DEC faced around the time of the Alpha. The third and last part will compare the instruction set difficulties facing out-of-order VAX, x86 and s/360 microprocessors and bring everything together.

The Design Cost of Microprocessors

Suppose it costs $50,000,000 to design a chip and get into to production. The amortized cost per chip for engineering versus total unit volume is shown below in Figure 1.

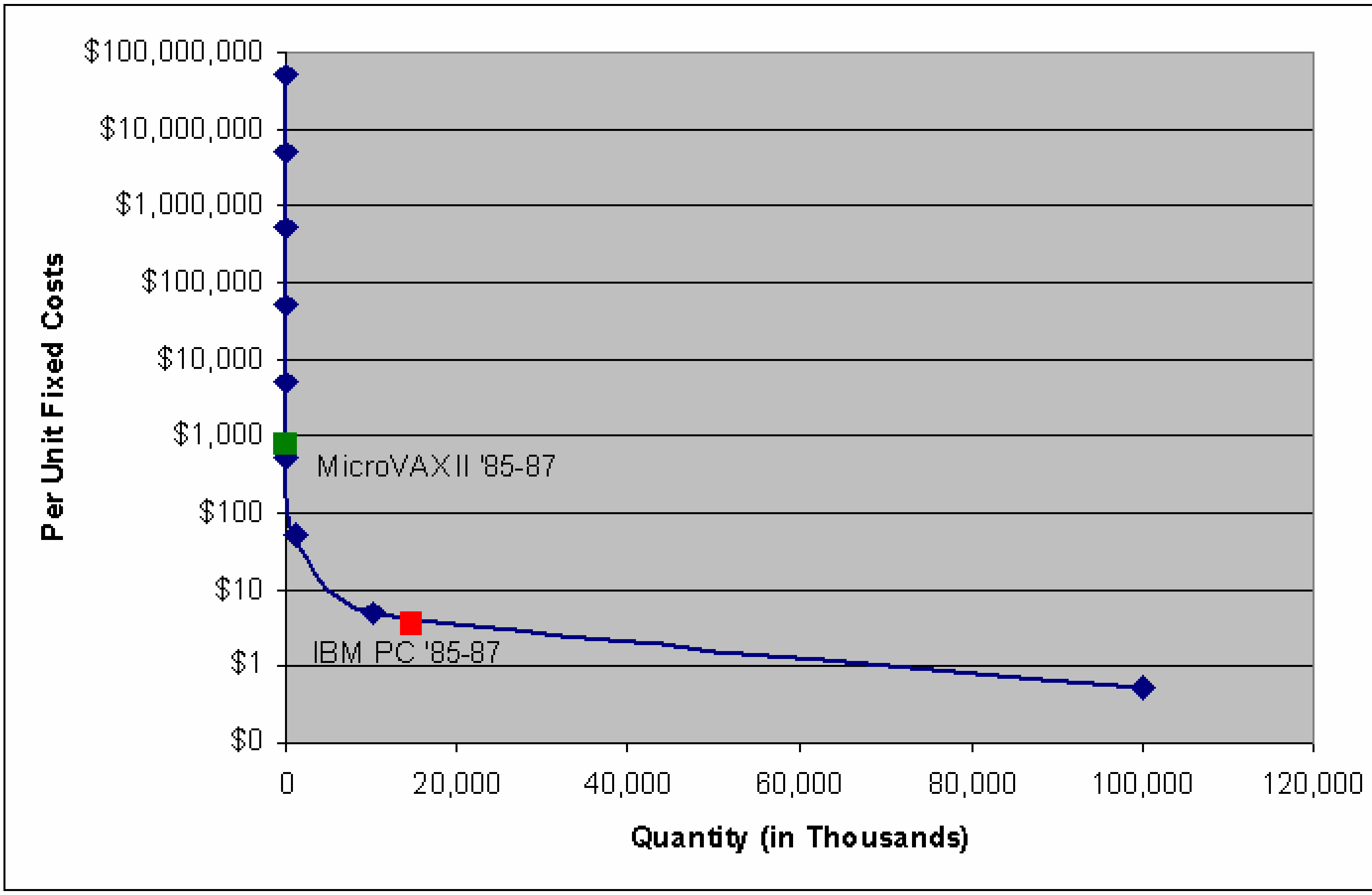


Figure 1 – Hypothetical Development Costs per CPU

Alternatively, if you can sell 5,000,000 chips, then you could spend $500,000,000 on development, and still only have an engineering cost per chip of $100. It is easy to see that as a result of these economics, high volume designs achieve high performance with relative ease. Additionally, higher production at a fab increases the quality of silicon, further improving the performance of a high volume design.

To put this in perspective, I have included the MicroVAX-II and IBM PC volumes over the periods 1985-1987. Under this hypothetical model that assumes equal development costs, the difference in per unit R&D costs is roughly two and a half orders of magnitude.

x86s Are Fast Because They Have Volume

As a result of the economics outlined above, x86s tend to have a very distinct advantage against other MPU families. Anne & Lynn Wheeler recently pointed out in comp.arch that VAX unit volumes have traditionally been quite low. The MicroVAX II, a very successful product, shipped 65,000 units over 1985-1987; by comparison over 1985-1987, there was roughly 14,000,000 IBM PCs and clone systems sold. Intel's large x86 volumes mean that they can spend more time and effort optimizing their CPUs, and probably still have lower average costs.

If the unit volumes are inherently low, you have to get profit based on *system* value that yields unusually high margins, so that the system profit essentially subsidizes the use of such chips. This strategy works quite well for a system vendor when the market dynamics and customer switching costs allow high margins, i.e., IBM mainframes to this day, and VAXen in the 1980s. The first MIPS were designed using 2 VAX 11/780s, plus (later) an 8600, plus some Apollos; the VAXen seemed expensive, but they were what we needed, so we paid.

Systems Companies

Mainframes and minicomputer systems companies thrived when the design style was to integrate a large number of lower-density components, with serious value-added in the design of CPUs from such components. For an example of this, look at a VAX 11/780 set of boards.

As microprocessors came in, and became competitive, systems companies struggled to adapt to the change in economics, and to maintain upward compatibility. Most systems companies had real problems with this. There were pervasive internal wars, especially in multi-division companies. Each faction would espouse one of the following strategies:

- We can do this ourselves, especially with new ECL (a faster, hotter and more expensive circuit design style) or even GaAs (Gallium Arsenide, a faster, more expensive and fragile semiconductor material). This view was common among those who worked on supercomputers, mainframes and minicomputers chasing mainframes.
- We can build CMOS microprocessors that are almost as fast as ECL, much cheaper and with better features, functions, and performance than commodity products. IBM, DEC, HP and later Sun would all pursue this strategy.
- We should put our money in system design and software, and buy microprocessors. Apollo, Convergent, Sun, SGI, Sequent, and various divisions of older companies tried this approach.

Of course, later there came:

- We should buy microprocessors, add value in systems design, and use Microsoft. This lead to IBM’s PC division, Compaq, etc.
- We should minimize engineering costs and focus on distribution i.e. Dell.

Many companies pursued one strategy too many. Most of the minicomputer vendors went out of business. IBM, DEC, and HP were among the few that actually had the expertise to do CMOS microprocessor technology, albeit not without internal wars. I was involved in dozens of these wars. One of the most amusing was when IBM friends asked me to come and participate in a mostly IBM-internal conference at T.J. Watson, circa 1992. What they really wanted, it turned out, was for somebody outside IBM politics (i.e., "safe") to wave a red flag in front of the ECL mainframe folks, by showing a working 100Mhz 64-bit R4000.

Semiconductors, Systems and Fabs

In the 1980s, if you were a semiconductor company, you owned one or more fabs, which were expensive at the time, but nothing like they are now. You designed chips that either had huge volumes, high ASPs (Average Selling Prices), or ideally, both.

Volume production provides your fab engineers with process experience and improves the yields of chips per wafer. Volume also amortizes both the fab cost and the design cost. Unfortunately, systems vendors with only one fab to produce chips face a nasty problem. If the fab is not full, you have a lot of capital tied up that is not being optimally used and this will make management upset. To illustrate this point, Bob Colwell, the Chief Architect of the Pentium Pro, said in a lecture:

"At some point, they [process people] are ready to start running production silicon through their new process. At that very day, you had better have a chip for them to start cranking through their fab, because otherwise, Craig Barrett himself will call you on the phone and express his unhappiness with you and his predictions for your career at that point. It's really really expensive if the process is ready and the chip is not."

On the other hand, if the fab is full, you are capacity constrained and unable to fully satisfy your potential customers, which will cause the same problems, except that the sales people will be upset too.

When you build a fab, first you run high-value wafers. Once the fab has aged, and is no longer state-of-the-art, you can run parts that do not need the most advanced technology. For example, Intel has many older 200mm plants that once produced cutting edge MPUs, they are now used to create chipsets for MPUs made in newer fabs.

In the 1980s, if you wanted to have access to leading-edge VLSI technology for your own designs, either:

- You were a semiconductor company, i.e., the era of "Only real men have fabs". Motorola, Intel and AMD all fell into this category.
- You were a systems company big enough to afford the fab treadmill. For example, IBM always had great process technology. DEC usually had 1-2 CMOS fabs (more on that later) and HP had at least 1, but sometimes there was conflict between the priorities for high-speed CPU design and high-volume lower-cost parts. Fujitsu, NEC, etc. were also chip and systems companies. In any case, you must carefully amortize the fab costs.
- You were a fabless systems company that could convince a chip company to partner with you, where your designs were built in their fabs, and either you had enough volume alone, or (better) they could sell the chips to a large audience of

others. For example, Sun and TI have had an arrangement like this, which still exists.

- You were a small systems/chip company (MIPS) that was convincing various other systems companies and embedded designers to use your chips. A few chip partners would make long-term deals to produce the chips, sell most of them to other companies, and as desired, have licenses to make their own variations of the designs. The motivation was that a chip salesperson could get in the door with CPUs, and be able to sell many other parts, like SRAMs (this worked for IDT with MIPS and Cypress with SPARC), or be able to do ASIC versions (this worked for LSI Logic).

In MIPS case, the first few years, the accessible partners were small to medium chip vendors, and it was only in 1988 that we were able to (almost) do a deal with Motorola and were able to do ones with NEC and Toshiba, i.e., high-volume vendors with multiple fabs.

Why wouldn't a company like Sun just go to a foundry with its designs, or in MIPS case, why wouldn't it just be a normal fabless semiconductor vendor? The answer is that accessible foundries, geared to producing outside designs, with state-of-the-art fabs didn't really exist. TSMC was founded in 1987, and it took a long time for it to become a truly attractive offering.

To Fab or Not To Fab

If you own the process, you can alter it somewhat to fit what you are building. If your engineers are close with the fab process people, and you have clever circuit designers, you can do all sorts of things to get higher clock rates. Otherwise, you use the factory design rules or maybe you can negotiate a little with them. In any case, there is a tradeoff between committing the capital to own a fab and achieving higher clock rates, or preserving your capital, not owning the fab, and losing the ability to produce highly tuned designs. To give a rough notion of the trade-offs, in the same process size, the speed of a process (measured in FO4 latency) can vary by a factor of 3, between the best processes at foundries like TSMC or UMC, and fabbed manufacturers like Intel or IBM.

An Overview of Systems Companies

There are two major approaches to system and CPU design, the proprietary approach:

- We're designing these chips for our systems, running our OSes and compilers. We might sell chips to a few close partners for strategic reasons.

and the merchant approach:

- We expect to use these chips in our systems, but we expect that large numbers will be used in other systems, with other software. We will include features that will never be used in our own systems. We will invest in design documentation, appropriate technical support, sampling support, debugging support, software licensing, etc. to enable others to be successful.

IBM still has fabs, but of course IBM Microelectronics does foundry work for others to amortize the fab cost. POWER was really geared to RS/6000; PPC made various changes to allow wider external use, and IBM really pushed to achieve high volumes.

Sun never had a fab and did a lot of industry campaigning to spread SPARC. However, outside of a few close partners, most of the sales of SPARC chips went to Sun.

HP has sold PA-RISC chips to a few close partners, but in general, never was set up in the business of selling microprocessors.

MIPS started to do chips, but had enough work to do on systems that it needed to build systems. Going back to the volume issues addressed earlier, MIPS needed systems revenue, since there was not enough volume to make money on chips alone. This is a classic example of how a systems business can work at low volumes, whereas the chip business does not.

DEC was somewhat naturally positioned as a systems vendor, based on its original business (modules). DEC never really had a merchant chip mindset, although the Alpha, of course, was forced to try to do that to achieve sustainable volume. Somebody suggested they should have been selling VAX chips, and that may be so; however it is really hard to make that sort of thing happen. It requires serious investment to enable other customers to be successful, it requires the right mindset, and it is really hard to make that work in a big systems company.

As a hypothetical example to illustrate the stark difference between the two business models, suppose I am DEC and I sell VAX chips to a third party. What OS will they run? VMS; OK we will license that. What sort of systems do they want to build? Lower-cost minicomputers to sell to the VAX/VMS installed base…actually, it looks like we're out of VAX chips this quarter, sorry.

Just in case you don’t think this a realistic example, consider this example from the real world. At one point, Sun convinced Solbourne to use SPARC, and Solbourne designed their own CPUs and built SMPs. If a Sun account wanted an SMP and somebody like SGI was knocking at the door, Sun would point at Solbourne to keep SPARC in place. But if Solbourne was infringing on a Sun sale, it was not so friendly – I once got a copy of a Sun memo to the sales force about how to clobber Solbourne.

It is very difficult for big systems vendors to sell their proprietary CPUs. The only option that makes business sense is to find other, non-competitive or complementary vendors that would be interested in using such CPUs for their own systems. The most successful example of this sort of partnership is Apple’s usage of IBM’s PPC architecture. Probably the most interesting case for the Alpha was its use in the Cray T3 systems, fine supercomputers, but not exactly high-volume.

Chip Design and Economics

Most people probably know the old project management adage: "features, cost, schedule: you pick two, I'll tell you the other."

In CPU design, there are currently several alternatives, arranged in order from cheapest to most expensive to design they are:

- FPGA
- Structured ASIC
- ASIC, full synthesized logic
- Custom, with some synthesized and some custom logic or layout design and maybe with some special circuit design.

Better tools help, but they are expensive, especially because people pushing the cutting edge tend to need to build some of their own. An FPGA will be big, use more power, and run at lower clock rate than the alternatives. The more customized a chip is, the faster it can go, but it either takes more people, or longer to design, and usually both.

Larger companies, like Intel, sometimes have one design team produce an original device using a lot of synthesized logic, and then another team comes right behind them. The second team will use the same logic, but tunes the critical paths for higher clock rates, shrinks the die using more custom design, works to get higher yields, etc. Put another way, if you have enough volume, and good ASPs, you can afford to spend a lot of engineering effort to tune designs, enough to overcome ISA problems.

Acknowledgements

We would like to take this opportunity to thank several people who have been kind enough to provide us with input on this series of articles:

- Bob Colwell
- Andy Glew
- Andy Goldstein
- Dileep Bhandarkar
- Sue Skonetski

We appreciate your input and expertise in your respective areas.

A Historical Look at the VAX: DEC, NVAX, Alpha and Competitors [Part II]

By: John Mashey (jmashey@realworldtech.com)

Updated: 06-20-2006

A Historical Look at the VAX: DEC, NVAX, Alpha and the Competition [Part II]

Editor's Note and Introduction:

This series of articles first appeared last summer in the comp.arch newsgroup, in a series of posts by John Mashey. Since then, it has been updated, edited and enriched with additional material and graphs by David Kanter, all with permission of the author. John Mashey is a well respected operating system developer and computer architect; he has worked at Bell Labs, Convergent, MIPS and retired from SGI as VP and Chief Scientist.

The first article in this series largely dealt with the economics of microprocessors and computer systems. In particular, the trade-offs between a fabbed and fabless model were discussed, and the incredible benefits of volume for a semiconductor company. This second article looks at the competitive position of the VAX during the rise of RISC architectures, in the late 1980's and early 1990's.

DEC (at least some people) understood the importance of VLSI CMOS and that good silicon design can have excellent results. DEC had excellent CPU and systems designers, software people, and invested in fabs (for better or worse – some of us could never quite figure out how they could afford the fabs in the 1990s). They had some superb circuit designers, who even impressed some of the best circuit designers I've known.

However, in the 1980s, they never had more than about 100 VLSI CPU designers, which in practice meant that at any one time, they could realistically be doing one brand new design, and one shrink or variation. Of course, they were also doing the ECL VAX9000 mainframe, but that was a whole different organization.

The problem that DEC faced was that their VAX cash cow was under attack, and they simply couldn't figure out how to keep the VAX competitive. This was first seen in the technical markets, where PA-RISC, SPARC, and MIPS started out. Eventually, PA-RISC migrated to the commercial market and began to pose a threat there as well. I think Robert Supnik's foreward in the Digital Technical Journal described this reasonably well.

As a good head-to-head comparison, NVAX and the Alpha 20164 were built:

- in same process

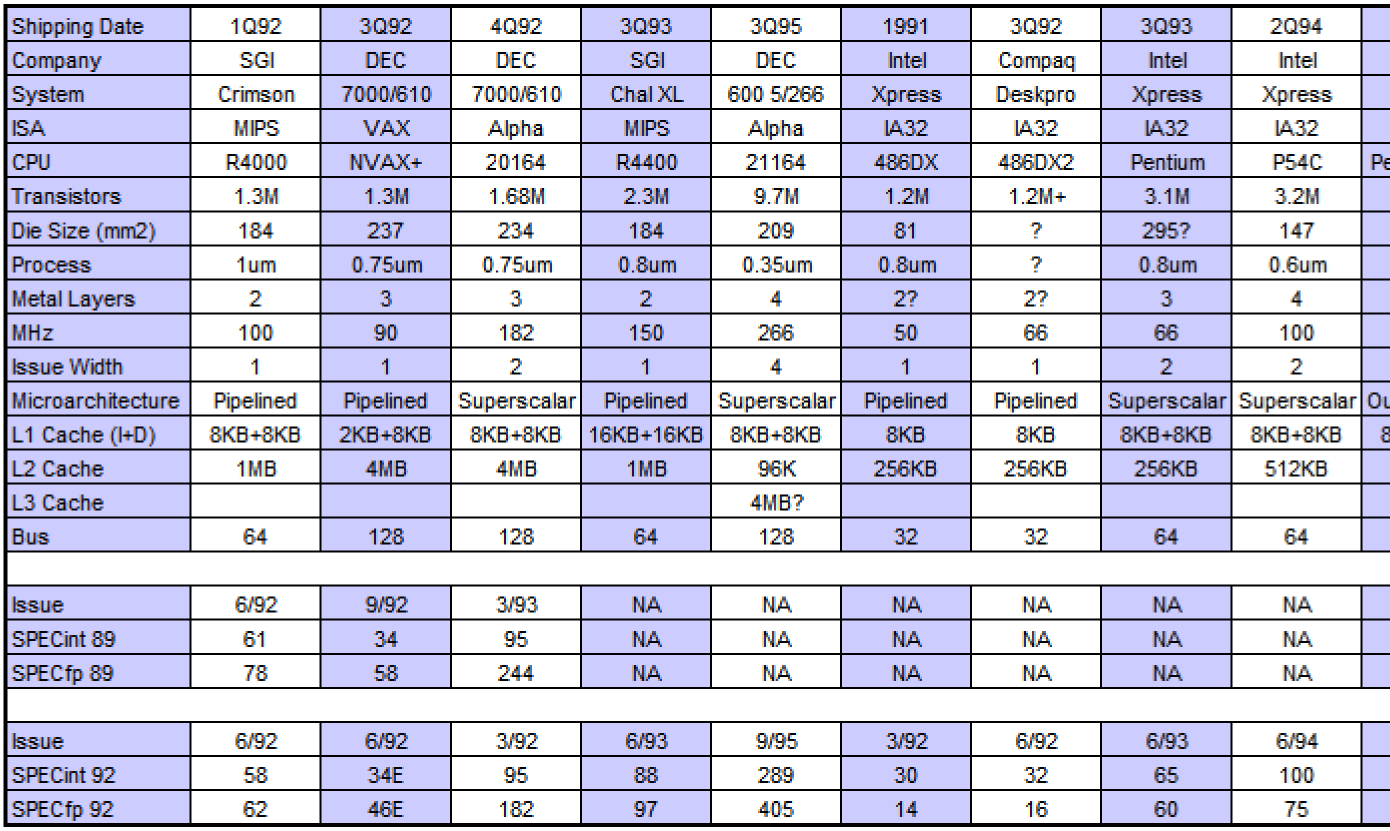
- about the same time
- with the same design tools
- with similar-sized teams

Moreover, the NVAX team had already implemented pipelined CMOS VAXen before, and had a long history of diagnostics, test programs, behavioral statistics, etc. while the Alpha team had far fewer resources and experience to call upon. So the extent that there was an advantage, it was certainly in favor of the NVAX team.

As a result of the ISA differences between VAX and Alpha, the NVAX team had to spend a lot more effort on micro-architecture, whereas the Alpha team could spend that effort on aggressive implementation. The NVAX/NVAX+ was only able to hit 80-90MHz, while the 21064 was running at 200MHz and was superscalar to boot. The end result was that the 21064 vastly outperformed the NVAX/NVAX+.

The Competition's Performance Vexes DEC

The prior example of the NVAX/NVAX+ and the Alpha 21064 serves to illustrate the implementation advantages of a clean RISC architecture. Despite similar resources, which if anything favored the NVAX/NVAX+, the Alpha 21064 performed roughly 3x better in SPECint89 and 5x better on SPECfp89. RISC designs were much simpler which lead to substantial performance benefits, consequently this created a difficult competitive environment for the VAX. The table below includes quite a few systems from the early 1990’s, most of which were released around 1992 (give or take a year) and several systems from mid 1990's as well.



Shipping Date	1Q92	3Q92	4Q92	3Q93	3Q95	1991	3Q92	3Q93	2Q94	
Company	SGI	DEC	DEC	SGI	DEC	Intel	Compaq	Intel	Intel	
System	Crimson	7000/610	7000/610	Chal XL	600 5/266	Xpress	Deskpro	Xpress	Xpress	
ISA	MIPS	VAX	Alpha	MIPS	Alpha	IA32	IA32	IA32	IA32	
CPU	R4000	NVAX+	20164	R4400	21164	486DX	486DX2	Pentium	P54C	Pe
Transistors	1.3M	1.3M	1.68M	2.3M	9.7M	1.2M	1.2M+	3.1M	3.2M	
Die Size (mm2)	184	237	234	184	209	81	?	295?	147	
Process	1um	0.75um	0.75um	0.8um	0.35um	0.8um	?	0.8um	0.6um	
Metal Layers	2	3	3	2	4	2?	2?	3	4	
MHz	100	90	182	150	266	50	66	66	100	
Issue Width	1	1	2	1	4	1	1	2	2	
Microarchitecture	Pipelined	Pipelined	Superscalar	Pipelined	Superscalar	Pipelined	Pipelined	Superscalar	Superscalar	Ou
L1 Cache (I+D)	8KB+8KB	2KB+8KB	8KB+8KB	16KB+16KB	8KB+8KB	8KB	8KB	8KB+8KB	8KB+8KB	8
L2 Cache	1MB	4MB	4MB	1MB	96K	256KB	256KB	256KB	512KB	
L3 Cache					4MB?					
Bus	64	128	128	64	128	32	32	64	64	
Issue	6/92	9/92	3/93	NA	NA	NA	NA	NA	NA	
SPECint 89	61	34	95	NA	NA	NA	NA	NA	NA	
SPECfp 89	78	58	244	NA	NA	NA	NA	NA	NA	
Issue	6/92	6/92	3/92	6/93	9/95	3/92	6/92	6/93	6/94	
SPECint 92	58	34E	95	88	289	30	32	65	100	
SPECfp 92	62	46E	182	97	405	14	16	60	75	

Table 1 – Performance and Characteristics of VAX and Competing Systems

I started with SPEC89 numbers for VAXen, because I cannot find SPEC92 numbers. Then I made a gross estimate of the equivalent SPEC92 scores, so that all of the machines are on the same scale, noting of course that benchmarks degrade over time due to compiler-cracking. The asterisked SPECfp89 scores include the effect of the heightened matrix300 numbers (NB: matrix300 was a subtest that certain compilers were able to 'crack', producing what some felt were unrealistic performance gains, this also happened to 179.art in SPECcpu2000). I used the highest NVAX numbers I had handy, from my old paper SPEC Newsletters. I am ignoring costs, and the dates in the table must be taken with lots of salt, for numerous reasons, and as always SPECint and SPECfp are not everything (spoken as an old OS person).

The NVAX shipped in 4Q91, and the NVAX+ in 1992. The first R4000s shipped in systems in 1Q92, so these are contemporaneous, as they are with 486DX and 486DX2.

The NVAX+ performs at about 75-80% of a MIPS R4000 on integer and FP here, despite using a better process (0.75um, 3-metal versus 1.0um, 2-metal), a larger die (237mm^2 versus 184mm^2), and being 32-bit rather than 64-bit. As a side note, it is somewhat an accident of history and business arrangements that the R4000 was done in 2-metal. The original plan called for a 2-issue superscalar MPU, but the 2-metal process forced the designers to opt for a super pipelined, 1-issue instead. As a result, the R4000/R4400 often had lower SPECfp numbers than the contemporaneous HP and IBM RISCs. To compensate, it sometimes had better integer performance, and sometimes could afford bigger L2 caches, because the R4000/R4400s were relatively small.

In any case, with respect to SPEC FP performance, in late 1992, the fastest NVAX+ was outperformed by IBM, HP, MIPS, Sun (maybe), and Alpha (by 3-4X). The NVAX+ was 3X faster than a 66MHz 486DX2. In SPECint, in late 1992, the NVAX was outperformed, generally by the RISCs...but worse, there wasn't much daylight between it and a 66MHZ 486DX2, or even a 50MHz 486DX.

The real problem of course (not just for the VAX, but for everyone), was the last entry in Table 1. Intel had the resources and volume to "pipeline" major design teams (Santa Clara and Portland) plus variants and shrinks, and there was an incredible proliferation in these years. Hence it is also instructive to compare the NVAX+ with Intel's entries, which are found in the right part of the table.

Suppose you were a VAX customer in 1992. If you were using VAX/VMS for:

- Commercial application, then you were committed to VMS for a long time.
- Technical applications (FP intensive), then RISC competitors keep coming by with their rather attractive numbers

If you were using Ultrix (Digital’s UNIX for VAX) for:

- FP applications, then the RISC competitors were looking pretty attractive
- Integer applications, not only are the RISC competitors looking good, but Intel was getting close to parity on performance

Could DEC Have Done it?

I'm not going to comment on DEC's handling of Alpha, fabs, announcements and alternate strategy variations. But this part should make clear that there was real pressure on the VAX architecture, from above (in terms of performance) and below (Intel coming up).

One might imagine, that had there been no Alpha, and had everybody at Hudson had kept working on VAXen, that they could have gotten a 2-issue superscalar (like the Pentium), in 1994. Alternatively, they could have made an out-of-order CPU (like Pentium Pro) in 1996 perhaps as well as doing the required shrinks and variants.

From the resources I've heard described, I find it very difficult to believe they could have done both a Pentium-class chip and a Pentium Pro-class chip simultaneously (and note, world-class design teams do not grow on trees). I could be convinced otherwise, but as one of the NVAX designers says, only by "members of the NVAX and Alpha design teams, plus Joel Emer", i.e., well-informed people.

Moreover, technical feasibility is not enough by itself. The real challenges would have been sustaining a customer base in face of incredibly strong competitive pressure mentioned previously and getting these hypothetical superscalar and out-of-order VAXes to market in a timely fashion.

In Part III, I'll sketch some of the tough issues for implementing the VAX, as best I can. In particular, I will note the ISA features that might make things harder for VAX than for x86, to do 2 or 4-issue superscalar, or a full blown out-of-order design. In particular, what this means is that you can implement a type of microarchitecture, but it gains you more or less performance dependent on the ISA and the rest of the microarchitecture. For instance, the NVAX design at one point was going to decode 2 specifiers per cycle, but it was found to add too much complexity and only get a 2% performance increase.

Robert M. Supnik,
Corporate Consultant, Vice-President,
Technical Director, Engineering

It all started with eight people in a conference room. [1]

The time was the summer of 1988. Digital Equipment Corporation had just closed the best fiscal year in its history, with record revenues and profits. Digital's VAX systems were the most widely used timesharing systems in the industry and were the "blue-ribbon standard" for mid- range computing. Digital was the second-largest workstation vendor. The company had just introduced the VAX 6000 system, its first expandable multiprocessor, was developing a true VAX mainframe, and had decided on a rapid thrust into RISC workstations to capitalize on that growing market. What could possibly go wrong?

Nonetheless, senior managers and engineers saw trouble ahead. Workstations had displaced VAX VMS from its original technical market. Networks of personal computers were replacing timesharing. Application investment was moving to standard, high-volume computers. Microprocessors had surpassed the performance of traditional mid-range computers and were closing in on mainframes. And advances in RISC technology threatened to aggravate all of these trends. Accordingly, the Executive Committee asked Engineering to develop a long-term strategy for keeping Digital's systems competitive. Engineering convened a task force to study the problem.

The task force looked at a wide range of potential solutions, from the application of advanced pipelining techniques in VAX systems to the deployment of a new architecture. A basic constraint was that the proposed solution had to provide strong compatibility with current products. After several months of study, the team concluded that only a new RISC architecture could meet the stated objective of long-term competitiveness, and that only the existing VMS and UNIX environments could meet the stated constraint of strong compatibility. Thus, the challenge posed by the task force was to design the most competitive RISC systems that would run the current software environments.

Key groups in Engineering responded to this challenge. A cross-functional team from hardware and software defined the basic architecture. Advanced development teams began work on the knotty engineering problems: in the semiconductor group, the specification and design of a fast microprocessor, and the automatic translation of executable binary images; in the operating systems groups, on the porting of ULTRIX and of VMS (which was not portable!); and in the compiler group, on superscalar code generation. In the fall of 1989, Alpha became an officially sanctioned advanced development program.[2] In the summer of 1990, it transitioned to product development.

From the original core in semiconductors, operating systems, and compilers, work expanded throughout Engineering. The server and workstation hardware groups specified and started designing a family of systems, from desktop to data center. The networks group began porting DECnet, TCP/IP, X.25, LAT, and the many other networking products. The layered software group inventoried the existing portfolio of products and prioritized the order and importance of delivery. The research group pitched in by designing an experimental multiprocessor as a software development testbed.

In parallel with the engineering work, marketing, sales, and service teams worked closely with business partners and customers to shape the deliverables and messages to meet external requirements. These teams briefed key customers and partners early in the development process and incorporated their advice into the development program. Ongoing partner and customer advisory boards provided regular feedback on all aspects of the program and helped shape two critical extensions of the original concept: the open licensing of Alpha technology; and the porting of Windows NT.

Taken together, the scope of the Engineering effort, the need for Marketing, Field, and Service involvement, and the high degree of customer and business partner participation, posed unique management challenges. Rather than organize a large scale hierarchical project, the company chose to manage Alpha as a distributed program. A small program team used enrollment management practices and strict operational discipline to coordinate and inspect activities across the company. This networked approach to management gave the program both flexibility and resiliency in the face of rapidly changing business and organizational conditions.

The work of Engineering, Manufacturing, Marketing, Sales, and Service came together in November 1992 with the announcement of the Alpha AXP systems family: seven systems, three operating systems, six languages, multiple networks, migration tools, open licensing of technology, hardware and software partnerships, and more than 2000 committed applications. Today, Alpha AXP embodies a fundamental repositioning of Digital Equipment Corporation to be the technology and solutions leader in twenty-first century computing: a company dedicated to meeting customers' needs with the best computing, business, and service technology available. The delivery of Alpha AXP required the largest engineering program in Digital's history, spanning more than twenty Engineering groups worldwide. This issue of the Digital Technical Journal documents just a few of the hundreds of projects involved in bringing Alpha to fruition; future issues will continue the story.

1. The Corona Borealis conference room in the LTN1 facility in Littleton, Mass. LTN1 was chosen because it was the geographic epicenter of the arc of Digital engineering facilities on Massachusetts Route 495, the Corona Borealis because it was the only conference room with windows.

2. After going through more than one name change. The original study team was called the "RISCy VAX Task Force." The advanced development work was labeled "EVAX." When the program was approved, the Executive Committee demanded a neutral code name, hence "Alpha."