

GP1.21

| d | i | g | i | t | a | l |

INTEROFFICE MEMO

To: Charlie Barker  
Demetrios Lignos  
Jim Mars  
Bob Pedersen  
Nelson Velez

Date: 16 March 1982  
From: Geof Potter  
Dept: Low End VAX Sys Dev  
Ext.: 247-2181  
Loc/Mail Stop: TW/B02

CC: Distribution

Subject: ALL THOSE VOLTAGE CHANNELS on BI

A point of information. In order to support Unibus options, AZTEC, TU58, and BI modules, we will require up to 12 voltage sources.

	+5V	Main Logic Channel	
	+5B	Battery Supported Channel for Memory	
	+LLV	Low Logic Voltage Channel for Gate Arrays	
	+12V	TU58 Motor, NI, EIA Serial Lines	
pich <sup>2</sup> {	-12V	EIA Line Drivers (if on-board regs don't go)	
	+15V	Unibus Comm	
	-15V	Unibus Comm	
	+24V	AZTEC Servo	} AC
	-24V	AZTEC Servo	
	-5V	ECL Devices on Higher Performance Modules	
	FAN	Source for Fan Drive, Probably Battery Supported	
	TOY CLK	Battery Supported Channel for TOY Clock	

*or put it on a TOY Board!*

This does not account for any special isolation needs that may arise due to pulse loading of individual channels. There may be more channels required to provide such isolation.

The system fans will be powered from available channels such as +/-24V, but battery support will probably be necessary causing complications.

It is possible to eliminate +/-15V by substituting +/-12V on the Unibus backplane. That leaves 10 different channels.

There will be some Unibus options that won't work on 12V, but that is not expected to be much of a problem. Most are EIA Comm no memory.

The BI backplane is intended to have only three channels to attempt to build some discipline into system design for the future. I worry that any possible discipline will disappear with the first Scorpio product featuring "all those voltage channels" necessary to support Unibus also.

The above speaks for system design as a desperate need if BI is to become an economically viable structure.

↳ yes

MAR 18 1982

To: Demner  
Streich  
Demetrios  
Niko Tilleban  
This is ridiculous!!  
Gordon

FYI BS

MAR 17 1982

OT2.5

| d | i | g | i | t | a | l |  
+-----+

I N T E R O F F I C E M E M O

To: Distribution

Date: 5 March 1982  
From: Omur Tasar *Omur*  
Dept: Low End VAX Systems Dev  
Ext.: 247-3027  
Loc/Mail Stop: TW/B02

Subject: MINUTES OF BI BUS SPECIFICATION REVIEW

A meeting was held on 8 February 1982 to review the BI bus specification as it is defined today, identify its pros and cons, and make a recommendation to keep BI as is or not. Those in attendance were:

Dileep Bhandarkar  
Frank Bomba  
Ted Gent  
Brian Hannon  
Dick Hustvedt  
Stan Lackey  
Wayne Parker  
Dave Rodgers

Steve Rothman  
Barry Rubinson  
Ted Semple  
Bob Stewart  
Bill Strecker  
Omur Tasar  
Mike Titelbaum  
Dale Troppito

In summary, the task force considered two alternatives: 1) BI as is, and 2) BI prime as a packet bus. With two exceptions, the task force recommended to go with the first alternative. It was felt that BI was a technically adequate solution and that its shortcomings could be overcome. Going with Alternative 2 was thought of as a better technical solution but a longer term, higher risk project. Baring in mind time to market and 2.5 years of investment into BI, the recommendation was to stay with it.

The Scorpio Program Office has since decided to proceed with the BI as is (see memo from D. Lignos, "BI Task Force Recommendations and Implementation Direction", dated 10 February 1982) and our plans have been revised to reflect that decision.

The purpose of the attached document is for your records only. It is almost a transcript of what transpired at the task force review. The conversations were recorded and the transcript has been edited only because at times the distant voices were not audible enough for the recording machine. A summary of the discussion was also done where feasible.

---

The meeting started with a presentation by Frank describing where we are with the BI spec and BIIC development. Questions were asked on BIIC schedule, die size, testability features and cost. Wayne reported that first pass BIIC will come out in Q1 FY'83 and second pass in Q3 FY'83. The die size is 310 x 310 mil. The BIIC has extensive RAMP features as well as data integrity provisions. Wayne said that the cost is estimated to be about \$72/chip but that the cost is a function of volume.

Issue: Cost of BIIC and Interfacing to BIIC

Barry was concerned that the cost of BIIC is unknown for FY'85 because we don't know who will use it and, therefore, what the volumes would be. He also added that we would need other chip(s) to interface to BIIC which would add to the cost. Today Unibus interface costs \$25, the BI would be much higher than that. And high cost would demotivate its use. Ted argued that if we make BI public and let the DEC customers use it, the volume will go up and cost will go down.

Next the complexity of interfacing to BIIC was discussed. The BI required memory address translation to be done at the controller end. Bob thought that memory sitting directly on BI makes memory translation more difficult, if the bus architecture was CI like then the disk would not have to do physical address translation right on the I/O bus and, therefore, would be easier to attach to. Barry said that if we use BI as is then we should design a memory management chip to be used at the controller end. Bill thought that was an implementation issue not a BI issue.

Issue: What Do We Want the BI For?

Since most of the users of BI went away except Scorpio, the group wanted to address the application space for BI before we looked at the BI related problem. Dave said we need BI for terminal size computers and the traditional mid-range systems that can do multiprocessing. The competition is the Japanese and IBM building personal computers and small systems in a networking environment. Dave thought BI was not good for the high end. DEC's strength is its familiness of products and ease of migration across time and size. Migration is achieved through the use of common application software. An argument was made, however, that high volume products such as table top computers will have their own bounded design and to be cost effective will use nothing standard. Therefore, they will have their own optimized bus. Migration is important for mid-range systems. For Micro's, migration is also very important because their customers design special controllers. Then an argument was made that if the table top market was not going to use BI, then for other systems we would have to worry about more than one BI because its performance will not be enough.

Issue: Bandwidth Not Enough--Especially With One "Stall Cycle" Added

Bob said that if Nautilus only had one BI, it would not work, BI has at least one stall cycle, sometimes you may get two. Steve asked how

long it would take to do 1 megabyte transfer. There was no answer to that because we haven't done system performance simulations with BI. However, for typical timesharing applications there is enough bandwidth. If we want to compete with SEL, there probably won't be enough. Bob didn't know what percent of the users wanted more than "X" bandwidth. Steve said 99 percent of the applications won't need more than 8-10 megabytes per channel. But there are applications requiring multi-channels. Ted worried about the bandwidth requirements of image processing. Barry said he had data on what the disk requirements will be in the future. He estimated:

6 megabytes in '88  
8 megabytes in '91  
11 megabytes in '95

and that all these were very conservative numbers. Also, he thought the CPU performance will go up and, therefore, demand for I/O bandwidth will go up. Therefore, we must have a strategy for multiple BI. Barry argued that every processor will have its own private memory interconnect and SI (storage interconnect) will sit on it. Steve said that a goal of BI should be a common attachment to storage. Dave added that BI also should serve as a way for customers to interconnect to a common bus. But if BI was only for storage, then we reinvented the Massbus.

We couldn't answer whether BI had enough bandwidth because we didn't know how it was used in a system environment. Bob said, if we go with BI and assume it is a common storage interconnect then we are stuck with it for high end as well. In a multi-BI system we need a cable between the PMI and each BI. Dual processor Nautilus or Venus II will not fit into one cabinet. Therefore, 5 foot length may not be enough.

Issue: Length of BI--5 Foot or 20 Foot or 1 Foot

Steve said if the bus was longer than 5 foot, we may require two stall cycles which might reduce the bandwidth to 6 megabytes. For large systems the requirements of longer BI and higher bandwidth are contradictory. Bob asked what would be wrong with dividing the bandwidth into byte size cost effective chunks that the customer can get to? Steve said that most people don't want DR32. All customers now want Unibus and Q-bus like interfaces. BI logically performs as if memory is on the bus. We anticipate that the use of BI on the high end will be like the Unibus.

Then it has to come out of the cabinet. How fast should the extension be? There is a hypothesis that some customers may want it very long and slow but the throughput is there. Ted thought the customer may be left with the responsibility of designing the 20 foot extension. Then the question is whether to extend from BI or PMI. If we extend from PMI, BI could be "backplane length" long. Bill summarized that we can see a need for a very short BI and one that is 20 foot long, but 5 foot isn't an interesting number because it does not allow mid and high end to get around all of their modules and it is longer than you need if you are building a "shoe box". Perhaps the question is not

how long it is but how many drops it can have. But then we have the same problem because it is related to capacitance. If the wire is 20 foot long, it would be better to run one or two wires instead of 60; which will look like a CI.

Issue: Can We Solve Everybody's Problem With One Bus?

Bob asked whether we can solve everyone's problem at the same time. Dave said we could define it as follows: 1 msec random access, 5-10 megabytes bandwidth, 5-10 drops and meets the "get out of the cabinet" requirement. The backplane is not very long, but somewhere at the end of the system an isolated extension comes out. Bill hoped that we don't want something we cannot physically build, i.e., we want to make it longer and it needs to be electrically isolated, then maybe it needs to run serial, and then it needs high bandwidth..... We are generating a wish list and going down the rats nest.

Issue: Competition Test

Who does it faster? Nautilus must compete with SEL. SEL's bus is equivalent to Nautilus' PMI, but they decided to make it public. What made Unibus desirable is that we made it public and it stayed the same from one system to another. Any customer who interfaces to PMI takes the risk of generating a point product because the next PMI may be different. It is also important for DEC to have a common bus to attach disks and comm devices.

Ted explained what the customer wants at the low end. The engines like V-11, Microvax, Nanovax, two or three of them would sit on BI with shared I/O and shared memory. Some customers will want RAM on the BI. An argument was made that BI allows all V-11 memory to be fully addressable so there is no need for RAM to be on BI. However, Bill thought in the long run memory will hang off in the local subsystems and it wasn't clear whether the BI needed to be a memory bus at all. It was clear that having BI be a memory and I/O bus was better than just having it be an I/O bus.

Issue: Should BI be a Memory and I/O Bus (i.e., Allow DMA) or Only an I/O Bus?

Dave said that we want to be able to do what Intel does which is to have shared memory, multiprocessing and I/O buses. Bill responded that each generation of CPU that we put out has multiprocessing capability on its PMI so you can always plug some number of CPU's on the PMI. But that is all you can do with it. If we let the customer attach to the PMI, we have no commitment to keep it the same from one generation to another. Dave said maybe what we need is a public memory bus and a public I/O bus. Bill's reaction was that we don't need more buses. If we try, we will end up with a compromise which is the BI. One can compromise one of several directions of which the current spec is a very good example.

A question was asked whether the customer wanted multiprocessing or a memory bus. Ted said "yes" to both. Cutler, representing Seaboard

software, thinks multiprocessing is essential and wants to keep BI as is. Stan believed that multiprocessing can be done over the PMI with higher performance. The current spec he generated for a new PMI can handle four Scorpio CPUs. Ted brought back the migration issue.

Steve asked what the cost difference would be between a memory plus I/O bus and I/O bus only. Barry said that if it was I/O only, we probably gravitate to a packet design. If we say we need shared memory on the I/O bus, then we would gravitate towards addressing data per cycle type design and leave out mid and high end. Dave thought we had requirements for both. Two public buses had less compromise. If we have two, then we could play with cost. Bill said two buses were unacceptable. He didn't believe we could afford both. We barely can afford to do one. We have identified two functions, that is a memory bus and an I/O bus. We are trying to produce a compromise. If you aim toward being a memory bus, you compromise I/O performance. If you do an I/O bus, you compromise the memory performance. Bill didn't think we can do two buses. The main thing that will get plugged into the bus is the I/O adapters and they can be plugged regardless of the model we have. Stan said the BI didn't meet the 10 megabyte requirement per octal word. Bob worried that if we stick the BI on a mid or high end system, we'd end up with a 6 mbyte bus which the customer can only put 3 mbytes on before he runs into problems.

Issue: Is it Better to Take a Schedule Hit and Do it Right?

Omur questioned the timeframe needed for a possible new BI. Barry said that BI will take 10 years from start to production. It should be well thought out and if it is almost ideal then we should go with it. But he thought the purpose of the meeting was to determine how far BI was from the ideal. If the deltas were large, let's take two more years and do it right. Barry said with BSA there was always a funding issue. Dave said that the current state of affairs would be not to do BSA because the only processor that will use BI is Scorpio, and it will be well served by an integrated I/O. Barry's reason to talk about funding was that some number of years from now there will be a bunch of peripherals and BI, but if there is no BSA nobody would use the BI. Bob said that it was clear that if the corporation does the BI, they would want to put something on it and the disks would be the obvious place to start. Omur asked if Nautilus will reconsider to put BI back in. Bob said that everytime the corporation reconsiders they get to reconsider their decision. But he was personally concerned about having to make a quick time to market project dependent on something that has already taken three years and hasn't shown any sign of real progress and doesn't seem to be perfectly suitable for his purposes. Dave argued that the whole reason the BI came back to life was because Unibus only strategies were not unacceptable.

The morning session was closed with a list of conclusions about BI that went as follows:

- BI is reasonably well suited for low end.
- BI does not have the desirable attributes for mid-range and high end.
- It may be unrealistic to expect the BI to meet the diverse needs of all products.
- BIIC as defined meets the requirements of the spec and is doable.
- BI needs to be a public bus.
- BI's performance in a system especially in a multiprocessing environment has to be understood. There may be enough speed but not enough bandwidth for multiprocessing of Scorpio size machines.
- BI is compromised towards a memory bus. Let's understand what we need to do to optimize an I/O bus.
- Scorpio system needs stability around the BI.

The afternoon session was dedicated to understanding what the I/O only bus would look like so that we could determine the worth of waiting for it without the presence of BI.

Barry presented the storage perspective. Looking over the next 20 years of the usage of this new bus, Barry said they'd like to see 4-5 Mbytes bandwidth per each adapter. For storage devices, they can see a need for two adapters summing up to a 10 megabyte bandwidth requirement. For ease of interfacing, a packet-oriented bus would be the best. This is the kind of a bus that gets out of the cabinet and is isolated, etc. So there is a bandwidth issue with BI.

The second issue is addressing. With the current BI, the BSA needs a certain amount of RAM to cache some of the pointers and timing is critical in bringing things in and out of the bus. Design of microcode was the major complexity in terms of mapping scheme.

Responding to a question, Barry said they don't need any interrupts on the bus and a two level prioritization is sufficient. The number of nodes on the bus could be between 10-16. They don't like the high speed buffering requirement of CI either. One might design a 20 Mbyte bus but the purpose would be to run 4 or 5 MB devices on it. Each device would get the bus once every four slots. In this scheme you would not have to use a superfast memory for buffering.

Bill said the question is whether one uses the main memory or packet buffers in order to store the information coming in under the wire. Barry said that large packets made things easier for them. But there were comments about technological problems to implement large packets inexpensively.

Bill commented that the problem is whether the bus is run at a fixed beat which places a requirement to take the data at the right rate or you have stalls which complicates the protocol. Barry argued that one would bargain for a new stall architecture. However, in any architecture one needs to guarantee data delivery to memory cycle after cycle and that is probably hard to do without packet buffers. Barry's position is that if we use commodity RAM chips, we would not have a real cost problem.

Barry said that one idea that came up was Time Division Multiplex (TDM) type design, where buffering and mapping is done in each node instead of for each node. One of the problems with BI right now is that if you stall you basically cause congestion on the entire bus for everybody who wants to use it.

Barry explained why the BI-AZTEC interface looked like it was going to be very expensive. The current AZTEC design uses the low end storage interconnect called LESI. Because of the similarities, LESI is defined to recognize the Unibus and Q-bus ports and the system software sees the same port. The microcode in the AZTEC controller implements the interface to LESI. For the optimal design, BI-AZTEC interface implements a VAX port architecture which necessitated changing the microcode in the AZTEC controller. There is a limited RAM and ROM there so you can't have both microcodes. What you have is an AZTEC prime that interfaces to BI. Ted asked if it is possible to go from BI to a BUA to Unibus then go to Unibus/AZTEC interface. The answer was to interface BI-AZTEC to LESI. The only disadvantage with that is that VMS would see a Unibus port instead of an elegant VAX port.

The discussion steered towards exploring what adapter architecture was to be used independent of what the new bus looked like. Two possible architectures were: VAX Port Adapter and an adapter with map registers. The advantage of VAX ports is better performance but at the low performance peripherals one wouldn't see a difference. Dick explained that one is limited by the number of map registers than anything else. The UBA and MBA on current VAXs have 496 map registers. Sometimes you run out of map registers. Barry mentioned that it is also a matter of mix of products. If the configuration has things like TU81, floppy disks and MAYA, then we probably are doing the wrong thing with the BI-AZTEC interface. We could easily go use a LESI interface to BI by just redesigning that piece.

Next, Bill presented his vision about BI and BI prime. He said that the company really cannot afford to have two buses in this space. I/O and memory needs may be different, but the purpose of this bus is to attach peripherals to VAX systems. Bill didn't think we could have two buses that attach peripherals to VAXs. What one needs to do is to pick a compromise, and like all compromises it won't leave everyone happy. But the compromise can be made to augment what we will have: CI and NI and what we have: Unibus, Q-bus, etc.

Bill continued, "If one wants a compromise towards a memory bus, the current BI definition is just fine. Quite a good job has been done to



get a very nice bus within a whole bunch of constraints in terms of the number of things and how many chips you can have. We won't gain anything by changing that definition. BI is 99% where it ought to be. The other emphasis which has been talked about from time to time is making it look more like an I/O bus. This was basically not considered strongly at the interconnect task force because of the assumption that the BI was going to be used for a lot of things it turned out not to be used for. Frank gave that list and you saw a host of projects who might have used it. Particularly Mercury, HSC, J-1, Venus and whatever the Fl1. So there were about five projects around, all of which were predicated upon memory box-like structures, and they were all going to use it so it seemed very clear at the time this was the proper orientation to go with and not an I/O orientation. Given that there's a substantial decommitment from using it in that fashion, I think it's appropriate as an exercise, hopefully, contained to just today and not further than today, to look at what it needs to do as an I/O bus."

Bill continued his presentation. "Barry gave a basic overview. I wouldn't want a higher bandwidth. If you look at the current BI design and assume that you had very long transfers, the bandwidth would be closer to 20 megabytes than 13. And further, if you assume that each of the nodes that are interfacing to this thing had packet buffers and which I think is the only reasonable assumption to do, then in fact you wouldn't have stall cycles so there would be 20 megabytes available on the bus. Right now we have with the current BI a definition of 13 megabytes maximum; but in actual operations it will be much less than 10. With the definition of the packetbus and packet buffers, there would be presumably available 20 megabytes so there's potentially a significant difference. Maybe as much as 3 to 1 advantage in operations. What else would we do differently?"

"The other thing is basically a more simple protocol. The memory mapping is presumably centralized. There will be some kind of port between the processor and this bus and all the memory mapping would take place there."

Barry asked if Bill was assuming there would be more than one processor allowed on that bus? Bill explained that what goes on this bus is basically computer memory, processors, etc. I think to the first approximation the definition of an I/O bus would be a CI-like structure implemented in some sort of parallel form and restricted to lie within the cabinet or cabinets that are bolted together. Barry said that this would change the chip fairly radically. Bill responded that it changed interrupts and interlocks. Dave said the BI by itself was not enough, you also have to invent the MSCP and the CI port architecture, etc. Bill answered that was already defined. Barry added that a real VAX support had to be implemented that does the DMA functions as well as the non-DMA functions and plays with the maps and all the rest of periphery.

Bill summarized that we would end up with a bus that is simpler in its design, that is, higher bandwidth with the same technology you put into it and for peripheral devices it solves the memory mapping

problem. What this does is if we went with this as opposed to this BI definition, we are sort of selling a slightly different class of systems--there's a slightly different emphasis. This is oriented towards multi-computer systems that are typically designed in a set of subsystems which are sort of self contained and they are linked together by this BI prime. The peripherals are message interprocess communication-oriented as opposed to memory reference oriented. Bill thought that over the long term that is clearly the way to go.

If we are selling systems in 1990, most people are not going to want to talk about buses that are memory reference oriented. Barry said that the other side of the argument is that Cutler wants a memory bus. Steve argued that if you want a Unibus then you put it on the PMI. You would either have to make the PMI available or you make something to hook through PMI.

Stan asked whether the packet lengths for the BI prime would be the same length as the BI. Bill said probably so. The CI allows up to 4000 byte packet sizes. Given that you are using messages as a synchronizing mechanism you've got to make sure it is at least big enough to hold one of the synchronizing messages.

Going back to the common PMI issue, Bob said it was hard to believe that the Scorpio and the high-end VAX really have any use for the same PMI. Steve responded that we don't have to pick the extremes. The question was, should a PMI be consistent across several members of the family? Bob asked how many members of the family can be out there at once? Dave said from a market point of view there are somewhere between 5 and 10. Bob added that there are some old ones and some new ones. The issue was that the group thought we would need a common and public PMI if we went with BI prime. Bill did not believe that the things we want to do, have to be implemented on the PMI. He said that if we look at the set of peripherals that we are proposing, we think we need SI-like adapters, comm adapters, NI adapters, things like that, that all of them will perfectly will go on a packet bus. We don't need this memory style to implement our own peripherals. Bob disagreed with the last statement.

Mike asked what George Champine will need for his Workstation when he replaces the Unibus inside the box and goes directly to some VAX. Bill answered that the graphics adapter has its own memory. George needs a high speed port to memory but it is not clear that he needs random to host memory.

Bill did not believe that for any of the peripherals we are making ourselves that they would be seriously inconvenienced by having a packet kind of bus; in fact, they will probably be inconvenienced by having it. Dick commented that any two Unibus options follow anything comparable or that anything else that offers you register bits is going to behave uniformly. Bill said until somebody puts up the name of something we want to adapt to, that this seriously inconveniences the bus structure. He thought Dick's comment was, "Does this provide a discipline?". It provides a discipline that really allows these things to be engineered independently of one

another. We have done things in the past where everybody is in everybody else's hair and we have a lot of real time constraints and so forth. The packet bus is a longer term solution and because of that there is some higher risk associated with it.

Bob asked about the comm devices on the packet bus. And Dave asked if Bill assumed that there is only one kind of processor out there with virtual memory. Bill explored the kind of components that we may produce in quantity. Bill said if we look at the bulk of DEC's business and asked what peripherals we want to make, it will not be more easily done on this bus than the other one.

If we look at our terminals strategy, we are not going to build character type terminals anymore. He could not see many peripherals adapters that would be hooked onto this packetbus for which it would not be highly suitable.

Steve brought up the original SUVAX concept where it was very important that the memory was in the BI address space and he said that they already have a problem because they can't do that with Unibus. Bill said they have an architecture that is based upon the Unibus. They do block transfers back and forth between the frame buffer and the Unibus. So it looks more like a block transfer device than a bit map that you write by the processor. Steve asked if the reason to change was Unibus limitations. Bill said there were several things. There are some architectures that have CPUs running directly into bit maps, however, that does not work too well if you want to provide a higher level abstraction of software.

Steve asked if the discipline enforced by the BI ends up costing us something. The answer was yes, if you look at the SBI the imposed discipline required to develop the pieces independently. Dave said it was absolutely clear that we paid dearly above the transfer cost and development cost for that but we also got the product out and got the pieces working independently. What we get with a more integrated systems design in the whole thing has to work together and it takes longer to make it work. Mike said, in a low cost product scheme we can put on the PMI the AZTECs, RNDs, and whatever else. Bill responded, they can go either place. The AZTECs will work fine on BI prime bus. Barry commented that until we get enough peripherals out there to cover all the things that Unibus did for customers, they're going to be Unibuses. The whole reason the Q-bus did not succeed very well was because there weren't just enough peripherals out there.

Dave was concerned that the thing that makes BI prime strategy risky among other things are that anybody can target the point from an integrated point, can bounce this off because we are basically trading storage for communications in a packet-bus strategy. The result is that there is more storage cost which means somebody could build a point product for that application. Bob asked if we don't get anything back when we buy those memory chips. He said there was a claim to get a factor of two to three in performance which may or may not be useful on Dave's particular system. Dave didn't agree that we get performance back. He said that we would end up executing the

packet protocol twice; once in the processor and once in the controller. If we could reduce that execution, then the storage would be twice. Bob said Dave was talking about the latency to get from a disk into the main processor memory and get the process started. He asked if that is of any particular interest to us? Dave and Barry said yes. None of the controllers we're building now are low latency. We would trade throughput for latency.

Dave was not sure that we gained or lost anything with BI prime. We saved some microcode in the controller; we added some microcode to the host. He thought it was about a wash. Bill said, Dave's contention was that BI prime was a more expensive way of doing things. Dave was sure it is more expensive because there is more memory overall.

Stan argued that it is got to be cheaper to build a dumb packet buffer than to build the translation buffers, etc. Bob said it was not obvious why the packet buffers can't be the same buffer to buffer the disk blocks. Barry explained because we do things in parallel. There are not enough guaranteed cycles in there to get things at the speed that they want to run.

Stan asked how much buffering Barry needed to keep the addressing function. Barry said that the address translation was not more than 128 bit words. Dick commented this kind of discussion shows the fall of not designing the whole system from the host down to the interconnect, because what this is talking about is a tradeoff of where do you factor the system. The two schemes have potential merit, but we're not able to delineate the concrete tradeoffs here. We have a design that's gone for two years and here we are.

Dileep asked why not make the BI the standard PMI at least for the lower mid-range. We could do simple adapters on BI like BI comm and BI-to-LESI interface and maybe at a later time when we decide that we want the long term BI prime we design BI prime adapters and put everything on that. Bob said BI has no where near enough bandwidth for mid-range PMI. Bill iterated that the only way we are going to survive is to have only one thing.

Barry offered that the only possible short term solution is to go with the existing BI and then start shoveling some money towards shrinking the CI cost to something we can live with for an I/O link later on. We won't get the CI in the near term, but at least we will have the BI out there.

Dave supported him saying that we need a random memory access bus of some kind and it is desirable to make that same across the multiple implementation for our customers reasons. Steve added that there is a large set of customers today that want it.

Stan asked if they understand that they have to build a VAX port architecture interface. Steve said they don't have to, they'll send that stuff to the physical memory. Barry added that they could put a small mapping register set like you have in the UDA.

Omur asked if anyone was suggesting we should do BI prime instead of BI. Bob believed it definitely was an improvement. Dave said it was a good thrust for the mid-range, but he was not convinced we have solved the problem of connecting to a PMI. Dave thought the right thing would be having both. He said, we are flip flopping between which one we toss and which one we redefine.

Dick said we are already doing the CI and NI and this is the third thing. Exactly what it ought to do is what we're trying to debate and that debate changes in some degree dependent on what we might do to the CI, for example, if we made the CI come down in price without way architecturing it then in fact that relieves this I/O bus versions because that is already a designed solution. Then suddenly the BI looks more right for what it is supposed to do.

Bill added that if you take the current BI, we better work on cost reducing the CI. He thought the other alternative is to go with this BI prime, and that implies reliance on Q-bus and the Unibus until we can convince people they don't need random access memory. Current BI is shorter term and lower risk and BI prime is longer term and higher risk. The risk is having to convince customers that they can exist on a packet bus as opposed to having a memory access bus.

Omur wanted to explore what would happen if we do the BI prime. Dave said we still have a commitment to do the peripherals. Omur worried that there won't be any processors to commit to using BI prime. If the problem is time to market and we have to get there early FY'85, we can't depend on something that is just starting.

Omur asked if Nautilus would commit to use BI prime. Bob said, in addition to the Unibus, he might consider it. Bill said Scorpio would be in the same situation. We would not really build a BI machine, we would build a Unibus machine. Steve added that Ted would end up with a backplane that has a BI module, a processor module, and a memory array plugged into it, and his customers can plug in modules that interface with the system (BI prime).

Next, Bob covered the needs of Nautilus like machines. He said what Bill proposed has valid, useful, sensible points, and that we have been ignoring the packaging issue quite a bit. From his perspective, it was clear that one, two, or three BIs probably are not enough for a mid-range machine and no where near enough for a high-end machine. He feared that independent of whatever decision was made we would see a continuing failure to do the system analysis.

Dave asked if Bob thought that BI prime will be a satisfactory solution for mid and high-end machines. Bob responded that it appeared to do the things that at least most of our peripherals want done in a reasonable sort of manner without depending on a bunch of facilities that mid and high-end are going to be unable to provide. He said, the plain BI is going to start dropping more and more stuff we put between it and the real memory and the bigger the system the more stuff that will be in between as you use a number of BIs. Barry agreed with Bob that if we go with the current BI, Bob still has a

problem with packaging that is serious. He recommended that, someone should go off and solve the packaging problem and make sure that whatever BI we produce, it will be something close to the current one and won't impact the schedule too badly.

Barry's second issue was the system analysis. He said we don't really know if the current BI works. It would be important to make sure it works and then add the task of cost reducing CI. Bob agreed that the CI would appear to be a fairly obvious bus to try and get our customers to use. It solves our packaging, installation problem. Stan said he would vote for BI prime because he didn't think CI is going to ever get down to a Scorpio price range. The BI prime would be inherently cheaper because it is 5 volt only and it doesn't have 7 nanoseconds per bit, etc.

Bill said, when we say the current BI plus a cost reduced CI we are still talking about a CI that is fundamentally a between cabinet bus; coaxial cable. Instead of being \$4K or \$2K maybe it is \$1K or \$900 but it is not \$50 and it is not a bus designed in any sense of the word to be optimal within the cabinet. Whereas the BI prime is a bus designed to work primarily within a cabinet or a set of cabinets that are bolted to one another so it's entirely different from a cost reduced CI.

Bill believed the BI prime is likely to be a lot cheaper than going out through transformers and coaxial cables, etc. He said, as soon as we want to go out of a cabinet, it is clear we don't want to use the parallel bus. We talked about five components to build on the BI: BNA, BCA, BSA, BI Combo, and BI AZTEC—not one of those would he like to have come out of the box. Everyone of those are presumably packaged with a processor and has a cable coming out of the box.

Stan said, Scorpio has a PMI and the BI is strictly an I/O bus and the question is whether he has a good I/O bus or a bad one. Bob added that BI is not going to multiprocess the thing that follows Scorpio and might not even do Scorpio, and is not going to solve any requirements for bandwidth in excess of about six megabytes depending widely on which system your going to plug it onto with the high performance systems showing less performance. It isn't also going to be, at least for the moment, convenient to package; it will put a lot of packaging constraints on the medium and high-end systems because you can't get enough of them stacked around the cabinet. It is going to have the unpredictable configuration dependent property of the Unibus.

Bill argued that BI is not nearly that bad; the BI is not a fatal error—that is clear. He thought for a shorter timeframe, definitely the current BI is lower risk and a better choice. From a longer term perspective he believed the BI prime is a better choice.

The meeting concluded by voting on the two alternatives. Bob and Stan recommended to go with BI prime. The rest accepted the choice of staying with BI as is. Bill recommended that we take the alternatives to management and have them make a business decision, whether they wanted to solve the short term or longer term problem. Omur took an action item to do that.

JUN 28 1982

TS1.8

+ + + + +  
| d | i | g | i | t | a | l |  
+ + + + +

INTEROFFICE MEMO

To: List

Date: 22 June 1982  
From: Tom Sherman  
Dept: Low-End VAX Sys Dev  
Ext.: 247-2418  
Loc/Mail Stop: TW/B02

Subject: PHASE REVIEW ANNOUNCEMENT - AGENDA

PHASE REVIEW ANNOUNCEMENT

PHASE: 0

PLACE: Site Management Conference Room, TW

DATE: 29 June 1982

TIME: 8:30 a.m. - 12:00 noon

PRODUCT MANAGER: Tom Sherman

On Tuesday, 29 June, a Phase Review meeting will be held to reach agreement that Phase 0 has been satisfactorily completed for the Scorpio Program.

The documents required for Phase 0 are:

Business Plan  
Product Requirements  
Market Requirements  
Alternatives/Feasibility Study  
Manufacturing Impact Statement  
Customer Services Impact Statement

The business plan is attached. The other documents will be available at the meeting or on request.

TS:clc

Attachment

DIGITAL EQUIPMENT CORPORATION

SCORPIO PROGRAM BUSINESS PLAN

At Exit From Phase 0


Prepared By:

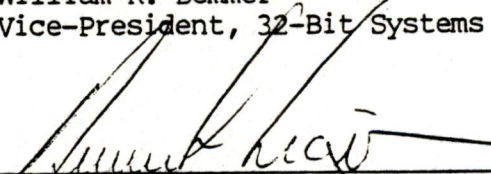
Thomas H. Sherman  
Scorpio Product Manager

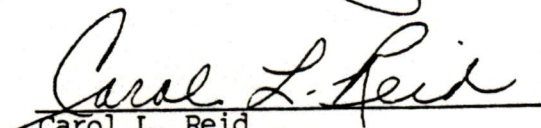
Revision Number: 0.7

22 June 1982

Approved For Publication

  
William R. Demmer  
Vice-President, 32-Bit Systems

  
Demetrios Lignos  
Scorpio Program Manager

  
Carol L. Reid  
32-Bit Systems Financial Manager



# SCORPIO PROGRAM BUSINESS PLAN

## TABLE OF CONTENTS

SCORPIO PROGRAM TEAM .....	iv
PHASE REVIEW MANAGEMENT APPROVAL .....	v
INTRODUCTION .....	vi

### CHAPTER 1 EXECUTIVE SUMMARY

1.1	PRODUCT DEFINITION .....	1-1
1.2	PRODUCT GOALS .....	1-1
1.2.1	Marketing Goals .....	1-1
1.2.2	Engineering Goals .....	1-2
1.2.3	Manufacturing Goals .....	1-2
1.2.4	Service Goals .....	1-3
1.2.5	Financial Goals .....	1-3
1.3	RELATIONSHIP TO CORPORATE PRODUCT STRATEGY .....	1-4
1.4	NUMERICAL HIGHLIGHTS .....	1-5
1.5	MARKETING STRATEGY .....	1-6
1.5.1	Market Size .....	1-6
1.5.2	Applications .....	1-6
1.5.3	Geographic Considerations .....	1-7
1.6	PHASE PLANNER .....	1-8
1.7	RISK ASSESSMENT .....	1-8
1.7.1	Technology .....	1-8
1.7.2	Manufacturing .....	1-8
1.7.3	Service .....	1-9
1.7.4	Marketing .....	1-9
1.8	PROGRAM ISSUES .....	1-9

### CHAPTER 2 PRODUCT DESCRIPTION

2.1	PRODUCT LIFE CYCLE OVERVIEW .....	2-1
2.1.1	Product Offerings and Potential Migration Strategy .....	2-1
2.1.2	Affected and Successor Products .....	2-1
2.2	KEY FEATURES--INITIAL PRODUCT OFFERINGS .....	2-2
2.2.1	Initial Scorpio Architecture .....	2-2
2.2.2	System Component Descriptions .....	2-3
2.2.2.1	Processor Module .....	2-3
2.2.2.2	Memory Array .....	2-3
2.2.2.3	Backplane Interconnect (BI) Bus .....	2-3
2.2.2.4	BI-to-Unibus Adaptor (BUA) .....	2-4
2.2.2.5	Power and Packaging .....	2-5
2.2.3	Primary Product Deliverables .....	2-5
2.2.4	Marketable Features -- Relating to Customer Utility .....	2-9
2.2.5	Features to Reduce Life Cycle Cost to Digital .....	2-10
2.3	FUTURE SYSTEMS/FOLLOW-ON PRODUCT OFFERINGS .....	2-12
2.3.1	Ultimate Scorpio Architecture .....	2-12
2.3.2	Future Scorpio Peripherals .....	2-12
2.3.3	Future Scorpio Systems .....	2-13

## SCORPIO PROGRAM BUSINESS PLAN

### TABLE OF CONTENTS (Cont'd)

2.3.4	Future V-11 Based Products .....	2-14
2.4	SYSTEM/PRODUCT POSITIONING .....	2-14
2.4.1	Digital Products .....	2-14
2.4.1.1	Digital Systems .....	2-14
2.4.1.2	Digital Board Sets .....	2-17
2.4.2	Competitive Products .....	2-17
2.4.2.1	Market Size .....	2-17
2.4.2.2	Competing Systems .....	2-18
2.4.2.3	Competing Board Sets .....	2-19
2.4.3	Price Positioning Charts .....	2-19
2.4.3.1	System Price Bands Against Digital Products .....	2-19
2.4.3.2	System Price Bands Against Competitive Products .....	2-20
2.5	TECHNOLOGY .....	2-20

### CHAPTER 3 SHIPMENT FORECAST

3.1	WORLD-WIDE FORECAST .....	3-1
3.1.1	Scorpio Boards and Systems Forecast .....	3-1
3.1.2	Scorpio VAXstations Forecast .....	3-2
3.2	DETAILED WORLD-WIDE FORECAST BY PRODUCT GROUP .....	3-2
3.2.1	Detailed Scorpio Boards Forecast .....	3-2
3.2.2	Detailed Scorpio Systems Forecast .....	3-3
3.2.3	Detailed Scorpio VAXstations Forecast .....	3-4
3.3	IMPACTED PRODUCTS: PHASEOVER PRODUCT SUMMARY .....	3-4
3.3.1	World-Wide System Phaseover Forecast Summary .....	3-4
3.3.2	Impact of Not Bringing Scorpio to Market .....	3-5

### CHAPTER 4 ASSUMPTIONS

4.1	ENVIRONMENT .....	4-1
4.1.1	Competition .....	4-1
4.1.2	Geography .....	4-1
4.1.3	Regulatory Compliance .....	4-1
4.1.4	Economic Factors .....	4-1
4.1.5	Operating Environment .....	4-2
4.2	TECHNOLOGY .....	4-2
4.2.1	New Processes and Components .....	4-2
4.2.2	Design Tools .....	4-3
4.2.3	Key Suppliers of Components .....	4-3
4.3	INTERNAL FUNCTIONS .....	4-3
4.3.1	Engineering Dependencies .....	4-3
4.3.2	High-Volume Manufacturing Dependencies .....	4-4
4.3.3	Final Assembly and Test Dependencies .....	4-4
4.3.4	Customer Services Dependencies .....	4-5
4.3.5	Marketing .....	4-5
4.3.6	Sales .....	4-6
4.3.7	Asset Management .....	4-6

SCORPIO PROGRAM BUSINESS PLAN

TABLE OF CONTENTS (Cont'd)

4.4 CUSTOMERS ..... 4-6  
4.4.1 Impact on Key Accounts ..... 4-6  
4.4.2 New Opportunities ..... 4-6  
4.5 SCHEDULE ..... 4-6  
4.5.1 Project Schedule ..... 4-6  
4.5.2 Phase Planners ..... 4-6

CHAPTER 5 FINANCIAL/SENSITIVITY ANALYSIS

5.1 SUMMARY AND CONCLUSIONS ..... 5-1  
5.2 FINANCIAL BASE CASE ANALYSIS--NUMERICAL SUMMARY ..... 5-2  
5.3 FINANCIAL MODEL ASSUMPTIONS ..... 5-2  
5.4 CUMULATIVE CASH FLOW GRAPH ..... 5-5  
5.5 BURP FINANCIAL STATEMENT ..... 5-5  
5.6 SENSITIVITY ANALYSIS ..... 5-7

CHAPTER 6 REFERENCE DOCUMENTS ..... 6-1

CHAPTER 7 BUSINESS PLAN HISTORY/CHANGES ..... 7-1

SCORPIO PROGRAM TEAM

Scorpio Program Manager: Demetrios Lignos

LOW-END VAX SYSTEMS DEVELOPMENT:

Systems Hardware Manager:	Mike Titelbaum
Systems Architecture:	Bob Willard
BI and BIIC:	Frank Bomba
BI Options:	Brian Hannon
Diagnostics:	Ernie Preisig
Systems Integration:	Ann Katan
Power and Packaging Manager:	Bill Schmidt
Micro-Packaging:	Jim Mars
Macro-Packaging:	Charlie Barker
Power Supply	Bob Pedersen
Manufacturing Introduction:	Nelson Velez
Operations Manager:	Omur Tasar
CAD Applications:	Peter Barck
Product Manager:	Tom Sherman
Financial Manager:	Kevin Reilly
Financial Analysis:	Doug Kellogg

MANUFACTURING:

Manufacturing Program Manager:	Ken Meissner
Volume Manufacturing:	Jim McWilliams
Systems Manufacturing:	Deiter Wolferserder

CUSTOMER SERVICES:

Customer Services Program Manager:	Doug Hanzlik
Field Service:	Ted Gent
Software Services:	Martha Sifnas
Educational Services:	Bob Campbell

OTHER ENGINEERING:

VAX Architecture:	Dileep Bhandarkar
Semiconductor Engineering Group:	Duane Dickhut
Storage Systems (Memory):	Ron Given
Storage Systems (AZTEC):	Carl Blatchley
Software:	Clark D'Elia

SCORPIO PHASE REVIEW MANAGEMENT APPROVAL

Michael Titelbaum

Michael Titelbaum - Scorpio Systems Hardware Manager

22 JUN 82

Date

Thomas H. Sherman

Thomas H. Sherman - Scorpio Product Manager

22-June-82

Date

Douglas R. Kellogg

Douglas R. Kellogg - Financial Consultant

22 June 82

Date

## INTRODUCTION

The Scorpio Program is a Central Engineering effort directed by the Low-End VAX Systems Development group of the 32-Bit Systems organization. The objectives of the program are twofold:

- Drive VAX systems into new low-end markets.
- Introduce the BI bus, the Unibus I/O replacement for future VAXs.

The program involves development efforts in several engineering organizations: (1) the V-11 and BI chip sets, and the CPU module by SEG, (2) memory array modules and AZTEC disks by Storage Systems, and (3) VAX/VMS, Midnight and Seaboard by BSSG. Low-End VAX Systems Development engineering responsibilities include specifications for the BI bus and BI interface chip, the BI physical interconnect, the BI-to-Unibus Adaptor, all power and packaging, and the integration of all of the above components into specific deliverables which include Scorpio CPU Board Sets, BI interface chip sets and Scorpio Systems (including an OEM box). Management of this program is accomplished through the Scorpio Systems Program Office (SSPO), a committee whose membership includes all of the development organizations, Customer Services, and Manufacturing.

This document, which follows DEC STD 130 (Business Plan, Rev B, 3-Dec-81), summarizes the product goals, defines and describes the deliverables and schedules, and provides preliminary forecasts and analyses.

## CHAPTER 1

### EXECUTIVE SUMMARY

#### 1.1 PRODUCT DEFINITION

Scorpio is a low-end VAX, its CPU being implemented on a single module. However, Scorpio as a generic term currently encompasses three separate Digital products: Scorpio board sets, Scorpio systems (including an OEM box), and Scorpio based VAXstations. Although this document focuses specifically on Scorpio boards and Scorpio systems, certain information and forecast data is presented on VAXstations (1) to identify the total market breadth and potential volume magnitude of the Scorpio CPU, and (2) to overcome a current product definition ambiguity of Scorpio systems in a workstation application vs VAXstations in the same application. However, unless the term "VAXstation" is specifically used, all information in this document refers to Scorpio boards and systems only. For additional information about the Scorpio VAXstation, contact Nasir Khan, VAXstation Product Manager.

The Scorpio CPU utilizes the V-11 chip set and interfaces to the new Backplane Interconnect (BI) bus. The BI has a 13.3 MB/sec bandwidth, implements a new Euro compatible module form factor, and directly supports multiple processors. An integral part of the Scorpio program is the development of a custom, two chip interface to this bus. A Unibus adaptor is available at FRS and initial packaging includes a combination backplane for both BI and Unibus modules. Refer to Page 2-2 for a system architecture diagram.

The design center for system packaging is a desk-height pedestal, compatible with open-office environments. Initially this enclosure houses one or two AZTECs; ultimately as mass storage technology improves, lower cost storage subsystems will be so mounted. System packaging also includes a 10.5" OEM box with slides, and SI mass storage based systems in 40" cross product cabinets.

#### 1.2 PRODUCT GOALS (Non-prioritized)

##### 1.2.1 Marketing Goals

1. Drive VAX systems into new low-end markets--specifically via board sets, VAXstations and open office compatible systems in hardware, and the "Seaboard" run-time system and "Midnight" operating system in software.
2. Introduce and ensure acceptance of the BI bus, the Unibus I/O replacement for future VAX system. Pursue making the BI an industry standard 32-bit bus.
3. Provide system expansibility via VAX/VMS asymmetric BI multiprocessing and Seaboard symmetric BI multiprocessing.
4. Announce a Scorpio VAX/VMS system (license-only) with an MLP of less than \$30,000; announce the Scorpio CPU module with a MLP of less than \$3,500.

5. Ensure local language "country kits" are available at FRS.
6. Establish a corporate open-office enclosure family common to Scorpio and Orion systems.
7. Ensure Scorpio systems are customer installable, diagnosable, and repairable; ensure BI options are customer field installable.
8. Achieve the above goals in conjunction with responding to the Product Groups' mandate of "time-to-market" and "cost/performance" as critical product/market requirements.

### 1.2.2 Engineering Goals

The primary engineering goal of the Scorpio Systems Program is twofold:

1. Development of the BI and BIIC (and associated specifications, documentation and validation tools) to provide the I/O interconnect for the Scorpio CPU module and systems products and for future VAX systems. The development will productize the BI and BI Interface Chip as an interface chip set product for the Micros board market concurrently with the availability of the Scorpio CPU module.
2. Development of a BI-to-Unibus Adapter (BUA) module and development of an OEM Box and AZTEC based Unibus VAX system products that will surpass 11/750 performance and be cost optimized for a 1-8 user VMS system.

The engineering goal set also includes:

- Specification development and verification of symmetric and asymmetric multiprocessor configurations of Scorpio with VMS and Seaboard software systems.
- Development of Native BI communications and LESI storage adapter options and the specifications of supportable I/O adapter architectures for future developed BI adapter designs.
- Development of the required simulations, diagnostic, and hardware tools to affectively complete the development on schedule and support of these tools after product introduction.
- Establishment of a systems engineering group capable of developing the BI and Scorpio Systems and options on the current schedule, supporting the transitions to manufacturing and supporting other engineering users of the BI or Scorpio system modules.

### 1.2.3 Manufacturing Goals

1. Life Cycle Cost: The primary manufacturing goal is to minimize this cost for the corporation.



2. Stage 2 Inventory: Computer Systems Manufacturing's (CSM) goal, as delineated in its LRP, is to have a maximum of 8.5 weeks of inventory in the pipeline.
3. Quality: Product certification to be accomplished within 90 days after the completion of PMT. Product quality will be achieved via process control rather than extensive test and inspection.
4. Transfer Cost: The goal is to achieve a 90% learning curve net after inflation on the total transfer cost.
5. Deliveries: CSM's delivery goal is to be 100% quarterly for FY'85, improving to 100% weekly by mid-FY'86.
6. Site Mergability: Eliminate FA&T for all Scorpio systems by ensuring all system components are site mergable.
7. Customer Installability: Scorpio packaged systems will be customer installable and BI options will be customer field installable.
8. Time to Market: CSM shares this corporate responsibility for timely introduction by reducing the duration of Engineering/Manufacturing hand-off.

#### 1.2.4 Service Goals

1. The primary service goal is to reduce the overall cost of ownership of a Scorpio system.
2. Scorpio systems will be customer installable.
3. Field Service performed preventative maintenance will not be required on any Scorpio systems.
4. Implementation of customer diagnosis to reduce user related calls.
5. Assuming a single AZTEC Scorpio pedestal packaged system with CPU, 2MB memory, BUA, single AZTEC and LA200/LK201, the service goals are:
  - MTBPR (Mean Time Between Parts Replacement) = 2015 hours
  - MTBC (Mean Time Between Calls) = 1752 hours
  - MTTR (Mean Time to Repair) = 1.5 hours
  - System Availability = 99.8%
  - BMC (Basic Monthly Contract) = \$170.00

#### 1.2.5 Financial Goals

The following financial metric goals are the results of a BURP analysis that incorporates current Scorpio goals such as transfer cost and markup.

1. NPV = \$1.69M. The goal is to maximize NPV given the constraints of Scorpio both introducing a 32-bit board product and a new VAX I/O architecture, and increasing market share in open office and workstation applications.
2. IRR = 41%. The goal is to exceed the engineering new product hurdle rate of 40%.
3. PBT (BURP) = 33%.

### 1.3 RELATIONSHIP TO CORPORATE PRODUCT STRATEGY

Corporate Product Strategy Statements (per Engineering Strategy Overview -- March 1982) and Scorpio relationships:

1. "Adopt a single VAX-11/VMS architecture."

Scorpio, in addition to providing complete conformance to VAX-11/VMS architectural specifications also extends this architecture via the introduction and definition of the BI bus and its physical interconnect implementation. The BI, which inherently supports multiple processors, is the primary bus structure for both Scorpio and Micro-VAX. In addition, on future VAXs the BI (through bus adaptors) will replace the Unibus as a system I/O bus.

2. "Implement a wide price range of products covering the computing styles of personal (individual) computing, timeshared departmental computing, and central computing."

Scorpio extends VAX applicability into new low-end markets via three new low-end products: boards, systems and VAXstations. Additionally, further low-end extensions may be achieved through bounded system implementations of the V-11 chip set.

3. "Interconnect these in a homogeneous network, including the formation of personal computer clusters."

Scorpio is the first VAX to contain an integral NI interface, requiring only an H4000 transceiver to complete a network connection. NI interconnect via an I/O processor is provided initially by the UNA, and ultimately by the BNA when available. Development of a BI-to-CI adaptor (BCA) will enable Scorpio to join the Digital cluster family.

4. "Build critical and unique applications."

The Scorpio pedestal system is the first VAX specifically designed for open-office compatibility. Scorpio VAXstations will be marketed as our first single user VAX workstations. Scorpio boards with Seaboard will become customer tools for unique application implementation.

1.4 NUMERICAL HIGHLIGHTS\*

FY'85 - FY'90 Base Data		Schedule: Q/FY	Financial Metrics
# CPU Boards Shipped	29,500	Announcement: <u>FY'85</u>	# Qtrs to Payback from System FRS: <u>14 Qtrs</u>
# Systems Shipped	74,500	CPU Boards FRS: <u>Q1 FY'85</u>	
FY'85 - FY'90 NOR	\$ 4,629M	Systems FRS: <u>Q3 FY'85</u>	Development Cost % NOR: <u>3.7%</u>
Gross Margin %NOR	57%	Prod Availability: <u>TBD</u>	NPV @ 40%: <u>\$1.69M</u>
PBT % NOR	33%	Cash Payback: <u>Q1 FY'89</u>	IRR: <u>41%</u>
		Last Ship: <u>FY'93</u>	

PACKAGED SYSTEM CONFIGURATIONS (VAX/VMS LICENSE-ONLY)

	Entry System	Typical System	Large System
Maynard List Price	\$29,900	\$41,900	\$89,900
Transfer Cost	\$ 8,890	\$11,688	\$20,977
Memory (MB)	2 MB	2 MB	2 MB
Disk Type - MB	AZTEC—21F/21R	2 AZTECS—42F/42R	RA81—456 MB
Tape Type	None	None	TU81
Performance (11/780 = 1.0)	0.7	0.7	0.7

CRITICAL DEPENDENCIES

Critical dependencies for a successful Scorpio program include:

1. Sufficient program funding.
2. Semiconductor technologies and CAD tools to ensure the availability of the V-11 chip set, TAT-020 gate array, BI interface chip (BIIC), and NI chip set.
3. Achieving transfer cost goals of \$515 for the V-11 chip set (8 chips) and \$45 for the BIIC in FY'86.
4. Optimal architecture and implementation of the BI bus physical interconnect.
5. Availability of VAX/VMS (including MP), Midnight, and Seaboard (including MP).
6. Availability of a full complement of native BI peripherals and controllers (current Unibus offering, plus DR and CI) prior to the availability of same from external vendors.

\* Forecast data includes Scorpio boards and systems only (VAXstations are not included). Forecast source is Scorpio systems product management.

1.5 MARKETING STRATEGY

1.5.1 Market Size

The following market share comparison of 32-bit systems utilizes International Data Corporation's "32-bit minicomputer market" definition as the market for Scorpio positioning. This IDC definition excludes board sets, desk top systems, general purpose computers, and small business computers, and includes only U.S. vendors.

NES Revenues (\$B) & Marketshare	FY84	FY85	FY86	FY87	FY88	FY89	FY90
Total 32-Bit Mkt	20	25	32	39	48	58	70
32-Bit Mini Mkt	5.00	7.00	9.40	12.50	16.00	20.00	25.00
Scorpio Systems	—	0.01	0.26	0.59	0.87	1.05	0.90
Market Share:							
Scorpio	0%	0.2%	2.7%	4.8%	5.5%	5.3%	3.6%
DEC 32-Bit	44%	41%	40%	39%	39%	39%	39%
Prime	<-----	-----	-----	10-15%	-----	-----	----->
DG	<-----	-----	-----	5-10%	-----	-----	----->
SEL	<-----	-----	-----	5-15%	-----	-----	----->
PE	<-----	-----	-----	5-10%	-----	-----	----->
HP	<-----	-----	-----	10-15%	-----	-----	----->
Other	<-----	-----	-----	10-20%	-----	-----	----->

Currently, TVG-Micros does not characterize board set market share by NES. However, a marketshare comparison by unit shipments is presented in Section 2.4.2.

1.5.2 Applications (Market Segmentation)

The following is a summary of targeted applications areas:

<u>Application/Segment Description</u>	<u>Marketing Message</u>	<u>Channels (Product Group)</u>
Components OEM; End User In-House System Assembly (many actual application areas)	First VAX board set.	TVG--Micros
Technical System OEM; Systems House (many actual application areas)	VAX system for < \$30K	TVG--Systems
"Seaboard" Program Development	Least expensive VAX system	TVG--Micros, TVG--Systems

Professional* Workstation	Least expensive single user VAX workstations with integral NI.	ECS, ESG, LDP, PBI
General Business/EDP and Office Automation	Small system member of large family.	COEM, CSI, ECS, GSG, PBI
Front-End Real-Time	Best VAX price/performance; integral NI	GSG, LDP, TIG
Network Servers	Best VAX price/performance	CSI, GSG, LDP, PBI

\* Scorpio systems (in addition to Scorpio VAXstations) will be purchased as "workstations".

### 1.5.3 Geographic Considerations

1. Distribution: Scorpio boards and Scorpio systems are planned to be sold in every country served by Digital and its distributors. Anticipated distribution by area is as follows:

- Scorpio Boards

U.S.	60%
Europe	30%
GIA	10%

- Scorpio Systems

U.S.	50%
Europe	35%
GIA	15%

2. Product Variants

- Country Kits: The implementation of local language kits is being pursued.
- Local AC Power: The Scorpio power supply AC voltage range design will necessitate system input transformers in Japan and Western Australia.

1.6 PHASE PLANNER

Scorpio phase transition dates are as follows:

<u>Phase</u>	<u>Date of Exit from Phase (Q/FY)</u>
0 (Strategy & Requirements)	<u>June 1982</u>
1 (Planning)	<u>Q2 FY'83</u>
2 (Implementation)	<u>Q4 FY'84</u>
3 (Qualification)	<u>Q4 FY'85</u>
4 (Production and Support)	<u>TBD</u>
5 (Retirement)	<u>TBD</u>

1.7 RISK ASSESSMENT

The asterisk (\*) on each of the following scales indicated the relative risk associated with that subject.

	<u>Low Risk</u>	0 1 2 3 4 5 6 7 8 9 10	<u>High Risk</u>
1.7.1 Technology			
- For Industry			
Well established, know how to do		*	Never done before
		0 1 2 3 4 5 6 7 8 9 10	
- For Digital			
Well established, know how to do		*	Never done before
		0 1 2 3 4 5 6 7 8 9 10	
1.7.2 Manufacturing			
- Test Procedures			
Well established		*	New Techniques
		0 1 2 3 4 5 6 7 8 9 10	
- Processes			
Well established		*	New
		0 1 2 3 4 5 6 7 8 9 10	

1.7.3 Service

Similar to many products, high ease of diagnosis	* _____ 0 1 2 3 4 5 6 7 8 9 10	New techniques, new service procedures, . much training, sparing
--	--------------------------------------	--

1.7.4 Marketing

- Distribution Channels

Traditional end-user direct sales, or OEM/reseller; standard terms and conditions	* _____ 0 1 2 3 4 5 6 7 8 9 10	New distribution channels, new terms and conditions, new order processing procedures
---	--------------------------------------	--

- Customer Base

Current Digital customers	* _____ 0 1 2 3 4 5 6 7 8 9 10	New Digital customers
---------------------------	--------------------------------------	-----------------------

- Customer Capability

Highly technical	* _____ 0 1 2 3 4 5 6 7 8 9 10	Computer naive
------------------	--------------------------------------	----------------

- Applications

Well understood, done before	* _____ 0 1 2 3 4 5 6 7 8 9 10	New applications
------------------------------	--------------------------------------	------------------

1.8 PROGRAM ISSUES

The following items are currently identified as Scorpio program issues requiring resolution:

- BI as an Industry Standard: Should Digital attempt to make the BI an industry standard bus?
- BI Propriety: Should the BI be a public or licensed bus?
- BI Licensing Mechanism: If the BI is to be licensed, what is the licensing mechanism?
- Console Load Device: Desire to change the system console load device from a TU58 to an RX50 or other floppy.
- Packaged Systems vs System Building Blocks: How should the H9640 cabinet based systems be packaged/marketed? Note that this business plan assumes "packaged systems".
- BI Line Printer Interface Strategy: Desire to interface line printers to an asynchronous or synchronous line rather than a special parallel interface.

CHAPTER 2

PRODUCT DESCRIPTION

2.1 PRODUCT LIFE CYCLE OVERVIEW

2.1.1 Product Offerings and Potential Migration Strategy

The following matrix defines the FRS and primary shipment years (denoted by an "X") of the anticipated Scorpio products:

SCORPIO PRODUCT	FY85	FY86	FY87	FY88	FY89	FY90
CPU Board Set	Q1	X	X	X	X	X
OEM Box	Q3	X	X	X	X	X
Single AZTEC System	Q3	X	X	—	—	—
Dual AZTEC System	Q3	X	X	—	—	—
Dual RA60 System	—	Q1	X	—	—	—
RA81/TU81 System	—	Q1	X	—	—	—
RDXX/Maya System	—	—	X	X	X	X
Single AZTEC II System	—	—	X	X	X	X
Dual AZTEC II System	—	—	X	X	X	X
Dual RAXY System	—	—	X	X	X	X
RAXX/TA81 System	—	—	X	X	X	X
Scorpio VAXstation	—	Q1	X	X	X	X

2.1.2 Affected and Successor Products

The primary products affected by Scorpio are the VAX-11/730 and VAX-11/750. High-end Orion systems also may be affected, but to a much lesser extent. Potential successor products at the low end include a Micro-VAX "system" (if so strategized) and Peevee. However, Nano-VAX should be the ultimate Scorpio replacement. Phaseover dates cannot be identified at this time.

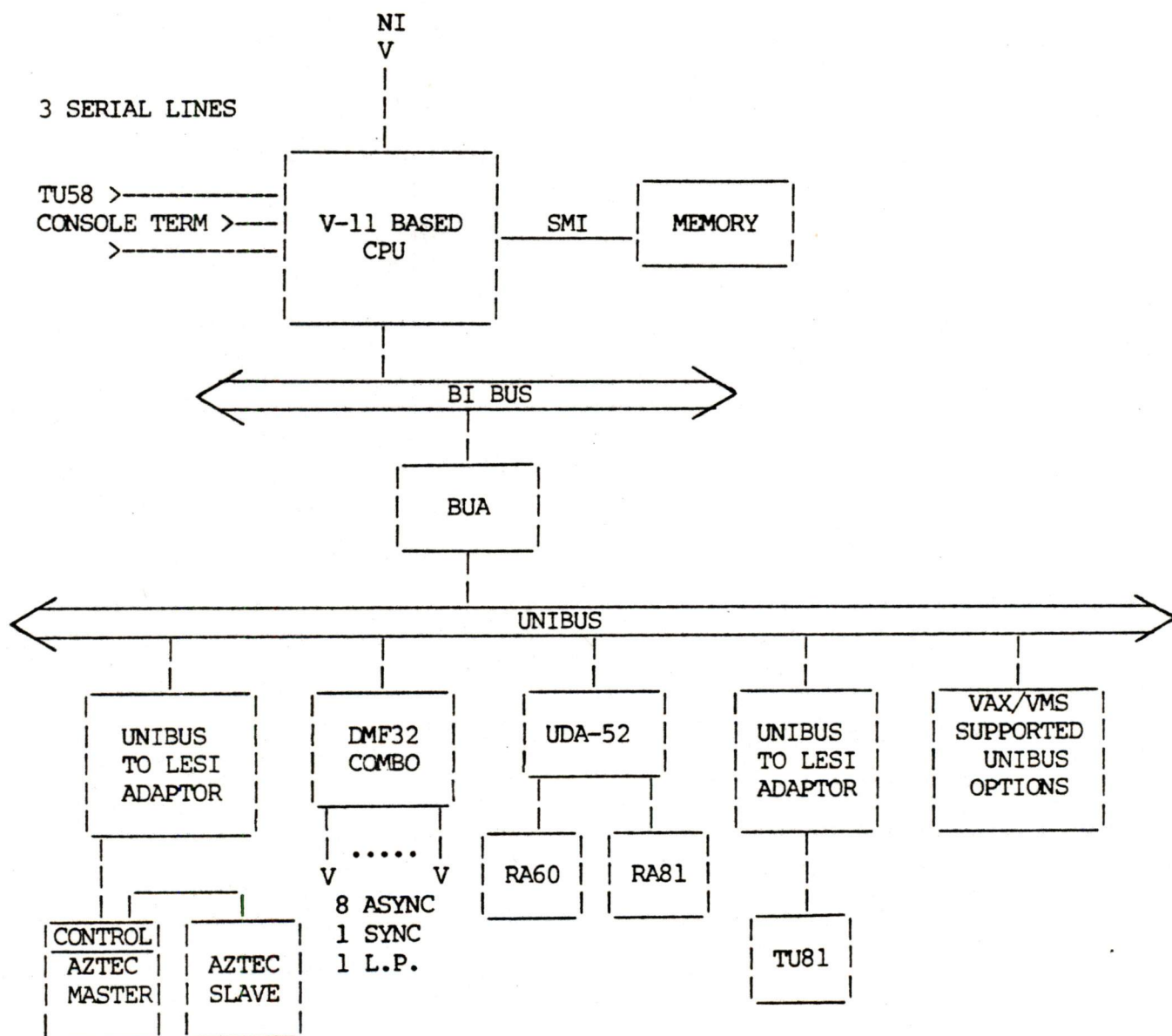


## 2.2 KEY FEATURES — INITIAL PRODUCT OFFERINGS

### 2.2.1 Initial Scorpio Architecture

The heart of the Scorpio architecture is the new Backplane Interconnect (BI) bus, a bus optimized for bandwidth, multiple processors, integrity and RAMP. Integral to the BI is a new physical interconnect scheme which includes a new Euro compatible module form factor of 8.00" x 9.18".

The processor, memory, and BI-to-Unibus Adaptor (BUA) modules conform to this new module size. The processor and BUA connect directly to the BI, while memory connects to a private Scorpio Memory Interconnect (SMI) bus. Initially, Scorpio peripherals connect to the Unibus, only because of the unavailability of equivalent BI peripherals.



## 2.2.2 System Component Descriptions

### 2.2.2.1 Processor Module

The Scorpio central processor is contained on a single Euro compatible module. Two custom VLSI processor chips (I/E and M chips) and a control store implement the VAX instruction set including warm floating point instructions that support S, D, G, and H formats. A third, optional, custom VLSI chip (F chip) provides floating point instruction acceleration. Instruction execution speed of Scorpio with the F chip is characterized as 70% of a VAX-11/780 with FPA.

Scorpio control store, which is 40 bits wide, is implemented by five custom RAM/ROM chips to provide 16K 40-bit words of ROM and 1K 40-bit words of RAM. The patchable RAM area is used for microcode updates, thus precluding the necessity of ROM replacement.

Additional processor functionality includes: (1) a cache with an 8KB, single set, direct map architecture, (2) three serial lines, one for the console terminal, one for the TU58 console load device, and the third for general user interface, (3) VAX console compatibility, and (4) self test diagnostics in hardware. Writeable control store (WCS) is not available on Scorpio.

A memory controller is also included on the processor module. This logic creates the Scorpio memory interconnect (SMI) bus (with 30-bit addressing) to which memory array modules attach.

Unique to Scorpio is an integral NI interface on the processor module. This implementation contains an Ethernet protocol and Ethernet driver chip, utilizes local processor memory for buffers, and employs the Scorpio CPU for NI processing. Only the H4000 transceiver must be added to complete the NI connection.

Lastly, the processor module contains an interface to the BI, provided by the BI interface chip (BIIC) and clock receiver chip.

### 2.2.2.2 Memory Array

Both a 1/2 MB (64K RAM) array and a 2 MB (256K RAM) array module are planned for Scorpio. The 1/2 MB module is intended primarily for TVG-Micros components sales, while the 2 MB array is included in systems. Both modules utilize the same printed circuit board of Eurocard form factor. Each array contains ECC and two-way interleaving on the module, and interfaces to the 13.3MB/sec SMI bus.

### 2.2.2.3 Backplane Interconnect (BI) Bus

The BI is a clocked synchronous, interlocked bus with a 200 nsec cycle time. The bus supports 30-bit physical addressing, multiple processors, up to 16 total nodes, and a bandwidth of up to 13.3 MB per second. BI bus integrity and RAMP are achieved through (1) bus parity, (2) BI node diagnosability, (3) a standard two chip interface that performs all bus protocol, and (4) controlled AC loading characteristics. Bus length can be up to 5.0 feet (1.5 meters) maximum.

While the BI is the "system" bus for TVG-Micros' Scorpio board sets, the BI initially serves as an internal bus for Scorpio systems. However, as native BI peripherals replace their Unibus counterparts, the BI will become the primary bus for Scorpio systems and the BUA will become an optional adaptor. Ultimately, future VAXs will contain multiple BI buses (via adaptors) for I/O, replacing the current Unibus.

The implementation of the BI physical interconnect is key to the success of this new bus. In fact, many of Scorpio's inherent features, and goals depend on this interconnect. Backplane slot independence, multiple processor/multiple local memory support, ease of backplane expansion, customer option installability and repairability, and the absence of I/O cable connectors on modules, to name a few, all depend on the physical interconnect.

The physical interconnect begins with the new 8.00" x 9.18" Euro compatible module. This form factor was selected to (1) satisfy TVG-Micros board size product requirements, (2) provide a "Euro" marketing presence for European board sales, (3) provide adequate real estate for system options, and (4) optimize raw board costs. Associated with this module is a new 300 pin connector (along the 9.18" edge) and resultant backplane. Connector/backplane goals include a ZIF design implementation and I/O cable exit from the backplane, in addition to those previously mentioned. Note that this BI design is the first totally new system interconnect since the PDP-8 "black block" implementation in 1965.

The BI is public/licensed bus. The availability of interface chips, backplane/connectors, interface specifications and application notes provide a user with the necessary tools to interface to this new bus. Thus, Scorpio becomes the first VAX to allow a user to design an interface to a bus with greater bandwidth than the Unibus. These interface tools being available to external customers in Q1 FY'85 should provide adequate incentive for Digital design organizations to expeditiously bring native BI peripherals to market.

#### 2.2.2.4 BI-to-Unibus Adaptor (BUA)

This adaptor is implemented on a single Euro compatible module, and is similar in functionality to the Unibus adaptors on the VAX-11/780 and VAX-11/750. BUA functionality includes (1) mapping from Unibus address space to BI address space and from BI address space to Unibus address space, (2) protocol translation for data transfers and interrupts from Unibus to BI, (3) Unibus priority arbitration and interrupt fielding, and (4) optimization of Unibus transactions. The adaptor contains one direct data path and a single octaword buffered data path.

Although multiple BUAs are architecturally possible, this functionality is not committed.

### 2.2.2.5 Power and Packaging

Scorpio power and packaging provides for a transition from BI/Unibus systems to BI only systems. Initially, all systems contain a combination BI and Unibus backplane and the associated power distribution. The Unibus, an integral system component, is the system I/O bus for both Digital peripherals (due to the lack of BI peripherals) and existing customer designed peripherals. Ultimately, as BI options become available, packaging will migrate to BI only with the BUA available as system option and all Unibus peripherals mounted in an external Unibus expander box.

The Scorpio system design center is a new desk height pedestal package that contains the combination BI/Unibus backplane and either one or two AZTEC disks. Compatibility with open-office environments is addressed by specific attention to cosmetics, form factor and acoustic management. The current pedestal design is only 26 inches high and 24 inches deep allowing it to fit under a table or work surface without protrusion. Acoustic management is achieved by removal of the AZTEC fans and employment of a single custom designed, low acoustics air mover for the entire enclosure. In addition, both the pedestal size and cost are reduced by the removal of the AZTEC power supplies in favor of a single universal power supply for the complete enclosure. The power supply design concept is that of a central supply to power all internal systems components, including the AZTEC disks. The supply is comprised of three separate hybridized regulators, that generate all voltages (BI, Unibus, and Storage). The pedestal power supply requires a 20 amp maximum line cord and wall receptacle at 120 VAC. Pedestal FCC compliance is achieved through RF gasketing and bulkhead I/O connectors.

A second Scorpio package is a 10.5 inch high OEM box with slides. This chassis also contains a combination BI/Unibus backplane and associated power distribution. The box serves both as the Scorpio OEM vehicle, and, when mounted in an H9642 cabinet, as the processor cabinet for dual RA60 and RA81/TU81 packaged systems.

Packaging for TVG-Micros board products consists of a six slot BI backplane and an inter-backplane connector/cable for BI extension. Currently, no power supply products are planned for board set products.

### 2.2.3 Primary Product Deliverables

FRS = Q1 FY'85

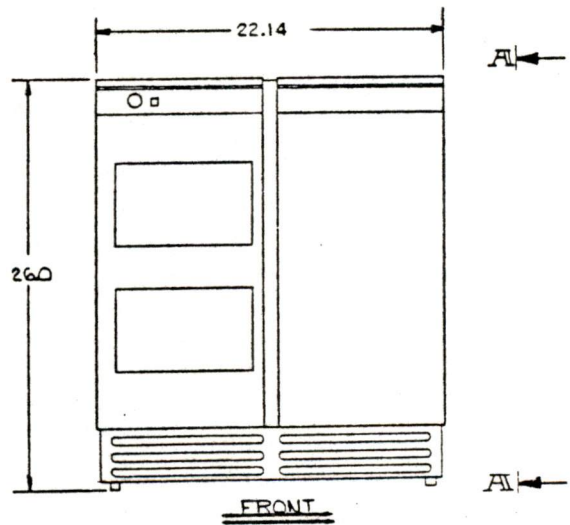
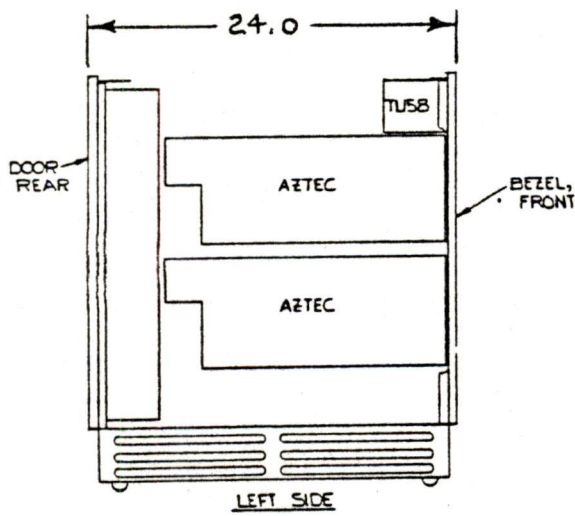
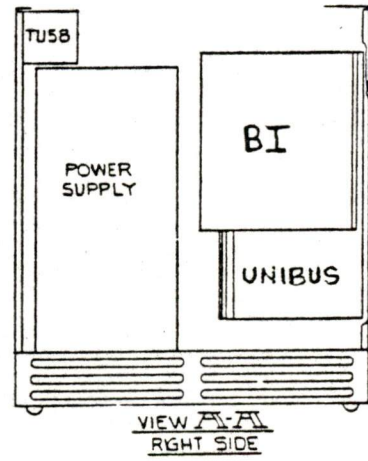
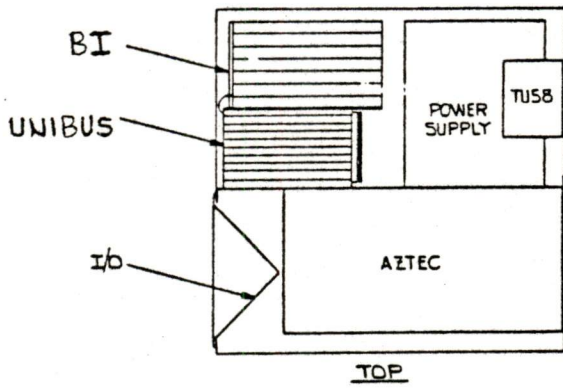
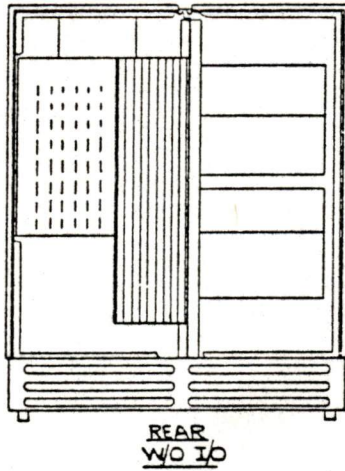
- CPU Module
- 1/2 MB ECC Memory Array Module
- BI Backplane/Module Connector
- "Seaboard" Run-Time System V2.0
- BI Interface Chip Set, Documentation, Application Notes

FRS = Q3 FY'85

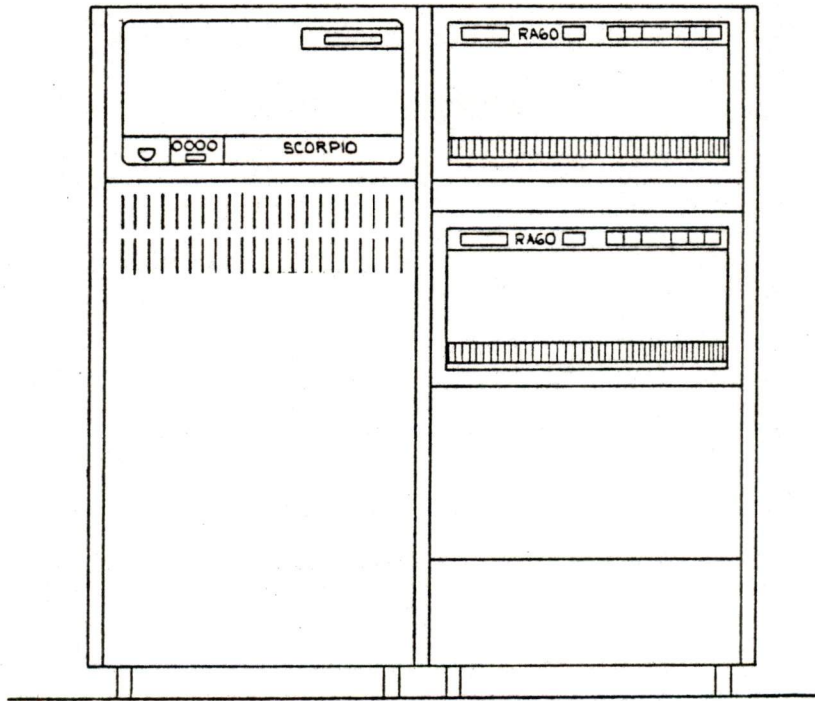
- OEM Box
  - CPU
  - 2 MB ECC Memory Array
  - Unibus Adaptor
  - 10 1/2" Box, Power Supply, TU58, Slides
  
- Single AZTEC System
  - CPU
  - 2 MB ECC Memory Array
  - Unibus Adaptor
  - Single Unibus AZTEC Subsystem (21MB F/21MB R)
  - Pedestal Packaging With TU58
  - Console Terminal
  - VAX/VMS V4.0
  
- Dual AZTEC System
  - CPU
  - 2 MB ECC Memory Array
  - Unibus Adaptor
  - Dual Unibus AZTEC Subsystem (2 x 21 F/21 R)
  - DMF32 Combo Module
  - Pedestal Packaging With TU58
  - Console Terminal
  - VAX/VMS V4.0
  
- 2 MB ECC Memory Array
  
- Battery Backup
  
- VAX/VMS V4.0
  
- Floating Point Accelerator Chip

FRS = Q1 FY'86

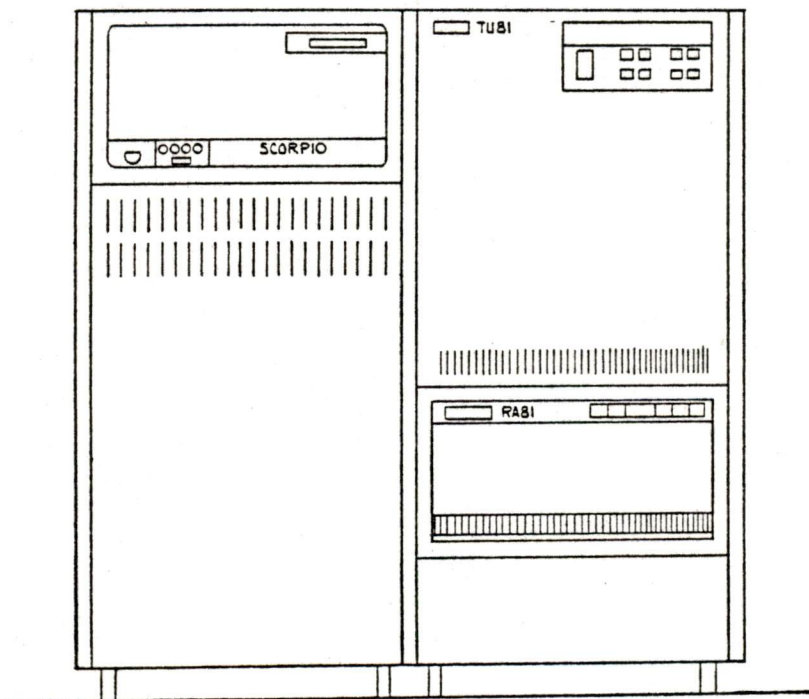
- Dual RA60 System
  - OEM Box Mounted in H9642 Cabinet
  - Unibus Combo Module
  - UDA-52
  - 2 RA60s in H9642 Cabinet
  - Console Terminal
  - VAX/VMS V4.0
  
- RA81/TU81 System
  - OEM Box Mounted in H9642 Cabinet
  - Unibus Combo Module
  - UDA-52
  - TU81 in H9642 Cabinet
  - RA81 Mounted in TU81 Cabinet
  - Console Terminal
  - VAX/VMS V4.0



Scorio Dual AZTEC Pedestal System



Scorpio Dual RA60 System



Scorpio RA81/TU81 System

#### 2.2.4 Marketable Features--Relating to Customer Utility

**Hardware and Packaging:** The open-office pedestal packaging with its acoustic management (an overdue departure from previous Digital designs) provides considerable customer utility. Additional features of importance include greater than VAX-11/750 performance on a single module, the integral NI, multi-processor support, greatly improved system integrity provided by the BI, and the ability of a user to interface to this 13.3 MB/sec bus. Scorpio board sets, utilizing only the BI bus (no Unibus), create a new benchmark for micro system integrity.

**Software:** VAX/VMS and layered product support of Scorpio, of course, extend the availability and applicability of VAX systems. In addition, Midnight further extends VAX/VMS into new, less sophisticated markets. However, Seaboard, with its real time tuning and symmetric multiple processor support, provides new Digital solutions to 32-bit application problems.

**Services:** Planned is a tailoring of service offerings to be more closely aligned with customer needs. Service features include:

- Module mailer
- Parts dispatch
- Carry-in service centers
- Telephone support centers
- Tailored contracts
- Distributed remote diagnosis as a system support tool

**RAMP:** Reliability, accessibility and maintainability features include:

- Self test diagnostics in hardware (to the 90% confidence level) for each BI option at power-up.
- High reliability (MTBF).
- Low repair time (MTTR).
- Visual fault indication on each BI option.
- BI peripheral I/O cable connector on backplane rather than module.
- Customer Runnable Diagnostics (CRD).
- The inherent BI bus integrity providing new levels of system integrity.
- New BI module form factor provides for more efficient cooling.
- Thermally efficient system packaging.
- Accessibility designed into the pedestal package and OEM box.



Price: Aggressive low end pricing is planned to ensure an increased market share in this 32-bit price band. AZTEC pedestal packaging, Midnight software and customer installability contribute heavily toward the ability to price Scorpio systems very competitively.

Customer Cost of Ownership Minimization: Features to ensure low cost of ownership include:

- Low purchase price.
- High reliability.
- Low BMC; tailorable service offerings.
- Reduced AC power consumption.
- Less stringent environmental requirements (acoustics and temperature).
- Smaller system footprint and volume requirements.

International: Primary international features are:

- Euro compatible module form factor.
- Universal power supplies.
- CSA certification; VDE compliance.
- Local language customer runnable diagnostics.
- Country kits with local language system documentation (a goal).

#### 2.2.5 Features to Reduce Life Cycle Cost to Digital

Maximization of corporate return on investment is achieved by specific attention to the following Scorpio program components.

Transfer Cost: Transfer cost is minimized by:

- Scorpio's CPU being a single, Euro compatible module with performance greater than a VAX-11/750 (four extended Hex CPU modules).
- Ultimate elimination of the Unibus as the standard I/O bus.
- Pedestal package design of single air mover, and single power supply (which also eliminates the need for an enclosure power controller).
- Process testing, rather than product testing, resulting in increased yields and a reduction of test times and test equipment.

- Utilization of high reliability components and requiring burn-in at the component level.
- Manufacturing team's early participation in the program to develop high volume manufacturing processes and to ensure timely maturity of these processes.

FA&T: The elimination of FA&T will be accomplished by:

- Ensuring Unibus options are site mergable.
- The ultimate realization of "BI only" systems.

Installation Cost and Quality: Cost is reduced while maintaining quality via:

- Customer installation of systems.
- Installation "hot-line" telephone support.

Warranty and Service Costs: These costs are minimized by:

- Customer installation of systems.
- Self test diagnostics in hardware.
- Customer runnable diagnostics.
- Low MTTR.
- Remote diagnosis support.
- Low parts count for branch inventory.

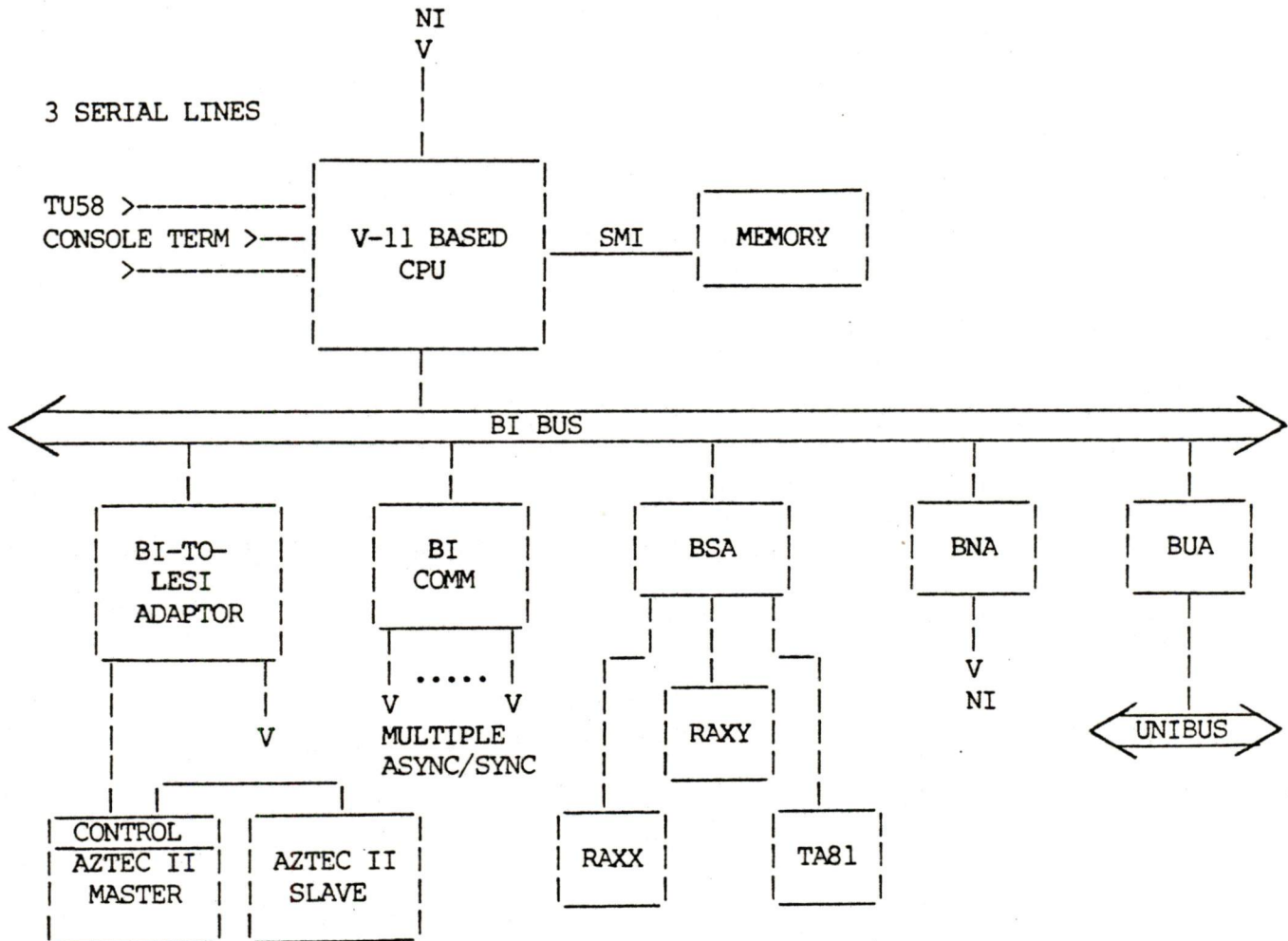
Manufacturing Inventory: The associated costs are minimized by:

- A reduction of Stage 2 inventory to 8.5 weeks.
- A reduction of Stage 1 inventory by requiring weekly delivery of raw material.

## 2.3 FUTURE SYSTEM/FOLLOW-ON PRODUCT OFFERINGS

### 2.3.1 Ultimate Scorpio Architecture

As native BI peripherals are developed, Scorpio systems will migrate toward the desired "BI only" status. The BUA still will be offered as an option, but it primarily will be used for attachment of customer designed Unibus options. Only at this time will the utility of the BI bus be fully realized.



### 2.3.2 Future Scorpio Peripherals

**BI-to-LESI Adaptor:** This single BI module adaptor is currently being developed by Low-End VAX Systems Development as a part of the Scorpio program. Unresolved funding issues preclude an availability commitment of FRS. The LESI port provides an interconnect to AZTEC, AZTEC II, TU81, and RD/RX storage products. A two port adaptor is being considered.

**BI COMM:** This one module BI communications multiplexor is "similar" to the Unibus DMF-32 combo. Like the LESI adaptor, this peripheral is also being developed within the Scorpio program and suffers from the same funding issues.

BSA (BI-to-SI Adaptor): This BI adaptor creates the SI bus for up to four SI disks or tapes (any combination). The product, which can be viewed as either a UDA-52 or superset or an HSC-50 subset, is being developed by Storage Systems Development. FRS is planned for FY'86.

BNA (BI-to-NI Adaptor): The BNA is an I/O processor based NI interface providing hardware protocol stripping and other intelligence. This option is being developed by Distributed Systems.

UPC (Utility Port Conditioner): This device, to be developed by Low-End VAX Systems Development, connects between the system and the AC power input to reduce mains distortion and to provide power factor correction.

4MB Memory Array: This single Euro module incorporates 256K MOS RAMS in surface mounted plastic chip carriers to achieve the packing density.

BCA (BI-to-CI Adaptor): The BCA, a requirement for Scorpio to join the Digital CI cluster community, is anticipated as a two Euro module device, even with a high degree of VLSI. The BCA is currently unfunded.

BI "DR32": Also unfunded is a general purpose BI interface. Product functionality probably should be between the DR780 and the DR11-W.

Other: To ensure timely and effective Unibus replacement, numerous other BI peripheral (e.g., A/D, D/A, programmable clocks) and/or alternative front end subsystem solutions need to be identified and funded.

### 2.3.3 Future Scorpio Systems

Packaged Systems: The anticipated system migration strategy (shown in Section 2.1.1) correlates directly to Storage Systems' product strategy. Changes in disk and tape offerings will result in corresponding changes in Scorpio packaged systems. However, a more important migration issue is that of Unibus elimination in favor of native BI systems. The success of this strategy correlates directly to the timeliness of BI peripheral development. The matrix in Section 2.1.1 assumes native BI systems in FY'87 which is much later than desired. The preferred Scorpio strategy is native BI systems at FRS.

BI Multiprocessing: While Seaboard V2.0 supports multiprocessing at board set FRS, VAX/VMS is not expected to support MP until a release after V4.0. This VAX/VMS support is for an asymmetric two processor system. The potential of adding a third processor will be investigated.

CI Clusters: A prerequisite for CI clusters, of course, is the currently unfunded BCA. Thus, clusters and the applicability of Scorpio program deliverables to "enhanced availability systems" will have to be determined at a later date.

#### 2.3.4 Future V-11 Based Products

Besides Scorpio boards, systems, and VAXstations, other Digital products will consider the V-11 for its engine. Specifically, the Advanced Development project "Pee Wee" is one such application. Pee Wee is envisioned as a quad module SBC interfaced to the expanded Q22 (Orion) bus creating an "odd couple" 16/32-bit LCP configuration potential.

Another potential product is a "CT" configuration. The V-11's compatibility mode could assist in a 16 to 32-bit CT migration strategy.

#### 2.4 SYSTEM/PRODUCT POSITIONING

More detailed product positioning data will be provided in future revisions of this plan.

##### 2.4.1 Digital Products

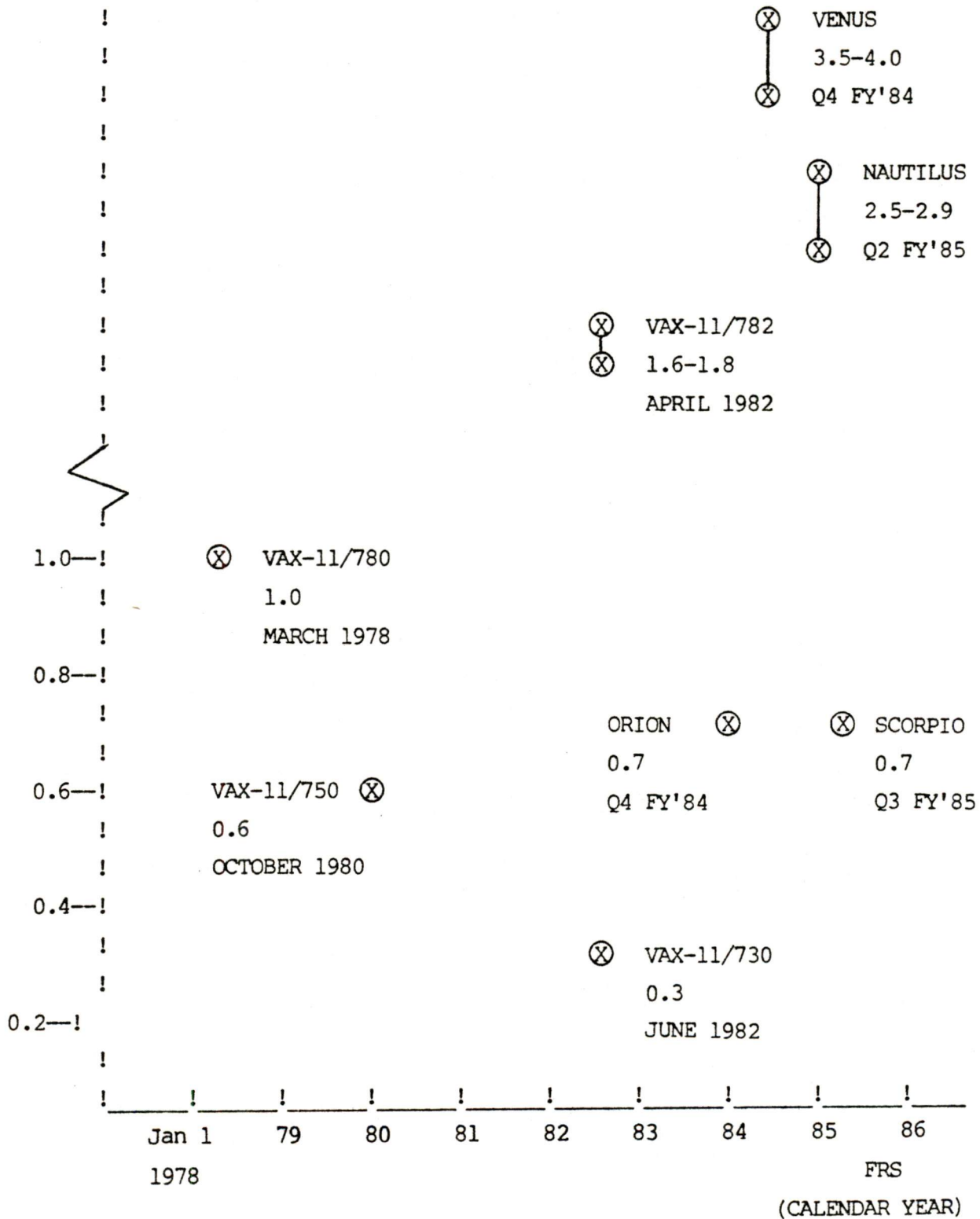
###### 2.4.1.1 Digital Systems

Relative processor performance is shown graphically on the next page. Relative processor/system specifics are summarized below:

###### VAX-11/780

- Processor performance is 1.0 vs 0.7 for Scorpio. However, with VAX/VMS multiprocessor support, the addition of a second single CPU module should raise Scorpio performance to 1.2.
- The 11/780 CPU has 20 extended Hex modules vs a single Euro module for the Scorpio CPU.
- The maximum 11/780 memory is 8MB (12MB with an MA780) vs 8MB for Scorpio (assumes 4 BI slots of 2MB each).
- The minimum 11/780 processor with memory and I/O bus is 28 extended Hex modules vs 2 Euro modules for Scorpio (native BI system).
- The 11/780 floating point accelerator is five extended Hex modules compared to a chip for Scorpio.
- WCS is an option on the 11/780 but not available on Scorpio.
- The 11/780 is packaged in H9600 corporate cabinets only, the CPU alone requiring a double-width high-boy. Scorpio is available in a pedestal, 10 1/2" box and H9640 cabinets.

PROCESSOR PERFORMANCE (RELATIVE TO VAX-11/780) AND SYSTEM FRS DATE



#### VAX-11/750

- Processor performance is 0.6 vs 0.7 for Scorpio.
- The 11/750 CPU has 4 extended Hex modules vs the single Euro module.
- The maximum 11/750 memory is 8MB.
- The minimum 11/750 processor with memory and I/O bus is 6 extended Hex and 1 Hex modules vs the 2 Euro boards for a native BI Scorpio.
- The FPA is an extended Hex vs Scorpio's chip.
- The WCS option is a daughter board that mounts on an existing CPU module.
- The 11/750 is packaged as a 31 1/2" cabinet mountable chassis (11/751), and in H9640 series cabinets, the CPU being housed in an H9645.

#### VAX-11/730

- Processor performance is 0.3 vs 0.7.
- The 11/730 CPU comprises 3 Hex modules vs one Euro.
- The maximum 11/730 memory is 5MB.
- The minimum system 11/730 processor with memory and I/O bus is 4 Hex modules.
- The FPA is an additional Hex module.
- 11/730 packaging includes a 10 1/2" box and systems in a single H9642 cabinet.

#### Orion

- Orion's performance is characterized as 0.7, identical to Scorpio.
- The CPU is a single quad board.
- Maximum Orion memory is 4 MB.
- The minimum Orion processor with memory and I/O bus is 2 quad modules for a Q22 systems and 3 quad modules for a Unibus system.
- The FPA is a chip addition to the CPU board.
- Packaging includes LCP for low-end systems, a 5 1/4" box with Q22 bus, a 10 1/2" box with Unibus, and H9640 cabinets for Unibus systems.

#### 2.4.1.2 Digital Board Sets

Relative board set specifics are summarized below. Expanded comparisons will be provided in future business plan revisions.

##### KDJ11

- The KDJ11 (Orion's CPU), although a PDP-11 board, is compared to Scorpio because of its anticipated equal performance and lower transfer cost (\$700).
- The KDJ11 is a single quad module that interfaces to an expanded Q22 bus.

##### Micro-VAX

- Micro-VAX is envisioned as Digital's first VAX subset offering, comprising a new single chip CPU with local memory interfaced to the BI on a single Euro module. More information will be provided in future editions of this document.

#### 2.4.2 Competitive Products

##### 2.4.2.1 Market Size

Systems: The total market in which our VAX systems compete is the total 32-bit systems market which encompasses all systems within a \$16K to \$100K priceband, with an address length of 24 bits or greater, and classified by International Data Corporation as either a minicomputer, a general purpose computer or a small business computer. Because this definition includes most of IBM's products, Digital's total corporate NES represents only about 5% of this market today. Work is underway to subset this market by priceband to provide a more focused market definition for positioning specific class systems. Note that our total 32-bit market excludes systems (mainly desktop) under \$16K because at this time no product exists in our portfolio with this definition.

In lieu of this priceband segmentation, we have chosen to use IDC's 32-bit minicomputer market (a subset of the total 32-bit market) to position Scorpio. Although this excludes most of IBM's products because they are not technically classified as minicomputer products (under IDC's definition), it still represents a useful metric for our system sales potential because it does represent our full minicomputer competition. The market sizes shown on the next page represent system vendor equipment revenues, and include world-wide shipments from U.S. vendors only.



NES Revenues (\$B) & Marketshare	FY84	FY85	FY86	FY87	FY88	FY89	FY90
Total 32-Bit Mkt	20	25	32	39	48	58	70
32-Bit Mini Mkt	5.00	7.00	9.40	12.50	16.00	20.00	25.00
Scorpio Systems	--	0.01	0.26	0.59	0.87	1.05	0.90
Market Share:							
Scorpio	0%	0.2%	2.7%	4.8%	5.5%	5.3%	3.6%
DEC 32-Bit	44%	41%	40%	39%	39%	39%	39%
Prime	<-----	-----	-----	10-15%	-----	-----	----->
DG	<-----	-----	-----	5-10%	-----	-----	----->
SEL	<-----	-----	-----	5-15%	-----	-----	----->
PE	<-----	-----	-----	5-10%	-----	-----	----->
HP	<-----	-----	-----	10-15%	-----	-----	----->
Other	<-----	-----	-----	10-20%	-----	-----	----->

In terms of our 32-bit "minicomputer" competition, Prime and HP should provide our stiffest competition, especially HP if they can overcome their late entry into the 32-bit systems market. Future profiles will attempt to incorporate more of the future revenues of IBM and Wang (among others) which are excluded here because of the "minicomputer" definition employed. In addition, an attempt will be made to include foreign vendors in these future profiles.

Board Sets: Digital currently characterizes market size and share in terms of units rather than NES. The world-wide board set market for 32-bit microprocessor boards is shown below. Competitive share will be presented in future editions.

Board Unit Ships (000) & Marketshare	FY84	FY85	FY86	FY87	FY88	FY89	FY90
Total 32-Bit Mkt	8.0	16.5	31.5	55.7	90.8	135.7	184.2
Scorpio Systems	--	0.5	1.3	3.3	5.4	8.1	11.1
Market Share:							
Scorpio	0%	3.0%	4.1%	5.9%	5.9%	6.0%	6.0%
DEC 32-Bit	0%	4.5%	7.5%	12.5%	17.5%	22.5%	27.5%

#### 2.4.2.2 Competing Systems

This section to be provided in future editions.

### 2.4.2.3 Competing Board Sets

The primary board set competition is expected to be:

1. Intel 432 and follow-on (386)
2. Motorola 68000 series (68020)
3. National 16000 series (16032)
4. Hewlett Packard 32-Bit MPU
5. TI 99000
6. Zilog Z80000
7. Bell MAC-32

Detailed competitive comparisons will be provided in future editions of this document.

### 2.4.3 Price Positioning Charts

The following system price band charts assume systems with operating system software (support-inclusive). Specific system pricing will be included in future revisions of this document. Price band charts are not currently developed for board sets.

#### 2.4.3.1 System Price Bands Against Digital Products

System Price Band	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90
\$100-\$250K	11/750 RA81/RA60 11/750 Dual RA60	Nautilus RA81/TA81 Nautilus Dual RA60	Nautilus RAXX/Adv Tape Nautilus Dual RAXY	Nautilus RAXX/Adv Tape Nautilus Dual RAXY	Nautilus RAXX/Adv Tape Nautilus Dual RAXY	Nautilus RAXX/Adv Tape Nautilus Dual RAXY
\$65-\$100K	11/750 RA80/RL02 11/730 RA81/RA60 11/730 Dual RA60	Nautilus Box Scorpio RA81/TU81 Scorpio Dual RA60	Nautilus Box Scorpio RAXX/TA81 Scorpio Dual RAXY	Nautilus Box Scorpio RAXX/TA81 Scorpio Dual RAXY	Nautilus Box Scorpio RAXX/TA81 Scorpio Dual RAXY	Nautilus Box Scorpio RAXX/TA81 Scorpio Dual RAXY
\$40-\$65K	11/730 R80/RL02 11/730 Dual AZTEC Orion RA81/RA60 Orion Dual RA60	Scorpio Dual AZTEC	Scorpio Dual AZTEC II	Scorpio Dual AZTEC II	Scorpio Dual AZTEC II	Scorpio Dual AZTEC II
\$6-\$40K	11/730 Box Orion Dual AZTEC Orion Single AZTEC Orion RD51/RX50 Orion Box	Scorpio Single AZTEC Scorpio Box Orion Dual AZTEC Orion Single AZTEC Orion RD51/RX50 Orion Box	Scorpio Single AZTEC II Scorpio RDXX/MAYA Scorpio Box Orion Dual AZTEC II Orion Single AZTEC II Orion RDXX/MAYA Orion Box	Scorpio Single AZTEC II Scorpio RDXX/MAYA Scorpio Box Orion Dual AZTEC II Orion Single AZTEC II Orion RDXX/MAYA Orion Box	Scorpio Single AZTEC II Scorpio RDXX/MAYA Scorpio Box Orion Dual AZTEC II Orion Single AZTEC II Orion RDXX/MAYA Orion Box	Scorpio Single AZTEC II Scorpio RDXX/MAYA Scorpio Box Orion Dual AZTEC II Orion Single AZTEC II Orion RDXX/MAYA Orion Box

### 2.4.3.2 System Price Bands Against Competitive Products

The source of this data is the 32-Bit Program Office. Note that Hewlett Packard along with Charles River Data, Symbolic Systems, etc. are considered to VAXstation competitors and thus are not included below.

System Price Band	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90
\$100-\$250K	IBM CATLIN 2&3 (4300 Type)  IBM S/38 4X, & 7	IBM CATLIN 2&3 (4300 Type)  IBM S/38 4Y, & 7X	IBM CATLIN 2&3 (4300 Type)  IBM S/38 4Y, & 7X			
\$40-\$100K	IBM CATLIN 1 (4300 Type)  IBM S/38-3X  IBM Sushine/ Sunstar (S/34 Upgrade)  NEW FASTER SEL 32/27  DG MV3000  FASTER PRIME 150/250  WANG VS95	IBM CATLIN 1 (4300 Type)  IBM S/38-3Y  IBM Sushine/ Sunstar (S/34 Upgrade)  NEW FASTER SEL 32/27  DG MV3000  FASTER PRIME 150/250  WANG VS95	IBM CATLIN 1 (4300 Type)  IBM S/38-3Y  IBM Sushine/ Sunstar (S/34 Upgrade)  NEW FASTER SEL 32/27  DG MV3000  FASTER PRIME 150/250  WANG VS95			
\$6-\$40K	PRIME RABBIT  PE 3210 NEW  DG MV1000	PRIME RABBIT  PE 3210 NEW  DG MV1000	PRIME RABBIT  PE 3210 NEW  DG MV1000			

### 2.5 TECHNOLOGY

Technology advancements critical to Scorpio include:

- ZMOS process for the V-11 chip set and BIIC.
- CAD tools necessary for the execution of custom MOS and TAT-020 gate array development.
- BI physical interconnect including new module connector, I/O connector and backplane.
- 256K MOS RAM chip for memory.
- BI printed circuit board packaging technology.
- Winchester technology for AZTEC disk.

- Custom pedestal air mover design--an existing technology, but new to Digital.
- Power supply hybrids--an existing technology, but new to Digital.

Product positioning relative to state-of-the-art is characterized as:

- ZMOS process is state-of-the-art for Digital.
- As a system, Scorpio is highly competitive, but not leading edge, state-of-the-art technology.

Likely replacement technologies include:

- CMOS Nano-VAX CPU.
- 1M bit MOS RAM for memory.
- Advanced Winchester technology for AZTEC II, RAXX and RAXY.

CHAPTER 3

SHIPMENT FORECAST

3.1 WORLD-WIDE FORECAST

3.1.1 Scorpio Boards and Systems Forecast (Product Management Supplied)

SCORPIO	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90	FY85-90 Total
Unit Shipments (000):							
CPU Board Sets	0.5	1.3	3.3	5.4	8.1	11.1	29.7
External Systems	0.4	5.6	12.5	17.0	21.0	18.0	74.5
Internal (IEG) Sys	TBD	TBD	TBD	TBD	TBD	TBD	TBD
<b>TOTAL</b>	<b>0.9</b>	<b>6.9</b>	<b>15.8</b>	<b>22.4</b>	<b>29.1</b>	<b>29.1</b>	<b>104.2</b>

Scorpio Net Revenues (\$M):							
Board Set	4.2	10.2	26.7	44.3	65.5	88.8	239.7
System	10.7	206.9	475.8	698.1	838.4	718.6	2948.5
Add-on	2.6	49.9	115.8	171.6	208.0	184.3	732.2
Layered Software	0.6	1.3	3.0	3.5	4.1	4.8	17.3
<b>TOTAL SCORPIO NES</b>	<b>18.1</b>	<b>268.3</b>	<b>621.3</b>	<b>917.5</b>	<b>1116.0</b>	<b>996.5</b>	<b>3937.7</b>
% Corporate NES*	0.3%	3.2%	5.9%	7.0%	6.8%	4.8%	5.7%
Service Revenues	1.0	19.7	63.7	110.5	233.0	263.5	691.4
<b>TOTAL SCORPIO NOR</b>	<b>19.1</b>	<b>288.0</b>	<b>685.0</b>	<b>1028.0</b>	<b>1349.0</b>	<b>1260.0</b>	<b>4629.1</b>
% Corporate NOR*	0.2%	2.4%	4.5%	5.3%	5.5%	4.0%	4.2%

BOARD SET % Unit Distrib. by Operating System:							
% VAX/VMS	20%	20%	20%	20%	20%	20%	20%
% Midnight	20%	20%	20%	20%	20%	20%	20%
% Seaboard	40%	40%	40%	40%	40%	40%	40%
% Non-DEC	20%	20%	20%	20%	20%	20%	20%
<b>TOTAL</b>	<b>100%</b>	<b>100%</b>	<b>100%</b>	<b>100%</b>	<b>100%</b>	<b>100%</b>	<b>100%</b>

\* Corporate NES & NOR beyond FY'86 are unofficial 32-Bit Program Office projections.

Scorpio (Cont'd)	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90	FY85-90 Total
SYSTEM % Unit Distribution by Operating System:							
% VAX/VMS	43%	60%	60%	63%	61%	61%	61%
% Midnight	37%	30%	27%	23%	23%	23%	24%
% Seaboard	20%	10%	13%	14%	16%	16%	15%
% Non-DEC*	0	0	0	0	0	0	0
TOTAL	100%	100%	100%	100%	100%	100%	100%

\* Every Scorpio system is sold with, at least, a "license-only" license.

### 3.1.2 Scorpio VAXstations Forecast (VAXstation Product Management Supplied)

Unit Shipments (000)	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90	FY85-90 Total
VAXstations	--	20.0*	35.0*	60.0	100.0	150.0	365.0

\* FY'86 and '87 VAXstation numbers represent the product strategy to migrate all VAXstations to Scorpio. However, actual Scorpio VAXstation based shipments may be less due to initial Scorpio manufacturing ramp-up.

## 3.2 DETAILED WORLD-WIDE FORECAST BY PRODUCT GROUP

### 3.2.1 Detailed Scorpio Boards Forecast (TVG-Micros Supplied)

Unit Shipments	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90	FY85-90 Total
CPU Board Sets	500	1,260	3,342	5,448	8,142	11,052	29,744
Geographic Mix % Shipments To:							
U.S.	60%	60%	60%	60%	60%	60%	60%
Europe	30%	30%	30%	30%	30%	30%	30%
GIA	10%	10%	10%	10%	10%	10%	10%

3.2.2 Detailed Scorpio Systems Forecast (Product Group Supplied)\*

Market/Product Group	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90	FY85-90 Total
Internal (IEG)	TBD	TBD	TBD	TBD	TBD	TBD	TBD
Commercial:							
CSI							
MDC							
PBI	10	50	75	75	65	50	325
TIG	55	1,000	2,000	2,620	2,000	—	7,675
Small Systems:							
COEM	250	800	1,200	2,000	3,000	2,000	9,250
Technical:							
ECS	65	150	300	400	500	625	2,040
ESG	150	300	520	900	1,600	2,140	5,610
GSG	228	800	1,120	1,500	1,500	1,500	6,648
LDP	—	210	550	700	700	400	2,560
MSG	50	150	200	250	250	250	1,150
Technical Volume:							
Micros	103	1,076	1,811	2,017	2,231	2,448	9,686
Systems	100	2,400					2,500
TOTAL EXTERNAL	1011	6936	7776	10,462	11,846	9,413	47,444
Geographic Mix** % Shipments To:							
U.S.	75%	55%	52%	50%	50%	50%	50%
Europe	25%	35%	35%	35%	35%	35%	35%
GIA	0%	10%	13%	15%	15%	15%	15%

\* Complete forecast data is not available from some Product Groups. Data provided may not include Europe due to recent reorganization.

\*\* Product management supplied.

### 3.2.3 Detailed Scorpio VAXstations Forecast (Product Group Supplied)\*

Unit Shipments	FY'85	FY'86	FY'87	FY'88	FY'89	FY'90	FY85-90 Total
Scorpio VAXstations	150	4,465	7,675	11,090	14,790	16,680	54,850

\* All product groups combined (data not available from CSI and MDC Product Groups). Forecast may not include Europe due to recent reorganization. Detailed breakdown available from Tom Sherman or Nasir Khan. Note: Some product groups have potential VAXstations, currently forecasted as Scorpio systems due to their uncertainty of the VAXstation product definition and its fit in their market.

### 3.3 IMPACTED PRODUCTS: PHASEOVER FORECAST SUMMARY

#### 3.3.1 World-Wide System Phaseover Forecast Summary

The primary products impacted by Scorpio are the VAX-11/730 and VAX-11/750. This data to be provided in a later revision of this document.

VAX-11/730	Actual			Forecast				
	FY'82	FY'83	FY'84	FY'85 FRS	FY'86	FY'87	FY'88	FY'89
Unit Shipments #								
NES	\$	\$	\$	\$	\$	\$	\$	\$
Service Revenues	\$	\$	\$	\$	\$	\$	\$	\$
NOR	\$	\$	\$	\$	\$	\$	\$	\$

VAX-11/750	Actual			Forecast				
	FY'82	FY'83	FY'84	FY'85 FRS	FY'86	FY'87	FY'88	FY'89
Unit Shipments #								
NES	\$	\$	\$	\$	\$	\$	\$	\$
Service Revenues	\$	\$	\$	\$	\$	\$	\$	\$
NOR	\$	\$	\$	\$	\$	\$	\$	\$



### 3.3.2 Impact of Not Bringing Scorpio to Market

Not bringing Scorpio to market has an adverse effect on all market segments for which Scorpio is intended. TVG-Micros would be even later to market with a 32-bit board set, relying on Micro-VAX as its initial and only product. The VAX-11/730 would bear the burden of satisfying for a longer period all low end and open office requirements necessitating cost reduction and repackaging efforts, or as an alternative, redirection of Micro-VAX into the systems space.

The VAX-11/750 would lose its ultimate "equal performance/less cost" replacement. Thus, pressure would be exerted on the Nautilus program to provide a less expensive version to satisfy this request. And finally, the BI development effort would be reunited with Nautilus. But since Nautilus no longer depends on the BI, this overdue replacement for the Unibus would lose focus and probably be further delayed.

## CHAPTER 4

### ASSUMPTIONS

#### 4.1 ENVIRONMENT

##### 4.1.1 Competition

Systems: Competitive assumptions will be provided in the next revision of this plan.

Board Sets: Major competitors are assumed as follows:

- National--The 16032 is expected to be the primary competition, due to its architectural features and anticipated availability date.
- Motorola--Although the 68000 series has become a well accepted architecture, the 68020 will not be available until after National's 16032. Thus, Motorola is assumed to be the number two competitor.
- Intel--Intel is expected to be the number three competitor. Although the 432 was the first 32-bit chip to market, its success has been less than anticipated. It now appears that this architecture is being placed "on the back burner" in favor of the 386, an 8080 compatible 32-bit upgrade.

##### 4.1.2 Geography

Scorpio will be sold in all countries served by Digital and its distributors. However, this plan is subject to U.S. State Department approval of Scorpio technologies being permitted into sensitive geographies such as Eastern Block countries.

##### 4.1.3 Regulatory Compliance

Scorpio products take no exception to DEC STD 060, and plan to be UL, CSA and FCC certified, and VDE compliant.

However, the current power supply design does take exception to the DEC STD 122 with regard to Japan and western Australia operating ranges, mainland Europe design range, and 20 amp service power factor.

##### 4.1.4 Economic Factors

Sensitivity of revenue forecast/projections to (1) fluctuations in U.S. and international economies, (2) price/performance trends in Digital's markets, and (3) price elasticity of demand cannot be defined at this time.

#### 4.1.5 Operating Environment

Scorpio board sets, power supplies, and the OEM box (excluding TU58 operation) all meet Class C requirements per DEC STD 062.

AZTEC, RA60 and RA81 disks all essentially meet Class B requirements (AZTEC—no exceptions; RA81—85% max humidity; RA60—80% max humidity). However, since the TU58 console load device meets none of the Class B humidity range, and only a portion of the Class B temperature range, Scorpio packaged systems must be classified as Class A.

The market requirements for Class B Scorpio packaged systems (non-magtape based) provide an additional incentive to eliminate the TU58 in favor of a Class B compatible floppy.

#### 4.2 TECHNOLOGY

##### 4.2.1 New Processes and Components

The following assumptions are made:

- The ZMOS process becomes mature and produces the necessary chips at cost goals, on schedule.
- The 50 mil center connector technology is available, reliable and meets specifications.
- NI lance chip available to specifications and on schedule.
- PCB and hybrid mounting technology can be manufactured cost effectively by Digital.
- Power supply hybrids available per specification and on schedule.

##### 4.2.2 Design Tools

The following tools are assumed to be in place:

- DECSIM—hierarchical logic simulator.
- CHAS—CAD tool data base manager.
- HILO—gate array test pattern fault grading.
- AUTODLY—timing delay simulator.
- MAPS or Project II—PERT and project management tool.

#### 4.2.3 Key Suppliers of Components

The following components are assumed to be available from their respective suppliers:

- TAT-020 Gate Arrays: Texas Instruments
- 256K MOS RAMS: The industry
- 300 Pin Connector: Burndy/AMP
- NI Chips: AMD and Mostek
- Power Supply Hybrids: Centralab, General Electric, and Siemens

#### 4.3 INTERNAL FUNCTIONS

The following assumptions highlight product requirements that cross functional and organizational lines.

##### 4.3.1 Engineering Dependencies

- VAX/VMS V4.0 with Scorpio support from BSSG.
- VAX/VMS to provide multiprocessor support for system engineering testing prior to system FRS.
- VAX/VMS to officially support MP as soon as possible.
- Seaboard V2.0 with Scorpio MP support from BSSG.
- AZTEC disks from Storage Systems.
- 2MB memory arrays from Storage Systems.
- Other Unibus peripheral support by combined BSSG/Systems Evaluation Lab efforts.
- Corporate standardization of LESI, NI, and SI buses.
- Battery back-up from 11/730.
- Acoustic packaging in H9640 series cabinets for RA60 and RA81.

##### 4.3.2 High-Volume Manufacturing Dependencies

- Engineering to supply BI and NI test equipment per specifications and delivery schedule.
- Engineering to design self test diagnostics to a 90% level for all BI modules.

- Engineering to supply micro and macro diagnostics that provide 100% test coverage.
- Advanced Technology Group (ATG) to provide design and qualification of all BI module hybrids as well as associated assembly and test processes.
- ATG to provide development and qualification of new BI connectors, and an inspection process for 128 pin chip carriers.
- ATG to provide development and qualification of a process for fine line/controlled impedance/multi-layer/H3D printed circuit boards as required.

#### 4.3.3 Final Assembly and Test Dependencies

- All system options to be site mergable.
- Mass storage cabinet assemblies to be site mergable.
- BI options to be customer installable.

#### 4.3.4 Customer Services Dependencies

- Customer installable systems that include appropriate system packaging design, documentation, and carton labeling.
- Optional Field Service installation available for a price.
- A UETP replacement, currently being defined in a new systems acceptance procedure.
- Product Lines to provide warranty funding:
  - End User--90 days on site.
  - OEM--30 days on site.
  - Boards--Return to exchange facility.
- Manufacturing to deliver a high quality product.
- Engineering to design reliable hardware with self test features (90% level) on each BI module.
- RAMP must be designed in.
- All system components to meet their individual MTBF (BMC) goals.
- Training courses implemented as defined.
- Manufacturing to supply the required CD kits and spares per Field Service plan.

- Manufacturing to provide repair facilities prior to MRC repair start-up.
- An "XCON" type process established to ensure all configurations are qualified.
- FCO rework to be implemented at MRCs.
- V-11 firmware updates to be distributed on console device load media.
- Revision control via machine readable registers for all BI and corporate interconnect components.

#### 4.3.5 Marketing

Development priorities, as defined by all Product Groups, are as follows:

1. Time to Market
2. Compatibility
3. Transfer Cost/MLP
4. Performance
5. Quality
6. Reliability/Availability

Product parameters of extreme importance to selected Product Groups include:

- Real time performance
- Office packaging
- Public BI bus

Advertising and promotion priorities will be defined in later revisions.

#### 4.3.6 Sales

Scorpio distribution channels are as follows:

<u>Market/Product Group</u>	<u>Distribution Channels</u>
Commercial:	
CSI	Direct end user, Data service companies
MDC	Direct end user
PBI	Direct end user, Indirect
TIG	Direct to BOCs, Western Electric
Small Systems:	
COEM	OEM/distributor, Industry hook

Technical:

ECS	Direct end user
ESG	Direct end user
GSG	Government end user, Government primes
LDP	Direct end user
MSG	Direct end user, Medical OEM

Technical Volume:

Micros	Direct sales, Industrial distribution
Systems	Direct sales, Distributors

4.3.7 Asset Management

The Stage 2 inventory goal is 8.5 weeks maximum. The overall inventory goal will be defined in future business plan revisions.

The Days Sales Outstanding (DSO) goals cannot be identified at this time.

4.4 CUSTOMERS

4.4.1 Impact on Key Accounts

To be provided in future revisions.

4.4.2 New Opportunities

To be provided in future revisions.

4.5 SCHEDULE

4.5.1 Project Schedule

A Scorpio program schedule will be included in future editions.

4.5.2 Phase Planners

Scorpio phase planners follow on the next three pages.

PHASE REVIEW PLANNER

22-June-82  
REVISION DATE

PROJECT NAME: SCORPIO

PROJECT TEAM

Product Manager: Tom Sherman  
Program Manager: Demetrios Lignos  
Manufacturing: Ken Meissner  
Customer Services: Doug Hanzlik  
Project Start Date (Actual): FY'81

<u>Phase</u>	<u>Title</u>	<u>Estimated/Actual Completion Date</u>
Phase 0	Strategy & Requirements	June 1982
Phase 1	Planning	Q2 FY'83
Phase 2	Implementation	Q4 FY'84
Phase 3	Qualification	Q4 FY'85
Phase 4A	Engineering, Manufacturing Transition	TBD
Phase 4B	Manufacturing Volume Production	TBD
Phase 5	Retirement	TBD

FIRST REVENUE SHIP:

Scorpio Board Sets	Q1 FY'85
BI Interface Chips	Q1 FY'85
Scorpio Systems	Q3 FY'85



PHASE REVIEW PLANNER

PHASE 0

STRATEGY AND REQUIREMENTS

PRODUCT NAME: SCORPIO

PRODUCT MANAGER: Tom Sherman      LOC: TW/B02      DTN: 247-2418

PROGRAM MANAGER: Demetrios Lignos      LOC: TW/B02      DTN: 247-2990

MILESTONE	RESPONSIBLE INDIVIDUAL	COMPLETION DATE		LATEST FORECAST
		ORIG PLAN	ACTUAL	
1. Business Plan (Preliminary)	T. Sherman			June 1982
2. Market Requirements Document	T. Sherman			June 1982
3. Product Requirements Document	T. Sherman			June 1982
4. Alternative/Feasibility Examined	M. Titelbaum			June 1982
5. Manufacturing Impact Statement	K. Meissner			June 1982
6. Customer Services Impact and Requirement Statement	M. Henderson			June 1982
Phase 0 Complete	T. Sherman			June 1982

Date: 22-June-82

Rev: 0

PHASE REVIEW PLANNER

PHASE 1

PLANNING

PRODUCT NAME: SCORPIO

PRODUCT MANAGER: Tom Sherman

LOC: TW/B02

DTN: 247-2418

PROGRAM MANAGER: Demetrios Lignos

LOC: TW/B02

DTN: 247-2990

MILESTONE	RESPONSIBLE INDIVIDUAL	COMPLETION DATE		LATEST FORECAST
		ORIG PLAN	ACTUAL	
1. Business Plan	T. Sherman	Q2 FY83		Q2 FY83
2. Functional Specification	B. Willard	Q2 FY83		Q2 FY83
3. Project Plan	O. Tasar	Q2 FY83		Q2 FY83
4. Manufacturing Plan	K. Meissner	Q2 FY83		Q2 FY83
5. Customer Services Plan (Preliminary)	D. Hanzlik	Q2 FY83		Q2 FY83
Phase 1 Complete	T. Sherman			Q3 FY83

Date: 22-June-82

Rev: 0

## CHAPTER 5

### FINANCIAL/SENSITIVITY ANALYSIS

#### 5.1 SUMMARY AND CONCLUSIONS

The objective of this system level financial analysis is to evaluate whether or not the Scorpio engineering product goals for prices, transfer cost, volume, hardware development, etc. will provide the corporation with a financially viable product. This analysis required numerous assumptions, due to it being early in the development cycle, and covers the period from development through the first five years of product shipments (FY'81-FY'90). FRS for Scorpio systems is at the end of Q3 FY'85.

Results of the analysis show the Scorpio combined system and board products yield 41% internal rate-of-return (IRR) and a net present value (NPV) of the annual discounted cash flows equaling \$1.69M at a 40% discount rate. Breakeven date is 14 quarters after the system FRS. These metrics are all lower than past 32-bit product analyses. Past system analyses indicate that IRRs exceed 50% at this phase of development.

The major assumptions are the average system and board set MLP (in FY'87 \$49.4K, \$12.5K respectively), the average system and board set transfer cost (in FY'87 \$12K, \$3.2K respectively) and shipment forecast of 74500 systems and 29744 boards in the FY'85-FY'90 timeframe. The shipments exclude VAXstation Scorpio sales and assume that a Microvax product impacts the total Scorpio board sales throughout the five year timeframe. Also, the base case assumes total BI and V-11 development costs with 100% of the Hudson facility and the equipment capital costs for the V-11 chip.

Sensitivities were performed around the major assumptions to determine why the returns are low compared to past 32-bit product analyses. These sensitivities show that even if half of the BI and V-11 chip development and capital costs are allocated away from Scorpio, the project returns only increase to 45% with \$9.5M NPV @ 40%. Analysis of the various sensitivities indicated that no one factor is responsible for the low project returns. Scorpio is targeted as a low end product with a low markup (about 4.0) but volumes are not commensurately high given the high level of development costs. A low end product should have high volumes to go with the low margins and anticipated high development costs. To improve the Scorpio product investment, the technology pioneered by Scorpio (both the BI and V-11) must be used in large volumes by other products within the timeframe of Scorpio.

The variables directly controllable during product development are time-to-market, hardware development costs, transfer costs, and inventory. Sensitivities around these variables show that if transfer cost for the V-11 chip set is increased an average of only \$300, the project falls below the corporate hurdle rate of 40%. If system hardware development costs are increased up to 5% in order to prevent a one month slip in FRS, it is a worthwhile tradeoff. Another worthwhile tradeoff

is a 10% increase in system hardware development costs to reduce the box transfer cost by 10% or more. These tradeoffs assume there are no diminishing returns for increased development expenses (i.e., one dollar more in development expense yields one dollar more in returns). Inventory changes significantly impact profitability. An inventory increase of four weeks from the 16 week base case causes the profitability to decline 200% to -\$1.6M.

## 5.2 FINANCIAL BASE CASE ANALYSIS - NUMERICAL SUMMARY\*

	<u>FY'81 - FY'90 (\$M)</u>
Total Net Operating Revenue (Equipment and Service)	\$4,629
Less: Cost of NOR	1,984
Gross Margin	2,645
Gross Margin as % of Total NOR	57%
Less: Development Expense	169
Selling, Marketing, G&A Expense	941
Profit Before Tax	1,535
PBT as % of Total NOR	33%
After Tax Net Cash Flow	831
After Tax NPV @ 40%	1.69
After Tax IRR	41%
Breakeven Date	Q1 FY'89 (July '88)
Number of Quarters to Breakeven from System FRS	14
Development Expense as % of Total NOR	3.7%

\* Analysis was performed using the Business Review Program (BURP).

## 5.3 FINANCIAL MODEL ASSUMPTIONS

The following major assumptions are used in the Scorpio BURP analysis:

1. Ship Forecast: The matrix below describes the ship forecast for Scorpio systems and Scorpio board sets from FY'85 to FY'90. The board forecast assumes the existence of a Micro-VAX product.

	FY85	FY86	FY87	FY88	FY89	FY90
Scorpio Systems	400	5600	12500	17000	21000	18000
Scorpio Board Sets*	500	1260	3342	5448	8142	11052
Total	900	6860	15842	22448	29142	29052

First Revenue Ship for Scorpio boards is (end of) Q1 FY'85, while FRS for systems is (end of) Q3 FY'85. The systems forecast is from Scorpio product management with some confirmation by product groups. The board set forecast is from TVG-Micros.

\* A Scorpio board set is an average configuration comprised of a CPU module, memory, disk and comm.

2. Transfer Cost: Scorpio product management provided the following system configurations and transfer cost goals for FY'87. The average board set configuration is forecast by TVG-Micros.

<u>System</u>	<u>FY'87 Transfer Cost Goal</u>
OEM Box (2MB)	\$ 4,949
Single AZTEC (2MB)	8,860
Dual AZTEC (2MB)	11,688
Dual RA60 (2MB)	16,877
RA81/TU81 (2MB)	20,977
Single AZTEC II (2MB)	8,360
Dual AZTEC II (2MB)	11,188
RDX/MAYA (2MB)	7,995
Dual RAXY (4MB)	15,887
RAXX/TA81 (4MB)	20,177
System Weighted Average	\$12,544

<u>Board Set Average Configuration</u>	<u>Qty</u>	<u>FY'87 Transfer Cost Goal</u>
CPU Module	1.0	\$1,000
.5MB Memory	0.35	625
2MB Memory	0.65	892
BI AZTEC (or future disk)	0.2	2,695
BI COMM	1.0	833
Board Set Weighted Average		\$3,171

3. MLP: System MLPs are proposed by Scorpio product management and board set MLPs are proposed by TVG-Micros. All system MLPs are held constant throughout their respective ship periods (refer to Section 2.1.1) but the board sets MLPs are lowered in the last two years for competition. All systems include VAX/VMS (license-only).

<u>System</u>	<u>MLP</u>	<u>Mark-up (FY'87)</u>
OEM Box	\$22,900	4.63
Single AZTEC	29,900	3.34
Dual AZTEC	41,900	3.58
Dual RA60	69,900	4.14
RA81/TU81	89,900	4.29
Single AZTEC II	31,900	3.82
Dual AZTEC II	44,900	4.01
RDX/MAYA	29,900	3.74
Dual RAXY	74,900	4.71
RAXX/TA81	94,900	4.70

<u>Board Set Average Configuration</u>	<u>Qty</u>	<u>MLP</u>	<u>Mark-up (FY'87)</u>
CPU Module	1.0	\$ 3,500	3.50
.5MB Memory	0.35	1,875	3.00
2MB Memory	0.65	2,676	3.00
BI AZTEC (or future disk)	0.2	10,700	3.97
BI COMM	1.0	2,500	3.00
Software			
VMS (license-only)	0.2	7,000	
Midnight (license-only)	0.2	2,500	
Seaboard Runtime (license-only)	0.4	100	
Weighted Average MLP		\$12,476	

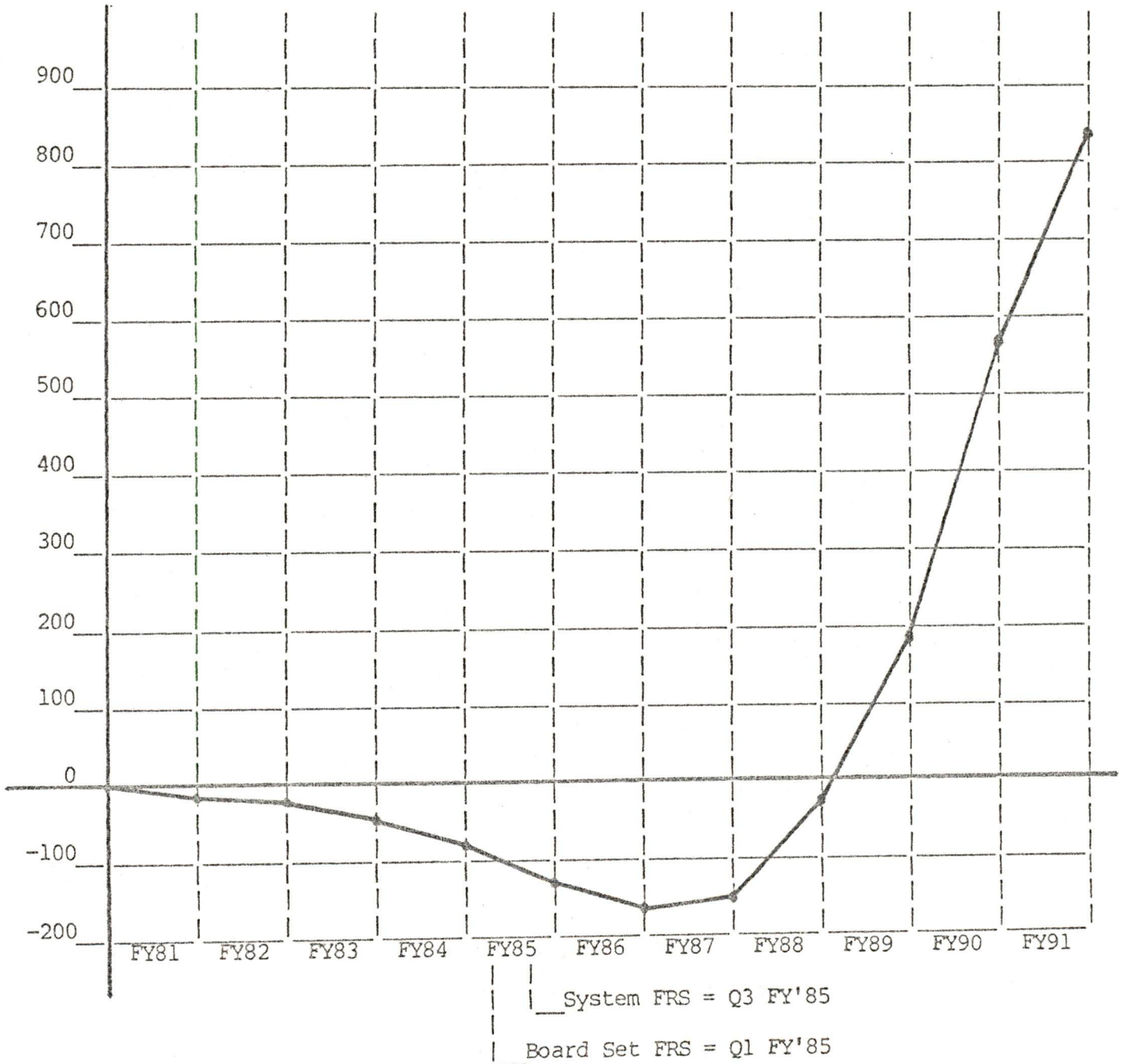
The average board set configuration is forecast by TVG-Micros. The MLPs are best estimates given that some products are just beginning development (e.g., BI COMM, Midnight).

Revenue for VAX/VMS installation and warranty is shown in the Pro Forma Financial Statement as Other Revenue. It is assumed that 40% of VAX/VMS shipments include the installation and warranty package. Other Revenue also includes revenue from the sale of Seaboard development software, a VAX/VMS layered product.

Add-ons, shipped with a system, are assumed to be 25% of system MLP with a 4.0 mark-up. The Board Set average configuration includes such add-ons.

5.4 CUMULATIVE CASH FLOW GRAPH

AFTER TAX  
 CUMULATIVE  
 NET CASH  
 FLOW (\$M)



5.5 BURP FINANCIAL STATEMENT

A copy of the Scorpio Pro Forma Financial Statement (from the BURP analysis) is provided on the next page.

Scorpio System & Board Financial Analysis  
Pro Forma Financial Statement

(\$M)	FY81	FY82	FY83	FY84	FY85	FY86	FY87	FY88	FY89	FY90	FY91	TOTAL
SHIPMENTS (UNITS)	0	0	0	0	900	6860	15842	22448	29142	29052		104244
EQUIPMENT NOR	0	0	0	0	17.5	267	618	914	1112	992		3920.5
FIELD SERVICE NOR	0	0	0	0	.1	7.3	26.2	57.4	97.4	137		325.4
S/W SUPPORT NOR	0	0	0	0	.1	2.4	14.7	20	94.9	89.3		221.4
OTHER NOR	0	0	0	0	1.4	11.3	25.7	36.6	44.8	42		161.8
TOTAL NOR	0	0	0	0	19.1	288	684.6	1028.0	1349.1	1260.3		4629.1
TRANSFER COST	0	0	0	0	6.4	90.1	198	279	341	305		1219.5
FS CONTRACT COST	0	0	0	.4	.6	8.1	22.2	42.3	68.7	94.5		236.8
SW SUPPORT COST	0	0	0	0	.1	1.8	10.9	14.8	68	63.9		159.5
FA&T COST	0	0	0	0	.4	8.0	18.5	27.4	33.3	28.7		116.3
FS WARRANTY COST	0	0	0	.7	1.7	6.5	14.3	20.4	26.5	21.4		91.5
SW WARRANTY COST	0	0	0	0	.6	8.6	21.1	33.1	43.6	42.5		149.5
MFG STARTUP COST	0	.7	2.4	4.5	1.0	1.2	.9	0	0	0		10.7
COST OF NOR	0	.7	2.4	5.6	10.8	124.3	285.9	417	581.1	556		1983.8
GROSS MARGIN	0	(.7)	(2.4)	(5.6)	8.3	163.7	398.7	611	768	704.3		2645.3
MARKETING EXPENSE	0	0	.1	1.2	.2	0	0	0	0	0		1.5
SALES EXPENSE	0	0	0	0	4.7	64.1	148	219	266	238		939.8
HARDWARE DEV EXP	3.5	15.5	23.8	35.1	29	11.8	7.9	0	0	0		126.6
SOFTWARE DEV EXP	0	.9	2.0	4.8	7.9	11.7	14.6	0	0	0		41.9
TOTAL EXPENSES	3.5	16.4	25.9	41.1	41.8	87.6	170.5	219	266	238		1109.8
PROFIT BEFORE TAX	(3.5)	(17.1)	(28.3)	(46.7)	(33.5)	76.1	228.2	392	502	466.3		1535.5
PBT % TO TOT NOR	0	0	0	0	(176)	26.4	33.2	38	37.1	36.9		33%
CUM PBT % TO TOT NOR	0	0	0	0	(676)	(17.3)	17.7	28.1	31.7	33.2		33%
LESS 46% TAX	(1.6)	(7.9)	(13)	(21.5)	(15.4)	35	105	180	230	214		704
PROFIT AFTER TAX	(1.9)	(9.2)	(15.3)	(25.2)	(18.1)	41.1	123.2	212	272	252.3		831
LESS CAPITAL	.1	3.7	26	9.8	5.7	3.7	1.5	.7	0	0		51.2
PLUS DEPRECIATION	.1	.5	4.5	8.4	8.2	7.0	6.2	4.4	2.4	1.4	.8	43.9
PLUS BUY BACK	0	0	0	0	0	0	0	0	0	0	7.5	7.5
A/R SHIFT	0	0	0	0	(3.9)	(55.4)	(81.7)	(70.5)	(66.1)	18.2	259	0
WIP SHIFT	0	0	0	0	(27.7)	(33.3)	(24.8)	(18.9)	10.9	93.9	0	0
ATNCF	(1.9)	(12.4)	(36.8)	(26.6)	(47.2)	(44.3)	21.4	126.3	219.2	365.8	267.3	831
ATDCF	(1.9)	(8.0)	(14.6)	(9.5)	(12.3)	(8.2)	2.8	11.8	14.7	17.6	9.3	1.69

AFTER TAX NET PRESENT VALUE @ 40% = \$1.69M  
 AFTER TAX INTERNAL RATE OF RETURN = 41%  
 BREAKEVEN DATE = Q1FY89 (14 QUARTERS AFTER FRS)  
 FRS = Q4FY85

ANALYSIS PERFORMED USING THE BUSINESS REVIEW PROGRAM



## 5.6 SENSITIVITY ANALYSIS

This sensitivity analysis was conducted around the major variables that will determine the financial performance of Scorpio systems and board sets. Also sensitivities were performed on variables which currently can be controlled during the product development. Use of information derived from these sensitivities can help identify opportunities to minimize product risks to Digital and maximize life cycle profitability.

Shipments: Life cycle profitability is less sensitive to volume changes (assuming all other variables are held constant) than changes in either transfer cost or price. Though shipments are less sensitive, less than a 10% decline in units drops the internal rate-of-return below the 40% corporate hurdle rate.

Price: Product profitability is most sensitive to variations in MLP. A 10% increase in price results in a 45% IRR or a \$12M increase in net present value (NPV).

Shipment/Price: The combination of both price and volume changes show that a 10% increase in price resulting in a 20% decline in volume still yields a level of profitability exceeding the base case. Conversely, if a price reduction increases shipments 20%, this reduction cannot be greater than 6% or the profitability of the product will fall below the 40% corporate hurdle rate.

Transfer Cost: Transfer cost is less sensitive than price, thus a 10% decrease in price would require nearly a 20% decline in transfer cost to maintain the same product profitability as the base case. The base case assumes a constant lifetime V-11 chip set cost of \$515. If the chip set cost increases an average of only \$300, the project falls below the corporate hurdle rate of 40%.

Hardware Development: The development costs assumed 100% of the BI and V-11 expenses. If half of the BI and half of the Hudson V-11 expenses are included, the IRR only increases to 45% with an NPV @ 40% of \$8.0M.

Hardware Development/Transfer Cost: The combination of these two factors show that a 10% increase in system hardware development expenses would require a 3% or greater reduction in total system transfer cost to maintain the profitability of the project. However, if the development is compared to the box, a 9.6% decline in the box transfer cost is needed to offset the 10% increase in system development costs.

Hardware Development/Time-To-Market: The comparison of these costs show a worthwhile tradeoff is a 5% increase in development expenses to move the FRS date in one month.

Inventory: Changes in inventory have a significant impact on profitability. An increase of 4 weeks inventory from the base case of 16 weeks drops the total product life profitability (NPV) to -\$1.6M (a 200% decline) and the IRR from 41% to 39% which is below the corporate hurdle rate.

Capital Expenditures: Changes in the level of capital expenditures have little impact on IRR but have a significant impact on total product life profitability (NPV). Dropping the Hudson building and equipment allocation from 100% to 18% raises the IRR from 41% to 43% while the NPV increases 279% (\$3.1M) to \$4.7M.

## CHAPTER 6

## REFERENCE DOCUMENTS

<u>Document</u>	<u>Rev</u>	<u>Date Published</u>	<u>Issued By</u>
<u>Low-End VAX Systems Development</u>			
Scorpio Product Requirements	1.0	23-Jun-82	Tom Sherman
Scorpio Market Requirements	1.0	23-Jun-82	Tom Sherman
Scorpio Alternatives/Feasibility Study			Mike Titelbaum
Scorpio Systems Functional Specification	0.85	11-Mar-82	Dileep Bhandarkar
Scorpio Systems Development Engineering Project Plan	0.2	01-Mar-82	Omur Tasar
BI Specificiation	F	12-Jan-82	Frank Bomba
BI Interface Chip Specification	1.2	Jan-82	Frank Bomba
Diagnostic Engineering Project Plan for Scorpio Systems	1.0	10-May-82	Ernie Preisig
Scorpio Program Pert	---	09-Jun-82	Corinne Thompson
Scorpio Financial Analysis--Phase 0	---	June 1982	Doug Kellogg
<u>Semiconductor Engineering Group</u>			
KDV11 CPU Module Specification	1.2	23-Dec-81	Bill Laprade
KDV11 Macro Programmer's Reference Manual	1.1	15-Jan-82	Bill Laprade
V11 IE Chip Specification	2.1	31-Mar-82	Ed Burdick
V11 M Chip Feasability	1.2	19-Dec-81	Bill Grundmann
V11 FBox User's Guide	1.3	08-Jan-82	Bill Brundmann
V11 Interchip Document	2.0	19-Apr-82	Stan Lackey
V11 Microcode User's Guide	1.1	03-Nov-81	Dick Sites
<u>Storage Systems</u>			
Scorpio Memory Array Business Plan	0	21-Jun-82	Dan Haley
SIPS vs DIPS For Scorpio Memory	0	15-Mar-82	Dan Haley
AZTEC (RC25) Business Plan, Phase 1	---	15-Mar-82	John Forde
RC25, AZTEC, Engineering Specification	5	09-Mar-82	Carl Blatchley
<u>Salem Volume Manufacturing</u>			
Scorpio Manufacturing Impact Statement	1.0	23-Jun-82	Ken Meissner
<u>Customer Services Support Engineering</u>			
Customer Services Scorpio Impact and Requirements Document	1.3	25-Jun-82	Mike Henderson

CHAPTER 7

BUSINESS PLAN HISTORY/CHANGES

Dev Phase	Rev Date	Rev No	System FRS	Phase Completion Dates						FY'85-FY'90 External Unit Ships (000); Boards/Systems	Gross Margin, % NOR (%)	IRR (%)	NPV @ 40% (SM)	Total Dev Budget-TW, HL (SM)	
				0	1	2	3	4	5						
0	6/22/82	0.7	03'85	6/82	02'83	04'84	04'85	TBD	TBD	TBD	29.7 / 74.5	57%	41%	1.69	55.0

Meeting between Gordon, Bill H., and Craig A

Agenda

- 0. Our Goal: A factor of 25 improvement in design capability
- 1. Advanced Development Plan

Gordon's Comments on the Plan

Why we chose the FPP chip

Any questions on bubblechart?

Comments on port concept for information flow

- 2. Structured chip design
  - a way of doing design; not a CAD tool
  - supported by some computer programs

- 3. Advanced Development Concept within OOD

→  $\mu$ VAX

Small Systems ←

→ MSD (Tewksbury)

L.C.G.

nMOS process development ←

- 4. DEC's long-term commitment to wafer fab.

CM

12/5/79

WHAT IS STRUCTURED CHIP DESIGN?

WE DON'T KNOW YET

BUT

(a) As FPP chip gets fleshed out, we will understand it (April timeframe)

(b) We believe each of the following techniques will be used

1. Regularity

- minimize the number of drawn transistors
- by (a) discipline
- (b) use of topologically regular functions

2. Wiring first, then transistors

3. Disciplined chip floor planning -  
    and follow through by step-wise refinement

4. Rigorous intra-chip interface management

5. Chip assembler

6. Hierarchical verification

7. One page of design rules

8. Designers who span architecture through layout

9. Library of parameterized circuits layout

10. Colored sticks notation

11. Design for testability

12. Think structural ( $\equiv$ Geometric), not logical

CM  
12/5/79

NEXT UPDATE OF ADVANCED DEVELOPMENT PLAN

(IN CONCRETE ON 12/1/79)

TABLE OF CONTENTS:

- RATIONALE / GOALS
- A STRUCTURE FOR INFORMATION FLOW
- PREDICTABLE COMMUNICATION FLOW --- THE DATES
- OUR FOCUS
- { BUBBLECHART
- { DELIVERABLES
- CAD SUITE --- DEFINITION BY EXCLUSION
- WRITTEN REPORTS AVAILABLE
- THE TEAM

CRAIG MUDGE  
SCORPIO ADVANCED DEVELOPMENT MANAGER  
11/13/79

NEXT UPDATE OF ADVANCED DEVELOPMENT PLAN  
=====

(TO BE FINALIZED AS 12/1/79 Update )

To be presented at Staff Meetings Tues 11/13/79

I. Introduction

The VLSI Advanced Development Group is growing a core or kernel (of MicroVAX and its design methodology) which will shape product development and manufacturing in FY81 and beyond.

As product development progresses the machine will be engineered by  
CSD/LSI  
MPD MOS engineering (circuits and logic)  
MPD LSI CAD  
Small Systems  
MOS Process engineering

and manufactured by  
Hudson Semi Plant  
a board plant  
a system plant

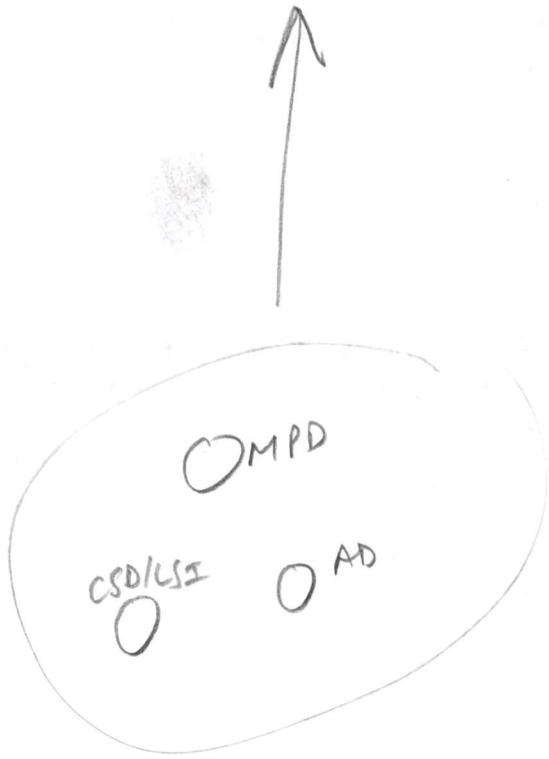
with support from the Advanced Development Group.

-----  
The REQUIREMENTS for MicroVAX and its design methodology are currently seen to be:

- MOS Process: 2-2.5 micron; additional low resist poly layer (polycide)
- Design time: 33 months from Q281 (see Bill J's data)
- Design capability for 75000 transistor chips
- Chip set of 4 to 8 chips
- Runs VMS
- Performance: 70% of VAX-11/780
- Design for Testability
- New bus (BI)

-----  
This update of the AD Plan gives

- 1. a structure for information flow
- 2. ports through which status is reviewed and info flows
- 3. latest bubble chart
- 4. list of deliverables and non-deliverables





The structure has the following goals:

- a. communicate outwards both intent and results of the AD group
- b. permit the status of the the AD group to be measured periodically
- c. update the AD group of relevant advances, decisions, etc. in the other engineering groups (CSD/LSI, Chips, Small Systems, MOS Process Dev) who are doing vital pre-product development work
- d. ensure a smooth transition of the MicroVAX program from Advanced Development mode to Product Development and Manufacturing modes.
- e. provide the necessary isolation/freedom that the AD group needs in order to achieve the revolutionary steps expected of it.

Over the last couple of months, we have all participated in a set of program-level Resource Planning Meetings guided by Steve Teicher. This intense activity has led to a top-level Work Breakdown Structure and soon to a top-level network.

This period has generated data, not only for post-FY80 planning, but also ideas and data for this next update of the AD Plan. In particular, it has enabled me to make this update more specific.

Moreover, a lot of momentum has been generated in the various engineering groups; this made it clear that I needed to develop a more formal structure to focus and nourish advanced development.

The structure, with its ports, reporting (in and out) mechanisms, and bubble chart, is given in the pages that follow and is being debussed by

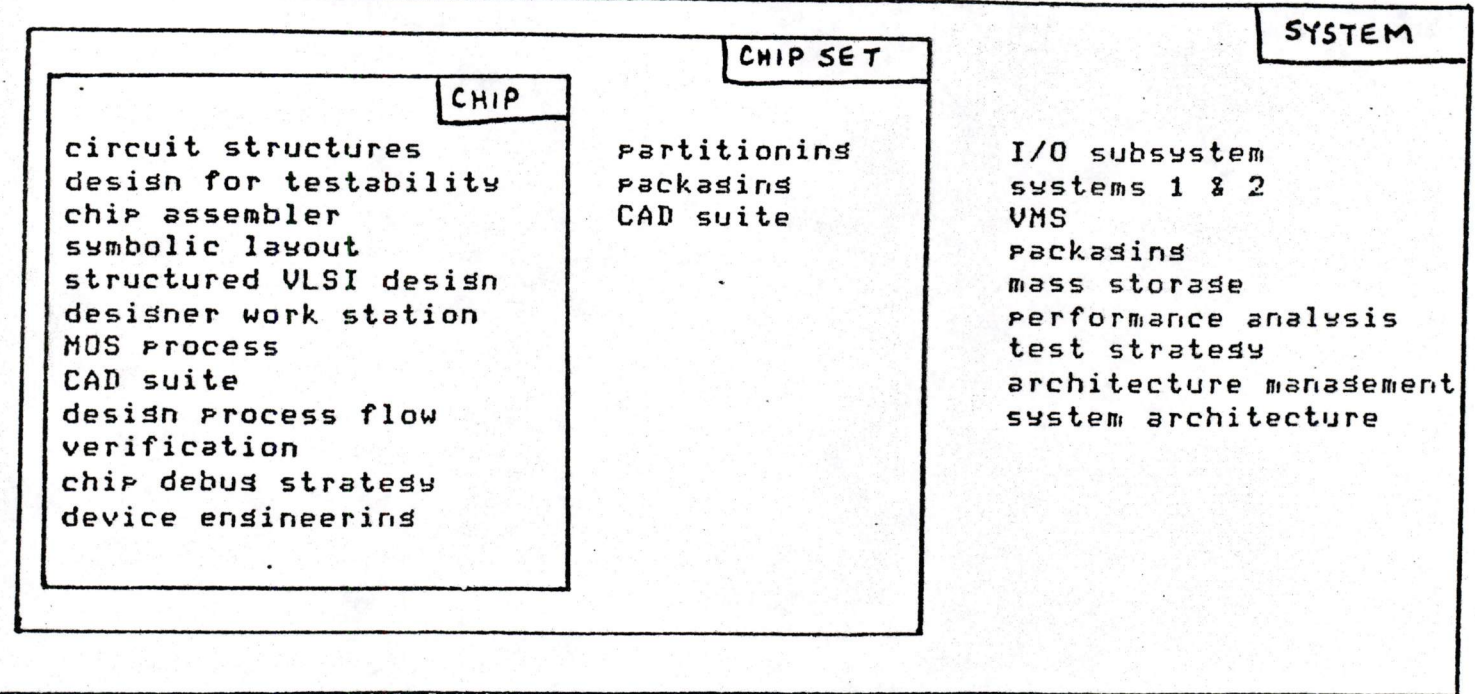
Duane Dickhut, Dave Morgan, Val Patel, Bill Johnson, Andy Wu, and Russ Moore with guidance from Steve Teicher and Ron Cadieux and buy-in from the AD team.

Crais Hudge

11/12/79

The first step in developing a structure was to list all the technical components of MicroVax:

TECHNICAL COMPONENTS OF MICROVAX



This diagram of the technical components of MicroVAX has been redrawn to give us

A STRUCTURE FOR INFORMATION FLOW

This structure, shown on the following page, shows:

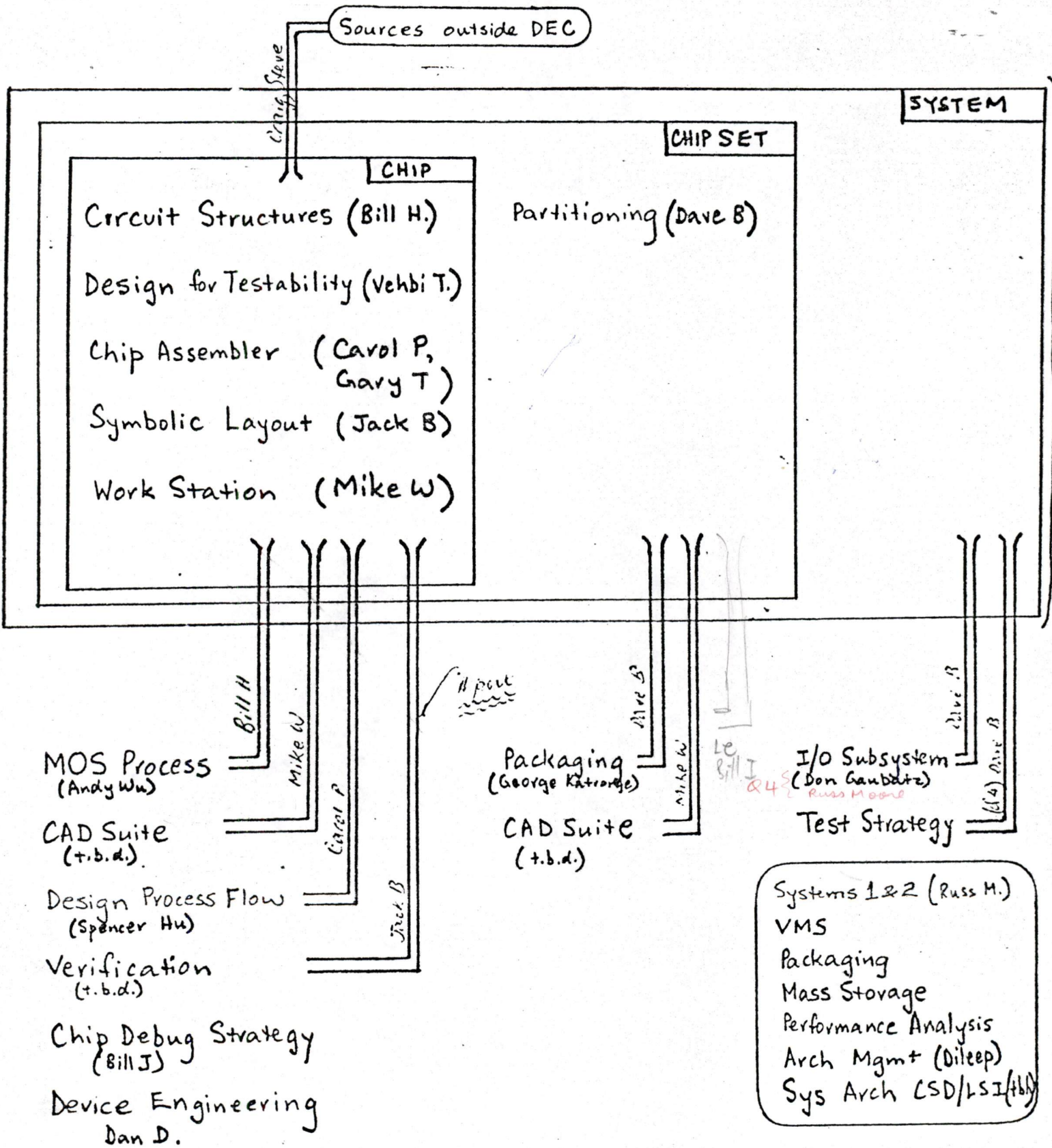
- \* a split between the Advanced Development Group on ML4-3 and other pre-product development efforts
- \* ports through which status is reviewed and information flows bi-directionally
- \* some postponements of interactions until FY81

NOTES:

1. Attached to each port is the name of a person, the port manager, who is responsible for facilitating the interface. Formal status will be communicated outwards on the dates given (in a subsequent page).
2. The ports are between those activities which, in our engineering's judgement, will need interaction. There is no port, for example, to Dileep's architecture management because we expect no changes (e.g., the addition of a new datatype) to the VAX SRM in FY80. If, however, the situation changes, we would immediately establish a port and a named port manager. Note that all reports, plans, etc., will be available to those groups, whether there is a port or not.
3. Tom Gunter will be joining us from the Motorola M68000 project and will be making a significant contribution. Although he won't be starting until the New Year, he plans a visit to DEC in the next few weeks. At that time, we will establish his role more specifically.

Please review this structure against the goals given on page 2.

# A STRUCTURE FOR INFORMATION FLOW



CM  
11/13/79

III. PREDICTABLE COMMUNICATION FLOW - THE DATES

1. Via ports

Outwards: every 3 months by a technical status report written  
by port managers

1/1/80      4/1/80      6/30/80

Inwards: briefings received (of decisions and advances made)

2/1      4/1      6/1

2. Updates of Adv Dev Plan

12/1/79      3/1/80

3. Reports at each milestone given on AD bubble chart

4. Miscellaneous

CAD symposium (January)

next MicroVAX workshop

AD plan briefings to management, e.g., to staff meetings of Zeh, Clayton,  
Demmer, Cudmore, OOD

SOME COMMENTS ON COORDINATION OUTSIDE OF THE STRUCTURE SHOWN

I have not shown any communication lines between the various activities going on outside of ML4-3.

It is my understanding that, in FY80, Duane and Dave (among others) are driving items that are not included in advanced development. For example,

Duane

=====

- chip packaging
- chip debug strategy
- systems architecture

Dave

=====

- MOS process
- design process flow
- device engineering

## FOCUS OF SCORPIO ADVANCED DEVELOPMENT

MOS PROCESS ←

CHIP ASSEMBLER

→ FPP CHIP

CHIP-SET PARTITIONING

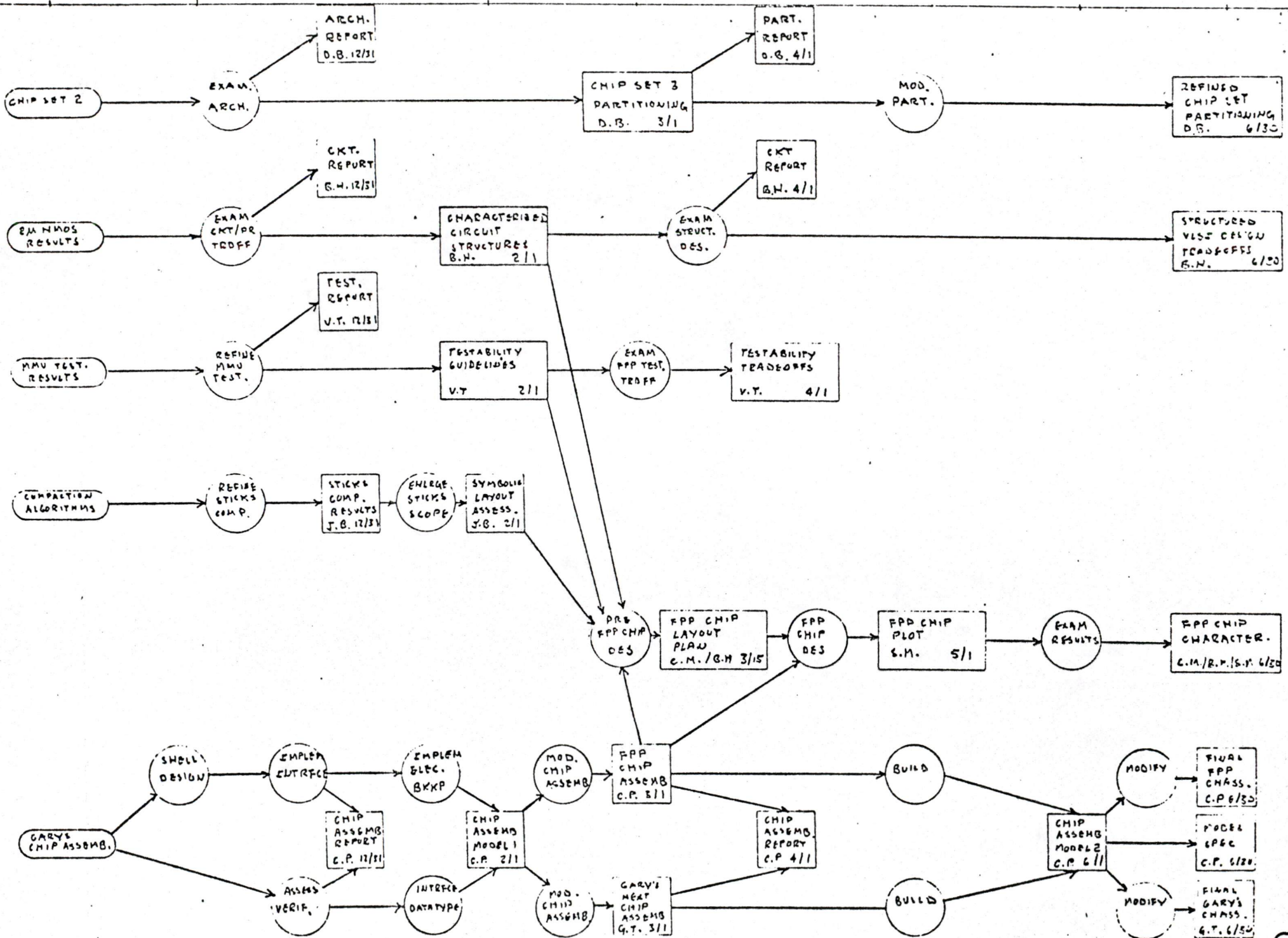
- o THE KEYSTONES ON WHICH PRODUCT DEVELOPMENT DEPENDS
- o BUBBLE CHART SHOWS STEPS, MILESTONES
- o FPP CHIP DRIVES MOS PROCESS, PROVES CHIP ASSEMBLER, AND REFINES PARTITIONING
- o RESOURCES (LAYOUT, PROGRAMMING, CIRCUITS) WILL BE REARRANGED AND ADDED TO FPP CHIP IF IT STARTS SLIPPING

CM  
11/13/79



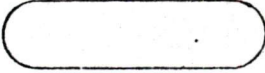

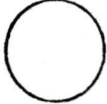
33 mths 1





11/11/79

## Workflow "Rules"

-  = inputs to SCORPIO effort
-  = tangible outputs or deliverables
-  = activities
- deliverables are the only "output" points to other "rows"
- deliverables can receive inputs only from their own "row"
- deliverables (except status reports) must be separated by activities
- activities can receive inputs from any deliverable or from other activities in the same "row"
- a "row" represents generally a refinement of activity from left to right
- only deliverables are associated with time (a milestone)

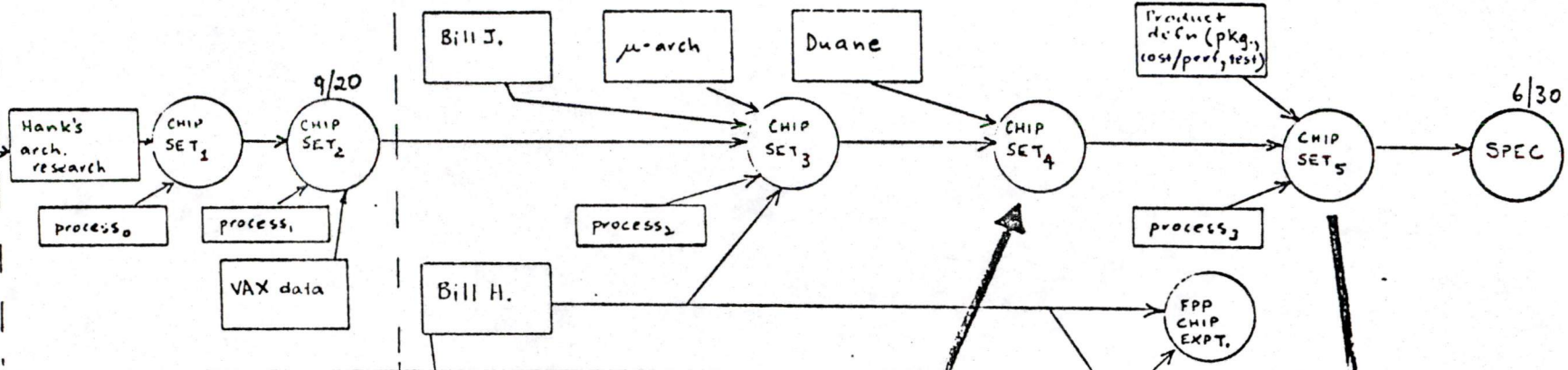
WORK FLOW & MILESTONES  
MICRO-VAX ADVANCED DEVELOPMENT

9/20/79  
VERSION

(FY79) ← SUMMER → | ← POST SUMMER (FY80) →

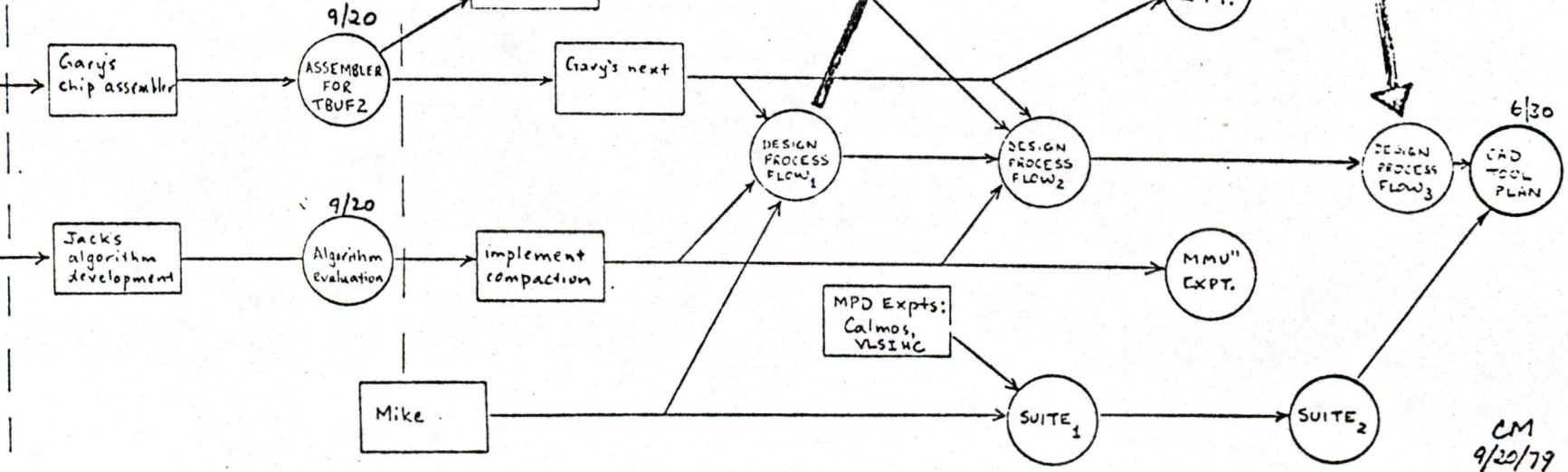
ARCHITECTURE

CHIP SET PARTITIONING RESEARCH (Craib)



CAD TOOL PLAN

STRUCTURED CHIP DESIGN RESEARCH (Craib)



CM  
9/20/79

V. DELIVERABLES

13

[list taken off bubble chart]

- 12/31 sticks compaction results (Jack B)
- 12/31 chip assembler report (Carol P)
- 2/1 2um circuit structures (Bill H)
- 2/1 testability guidelines report (Vehbi T)
- 3/1 chip set spec 3 (Dave B)
- 3/15 FPP chip layout plan (Bill H, Crais M)
- 5/1 FPP chip plot (ALL)
- 6/1 chip assembler model (Carol P, Gary T)

The most significant (in terms of getting product development) are:

- 5/1 FPP chip plot
- 6/1 chip assembler model

-----

In addition to the above, there are the scheduled reports from port managers.

DEFINITION BY EXCLUSION

We have found it useful, in defining our deliverables, to list the total CAD suite and show those things we are not working on.

CAD SUITE - the tools for microVAX (high-density hand-crafted MOS)

- \* chip assembler
- \* symbolic layout
- \* path analysis program
- circuit simulation
- logic simulation (SAGE3)
- DRC
- IV
- graphics input system
- geometric editor
- PG tape generation
- microcode assembler
- VOTE
- microcode simulator
- logical-physical verification
- host CAD computer (& network)

We are responsible for the items marked \* which is about 10% of the CAD suite.

In particular, we do not bear the following on our shoulders:

- a. the other 90%
- b. fitting the whole lot together
- c. a gate array design process
- d. DEC's future, e.g., the next 10 years, in CAD
- e. an integrated data base

In the CAD area, we are testing out concepts for MicroVAX and delivering a chip assembler (early FY81). If parts of our effort on symbolic layout are useful, then they will also be delivered.

WRITTEN REPORTS AVAILABLE

These are available from Del Thorndike (ML4-3) on 8-223-7667

- Hank Walker, Scorpio Chip Set Architecture Design Tradeoffs, 8/23/79
- Hank Walker, Design Methodology, 8/24/79
- Larry Seiler, A Pascal Machine Architecture implemented in Bristleblocks, a prototype silicon compiler, 7/3/79
- Gary Tarolli, Summer report on Chip Assembler for TRUF2, 10/23/79
- Everybody, MicroVAX workshop #1: Summer Results, 9/20/79 (viewgraphs)
- Charles Minter, Charles Terminal care package, 7/79
- Bill Herrick et al, HMOS J-11 Technical Feasibility, 8/79
- Crais Mudger, Chip Design using a Chip Assembler, 11/79
- Crais Mudger, On Circuit-to-pin Ratios, 11/79
  
- Ed McGrath, A Physical Design Rule Description, Caltech #3236, 10/29/79
- Irene Buchanan and John Gray, A model for structural integrated circuit design, Caltech #3230, 10/79
- Andy Wu, The 2um NMOS process, 11/79
- Mike Waters, CAD Suite (Rev 0), 11/79

-----

We will have a formal Advanced Development library of reports, books, etc., on ML4-3 by 1/15/79.

THE ADVANCED DEVELOPMENT TEAM ON ML4-3

The top-level WBS shown on the following page gives the overall MicroVAX structure. The team on 4-3 fits under Advanced Development and consists of the following:

- Jave Bernstein (chip-set partitioning)
- Jack Burness (symbolic layout - Sticks)
- Tom Gunter (x%)
- Bill Herrick (circuit structures)
- Crais Mudge (structured chip design)
- Carol Peters (chip assembler)
- Jary Tarolli (chip assembler)
- Jehbi Tassar (40%) (design for testability)
- Mike Waters (designer work station)
- (technician) to be hired
- (programmer)

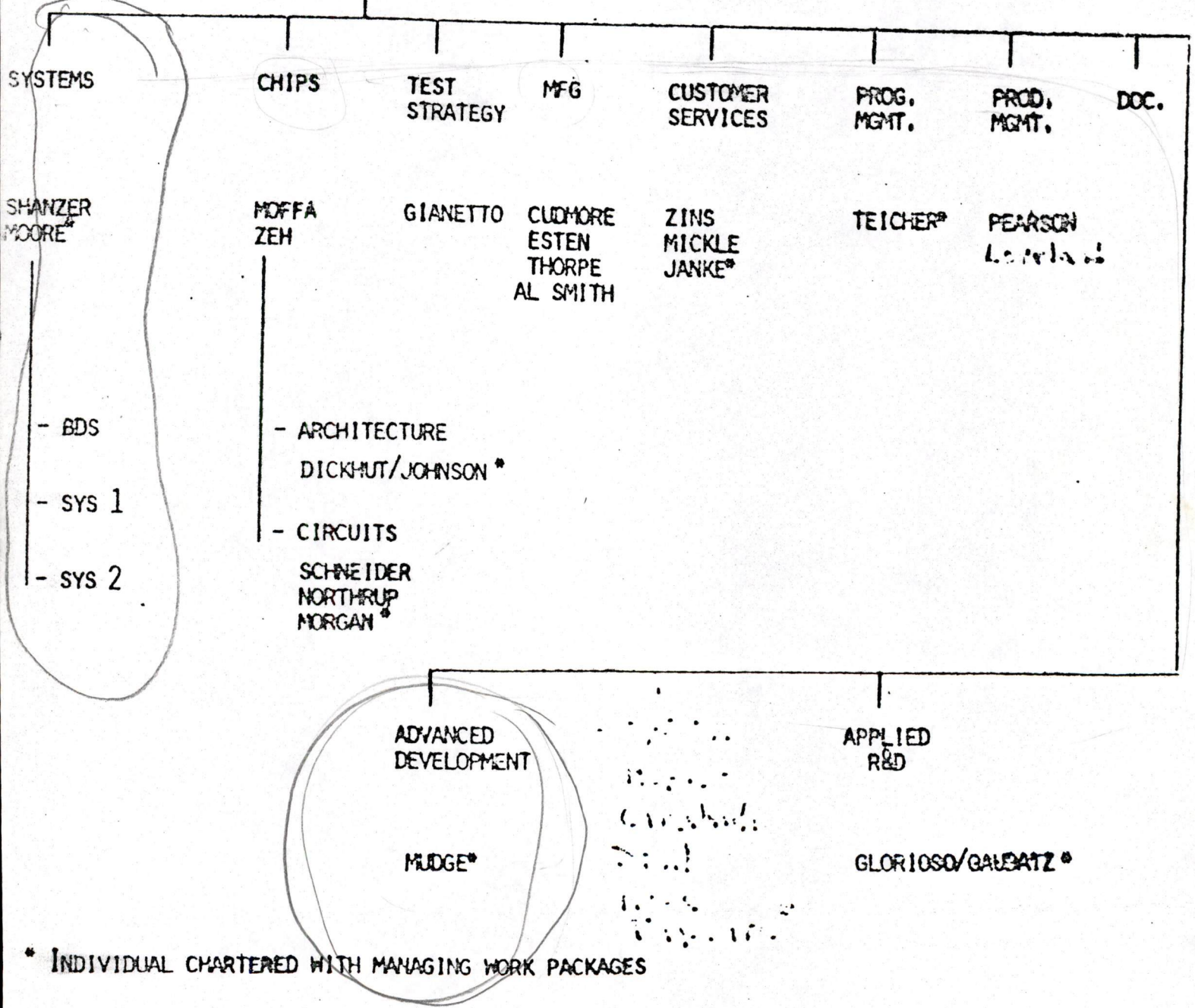
My expectation for my own time allocation through the end of FY80 is that I will be 50% management and 50% technical (FPF chip and structured VLSI design).

Crais Mudge, Scorpio Advanced Development Manager  
11/13/79

TOP LEVEL WBS

SCORPIO

TEICHER

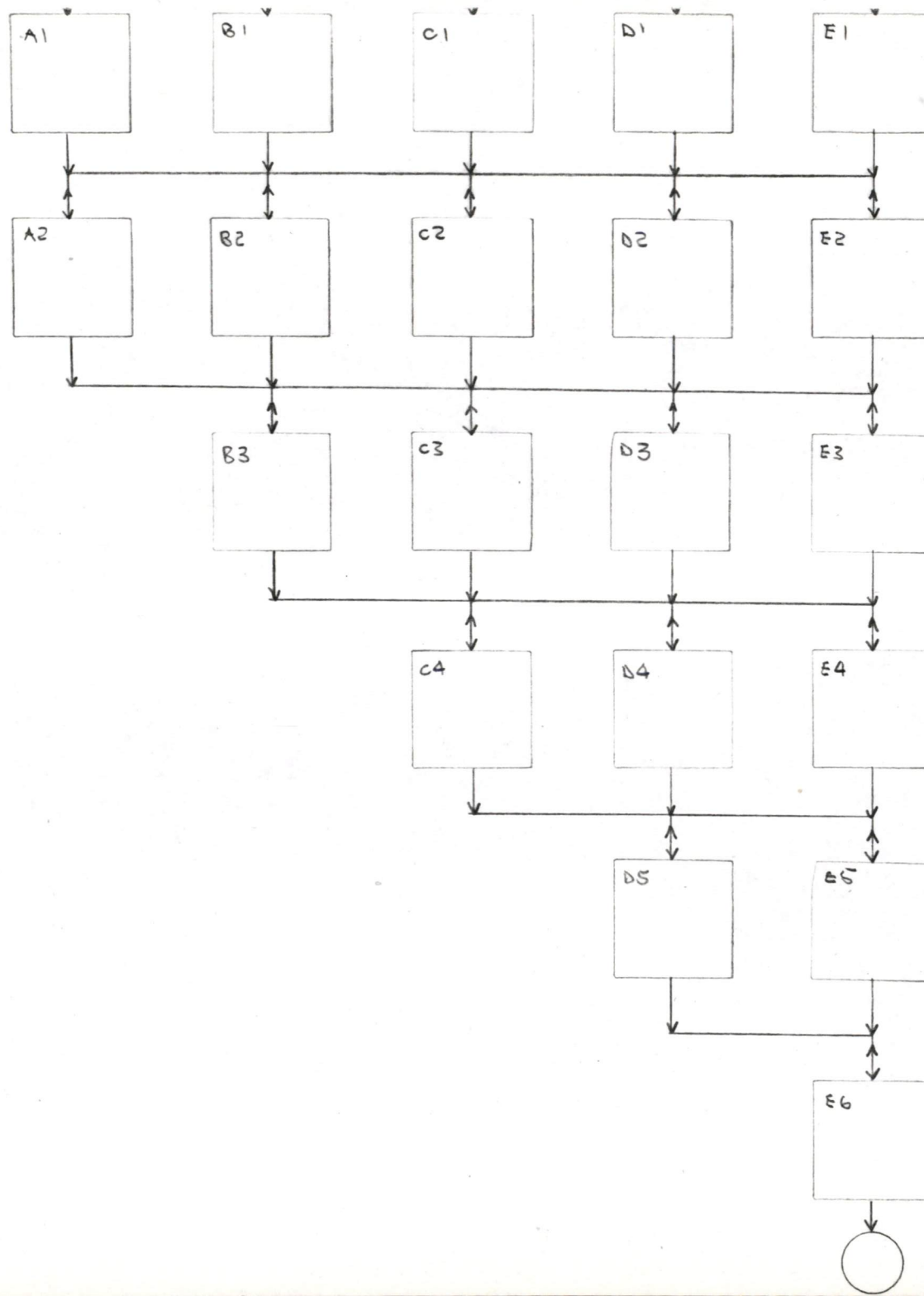


\* INDIVIDUAL CHARTERED WITH MANAGING WORK PACKAGES






Design State \ Discipline	System Definition	System Implementation	Logic Implementation	Circuit Implementation	Layout Implementation
Advanced Development	(A1) • Preliminary Product Goals	(B1) • Architectural Alternatives	(C1) • Logic Structure Alternatives	(D1) • Gross Process Parameters and • Gross Package Requirements	(E1) • Gross Process Design Rules Requirements
Product Definition	(A2) • Product Goals	(B2) • System Partition • Preliminary System Spec and Timing	(C2) • Usable Functional Elements	(D2) • Speed / Power Estimates	(E2) • Layout Density Estimates
System Design		(B3) • System Spec and Timing	(C3) • Preliminary Chip Specs, Eqs. and Timing	(D3) • Preliminary Key Circuits • Timing Feasibility	(E3) • Preliminary Layout Plans
Logic Design			(C4) • Chip Specs, Eqs. and Timing	(D4) • Unsized Circuit Schematics	(E4) • Layout Plans
Circuit Design				(D5) • Sized Circuit Schematics	(E5) • Major Section Layouts
Layout Design	Design Process Flow for Systems containing Custom LSI / VLSI				(E6) • Data Base Tape
	Bul Herude 12/5/79				



## The VLSI Design Process - Product Design Involving Custom VLSI

- Columns indicate the various design disciplines or skills involved. Moving "down" a column is essentially a refinement process - one of adding more detail, and not violating assumptions made "above" without invoking "expensive" rework
- Rows indicate the various design states or gross states of the product development. Moving to the "right" invokes activities in the various disciplines supporting the primary objective of that state (ie the "left" most item in the row)
- Main Diagonal  indicates the primary objective of the given design state
- Each Box indicates the primary output or major activity of that matrix location. These boxes must be expanded in detail to be useful for our understanding and for identifying CAD tool support.

12

• Detailed Boxes will include inputs, outputs, and the necessary transformations between them. In general, boxes may get inputs from their own state, the previous state, or a parallel development effort (ie process development). outputs may go to their own state, the next state or a parallel process. Costly consequences occur when outputs invoke changes in any previous state!

• Design Verification is a validity comparison between a given box and any box to its left in the same state, any box in the previous state, or a parallel process.

• Testability is considered an overlay to this design process. It invokes a bias in one's thinking about the details of the design implementation.

• Other Observations - All tasks in a given row are horizontally iterated or essentially completed simultaneously. Any box interacts with the previous, the next, or its own row only.

1/1

1/1

2/1

3/1

4/1

5/1

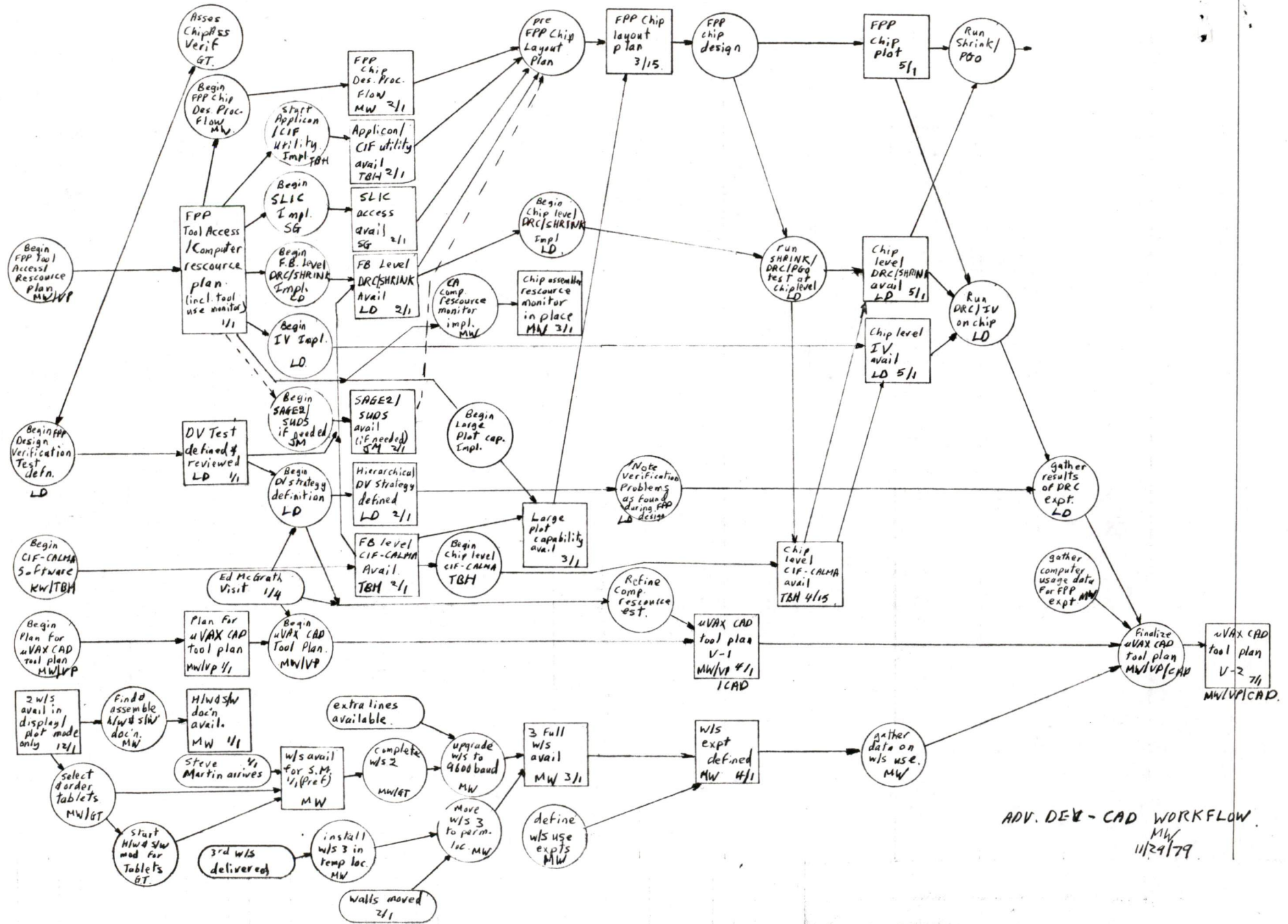
6/1

FPP Chip Design Plan.

mini CAD Suite For FPP expt.

uVAX CAD Tool plan

Workstation support



ADV. DEV - CAD WORKFLOW  
MW  
11/24/79

## Deliverables.

- 1/1 FPP CAD tool plan / ~~com~~FPP computer resource plan. (MW/VP)
- 1/1 Design Verification test plan (~~to~~ Len D)
- 1/1 Plan for  $\mu$ VAX tool plan &  $\mu$ VAX computer resource plan (MW/VP)
- 1/1-? Interactive color workstations (MW)
  - 2/1 Access to functional block ~~ti~~ ~~sm~~ level <sup>utilities</sup> tools<sub>n</sub> (LO, JM, TBH)
  - 2/1 Access to SLIC (SG)
  - 2/1 Hierarchical D.V. strategy (LO)
  - 3/1 Capability for large (Xynetics sized) plots (TBH)
  - 3/1 Monitor for chip assembler computer resource usage (MW)
  - 3/1 3 CALTECH workstations available for use (MW)
  - 4/1  $\mu$ VAX CAD tool plan V. 1. (MW/VP, CAD)
  - 4/1 Workstation experiments defined (MW)
  - 5/1 Chip level DRC/IV/P60 available (LO)
  - 7/1  $\mu$ VAX CAD tool plan V. 2. (MW/VP/CAD)

SEP 20 1979 42



# INTEROFFICE MEMORANDUM

TO: See Distribution

DATE: September 13, 1979  
FROM: Steve Teicher *stu*  
DEPT: Engineering  
EXT: 3175  
LOC/MAIL STOP: ML4-3/T34

SUBJECT: Work Breakdown For Scorpio Program

The attached work breakdown chart for the Scorpio Program should serve as a starting focal point for planning. I have listed as major tasks those pieces or chunks of the program that I believe will be the collection points for information flow among our managers and technical contributors. These major tasks also represent the way we are likely to allocate resources, i.e., budget. The attached diagram does represent the organizational reporting structure.

My expectation is that the "owners" of each of the major tasks will complete the lower levels of the work breakdown chart in sufficient detail that they will be able to:

1. Understand the tasks that they "own"
2. Acquire the resources to accomplish their portion of the program
3. Share with others, what they expect to deliver, when they expect to deliver, and how much budget versus time that they require

I understand the "TASK OWNERS" to be at this time as follows:

	<u>GROUPS</u>	<u>PEOPLE</u>
Chips	Small Systems/MPD (Moffa)	Dickhut/Johnson/Northrup
Boards	Small Systems (Shanzer)	
System 1	" " "	?
System 2	" " "	?
Program Management Support	Scorpio	Teicher/Cadieux/Shnitzler
Product Management	CSD Product Management	Loveland
Manufacturing	?	?
Test Strategy	Mfg. Engineering - Acton	Ed Gianetto
Documentation Architecture	Scorpio	
Customer Services	?	?



Advanced Development

Design Methods  
Architecture  
Applications

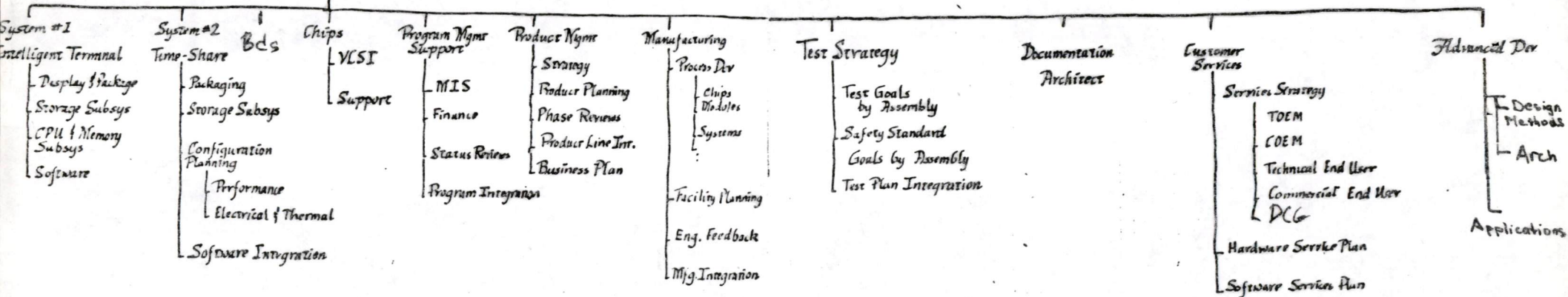
Scorpio  
Scorpio  
R&D

Mudge  
Mudge  
Glorioso

Note: Just a reminder that the Resource Planning Meeting will be held on Tuesday, September 18, 1979, at the Holiday Inn in Marlboro from 8:30 - 1:00.

cad

Scorpio



Rev 9.11-79 SH  
13

Distribution:

Keith Amundsen	ML3-3/E67
Jack Burness	ML4-3/T34
Ron Cadieux	ML12-2/E71
Duane Dickhut	ML1-2/E65
Don Gaubatz	ML3-2/E41
Ed Gianetto	AC/E48
Bob Glorioso	ML3-2/E41
John Groark	AC/E36
Bill Johnson	ML4-3/T34
Ray Mercier	ML1-2/H26
Craig Mudge	ML4-3/T34
Tom Northrup	WZ2
Jack Schneider	WZ2
Bert Shnitzler	ML12-2/E71
Steve Teicher	ML4-3/T34
Mike Waters	ML4-3/T34

cc:

Gordon Bell	ML12-1/A51
Dick Clayton	ML12-2/E71
Bill Herrick	ML4-3/T34
Dick Loveland	ML12-2/E38
Roy Moffa	ML1-2/H26
Russ Moore	ML1-2/E60
Carol Peters	TW/D08
Herb Shanzer	ML1-2/E60
Gary Tarolli	ML4-3/T34
Hank Walker	ML4-3/T34
Ted Webber	ML1-2/T29
Joe Zeh	WZ2

DISTRIBUTION:

Resource Planning Meeting Attendees:

Steve Teicher	ML4-3/T34
Craig Mudge	"
Jack Burness	"
Tom Northrup	WZ
Duane Dickhut	ML1-2/E65
Bill Johnson	ML4-3/T34
Bob Glorioso	ML3-2/E41
Jack Schneider	WZ2
Herb Shanzer	ML1-2/E60
Ron Cadieux	ML12-2/E71
Bert Shnitzler	ML12-2/E71
Ron Mercier	ML1-2/H26
Don Gaubatz	ML3-2/E41
Ed Gianetto	AC/E48
John Groark	AC/E36
Mike Waters	ML4-3/T34

cc:

Joe Zeh	WZ2
Dick Clayton	ML12-2/E71
Roy Moffa	ML1-2/H26
Ted Webber	ML1-2/T29
Dick Loveland	ML12-2/E38
Russ Moore	TW/C03

+-----+  
|digital|  
+-----+

INTEROFFICE MEMORANDUM

TO: See Distribution

DATE: 19 September 1979  
FROM: Del Thorndike  
DEPT: Project Scorpio  
EXT: 223-7667  
LOC: ML4-3/T34 *Del*

SUBJECT: Material From Resource Planning Meetings

Enclosed you will find all the material presented at the Resource Planning Meetings on September 18, 1979, Holiday Inn, Marlboro.

The people who presented ideas/thoughts were in order:

Steve Teicher (Project Scorpio)	Introduction
Ron Cadieux (CSD Staff)	Program Management Support
Crais Mudse (Project Scorpio)	MicroVAX Advanced Development FY80
Don Gaubatz (R&D)	Microarchitecture and Microcomputer R&D
Bill Johnson (Small Systems)	M/VAX Chip Development
Tom Northrup (Micro Products)	Resource Requirements
Ed Gianetto	Overall Test Strategist Roles
John Groark	Manufacturing Engineering

ACTION ITEMS:

1. CSD Review the week of October 15th
  - Status
  - Budget
  - People
  - Roush Plan

Preparation, October 8, 8:15AM - 2PM, Holiday Inn, Marlboro.

2. Tom Northrup, Bill Johnson, and Crais Mudse need to discuss a shared set of assumptions.
3. Systems group estimates need to be done. Keith Amundsen will take this message back and cause some action to be taken.
4. Bert Shnitzler to work on budget roll up and strategy.

I hope you find this information helpful. Please let me know if I can help you.

# GROUND RULES...

*Allow Presentations to Complete*

*Make NOTES on Issues*

## Expectations

*-Our Act is Just Beginning*

*-Holes and Overlaps Exist*

*-Groups do not "TRUST"*

*-Charters are UNCLEAR*

*-History is mixed*

SO WHAT'S NEW ...



# Expectations

*-Each Group will Display*

*-Their Own Task Breakdown*

*-Data on Interfaces*

*-Data on Resources*

*-In Own Areas*

*-In Other Groups*

GROUND RULES...

*List Presentations*

*Establish Time Limits*

ISSUES AND RESPONSIBILITIES

MICROVAX      ADVANCED      DEVELOPMENT

SUMMER      RESULTS

CHIP SET ARCHITECTURE  
- PARTITIONING

Hank Walker

CHIP ASSEMBLER FOR TBUF CHIP  
Gary Tavolli

STICKS COMPACTION ALGORITHMS  
Jack Burness

INSTALLATION OF CALTECH SOFTWARE  
- UTILITIES & COLOR GRAPHICS  
Gary Tavolli

FLOOR PLANS FOR STRUCTURED CHIP DESIGN  
Craig Mudge

[WORKSHOP #1      Sept 20, 1979]

CM  
9/18/79

MICRO VAX      ADVANCED      DEVELOPMENT

RESOURCE      PLANNING      FY 80

JUNE 30, 1980      OUTPUT:

1.    CHIP SET      SPEC
2.    CAD TOOL      PLAN

**WE      NEED**

SOFTWARE      CHIP      DESIGNER

GARY TAROLLI

SOFTWARE      CHIP      DESIGNER

CAROL PETERS

MOS ARCHITECT

CHIP SET      ARCHITECT

DICKHUT / JOHNSON

CAD      INNOVATOR

JACK BURNES

CAD TOOL      PLANNER

MIKE WATERS

TECHNICIAN

MOS      CIRCUIT      DESIGNER

↓ BILL HERRICK

20      FACILITIES      PROGRAMMER

CM  
9/18/79

# MICROVAX ADVANCED DEVELOPMENT

## FY80 EXPERIMENTS

### MMU'

MEASURES: AREA; SPEED; POWER;  
DESIGN TIME; YIELD

### ONE CHIP FROM CHIP SET SPEC (REV B)

MEASURES: as above but softer  
(extrapolate from 68000, 8086)

### DESIGN PROCESS FLOW

### TESTABILITY

CM  
9/18/79

PROGRAM MANAGEMENT  
SUPPORT

FINANCIAL

- BUDGETS
- MONTHLY REPORT OF ACTUALS
- COST PERF. ASSESSMENTS
- CONTRACTS
- FINANCIAL FORECASTING
- MIS DEVELOPMENT
- ADMINISTRATION

PLANNING

- OVERALL PROGRAM PLAN
- RESOURCE PLANNING
- COST ESTIMATING
- BEIGE BOOK INPUTS

SCHEDULE

- MASTER NETWORK
- FORMATS FOR LOWER LEVEL SCHEDULES
- PERIODIC SCHEDULE PERF. ASSESMENTS

PROGRAM INTEGRATION

- PERIODIC STATUS REVIEWS
- PROGRAM PROGRESS REPORTS
- YELLOW BOOK
- CSD STATUS REVIEWS
- ASST TO PRODUCT MGR FOR PHASE REVIEWS
- MAINT. OF MASTER CALENDAR

WBS DICTIONARY

LEVEL

2 PROGRAM MANAGEMENT SUPPORT

Provide the necessary services to assist in the planning, integration and control of the overall program. Co-ordinate activities of all development and support organizations.

3 FINANCE

Provides for financial support to the Scorpio Program

4 BUDGETS

Develop overall program budget and periodic updates - Record budgets on Engineering and Corporate systems.

4 MONTHLY REPORT OF ACTUALS

Provide a monthly report of actuals vs. budget and a variance report which highlights "Hot Spots".

4 COST PERFORMANCE ASSESSMENTS

Provide a periodic assessment of actual program progress vs. cost incurred to date ( integrates cost and schedule )

4 CONTRACTS

Assist in negotiation of outside contracts for support and prepare actual documents. Oversee the allocation of resources under the agreed to commitments of the contract.

4 FINANCIAL FORECASTING

Provide periodic financial forecasts which highlight trends in program spending and which forecast future spending based on past performance.

4 MIS DEVELOPMENT

Develop automated management Information systems to assist in the accomplishment of program management support objectives.

4 ADMINISTRATION

Track manpower requisitions and capital requisitions -- insure proper dissemination of financial information.

### 3 PLANNING

Provide for the coordination of all program related planning activities

#### 4 OVERALL PROGRAM PLAN

Develop a top level program plan which includes all reporting and supporting relationships, key development objectives, resource requirements and ~~market interrelationships~~, detailed program goals (cost/Technical/Schedule) - Feeds directly to Product Business Plan.

#### 4 RESOURCE PLANNING

Provide for the coordination and development of detailed resource plans for funds, manpower and facilities (Engineering, Manufacturing, and Customer Services)

#### 4 COST ESTIMATING

Provides for the development of detailed cost estimates for development, manufacturing, transfer cost, field services, etc. -Actual estimates are prepared by responsible agencies i.e. Engineering, Manufacturing, etc. This activity provides a coordinating function to assure uniformity of assumptions, estimating factors and format.

#### 4 BEIGE BOOK INPUTS

Provide for the collection of information and actual development of Beige Book inputs for entire program.

### 3 SCHEDULE

Provide for the development and maintenance of the Program Schedule and for the periodic assessment of performance against that schedule

#### 4 MASTER NETWORK

Develop a top level program network composed of all major program milestones. -Acts as a top level guide for the development of lower level schedules and networks.

#### 4 FORMATS FOR LOWER LEVEL SCHEDULES/NETWORKS

Develop a uniform set of symbols and techniques to allow the integrated development of lower level schedules/networks.

#### 4 PERIODIC SCHEDULE PERFORMANCE ASSESSMENTS

Prepare periodic assessments of how the overall program is doing vs. the planned schedule, highlight problem areas and develop potential work-around's.

### 3 PROGRAM INTEGRATION

Activities included here provide for the overall integration of all



activities associated with the development of Scorpio and for reporting the status of the program to all supporting organizations and to upper ~~level~~ management levels.

4 PERIODIC STATUS REVIEWS

Arrange for and manage periodic Program Status Reviews. Develop uniform and constant formats which accurately depict performance against plans.

4 PROGRAM PROGRESS REPORTS

Prepare periodic Program Progress reports for submission to CSD and corporate management <sup>level</sup> levels. Coordinate inputs from the development and supporting organizations.

4 YELLOW BOOK

Prepare monthly and Quarterly Yellow Book inputs .

4 CSD STATUS REVIEWS

Prepare status reviews for presentation to the CSD staff by the program manager.

4 ASSISTANCE TO PRODUCT MANAGER FOR PHASE REVIEWS

Provide assistance to the Product Manager in the preparation for and administration of Phase Reviews.

4 MAINTENANCE OF MASTER CALENDAR

Develop and maintain a master program calendar which lists and coordinates all key program meetings and reviews.

**digital**

R&D GROUP

MICROARCHITECTURE & MICROCOMPUTERS

PROGRAM SUMMARY

### PDP-11 GOALS

- CREATE PROCESSOR-INDEPENDENT (F-11, T-11, J-11, LSI-11) SOFTWARE-COMPATIBLE HIGH-PERFORMANCE LOW END SYSTEM ARCHITECTURE WHICH IS APPLICABLE TO
  - PDT SERIES
  - DESK-TOP SYSTEMS
  - APPLICATION-BASED SYSTEMS
  - PROFESSION ORIENTED SYSTEMS
- OPTIMIZE CONTRIBUTION TO SYSTEM FUNCTIONALITY AND PERFORMANCE MADE BY DIGITAL'S LSI PROCESSORS
- EXPLOIT INDUSTRY-STANDARD LSI CHIPS

## MICROVAX GOALS

- CREATE AN APPROACH TO LOW END (MICRO) VAX I/O WHICH WILL LAST FOR AT LEAST 10 YEARS AND MAKE OPTIMUM USE OF
  - VAX I/O STANDARDS, UNIBUS, BI,  
(AND BOUNDED)
  - INDUSTRY STANDARD LSI PERIPHERAL CHIPS
  - PDP-11 LSI CPUs AS I/O PROCESSORS
  
- USE MICROVAX I/O RESEARCH AS INPUT TO MICROVAX CHIP SET SYSTEM INTERFACE DESIGN

## LOW END SYSTEM ARCHITECTURE FOR PDP-11 LSI CPUs

- EMULATION-BASED
- SINGLE PROCESSOR  
(CPU DOES I/O EMULATION)
- PROCESSOR INDEPENDENT  
LSI-11, F-11, J-11  
MICROCODE EMULATION  
  
T-11  
MACHINECODE EMULATION
- SUPPORTS NATIVE AND EMULATED MODE  
PERIPHERALS

## F-11

- NONMASKABLE MICROCONTROL INTERRUPT
- MICROCODE DEVELOPMENT SYSTEM  
9 QUADS, 5 QUADS, OR 1 DUAL

## T-11

- REPOSITIONED HALT LINE PRIORITY NMI
- LSI EMULATING T-11

## PSC-11 (PERIPHERAL SUPPORT CIRCUIT)

- COLLECTS RANDOM I/O LOGIC
- CONTAINS EMULATION SUPPORT

**digital**

R&D GROUP

MICROARCHITECTURE & MICROCOMPUTERS

PROGRAM RESOURCES

GAUBATZ

- PROGRAM LEADER

TURCOTTE

- PROGRAM SUPPORT TECH (WCS)

PICCO

- EX-COOP (PSC-11)

NEW HIRE

- TREDENNICK REPLACEMENT  
(MICROVAX I/O)

**digital**

R&D GROUP

MICROARCHITECTURE & MICROCOMPUTERS

PROGRAM SUPPORT REQUIREMENTS

PERSON

- VAX SOFTWARE I/O

PERSON

- BACKSIDE OF THE BI

**digital**

R&D GROUP

MICROARCHITECTURE & MICROCOMPUTERS

PROJECT LOADING

INTEGRATED SYSTEM ARCHITECTURES

Q1, Q2 PDP-11

Q3, Q4 MICROVAX

MICROVAX I/O

NEW HIRE(S)

Q3, Q4 OTHERS AS AVAILABLE

COST-REDUCED PDP-11 SYSTEMS

Q3, Q4 FINISHED

T-11, PHASE TWO

Q2, Q3 FINISHED

F-11/J-11 MICROCONTROLLER

Q3, Q4 FINISHED

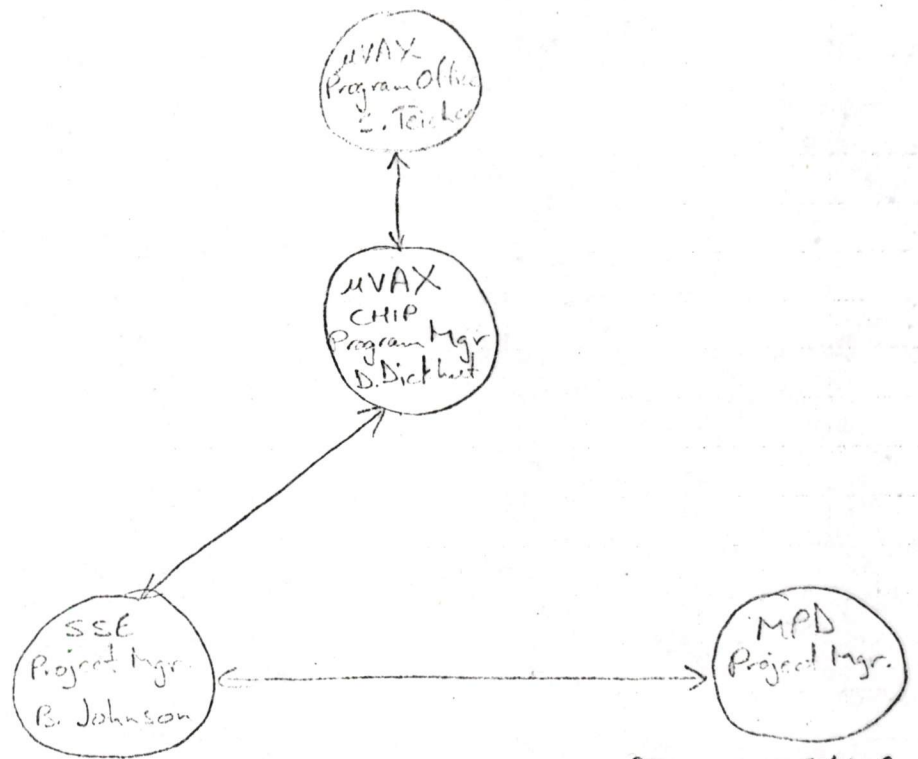
F-11 WCS/WLA

Q2 F-11 FINISHED

Q4 J-11 FINISHED



# μVAX CHIP DEVELOPMENT ORG.



## PROJECT TEAMS

- Arch. & Logic Design
- RTL/Gate Simulation
- Microcode Dev.
- Engineering Tester
- Diagnostic Eng.
- Systems Interface



## PROJECT TEAMS

- Circuit Design & Simulation
- Layout
- Product Eng.
- Test Eng.
- Process Eng.
- CAD support

KPI: Organize by function, not by chip ⇒ project team leaders need only to understand their function and in which which of roles need for globally

# ASSUMPTIONS

- 4 CUSTOM VLSI DESIGNS
- IMPLEMENT WITH NEW PROCESS
- MAXIMUM UTILIZATION OF IN-HOUSE RESOURCES
- PRODUCT DEVELOPMENT STARTS Q2 FY81

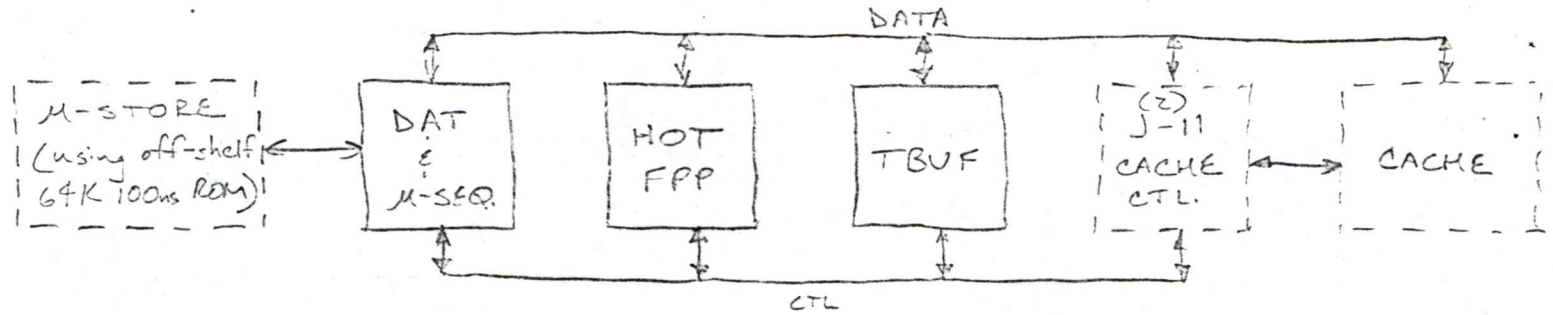
- \*  • CAD tools in place
- \*  • Preliminary process characteristics (+/- 20%)
- Requisite design resources
- chip set spec + layout plan
- System spec

- DESIGN EFFORT/CHIP WILL REMAIN A CONSTANT
  - Empirical Evidence
  - Learning curve offsets efficiency of new design tools and methodology

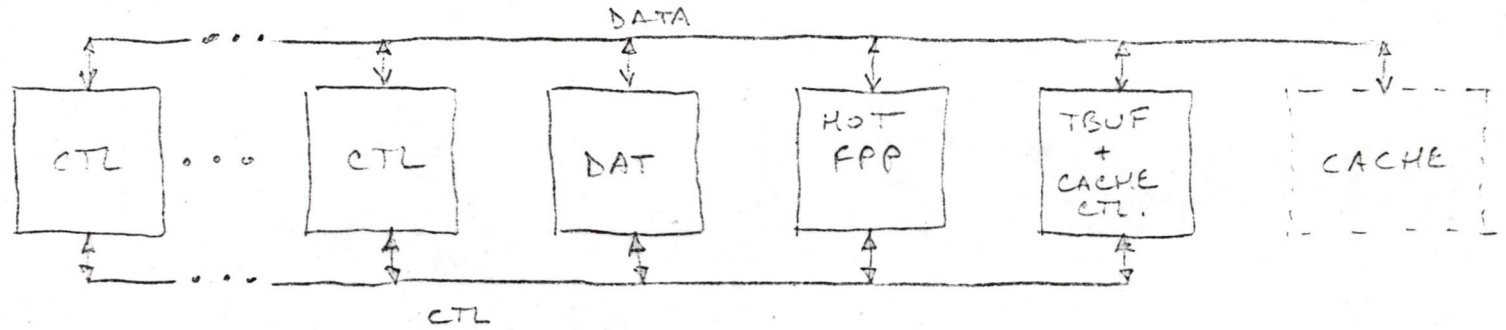
- 3 PASSES TO SHIPPABLE PARTS
- NO VMS REDESIGN OR SUBSETTING
- SOFTWARE BREADBOARDS ONLY FOR LOGIC DESIGN
- MINIMIZED TRAINING

# μ-VAX IMPLEMENTATION ALTERNATIVES

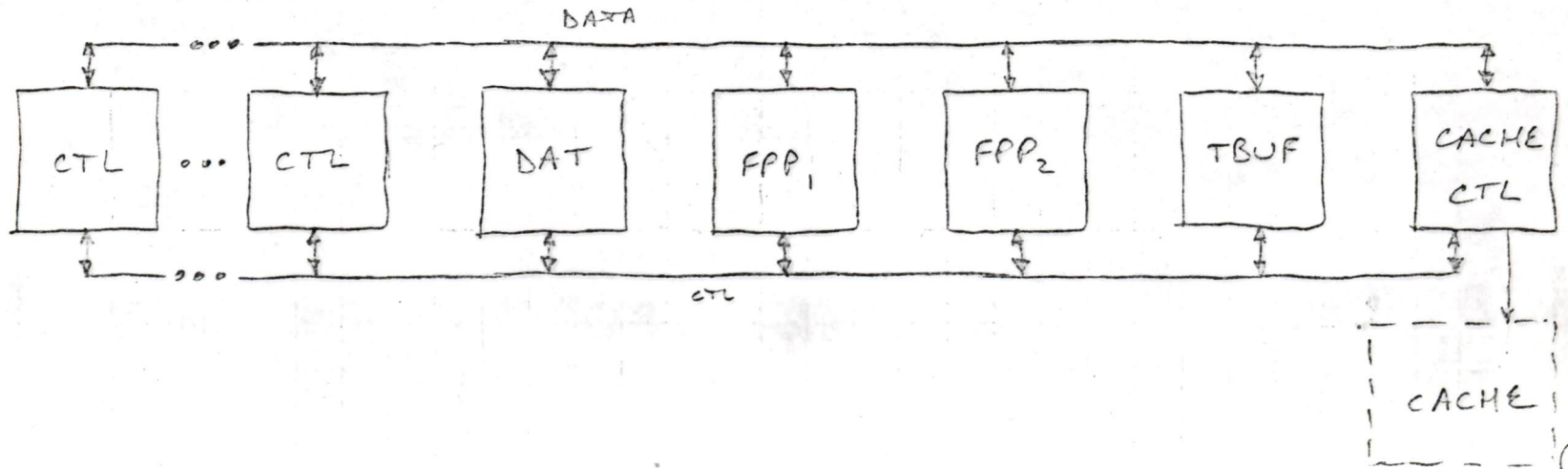
2-3 CUSTOM CHIPS



3-4 CUSTOM CHIPS



4-6 CUSTOM CHIPS



TASK FORCE OBJECTIVES

- I. BY Q4 F80 PRELIMINARY DESIGN RULES AND PROCESS PARAMETERS AVAILABLE TO START NEW PRODUCT CONCEPTUAL DESIGN.
- II. BY Q2 F82 CONFIRMED DESIGN RULES AND PROCESS PARAMETERS.
- III. BY Q4 F83 NEW PROCESS DELIVERING REASONABLE YIELDS FOR FIRST REAL PRODUCT CHIP.

# DESIGN TIME METRICS (man-months/clip) TO 1<sup>st</sup> PASS

	<u>LSI-II</u>	<u>F-II</u>	<u>J-II</u>
ARCHITECTURE & LOGIC DESIGN	7	8	9
CIRCUIT DESIGN	5	5	} 30
LAYOUT	12	15	
DIGITIZING & CHECKING ?		9	

# SSE LSI DESIGN RESOURCES (BY PROJECT TEAM)

- |   |     |
|---|-----|
| <p>① Architecture and Logic Design</p> <ul style="list-style-type: none"> <li>1 Chip set Architect/Project leader</li> <li>1 Architect-Logic Designer/chip</li> <li>2 Jr. Logic Designers</li> <li>2 Techs (documentation, data entry, etc.)</li> </ul>                                 | 7+2 |
| <p>② Microcode Development</p> <ul style="list-style-type: none"> <li>1 Senior microprogrammer/Project leader</li> <li>4 Base instruction set microprogrammers</li> <li>1 FPP microprogrammer</li> <li>2 Techs. (data entry, etc.)</li> </ul>   | 6+2 |
| <p>③ RTL/Gate simulation</p> <ul style="list-style-type: none"> <li>1 Project leader</li> <li>1-5 Gatelevel simulation engr./chip</li> <li>2 RTL simulation eng</li> </ul>  | 9   |
| <p>④ Diagnostic Engineering</p> <ul style="list-style-type: none"> <li>1 Test strategist/Project leader</li> <li>1 Diagnostic engineer/chip</li> <li>1 Systems diagnostics engin</li> </ul>   | 6   |
| <p>⑤ Engineering Tester</p> <ul style="list-style-type: none"> <li>1 Senior design engin./Project leader</li> <li>1 Junior design engin.</li> <li>1 Programmer</li> <li>2 Technicians</li> </ul>  | 3+2 |
| <p>⑥ Systems interface</p> <ul style="list-style-type: none"> <li>* 1 System Architect/Project leader</li> <li>* 1 VMS consultant</li> <li>* 1 CPU designer</li> <li>* 1 Substrate designer/packaging engr.</li> <li>1 Global Applications interface</li> <li>* 2 Technician</li> </ul> | 5+2 |

TOTAL 36 + 8

\* Eventually managed by SSE Systems Group

# MPD RESOURCES (BY PROJECT TEAM)

① Circuit Design and Simulation 9 + 4

- 1 Senior Designer / Project Leader
- 2 Circuit Designers / chip (S/P)
- 1 Tech / chip (simulation; data entry)

② Layout 9 + 4

- 1 Senior Layout Designer / Project Leader
- 2 Layout Designers / chip
- 1 Tech / chip (data entry; editing)

③ Product Eng. 5 + 2

- 1 Project leader
- 1 Product Eng. / chip
- 2 Techs

④ Test Eng. 6 + 2

- 1 Project Leader
- 1 Test Programmer / chip
- 1 Reliability engineer
- 3 Techs

⑤ Process Eng. 3 + 2

- 1 Device eng. / Project Leader
- 2 Process Eng.
- 2 Tech

⑥ CAD Support 8

- 1 Project Leader
- 1 Logic drafting system
- 2 Hierarchical modelling systems
- 2 circuit simulators
- 2 layout systems

---

40 + 14

# APPROXIMATE MILESTONES

t = 0 MONTHS : Start of detailed design

t = 12 MONTHS : Logic design review

t = 15 MONTHS : Circuit design review

t = 21 MONTHS : 1<sup>st</sup> pass data base complete

t = 24 MONTHS : 1<sup>st</sup> pass parts

t = 29 MONTHS : 2<sup>nd</sup> pass parts

t = 33 MONTHS : 3<sup>rd</sup> pass parts (QUALIFICATION)



# SSE MANPOWER COSTS

(9)



## PROJECT SCHEDULE (SEE DEC STD 008)

PROJECT(S) \_\_\_\_\_ BY \_\_\_\_\_

SHORT DESCRIPTION \_\_\_\_\_ DATE \_\_\_\_\_

USED ON \_\_\_\_\_

← ADVANCED DEVELOPMENT

PRODUCT DEVELOPMENT →

MONTH	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
YEAR	80	80	80	81	81	81	81	82	82	82	82	83	83	83	83
MASTER SCHEDULE PHASE END DATES															
PLAN ACTUAL															
ARCH & LOGIC DESIGN	1	2	3	4+2	5+2	7+2	7+2	7+2	7+2	7+2	7+2	7+2	7+2	7+2	7+2
U-CODE DEVEL.	0	0	1	1	1	2	3+1	5+1	6+2	6+2	6+2	6+2	6+2	6+2	6+2
RTL/GATE SIMUL.	0	0	0	1	3	5	7	8	7	5	3	3	3	2	2
ENGR. TESTER	0	0	0	0	0	0	0	0	1+1	3+2	3+2	3+2	3+2	3+2	1+1
DIAGNOSTICS	0	0	1	1	1	1	2	3	5	5	6	5	6	5	5
SYSTEM INTERFACE	1	1	2	2	2	2	3+1	4+2	4+2	5+2	5+2	5+2	5+2	4+1	4+1
TOTAL MANPOWER	2	3	7	9+2	12+2	17+2	22+4	27+5	30+7	31+8	30+8	30+8	30+8	26+6	23+6
TOTAL COST (\$K)	40	60	140	212	272	372	504	620	712	748	728	728	728	616	556

Product Dev. Start ↑

Logic Review ↑

CMT Review ↑

IC Pass Facts ↑

2nd Pass Facts ↑

DEFINITION, SPECIFICATION & SCHEDULE



ENGR.



DRAFTING



PROTO F&A



PROTO TEST



CORRECT REDESIGN



PILOT F&A



PILOT TEST



ACCEPTANCE, INSTALLATION, REDESIGN



RELEASE



DESIGN REVIEWS



PROJECT SCHEDULE  
(SEE DEC STD 008)

PROJECT(S) \_\_\_\_\_ BY \_\_\_\_\_

SHORT DESCRIPTION \_\_\_\_\_ DATE \_\_\_\_\_

USED ON \_\_\_\_\_

ADVANCED DEVEL.

PRODUCT DEVELOPMENT

MONTH	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
YEAR	80	80	80	81	81	81	81	82	82	82	82	83	83	83	83
MASTER SCHEDULE PHASE END DATES															
PLAN ACTUAL															
CKT DESIGN & SIMUL.	1	1+1	3+1	5+2	7+3	9+4	9+4	9+4	9+4	9+4	9+4	9+4	9+4	9+4	9+4
LAYOUT	1	1	2	3+1	5+2	7+3	9+4	9+4	9+4	9+4	9+4	9+4	9+4	9+4	9+4
PRODUCT ENG.	0	0	0	0	0	0	0	0	0	0	7	3+1	5+2	5+2	5+2
TEST ENG.	0	0	0	1	1	1	1	3	5	5	5	6+2	6+2	6+2	6+2
PROCESS ENG.	3+2	3+2	3+2	3+2	3+2	3+2	3+2	3+2	3+2	3+2	3+2	1+1	1+1	0	0
CAD SUPPORT	5	5	7	7	8	5	3	1	1	1	1	2	0	0	0
TOTAL MANPOWER	10+2	10+3	15+3	19+5	24+7	25+9	25+10	25+10	27+10	27+10	26+9	30+12	29+12	29+12	29+12
TOTAL COST (\$K)	232	248	348	460	592	644	660	660	700	700	664	792	772	772	772

Product Dev. Review

Logic Review, CAT Review

1st Pass Parts

2nd Pass Parts

DEFINITION, SPECIFICATION & SCHEDULE



ENGR.



DRAFTING



PROTO F&A



PROTO TEST



CORRECT REDESIGN



PILOT F&A



PILOT TEST



ACCEPTANCE, INSTALLATION, REDESIGN



RELEASE





PROJECT SCHEDULE  
(SEE DEC STD 008)

PROJECT(S) \_\_\_\_\_ BY \_\_\_\_\_

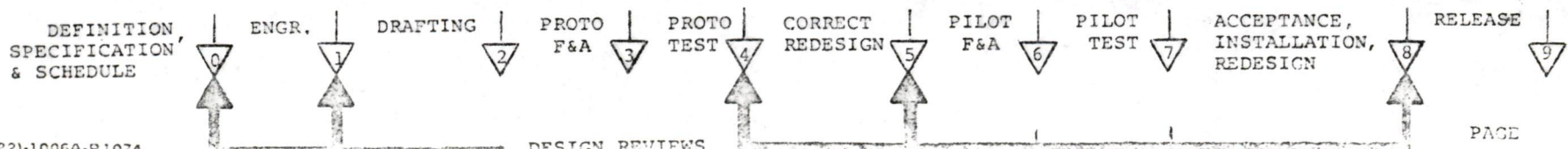
SHORT DESCRIPTION \_\_\_\_\_ DATE \_\_\_\_\_

USED ON \_\_\_\_\_

ADVANCED DEVEL.

PRODUCT DEVELOPMENT

MONTH	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
YEAR	80	80	80	81	81	81	81	82	82	82	82	83	83	83	83
MASTER SCHEDULE PHASE END DATES															
PLAN ACTUAL															
COMPUTER TIME	14	16	24	28	32	38	46	52	56	52	52	52	48	46	40
ENG. TESTER	0	0	0	0	0	0	0	0	0	30	30	30	0	0	0
TOOLING	0	0	0	0	0	0	0	0	0	0	0	60 <sup>(4)</sup>	105 <sup>(7)</sup>	135 <sup>(9)</sup>	0
FAB	0	0	0	0	0	0	0	0	0	0	0	48 <sup>(4)</sup>	84 <sup>(5)</sup>	108 <sup>(7)</sup>	0
CONTINGENCY															
CONSULTANTS	0	0	50	50	25	25	25	25	25	25	25	25	25	25	25
EXTRA PASS/CHIP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL COST (#K)	14	16	74	78	57	63	71	77	81	107	107	215	262	314	27



TOTAL PROJECT COSTS (\$K)  
(CSE LSI & MPL)

FY 80

# 1272

Q1	# 100
Q2	# 286
Q3	# 324
Q4	# 562

FY 81

# 3985

Q1	# 750
Q2	# 921
Q3	# 1079
Q4	# 1235

FY 82

# 5904

Q1	# 1357
Q2	# 1493
Q3	# 1555
Q4	# 1499

FY 83

# 6754

Q1	# 1735
Q2	# 1762
Q3	# 1702
Q4	# 1555

# 17915

⇒ \$18M

# CONCLUSIONS

• 8 CRITICAL ROLES

- SSE PROJECT MANAGER
- MPD PROJECT MANAGER
- CHIP SET ARCHITECT
- HEAD CIRCUIT DESIGNER
- DEVICE ENGINEER
- TEST STRATEGIST
- SYSTEMS ARCHITECT
- HEAD MICROPROGRAMMER

• ACUTE SHORTAGE OF RESOURCES TODAY;  
 LONG LEAD TIME ON PROWING RESOURCE



FIND RESOURCES SHOULD BE  
 #1 PRIORITY TODAY

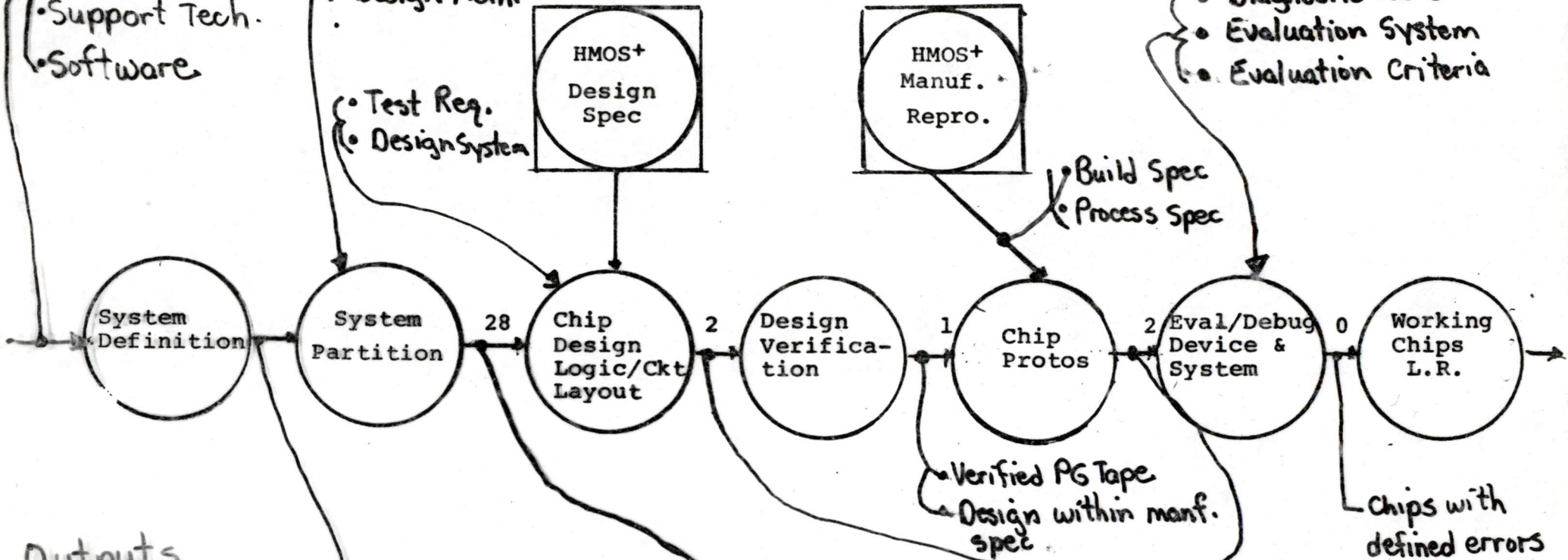
# Inputs

- Product Goals
- Implem. Tech.
- Support Tech.
- Software

- Process Alt.
- Logic Elem.
- Design Meth.

- Test Req.
- Design System

## COARSE PROJECT PERT



# Outputs

- Phase & Bus. Plan
- Project Plan

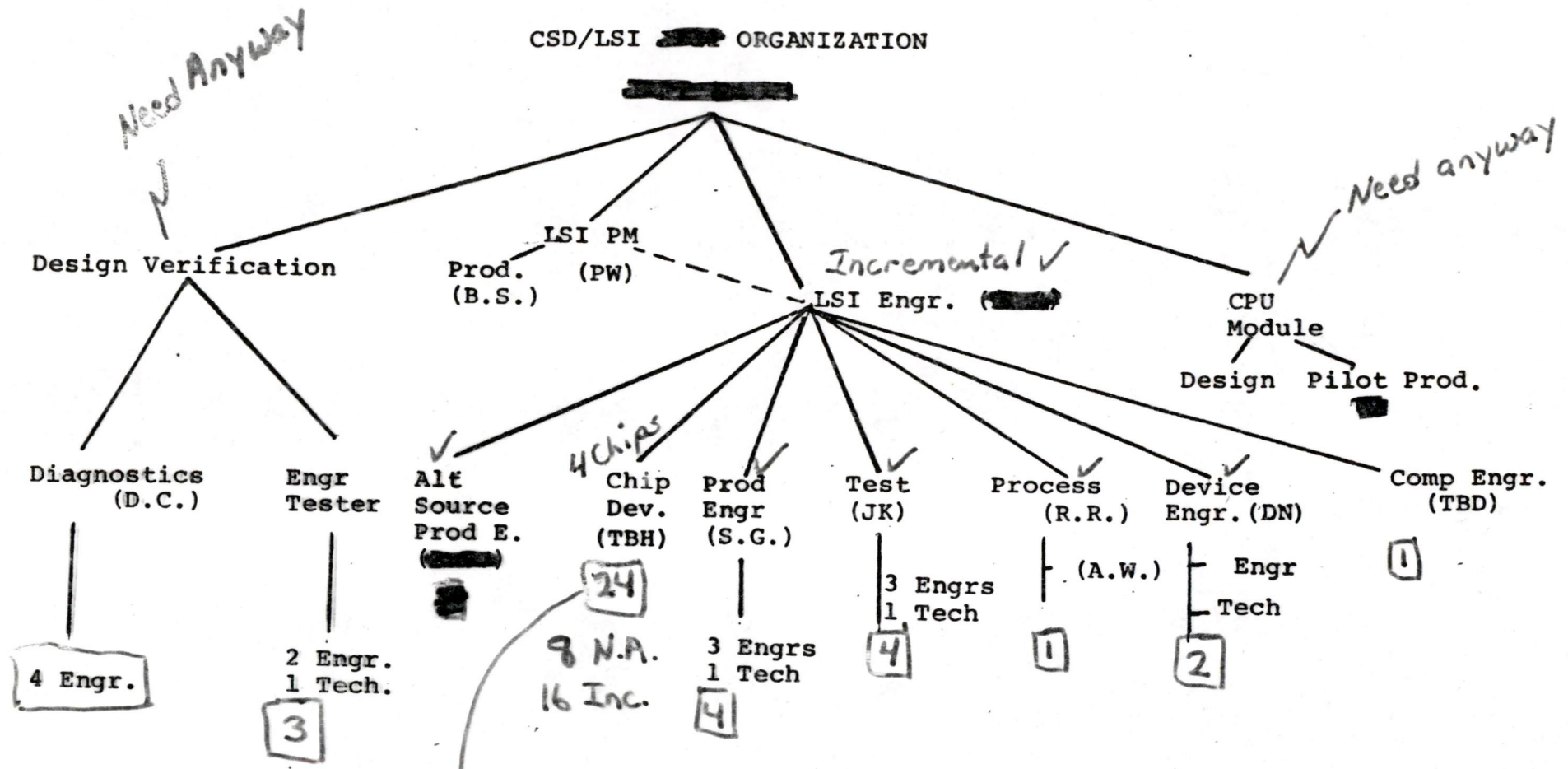
- Obj. Specs. Chips
- System Software
- Chip layout Plan (Ref)
- Process Selection

- Chip Data Base
- Refined Chip Specs.
- $\mu$  Code

- Chip within mfg. spec.

Chips with defined errors

### CSD/LSI ORGANIZATION



*Need Anyway*

*Need anyway*

*Incremental ✓*

*Critical Path*

### Program Resources

Need Anyway - 16 + CPU Product  
 Incremental - 28  
 44+

LSI Responsibilities

- Chip Design
  - o Logic
  - o Circuit
  - o Layout
  - o Participate in Partitioning
- Process Development
- Manufacturing
- Device Engineering
- Design Process & Tools [LSI]
- Packaging
- Chip Testing

Small System Responsibilities

- System Definition
- System Architecture
- Hardware Design Verification
- Microcode
- System Implementation
- Software
- Diagnostics



## Assumptions:

- Model is experience on F-11

• Complexity will increase but better design process will offset

• New Process Required

• For the purpose of this plan 4 chips 60k  $\rightarrow$  100k devices/chip

# Resource Requirements [81]

## Process Development & Manuf. [14]

- Process Dev.
  - 5 process Engr
  - 2 process Tech
- Ckt Test Chips
  - 1 ckt Engr.
  - 1 Layout
- Device Modelling & Process of Chips
  - 2 Engr
  - 1 Tech.
  - 2 Software Engr.
- Manuf. Resources (?)

## Design Methods & Tools [10]

- 2 ckt Engr.
- 2 Layout
- 1 Logic Engr.
- 5 Software Engr.

## Chip Design [37]

- 1 Project Manager
- 1 Project Engineer
- 8 Circuit Design
- 8 Logic Design / Verilog /
- 4 Technicians
- 8 Layout
- 3 CAD Tech
- 4 Software Sim

## Chip Test / Char. / Design Eval [17]

- 4 Product Engr.
- 4 Test Engr.
- 4 Test Tech
- 2 Engineers Design Verilog
- 2 Techs.
- 1 Project Engr.

## Packaging [3]

- 2 Engr.
- 1 Tech

JACK SMITH

WILL THOMPSON

ME/Q.A./ADV.  
TECH.

INTERCONNECTS  
JIM MELVIN

PROCESS  
MGAS

OVERALL  
TEST STRATEGY

TEST  
STRATEGIES

- CPU
- TERM
- MASS
- GENERAL

CPU PROJECT

# TEST STRATEGIES

GROUP MTGING - 1ST TIME EVER

OCT 25

STRATEGIES FOR INCOMING  
THRU

SYSTEM INTEGRATION

VISIBLE BY

DEC - JAN TIME FRAME.

---

IMPORTANT TO SCORPIO

LSI TEST - TOM MARMEN

MODULE TEST - RICH POWERS

SCORPIO TEST STRAT  
~~TEST~~ ~~TEST~~ - CPU - AL SMITH

---

IMPORTANT PROJECT

CPU - LOAD/TEST/DIAGNOSE  
PROJECT (E.D.G.)

---

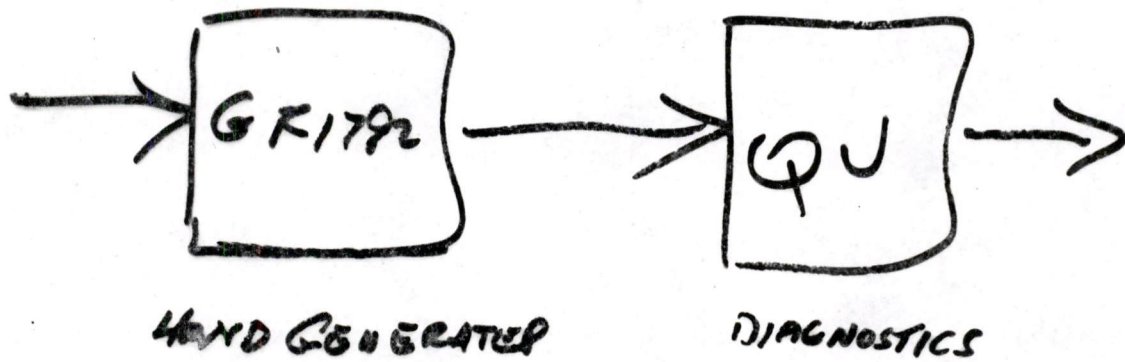
IMPORTANT TO MODULE TEST

MODULE FUNCTION/PERFORMANCE  
SPECS

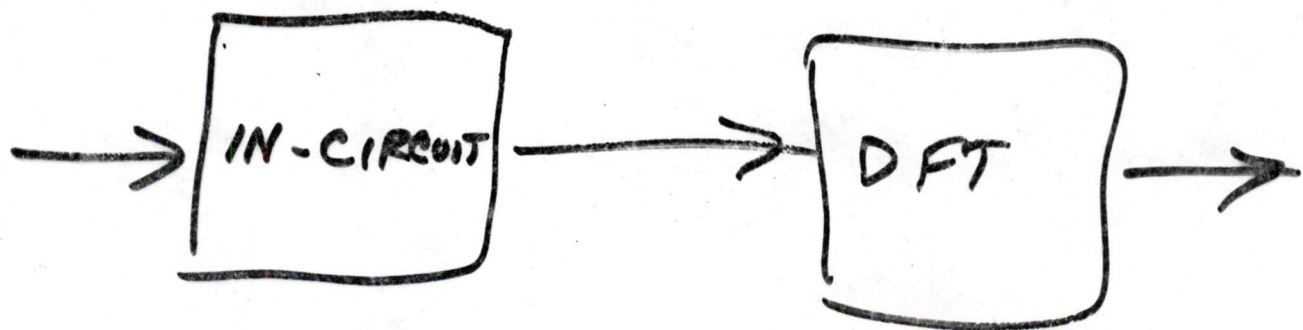
(ENGINEERING'S RESPONSIBILITY)

# MODULE TEST

PRESENT



FUTURE (?)



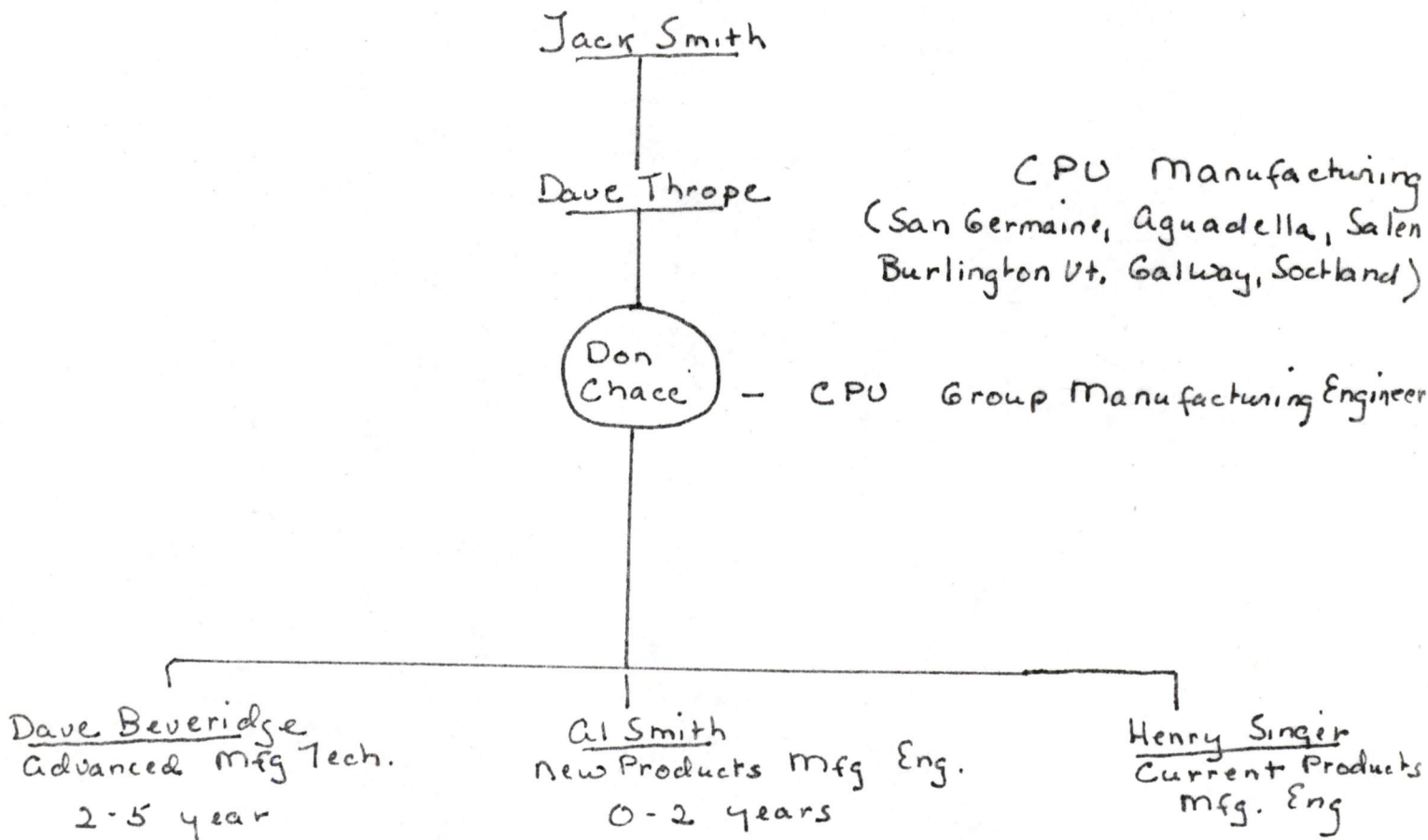
TEST PATTERNS  
↓  
SAME AS INCOMING (?)

TEST PATTERNS  
↓  
VOTE  
+

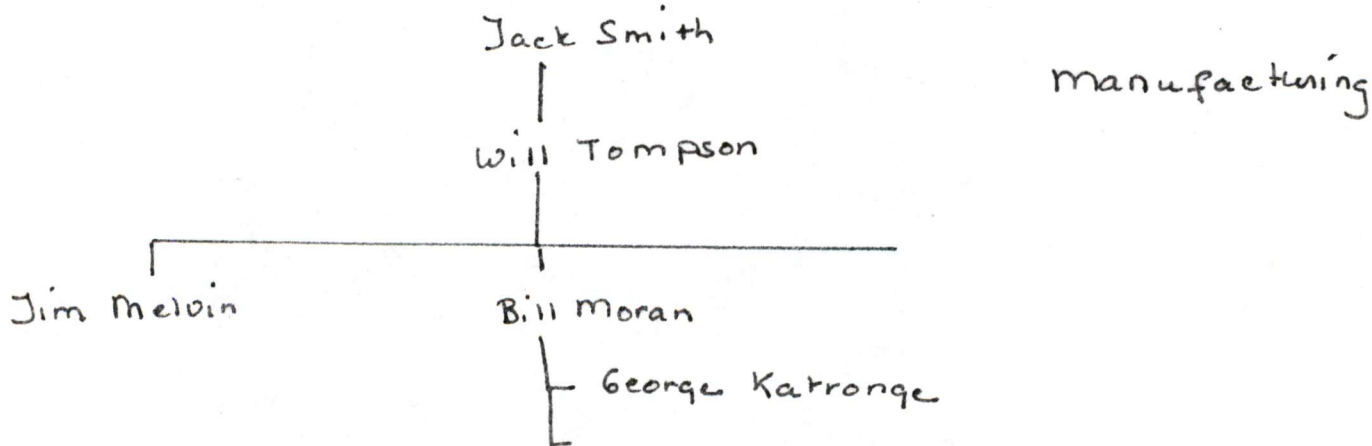
\* RISK - FIXTURING

From newsprint  
John Groark

### PRODUCT ORIENTED



### PROCESS ORIENTED



# APOLLO XI PRODUCT GOALS

- FULL VAX Functionality
- VMS Operating System
- Higher cost/performance than NEBULA
  - System
  - Box
  - Board

## • Typical product

Quad board :

CPU

512KB memory

console interface

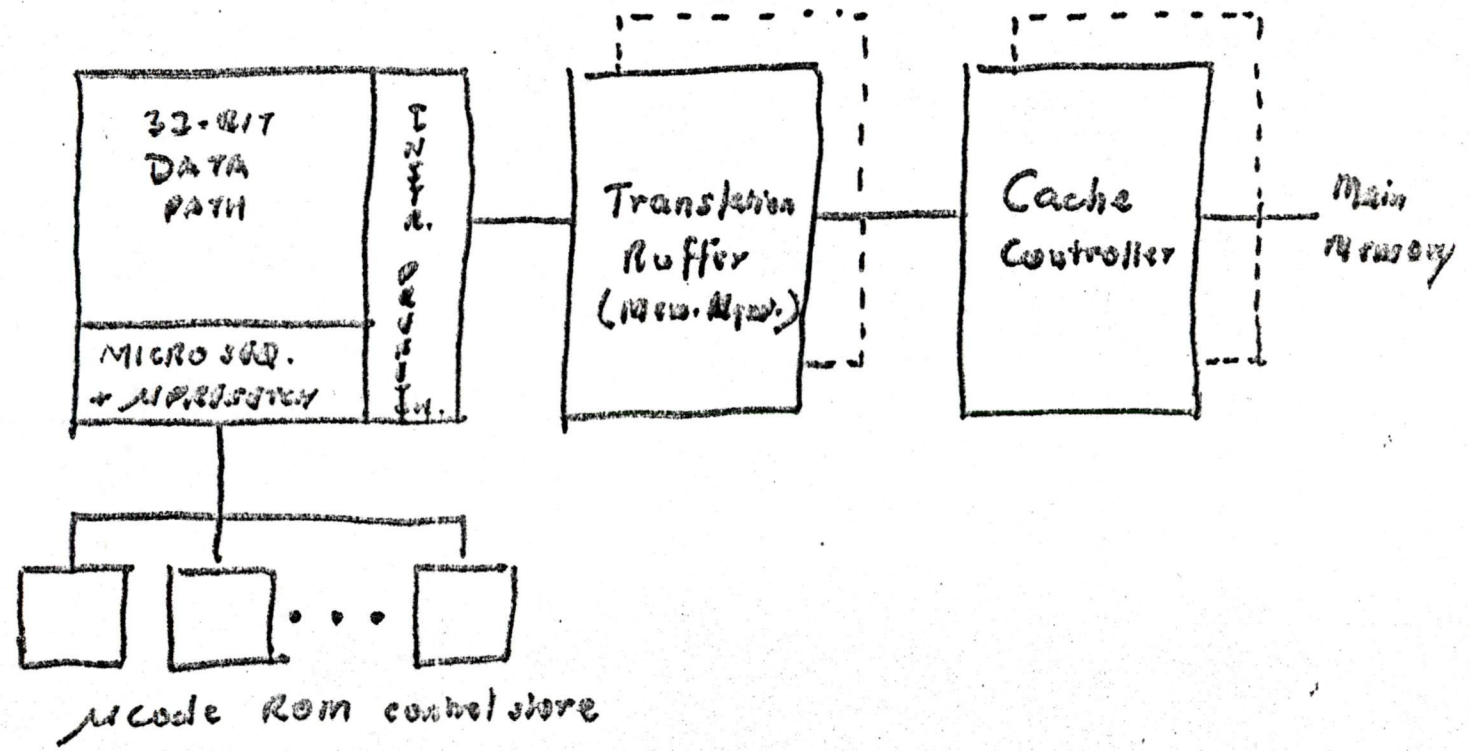
serial line interface

public bus interface

- FCS Date NOT KNOWN

DPB, 4-9-79

# POSSIBLE PARTITIONING





# COST / PERFORMANCE

	~ FY84/85 COST ESTIMATE		APOLLO II: Degr. for	
	<u>NEBULA</u>	<u>APOLLO II</u>	50% higher cost pufe	100%
SYSTEM (CPU, 1MB memory 10MB disk, console, comm)	\$ 4,300	3,600	1.4	1.7
BOX (CPU, 1MB memory, console)	2,400	1,700	1.1	1.5
BOARD (CPU, 1MB memory)	2,000	1,100	0.9	1.2

Note: Costs based on Supnik memo dated Mar 2, 1979.

DPB, 4-9-79

TECHNOLOGY

4 $\mu$

7 custom designs  
20 die

$\sim$  NEBULA perf.

3 $\mu$

5 custom designs  
15 die

1 = 1.5 x

2 $\mu$

4 custom designs  
10 die

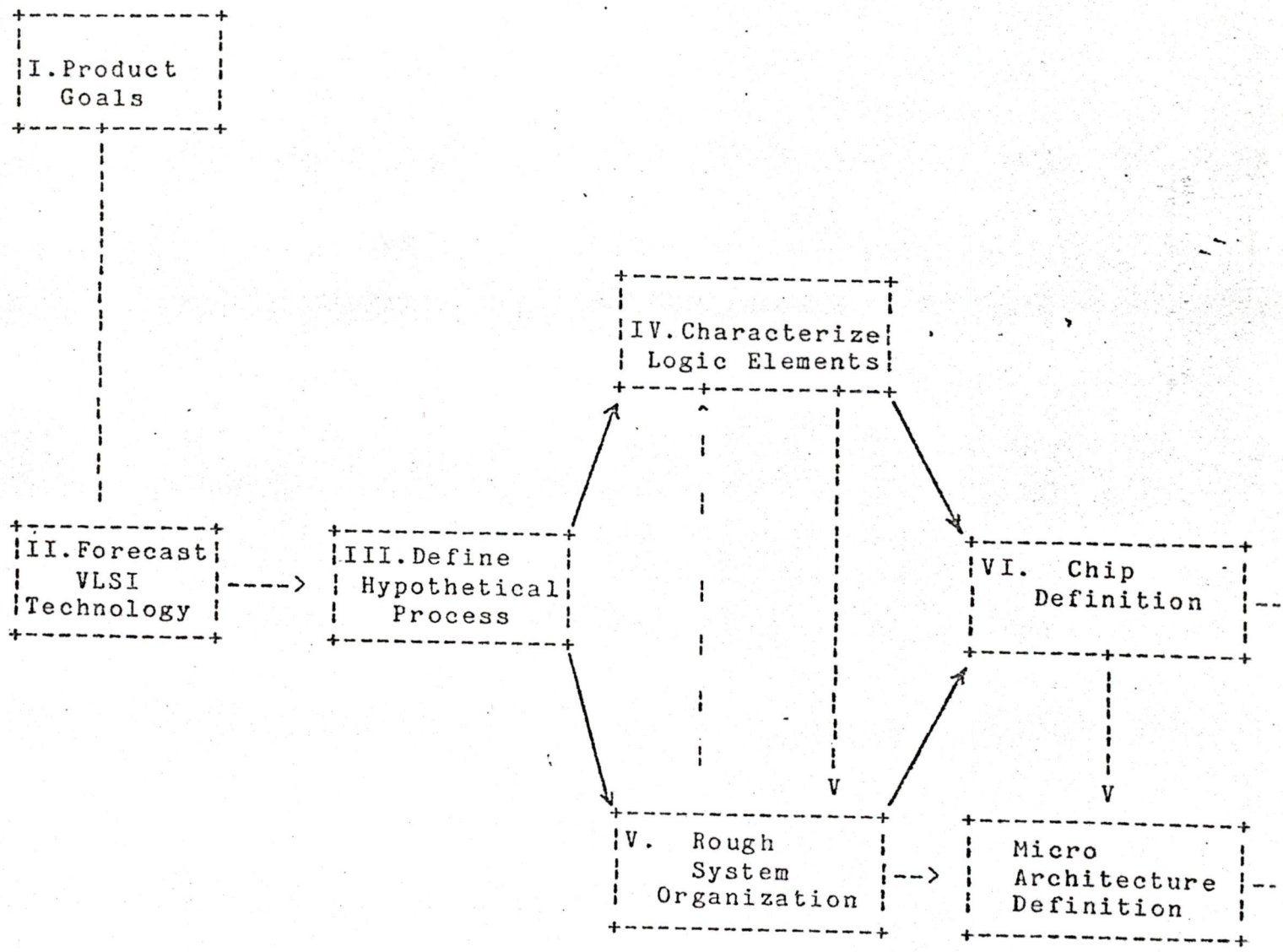
1.25 - 1.75 x

DPB, 4-9-79

COMPLEXITY

	<u>11/70</u>	<u>VAX</u>
General Registers	16 x 16	16 x 32
Floating Point	6 x 64	—
Mem. Mgmt.	96 x 16	4 x 32
Internal Proc. Reg.	8 x 16	8 x 32
ALU: Integer	16 bits	32 bits
FP/CIS	33	32
<u>Microcode</u>		
Low End	124 K (11/33)	384 K (Nebula)
Mid Range	170 K (11/44)	480 K (Comet)
High End	200 K (Bluebird)	500 K (11/70)

DPB, 4-9-79



PHASE I: INITIAL STUDY (ADVANCED DEVELOPMENT)

I. DEFINITION OF GOALS

- Approximate timeframe
- Cost
- Performance
- Functionality

II. FORECAST VLSI TECHNOLOGY

- List Technology alternatives consistent with Product goals
  - lithography - e.g., 3u, 2u
  - process NMOS, CMOS, etc.
  - RAM technology 256K bit
  - ROM technology 64K bit, 100ns
  - characteristics of (process, lithography)
- Identify no more than 2 or three most attractive candidates for further investigation

Prerequisite: Product Goals

Next Steps: Define Hypothetical Process.

Tools: Crystal Ball  
Technology Forecasts

III. DEFINE HYPOTHETICAL PROCESS

- Define and analyze theoretical process
- Generate process/device characteristics

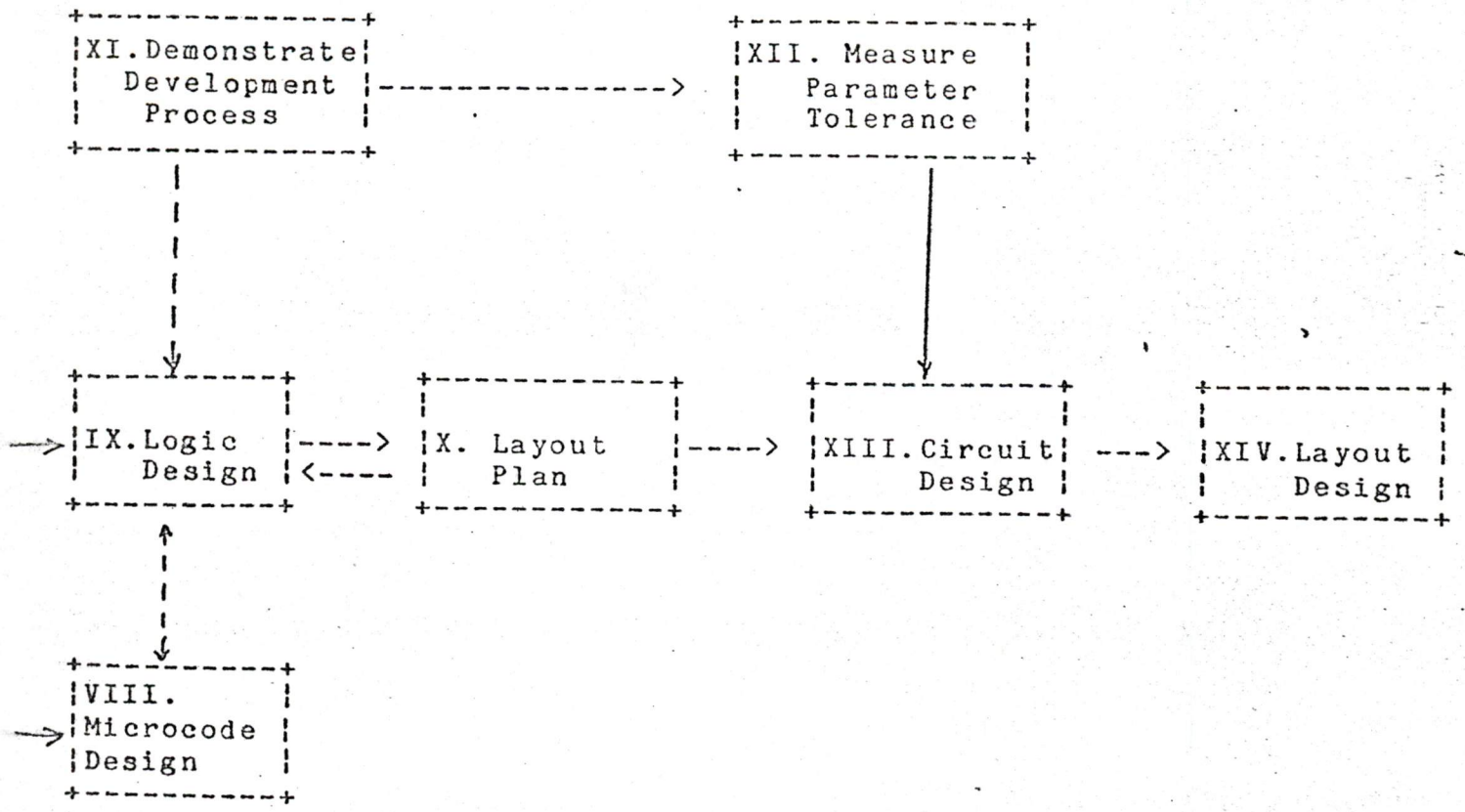
Prerequisite: Approximate process characteristics

Next Steps: Logic element characterization  
Rough system organization

Tools: SUPREME process simulation  
TANDEM  
SEDAN device simulation

IV. LOGIC ELEMENTS CHARACTERIZATION

- Create chip designers DATA Book
  - Define logic design rules
  - characterize logic building blocks
    - area
    - power
    - speed



PHASE II: LOGIC DESIGN  
LAYOUT PLAN

PHASE III: CIRCUIT LAYOUT  
DESIGN

Prerequisite: Process/device characteristics

Next Steps: Chip definition  
Rough system organization

Tools: SLIC Circuit simulation  
STICKS Building block definition/estimation

#### V. ROUGH SYSTEM ORGANIZATION

- System level definition/characterization
- Bus definition
- Partitioning into chips
- Rough chip definition
- Rough microarchitecture specification

Prerequisite: Logic element characteristics  
Process parameter estimates

Next Steps: Chip definition  
Microarchitecture definition

Tools: TUMS RT level simulation

#### VI. CHIP DEFINITION

- Rough logic design
- Rough layout plan
- Rough circuit characterization
- Power budget plan
- Test strategy

Prerequisite: Logic element characteristics  
Rough system organization

Tools: SUDS/SCALD      Design entry/management system  
          TUMS              RT level simulation

## VII. MICROARCHITECTURE DEFINITION

- Define microarchitecture

Prerequisites: System organization  
                  Chip definition

Next Step: Microcode design

## VIII. MICROCODE DESIGN

- Design/write/test microcode

Prerequisite: Chip definition  
                  Microarchitecture definition

Tools: TUMS              Microcode simulation  
          MICRO2            Microassembler

## IX. LOGIC DESIGN

- Detailed logic design
- Select appropriate logic elements from data book
- Identify circuit requirements for logic elements

Prerequisite: Chip definition  
                  Logic building block data book and design rules  
                  Measured process performance

Iterate with: Layout Plan

Tools: SUDS/SCALD      MOS Logic simulation  
          SAGE 2            Path identification between points  
          WIRECHECK        Delay calculator between points  
          DELAY

## X. LAYOUT PLAN

- Feedback parasitic effects to logic design
- Identify critical routing paths

Tools: ???



XI. DEMONSTRATE DEVELOPMENT PROCESS

- Necessary to increase confidence in process characteristics simulated/estimated

XII. MEASURE PROCESS PARAMETERS

- Detailed characterization of process parameter tolerances

XIII. CIRCUIT DESIGN

- Detailed circuit design
- Verify circuit performance/function with logical description

Prerequisite: Layout plan  
 Logic design  
 Process parameter tolerance

Next Step: Layout design

Tools: STICKS            Topological Circuit representation  
 SLIC                    Circuit simulation  
 logic/Circuit verification tool

XIV. LAYOUT DESIGN

-Layout

Tools: Geometric Editor  
 DRC                    Design Rule checking  
 DIVER                 Device interconnect verifier

DESIGN VERIFICATION

Levels of Description

1. System
2. CHIPS
3. RT level blocks
4. Logic/topology (gates)
5. Circuits (devices)
6. Layout

Pairwise comparison of descriptions may be needed to verify design.

TEST STRATEGY: Recognize critical need but not certain about best approach.  
*testability?*

SIMULATION STEERING COMMITTEE DISTRIBUTION

COMMITTEE MEMBERS:

Dick Beaven	ML21-3/E87
Steve Ching	ML 1-1/E24
Steve Greenberg	WZ-2
John Maenpaa	WZ-2
Will Sherwood	WZ-2
Ernst Ulrich	ML21-3/E87
Don Yelton	ML 1-1/E24
Bob Kusik	ML 3-5/H33

INTERESTED INDIVIDUALS:

Dick Albright	ML21-3/E87
Gordon Bell	ML12-2/A51
Bert Bruce	ML 1-1/E24
Phil Corman	ML 1-1/E24
Rattan Dhar	MR 1-1/M42
Dave Gross	WZ-2
Marv Horovitz	ML21-4/E10
Bill Johnson	ML12-1/T32
Maurice Marks	ML 3-5/E82
Val Patel	WZ-2
Bill Segal	ML 3-5/E82
Don White	ML 1-2/E60

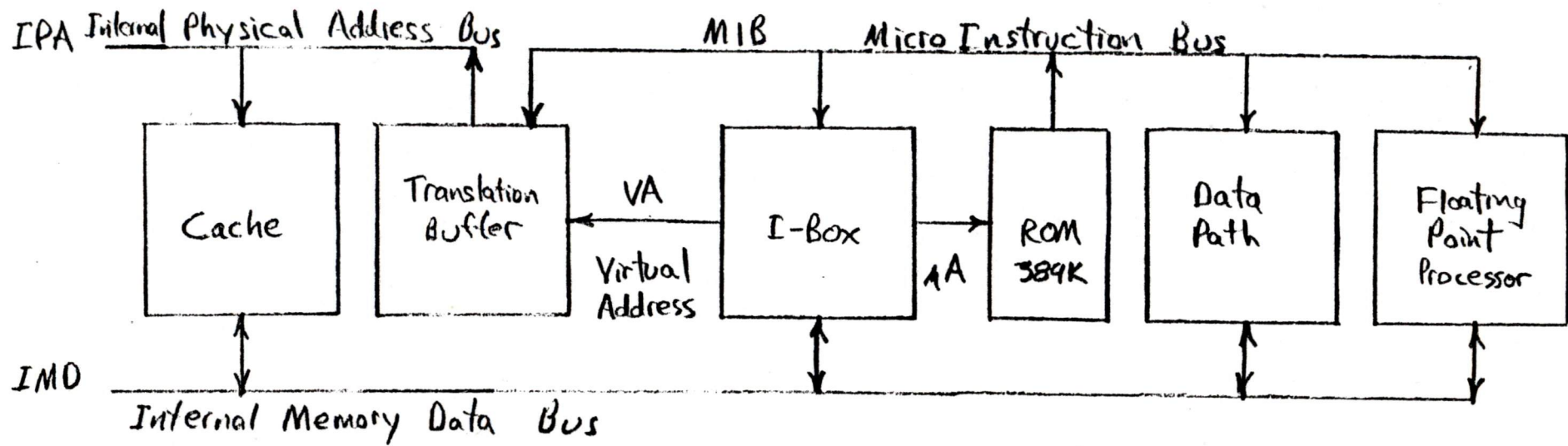


Figure 1

SCORPIO-REVISION 0

5 CUSTOM CHIP IMAGES

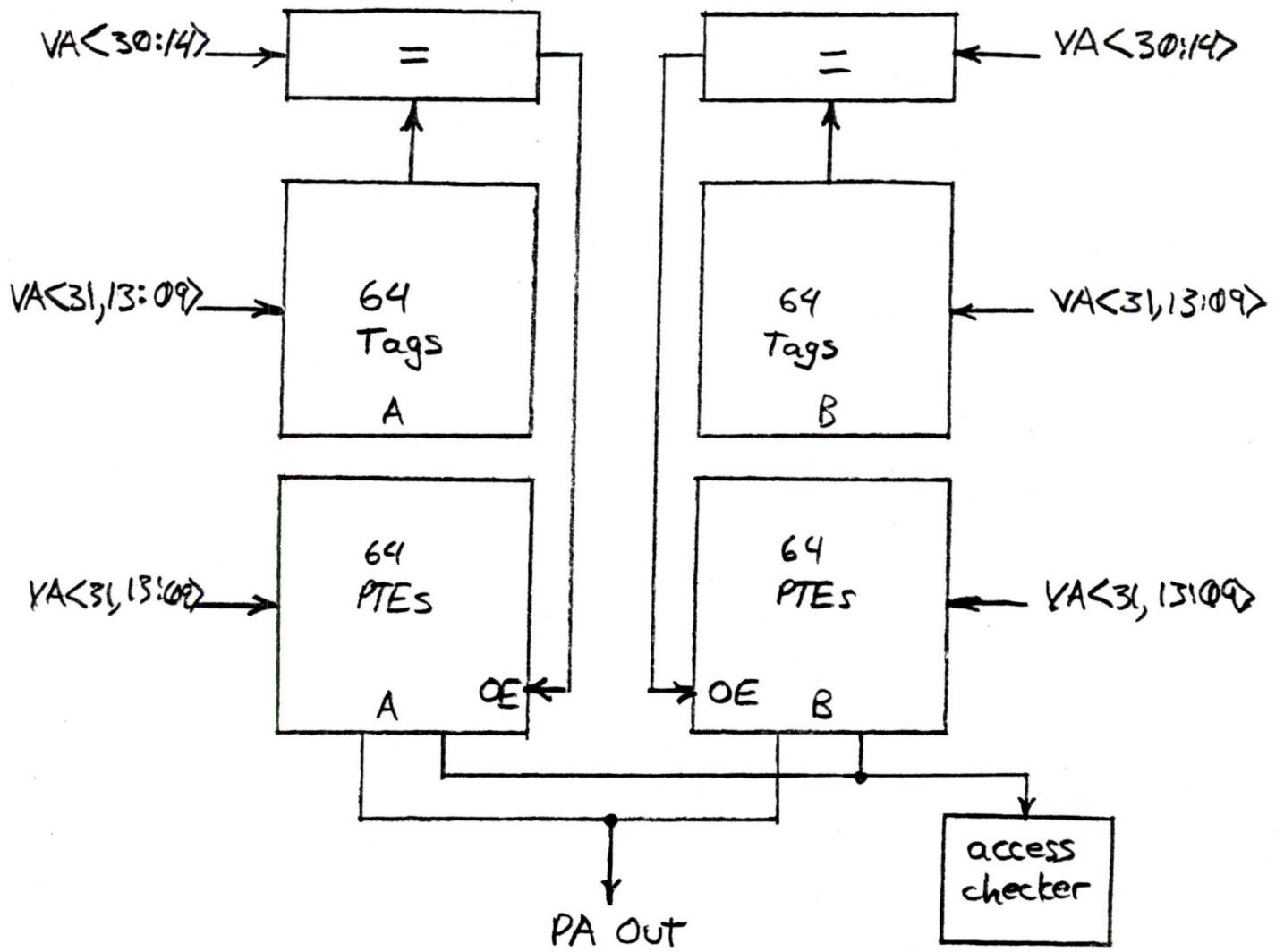
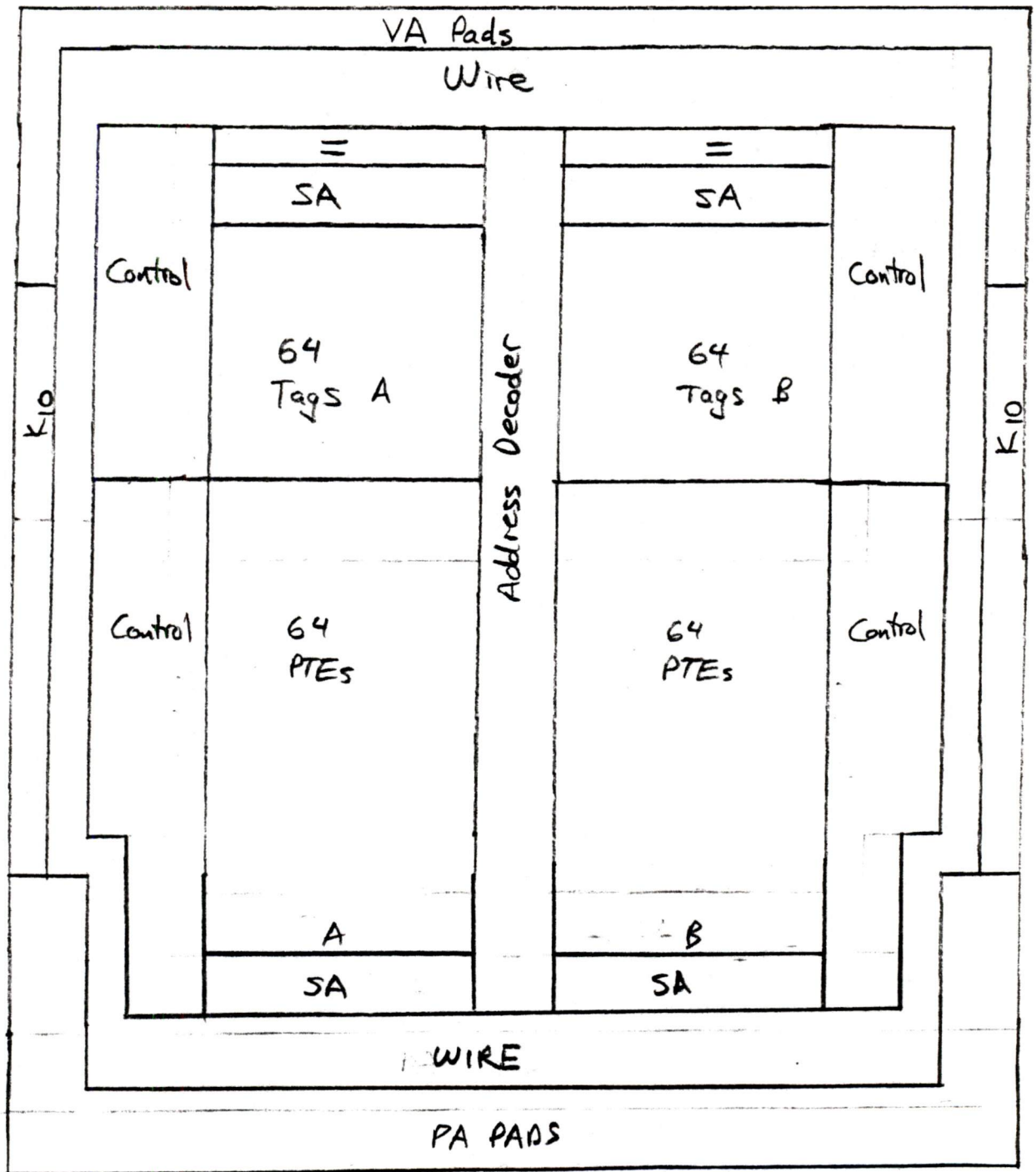


Figure 2



260 x 295 mils

Figure 3

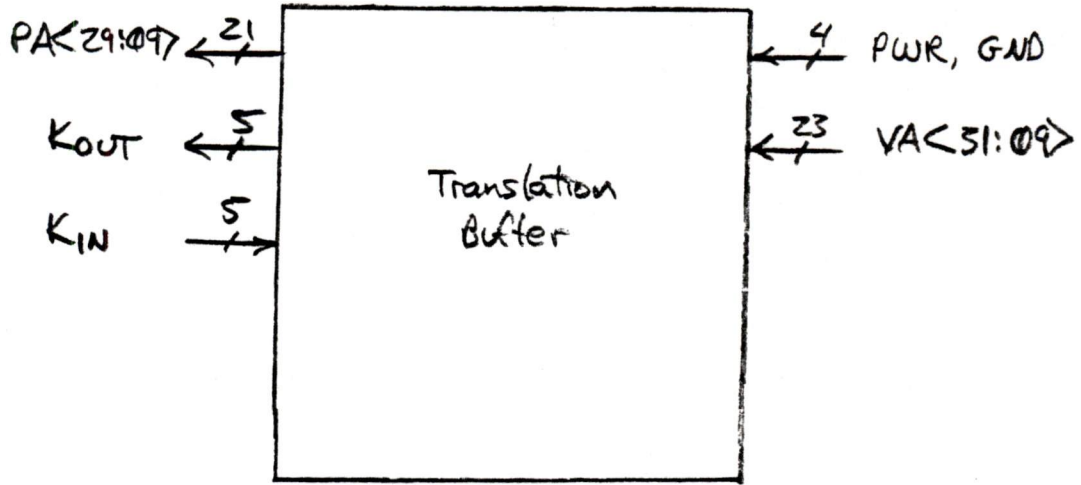
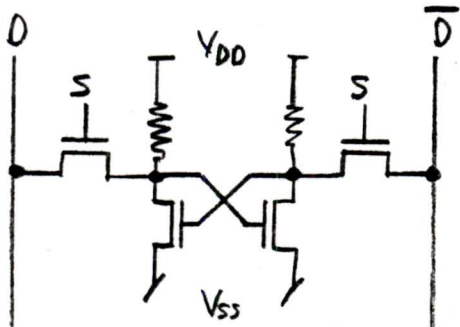
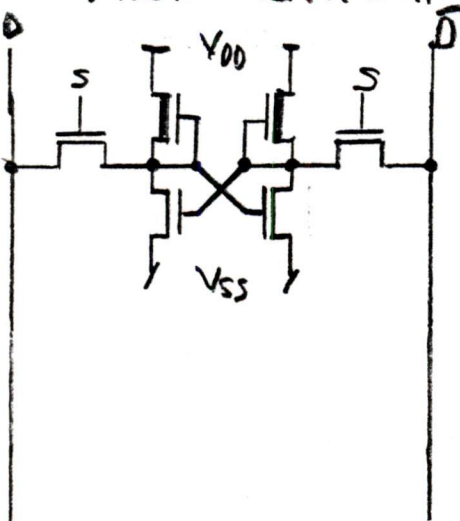


Figure 4

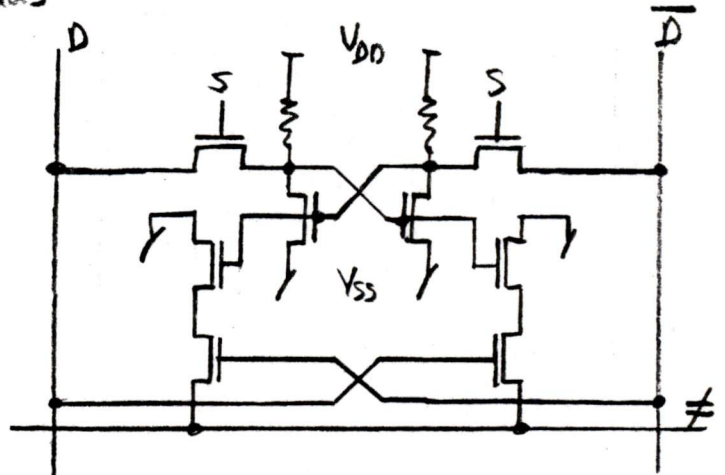
Poly Loads



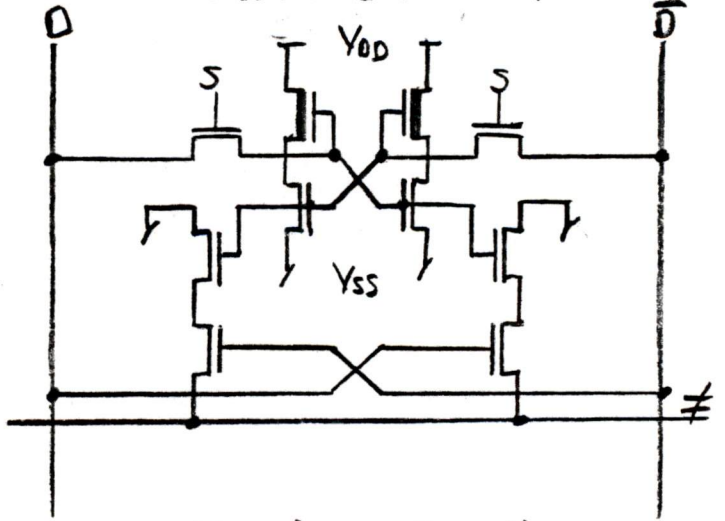
4 xistor RAM cell



6 xistor RAM cell



8 xistor CAM cell



10 xistor CAM cell

Figure 5

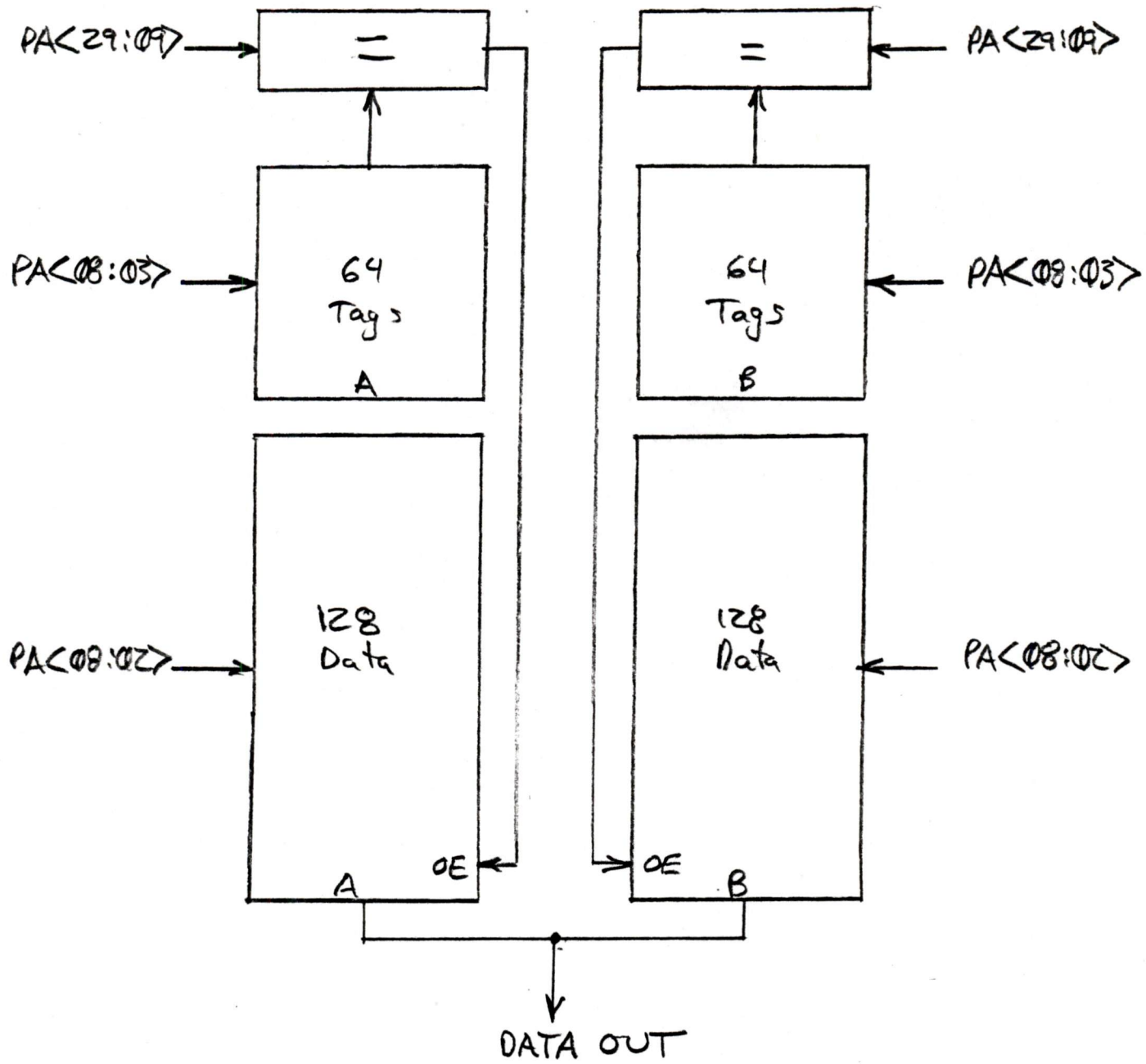


Figure 6

Table 1

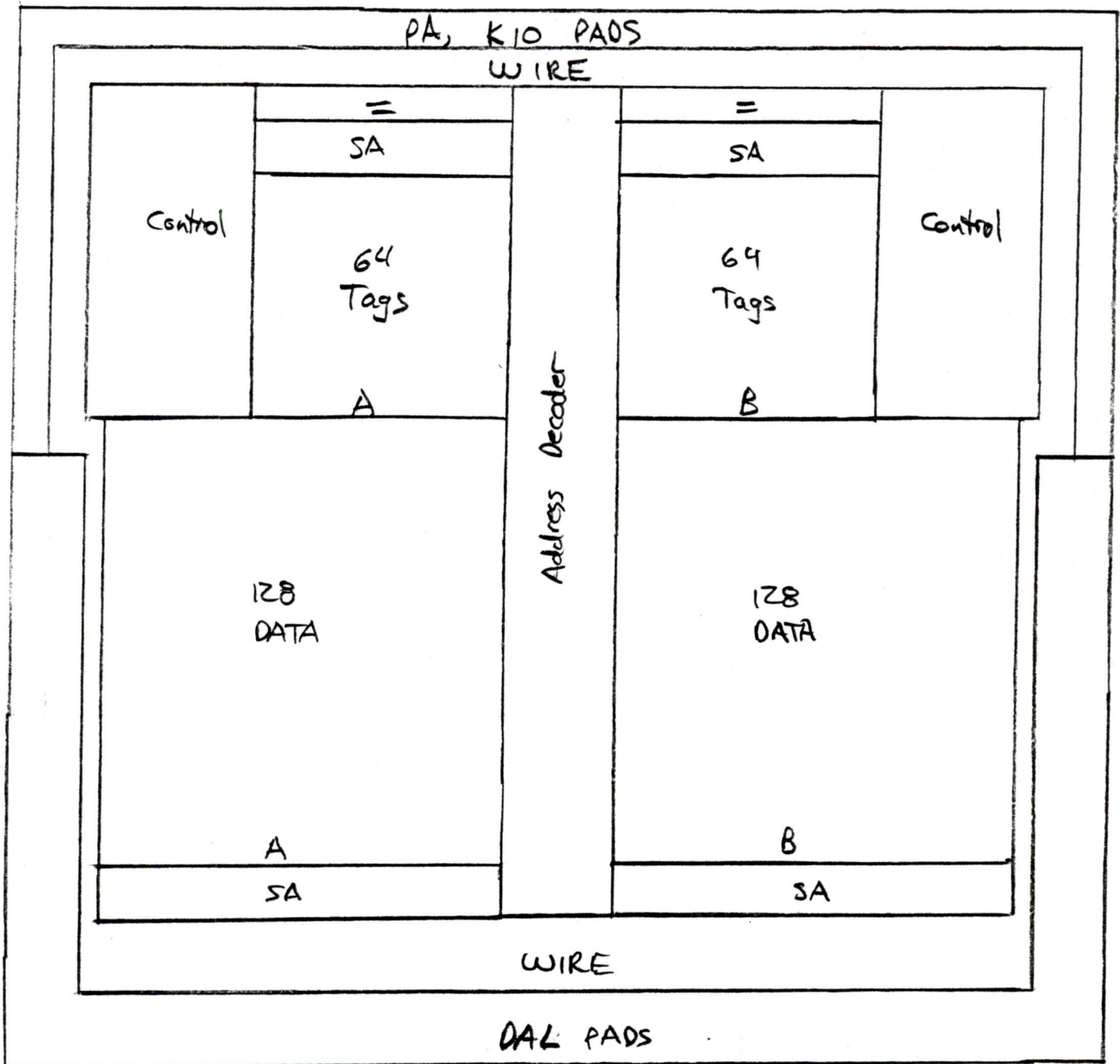
Mp Access Time	Effective Access Time
1000 ns	370 ns
900 ns	340 ns
800 ns	310 ns
750 ns (BI)	295 ns
700 ns	280 ns
600 ns	250 ns
500 ns	220 ns
400 ns	190 ns
300 ns	160 ns

$$\text{Effective Access Time} = \text{mr} \cdot \text{Mp Access Time} + (1 - \text{mr}) \cdot \text{Cache Access Time}$$

$$\text{mr} = \text{miss ratio} = 0.3$$

$$\text{Cache Access Time} = 100 \text{ ns}$$





300 x 285 mils

Figure 7

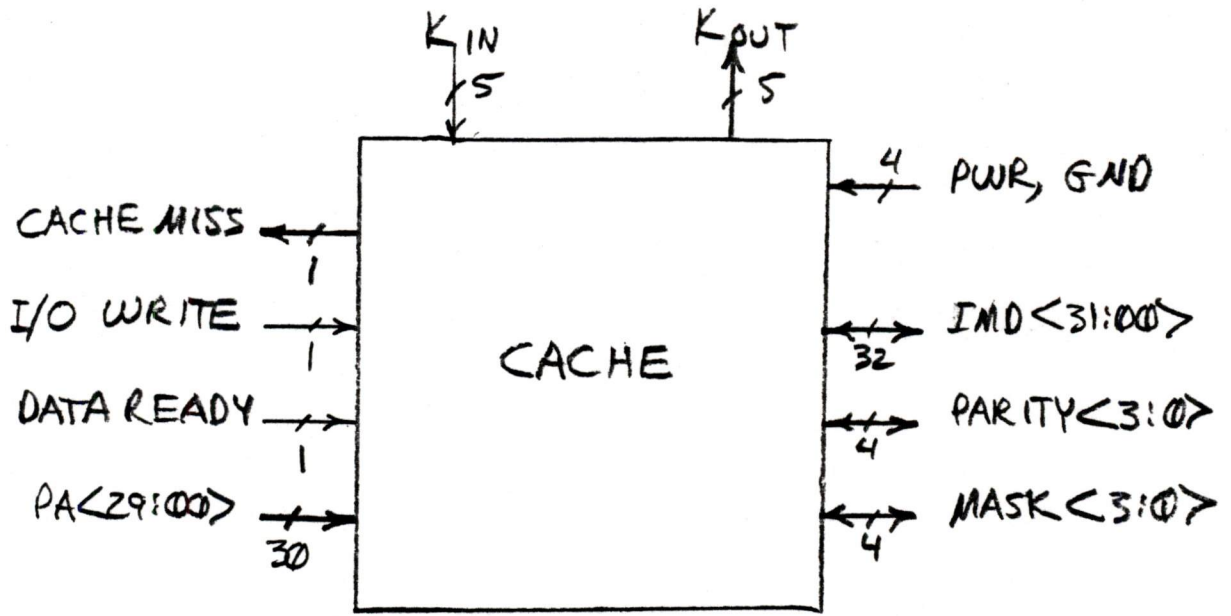
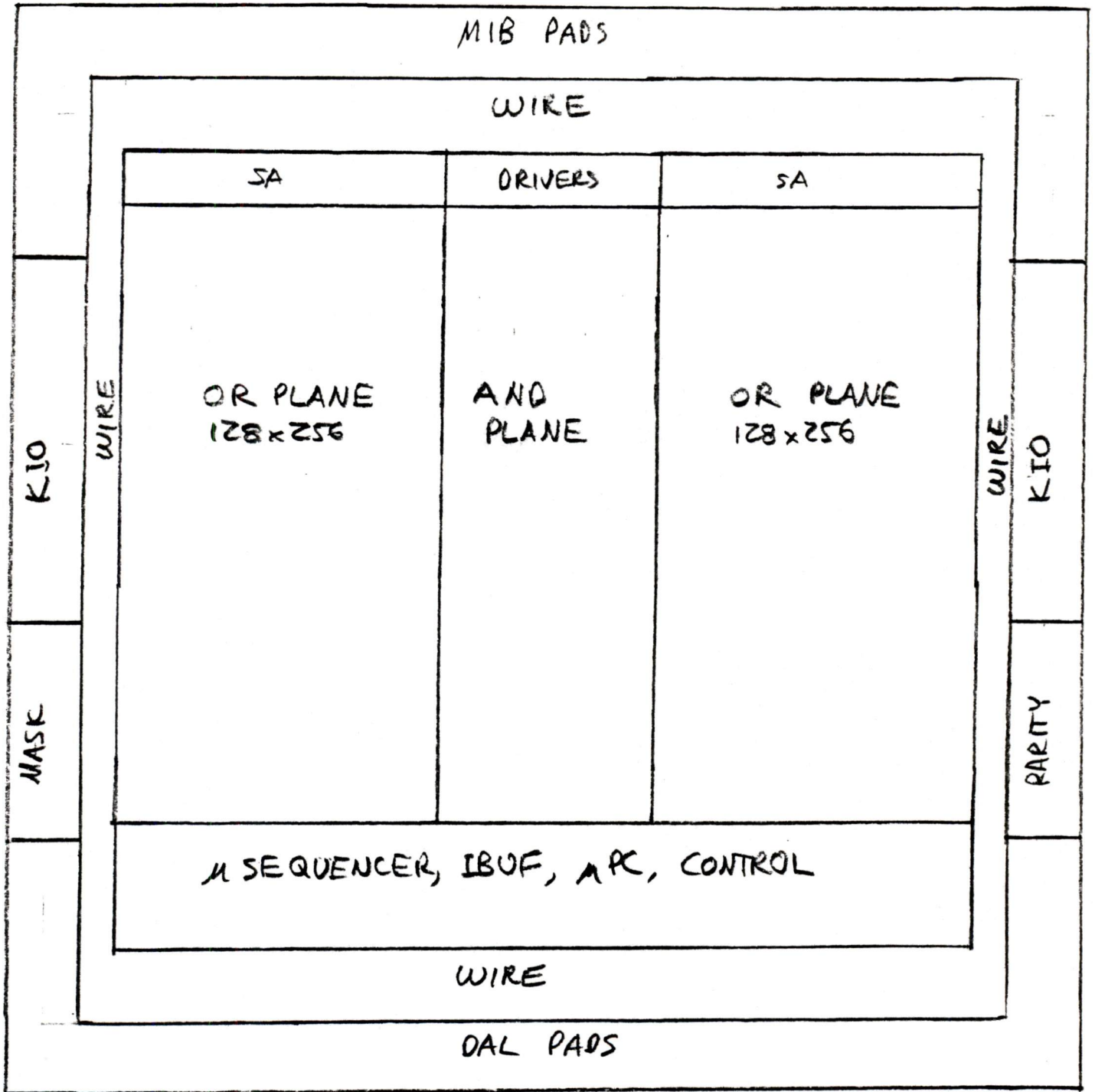


Figure 8



300 x 300 mils

Figure 9

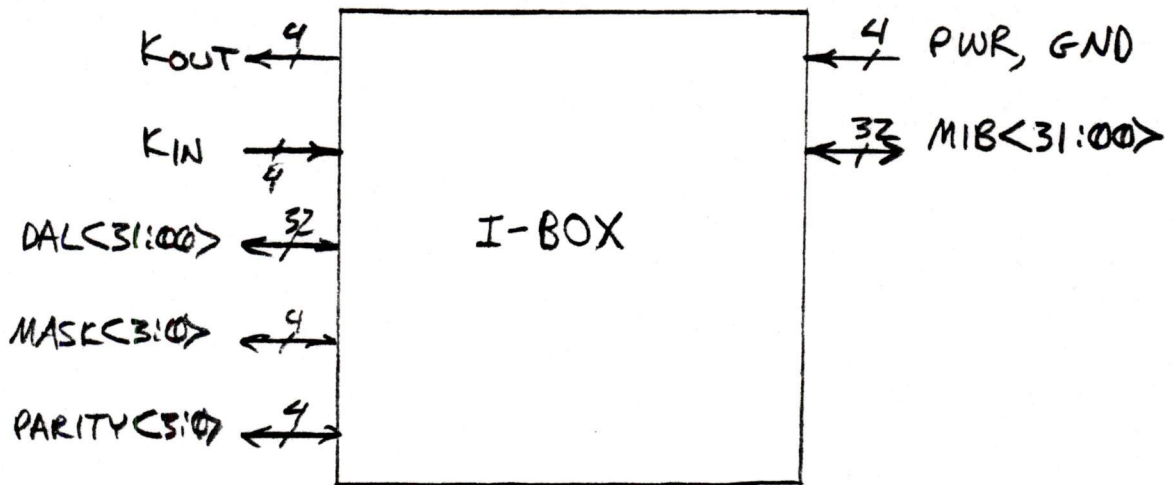


Figure 10

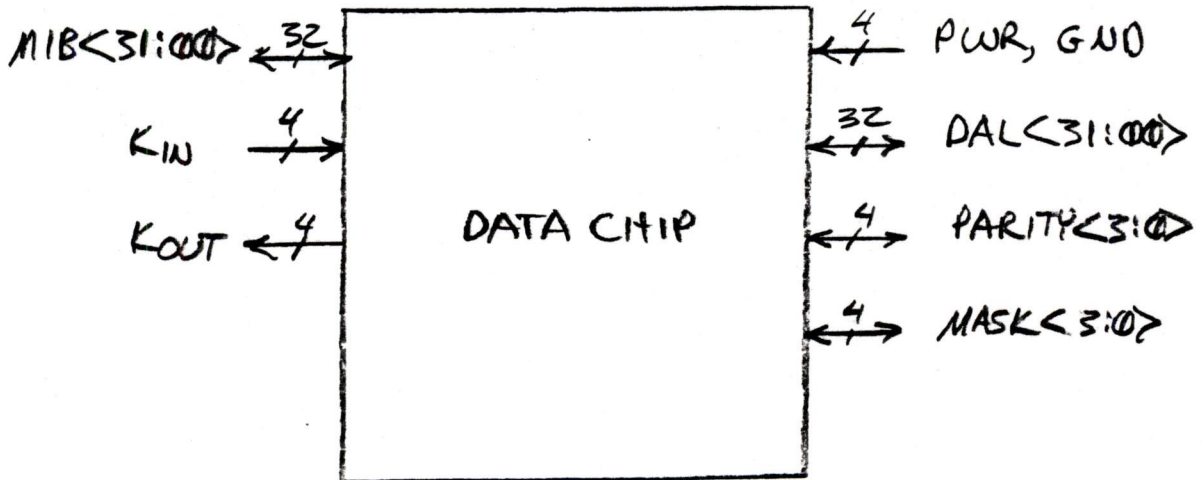


Figure 11

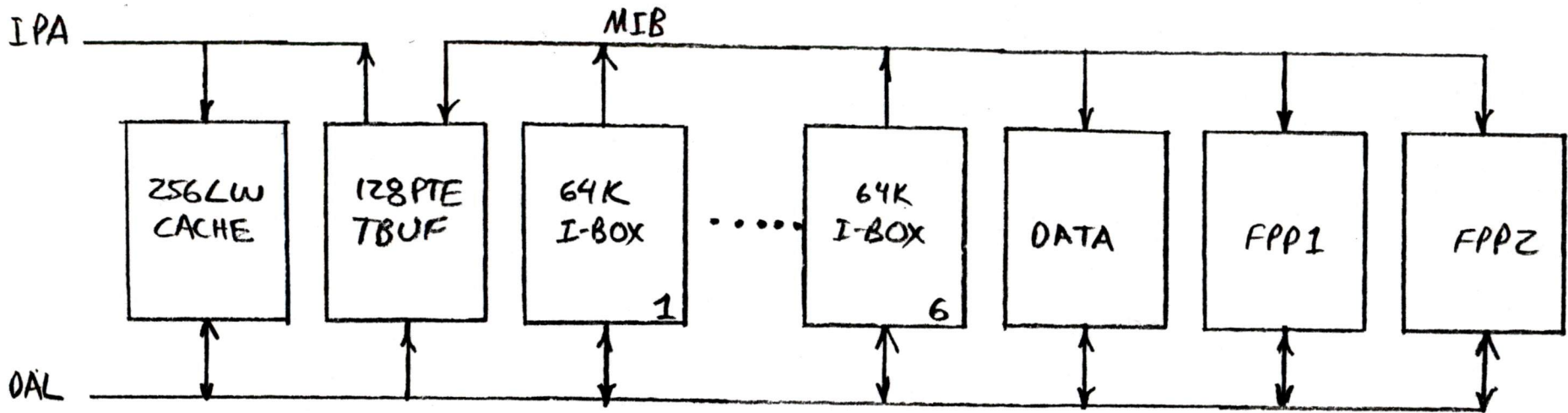


Figure 12  
 SCORPIO - REVISION 1  
 6 CUSTOM CHIP IMAGES

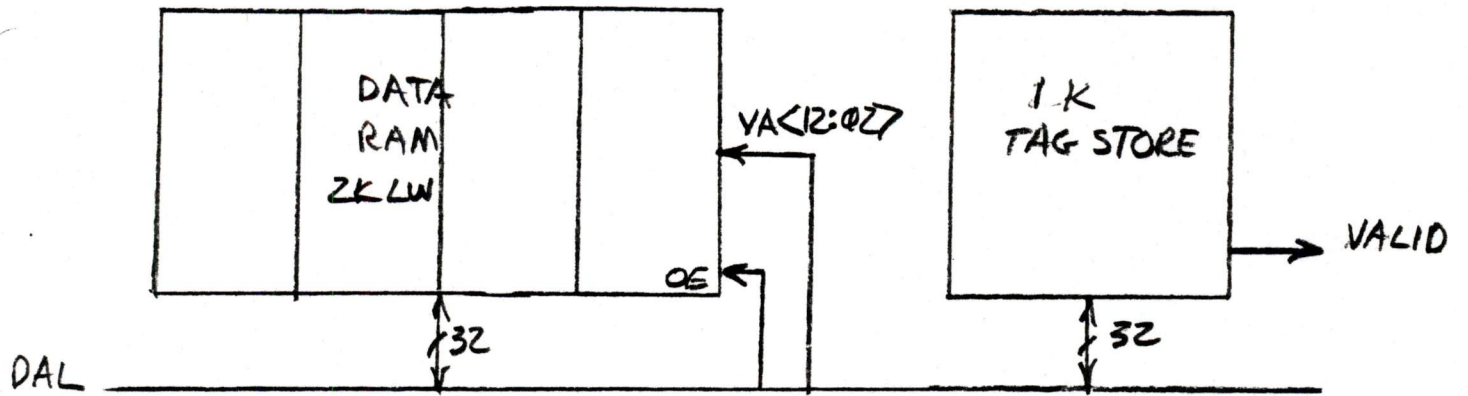


Figure 13

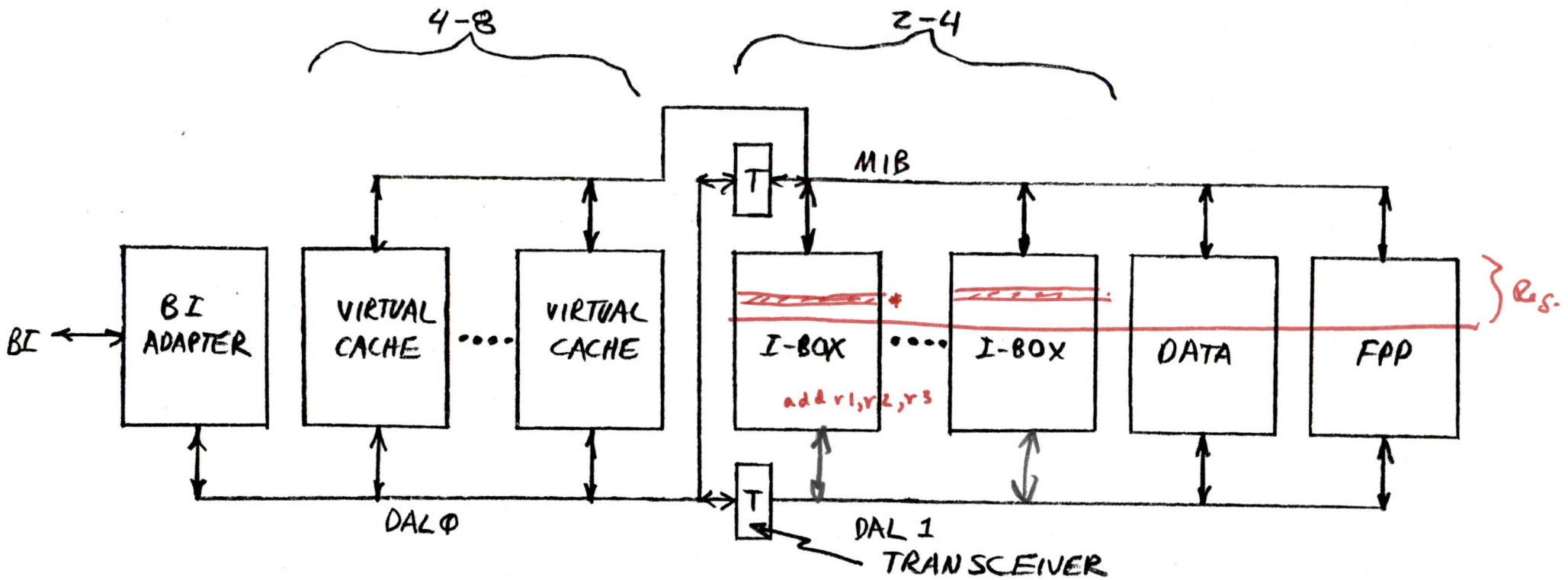
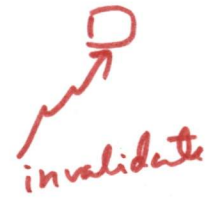
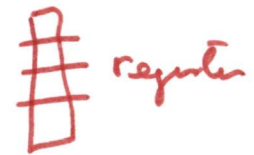


Figure 14

SCORPIO - REVISION 2

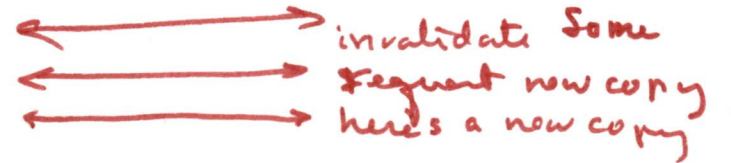
4 CUSTOM CHIP IMAGES + BIA  
8-14 CHIPS + BIA

valid bit



What about PC?

register #



+-----+  
|digital|  
+-----+

INTEROFFICE MEMORANDUM

TO: Distribution

DATE: August 23, 1979

CC:

FROM: Hank Walker

DEPT: Project Scorpio

EXT: 223-5465

LOC: ML4-3/T34

SUBJECT: Scorpio Chip Set Architecture Design Tradeoffs

*Hank Walker*  
*(BB)*

-----  
COMPANY CONFIDENTIAL DRAFT

INTRODUCTION

For the past two months I have been examining possible partitionings and design tradeoffs for implementing SCORPIO. The starting point of this work was the rough partitionings in Figure 1. This partitioning was done strictly on a functional basis with little thought given to chip complexities.

While studying this possible chip set architecture in more detail, I assumed the following capabilities:

- 68 to 84 pin packages
- 7100 micron (280 mil) square chips
- HMOS I process - 4 micron gates, 0.9ns gate delay
- 3 Watts dissipation/package

Based on Intel results, the use of HMOS I will allow the fabrication of 8K static RAMs, or datapaths with the complexity of the 68000 and a 200ns microcycle.

TRANSLATION BUFFER

I examined the internal details of each chip, starting with the translation buffer (Tbuf) because Crais Mudse had already designed this chip using Bristle Blocks at Caltech. Simulations show that Tbufs as small as 32 entries have hit ratios greater than 0.9. I did not consider fully associative Tbufs because it had not yet occurred to me. For performance, I chose the STAR Tbuf organization, the largest that easily fits on a chip. This Tbuf has a set size = 2, a block size = 1, and 128 page table entries (PTE) as shown in Figure 2. The measured hit ratio is 0.97. 4K of RAM is required for the PTEs, 2.3K for the tags, and 256 bits for parity, for a total of 6.6K of RAM. On-chip parity is included because of the possibility of soft errors in high-density RAMs. Besides RAM, the Tbuf also contains comparators, simple control logic, access checkers, and I/O buffers. A total of about 20 of the latter are required for internal physical address (IPA) output and control signals. These use about 6 percent of the chip area and 400mW of power as shown in the floorplan in Figure 3.

A possible addition to the Tbuf is to place the processor registers and logic necessary to handle Tbuf misses on the chip. This speeds up a miss, and eliminates the microcode necessary to handle the miss microtrap. This will only improve performance slightly and may not be worth the extra logic. The Tbuf fits easily on one chip, uses fewer than



68 pins, and has an access time of less than 100ns based on 2147 performance. The chip pinouts are shown in Figure 4.

## CACHE

I next examined possible cache chips. I considered caches with both the data and the tags on the same chip to reduce the chip count and cache access time. I studied a fully associative cache implemented with a CAM tag store instead of the set associative organization used on the 11/70 and STAR. Figure 5 shows that CAM cells use roughly twice the number of transistors and area as RAM cells, so CAM tags use twice the area of RAM tags. Since the maximum area of the chip is fixed, increasing the area of the tags reduces the area of the data, resulting in a smaller cache. Strecker's 11/70 cache simulations show that a fully associative cache has only a slightly higher hit ratio than a set associative cache. The smaller size of a CAM-based cache more than offsets this effect, for a lower overall hit ratio. There is little difference in access time between fully and set associative caches. This is seen by thinking of the CAM tag store as a programmable decoder which selects a word in the data store, so a CAM access takes as much time as a RAM access. Since the two RAM accesses in a set associative cache can be overlapped, there is no difference in access time between the two organizations. The conclusions are that fully associative caches have the same access time and lower hit ratios than set associative caches. The cache will therefore be set associative.

The VAX cache simulator was used to test various set associative organizations. A 256 LW cache was chosen because it is the largest that can be put on one chip. Increasing the set size from 1 to 2 reduces the miss ratio slightly, larger set sizes having little effect. This is unfortunate since increasing the set size only costs the area of more comparators, but supports the 11/70 results that show no advantage of full associativity over set associativity. Increasing the block size from 1 to 2 reduces the miss ratio by 0.02 and from 2 to 4 by 0.02, with less effect thereafter. Larger block sizes also reduce the tag store size. Large block sizes require more data to be fetched from memory on a cache miss. This could be done by multiple memory references, but is usually done by organizing the memory to have the same block size as the cache. In either case, large block sizes increase memory and internal data (ID) bus bandwidth and cache control logic complexity. Since the processor also used the ID bus for data transfers between chips, large block sizes may cause bus contention. The different write strategies and replacement algorithms have little effect on cache performance so random replacement, allocate-on-write, and write-through were chosen as the simplest. The final organization shown in Figure 6 is a 256 LW cache with a set size = 2 and a block size = 2. Simulations predict a miss ratio of 0.3. This is high, but possibly acceptable with fast (400ns access time) primary memory as shown in Table 1. This miss ratio is twice that of the same size 11/70 cache, suggesting that the locality of VAX programs is much less than that of 11/70 software.

Using separate IPA and ID outputs results in a chip of about 68 pins, including over 50 pad drivers. These use about 15 percent of the chip area and 1W of power based on J-11 data. The number of output drivers is reduced by multiplexing the ID and IPA buses. This does not affect

performance because the physical address (PA) is transmitted to main memory on a miss in the same time period that data ordinarily is returned to the processor. Multiplexing also requires modification to the Tbuf IPA output. If the PA and MD buses are not multiplexed, then multiplexing the internal buses will result in additional interface

logic. Figure 7 is a floorplan of the cache and Figure 8 shows the pinouts.

## INSTRUCTION BOX

The Instruction box (I-box) contains all of the logic necessary for fetching instructions, decoding them, and issuing microinstructions to the other chips over the microinstruction bus (MIB). The chip contains the instruction buffer (I-buf), the instruction decode PLA, the microsequencer, and prefetch logic. This chip runs the instruction pipeline.

The problem that immediately arises is that there are too many pinouts. Separate MIB, VA, ID, and microaddress buses result in more than 100 pins, so some of these buses must be multiplexed. Combining the VA and ID buses into a DAL bus seems to be the best choice since they have lower bandwidth requirements than the other two buses. The J-11, T-11, LSI-11, and FONZ-11 also use a DAL bus. Multiplexing still leaves more than 68 pins and requires modifications to the pinouts of the other chips.

An alternate design for the I-box was suggested by the FONZ team. They suggested that the instruction decode PLA and the control store ROM be combined into a ROM/PLA. This can be broken up into several chips with the I-buf, microsequencer, and prefetch logic duplicated on each chip. Assuming 64K ROM chip densities, this takes 6 chips, but only one image. This design also has higher performance because an instruction decode microcycle can be saved at the beginning of each instruction because the ROM and PLA combined has less delay. In addition, the on-chip microcode allows chaining to increase microJUMP speed without any increase in pin count. The PLA/ROM structure also reduces pinouts by eliminating the microaddress bus.

Simulations of VAX instruction streams show that the average instruction is less than 4 bytes long, so an I-buf of 8 bytes, the same length as the STAR, provides complete instruction prefetch. The simulations also show that 90 percent of all loops are less than 32 bytes long. I investigated the possibility of using a longer I-buf so that most loops can be contained in the buffer, as has been done on several 370 models. This is very difficult to do on the VAX because instructions are variable length. The I-buf must detect and save each instruction address in the buffer, and then recirculate the buffer during the loop, but prefetch at the end of the loop. Instead, if the access time of the cache is only 1 microcycle, then it can be viewed as a large I-buf. Having a larger I-buf only saves 1 microcycle per loop, which is an improvement of 5 percent for an average loop length of 15 bytes. A larger I-buf will also greatly reduce memory bandwidth. I decided that this gain did not justify the large increase in I-buf complexity.

I also investigated the possibility of operand prefetching. I restricted it to the cases where memory is not modified, and to the indirect operand specifier modes, such as (Rn) or (Rn)+, and to the literal and immediate modes, which are prefetched automatically. These restrictions prefetch the most common instructions while reducing the required complexity of the prefetch mechanism. Lengthy analysis of PDP-11 instruction and addressing mode frequencies and 11/45 microcycle times shows that at most, a performance increase of 10 to 20 percent is possible. Operand prefetching reduces the number of microcycles per instruction, so there may be insufficient bandwidth on the DAL bus to prefetch instructions, operands, and communicate between chips. Hardware

changes are also required so that the register file in the data chip can be read onto the DAL bus independent of the datapath operation. This also requires more control lines from the I-box to the data chip. Since the next instruction, for which operands are being prefetched, can lie in any I-box, some form of communication is needed between them to locate the next instruction, or each one has to decode the length of each instruction in order to remove it from the I-buf if it is being executed in another chip. If two consecutive instructions will be decoded by the same I-box, then either the instruction decoder and microsequencer must be capable of two independent operations, or no prefetching can occur for the second instruction. If higher chip densities allow the number of I-boxes to be reduced, this problem becomes more severe. Given these numerous problems, I have decided that operand prefetching is not worth the large increase in I-box complexity, unless there is a lot of free space on the chips. This may be the case, for example, if 384Kbits of microcode is required, but only 256K fits on one chip. Then two chips have to be used, each one 25 percent smaller than it can be. This extra space could be used to increase the functionality of the prefetch mechanism. In either case, a comparable performance increase can come more easily by increasing the functionality of the datapaths.

Another prefetch scheme is to prefetch instructions from possible branch targets. Since most instructions are found in the cache, only 1 microcycle can be saved if a branch is taken. A study of VAX and PDP-11 instruction frequencies and 11/45 microcoding shows that a 5 percent increase in performance can be achieved with a 10 percent increase in memory bandwidth. This mechanism has the same effect as containing loops inside of the I-buf, but with an even greater increase in complexity. The problem is the same as in operand prefetching in that instructions must be predecoded while the current instruction is executing. Therefore I don't think that the increase in complexity is worth it, except in the case cited above. Since the requirements on the I-box are similar in the cases of branch and operand prefetching, if one is implemented, then both of them can be with little increase in cost.

Another option is to reduce interchip communication by keeping copies of the register file in the I-boxes and floating-point processor (FPP). If a register file exists in an I-box, then an indirect operand address can be immediately placed on the DAL bus, rather than sending a microinstruction to the data chip that will cause the register contents to be placed on the bus, saving 1 microcycle. PDP-11 instruction frequencies and 11/45 microcycle data suggests a 1/3 savings for the operand specifier decode and 12 percent overall. VAX data confirms these calculations. A copy of the register file in the FPP saves a microcycle for register mode addressing, an improvement of 10 to 20 percent in FP performance.

The multiple copies of registers must be kept up to date. This can be done by transmitting the new value to the other chips each time a register is modified in any of the copies. Either the DAL or MIB bus must be used to update the registers because of pinout limitations. If data is prefetched on the DAL, then the MIB should probably be used for this and vice versa. Notice that now the I-box must have a connection to both the DAL and the MIB. Because it will be seen later that prefetching data on the DAL simplifies the cache design, I think that the register updates should occur over the MIB. The MIB must be 32 bits wide like the registers, which is desirable anyway because wider microcode usually means higher performance and flexibility. Many instructions modify more than 1 register, up to 4. These all have to be transmitted to other

chips. A complex datapath may be able to modify several registers at a time, such as 2 address pointers. Two microcycles are required to transmit the registers which only took 1 cycle to modify. This means that there is insufficient bandwidth to transmit updates over the MIB. The use of nanocode on the data chip, discussed below, may relieve this problem by reducing the number of bus cycles required to transmit microinstructions. Another solution is to examine current programming practice, which is to allocate certain registers, such as 0 to 4 for address calculation, a block for floating point (FP), and some for scratch storage. This means that the FP registers only have to reside in the FPP, the scratch registers in the data chip, and the address registers in the I-box. The I-box can keep track of the location of active registers. When a register is modified on one chip, the I-box checks for a copy in another chip, which requires an update to occur. Because most programs use each register for one function, updates will occur infrequently, reducing bus congestion. Local processing is needed to avoid updates. For example, if all address calculations require the use of the data chip, then there is no advantage in having the address registers in the I-box. A simple ALU in the I-box can perform address calculations so no off-chip communication occurs. However, the several I-box chips may have identical copies of the registers, that must be updated during address calculations. Exception handling must also be considered. This idea should be explored further since it promises a performance improvement of better than 10 percent with a minimum of logic. Programs that use registers randomly may find that they run slower, because more updating is required.

A chip floorplan of the I-box is shown in Figure 9 and the pinouts in Figure 10.

#### DATA CHIP

Little thought has been given to the data chip. Some differences between the SCORPIO and STAR datapaths are obvious. An FPP will be part of the base CPU, i.e., not an option, so the exponent section is no longer required. All integer multiplication will be done in the FPP so this logic can be removed too. A dual-port register file can replace the A and B register files. Part of the address calculation section, such as the prefetch instruction address, can be placed on the control chips. It was not clear how much of the data section, such as the Q and T

registers are needed in the presence of the FPP. A separate decimal ALU might be used for the packed decimal instructions. Another ALU or an incrementer/decrementer will allow multiple ALU operations. Additional ports on the register files and more registers improve performance too.

One concept that I want to try is nanocode, a second level of horizontal microcode on the data chip addressed by the I-box microcode. Higher performance can be achieved if microloops, such as packed decimal or character string instructions are controlled by nanocode. This avoids I-box to data chip communication, except to start the loop. If all conditional microcode instructions are placed in nanocode, then there is no need to transmit the PSW back to the I-box, which requires a MIB bus cycle or a dedicated PSW bus. In order to execute loops, a simple sequencer is required on the data chip. Nanocode allows the use of horizontal microcode to control the datapath internal to the chip, while keeping the external microcode bus narrow to reduce pin count, possibly as low as 16 pins. The amount of nanocode on the data chip will depend on the space available. A theory of how the nanocoding affects the size of microcode is also needed.

The J-11 team estimates that a chip of our size will have 1.5W power dissipation. They estimate that the delays in their datapath, which is very similar to ours, will be 40ns to read a register, 65ns in the ALU, 15ns for the shifter, 15ns to drive the data bus, and 25ns to write a register, a total of 160ns for a microcycle. A 32-bit ALU delay will be about 95ns, which suggests a 200ns microcycle. This figure is consistent with Intel and Motorola results. A diagram of the chip pinouts is shown in Figure 11.

#### FLOATING-POINT PROCESSOR

The floating-point processor has not been examined in great detail. The goal is to achieve very high performance and to support the F\_, D\_, G\_, and H\_ floating datatypes. In addition, I want to use the FPP for all integer multiplication. In order to achieve performance comparable to or better than the STAR, the datapaths will have to be just as complex. With HMOS I, two chips are needed to do this. The connections between them may be a severe bottleneck. The FPP in the STAR contains a 32 x 8 fraction multiplier, an exponent section, a fraction adder, register files, two 36-bit wide buses, and some 60-bit datapaths. A separate 24K nanocode store is also required to implement D and F datatypes. The implementation of these blocks will differ from the TTL version, for example, the multiplier will be a parallel array of adders, similar to the TRW multiplier, rather than the table-lookup multiplier currently used.

Figure 12 is a block diagram of revision one of the entire processor.

#### SECOND PASS

Microproducts is starting a project to develop a 2 micron process by 1983. This process will feature 7 micron wire spacings, 2 layers of low-resistance interconnect, poly loads, and reverse-tub CMOS. The new process should provide more than twice the density and performance of HMOS I. This will allow the fabrication of 32K static RAMs and 128 to 256K ROMs. It should be possible to design a datapath with a 100ns microcycle, based on an assumed 0.5ns gate delay and J-11 data. If the supply voltage is kept at 5V, then maximum chip dissipation may double to 3W. This figure is based on the fact that for a constant supply voltage and chip size, the power dissipation will double as circuit density doubles. We may reduce the power consumption by lowering the supply voltage to 3V. Another requirement of the SCORPIO is that it use the Backplane Interconnect (BI) bus as the memory bus.

The increased process density means that the cache and Tbuf can be placed on one chip. Now this chip accepts virtual addresses and returns memory data, i.e., it is a virtual cache. If the Tbuf and cache are just placed on the same chip, then the VA is used to select a virtual tag in the Tbuf, and the corresponding PTE. This is then used to select a physical tag in the cache, which selects the cache data. If virtual tags are used in the cache, then the address translation, and its serial delay, are avoided. The VA can be used by the cache to look up the data, while at the same time it is used by the Tbuf to check the protection, and to fetch the PA, which would be needed in the case of a cache miss. The previously serial operations now occur in parallel, so the virtual cache is twice as fast as a separate Tbuf and physical cache.

The virtual cache has several problems which did not exist in the physical cache. An obvious problem is I/O writes. On an I/O write, a cache entry must be invalidated or updated to avoid stale data. Since

the I/O system will transmit the PA of the data that was written, it must be translated to a VA in order to look it up in the cache. Some sort of reverse translation must occur in the cache, such as a reverse organization of the Tbuf. Another problem occurs when virtual pages are shared so several VAs map to the same PA and data. This results in several copies of a piece of data in the cache. If a write occurs to one of them, the others must be invalidated or updated. The solution is to allow only one of the entries to be valid at any one time. The J-11 and VENUS will use a virtual cache, so they must also solve these problems before we do. One last problem that the cache must take care of is to invalidate all entries on a context switch, since different processes can map to the same virtual space. It might be that only the process space entries should be invalidated. The new 2 micron process should allow a cache hit to occur in 100ns or one microcycle.

The virtual cache has 512 entries and a miss ratio of 0.15. The use of the BI for a memory bus will mean that the memory access time will be 750ns. With a 0.15 miss ratio, the effective memory read time will be 197.5ns or almost 2 microcycles, which is unacceptably slow. Using the BI means that a larger cache is needed. If an 8Kbyte cache has an 0.1 miss ratio, then the effective memory speed will be 165ns. If a 16Kbyte cache has an 0.05 miss ratio, then the effective speed will be 132.5ns, a 25 percent improvement. Decreasing the miss ratio to 0.03 will gain another 10 percent in performance. The latency of the BI means that a few percent difference in the cache miss ratio means a large performance difference, and that our cache must be at least 8Kbytes, and preferably 16Kbytes.

The actual cache implementation is uncertain now. Its organization will remain unchanged, except possibly using a set size = 1. I require

that all of these organizations involve only one custom chip design, and have an access time of one microcycle. The first possibility was suggested by Bob Surnik and is shown in Figure 13. The cache control chip contains the Tbuf and virtual tags. Its output is a valid signal back to the processor. The data will be stored in commercial RAM chips that always drive the bus on a cache read. This means that this organization will be just as fast as a single chip. In order to avoid a multiplexer or tristating the RAMs, the cache is direct-mapped, which has little effect on the miss ratio. The RAM chips will probably be available in organizations of 2K x 8 and 4K x 8, so that at least 4 chips will be required, regardless of the cache size or chip density. The single controller chip can hold the tags for an 8Kbyte cache. One possible extension to this idea is to make the tag chips pin-programmable so that several chips can be hooked in parallel, each one recognizing only part of the address space. One pin will allow the construction of a 16Kbyte cache, while 2 pins will allow 32Kbytes. The treatment of the Tbuf part of the chip is not clear to me now, perhaps it can be reduced in size, or expanded with the number of chips.

The J-11 uses the MIB bus to move prefetched instructions from the cache to the control chips. This means that the control chips do not have to be connected to the DAL bus, saving at least 32 pins. However it requires that the cache be able to connect to both the MIB and DAL buses. The J-11 solves this problem by including latches at the cache output to either of the buses, or the BI interface. This means that the cache is no longer directly connected to the DAL, which may increase the cache access time, while also adding TTL parts to the design. This problem might be solved by designing the cache chips to contain both the tags and data, say 256 LW or maybe 512 LW. The chip can be made expandable by pin programming. Three pins can support 8 chips for either

a 16Kbyte or 32Kbyte cache. Note that only one custom chip is required. An 8Kbyte cache will use 4 chips at most, one less than a cache with a tag chip. The advantage of avoiding commercial RAM is that larger chips can be built, using more power for higher speed. The pinouts can be varied to suit particular needs, such as having two outputs, one to the MIP, and one to the DAL. This would increase the pin count of the cache chips, but reduce the number of pins on the control chips, and the traffic on the DAL.

## CONCLUSION

The processor can be repartitioned because of the greater chip density. The FPP can be put onto one chip. The number of I-box chips can be reduced to 2 or 4, and more functionality and nanocode can be put on the data chip. This new partitioning is shown in Figure 14. With a 16Kbyte cache, the number of chips, excluding the BI interface, will be between 8 and 14, with 4 different chip images. These chips can be mounted on chip carriers in DIPs like the FONZ chips. Two chips per package would reduce the CPU to 4 to 7 packages, all less than 84 pins. The BI interface will either be a custom chip, or about 30 MSI packages. This means that the CPU, along with the BI adapter and clock logic, will easily fit on a dual extended-height module. SCORPIO has the same functionality as the STAR but twice the microcycle speed, so it should have twice the performance of the VAX-11/780.

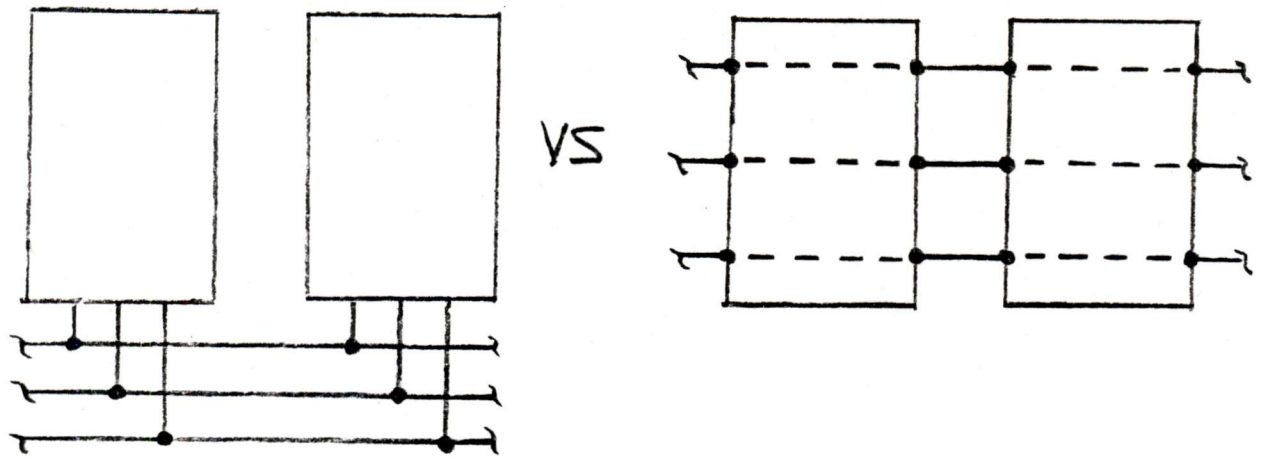
## COMMENTS

While exploring various architecture alternatives for SCORPIO, I found a disturbing shortage of data that I needed to make design decisions. A Tbuf or cache simulator was not available when I made most of my design decisions about sizes and organizations. I had to rely heavily on PDP-11 data. When simulation data became available after I had finished my work, it turned out that I had made the correct decisions about the Tbuf, but that the cache miss ratio estimates were off by a factor of 2. Since the STAR was designed quite some time ago, this data should have already been available. Simulations should not be run after-the-fact. I also had to rely on PDP-11 instruction frequency data, and the VAX data that I finally did get covered only a very narrow range of programs. The documentation on the STAR, COMET, and NEBULA was not very much help either because it is of poor quality or does not exist. The NEBULA notebook is supposed to be good, but it is out of date, and no one is updating it now. I think that on this project, and future ones, much more effort has to be put into documentation. Cheryl Wiecek is working on a VAX system simulator which will allow bus transactions, caches, and the processor at the microcode level to be studied, and should allow more design variations to be explored.

My goals for the SCORPIO design are to minimize the number of custom chips, often by using several of the same type; reduce the number of pinouts per chip because they are expensive in area, power, and packaging; and to minimize the interchip communication because of the 10 to 1, soon to become 20 to 1 ratio between off-chip and on-chip delays. The 2 micron process reduces the number of custom designs by allowing more logic to be placed on one chip, such as by reducing the FPP from 2 chips to 1. Chip types are also reduced by partitioning the design properly so that several of each can be used, such as in the control store and the cache. Several of these chips can then be placed in the same package. Partitioning is also used to minimize interchip communication, such as placing the microsequencer on the control chips, and maintaining multiple copies of the register files. A reduction in

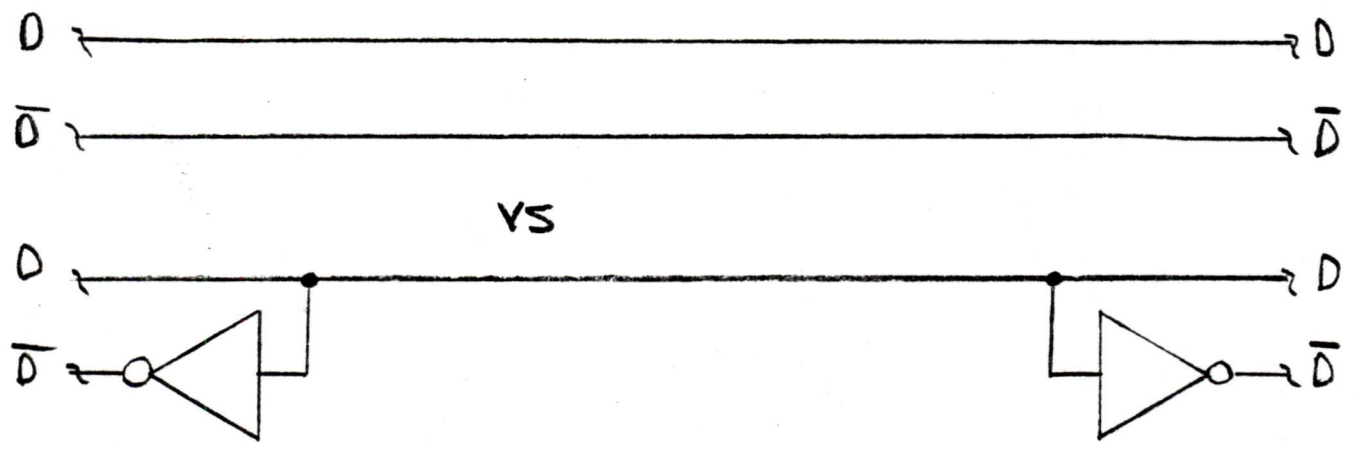
Pinouts is accomplished by multiplexing the buses and by combining related functions onto the same chip.





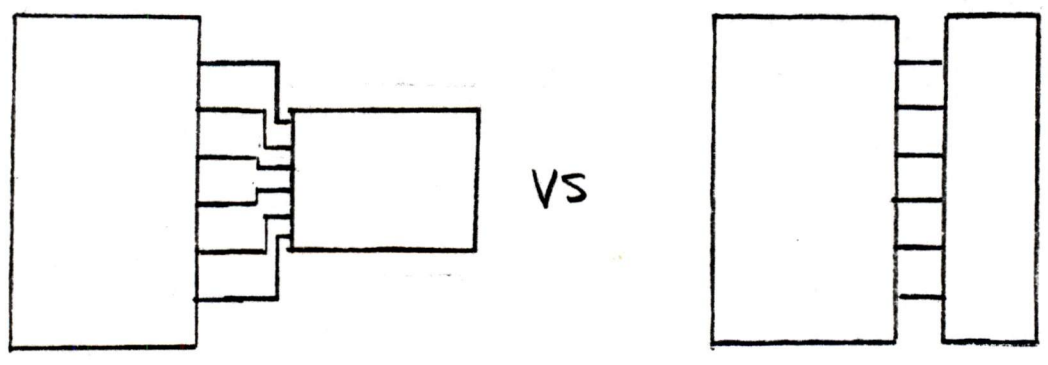
VS

Figure 1



VS

Figure 2



VS

Figure 3

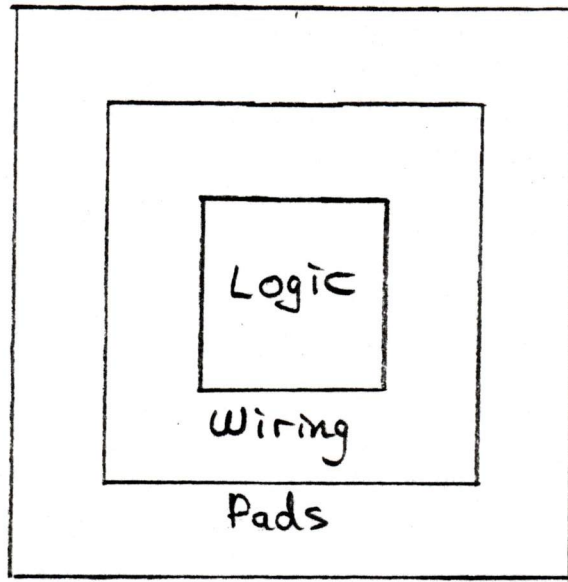
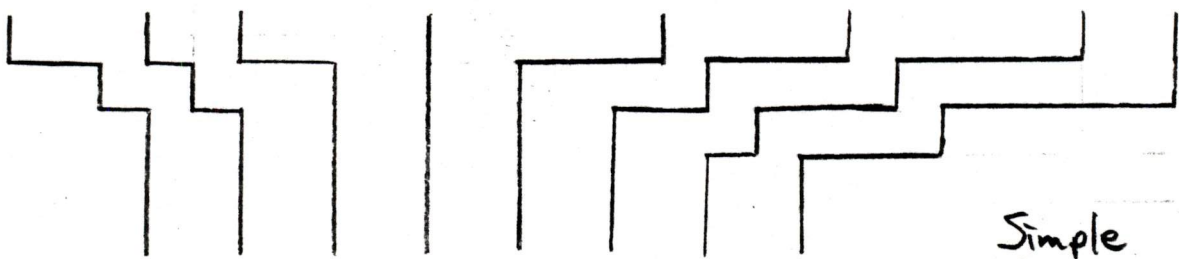
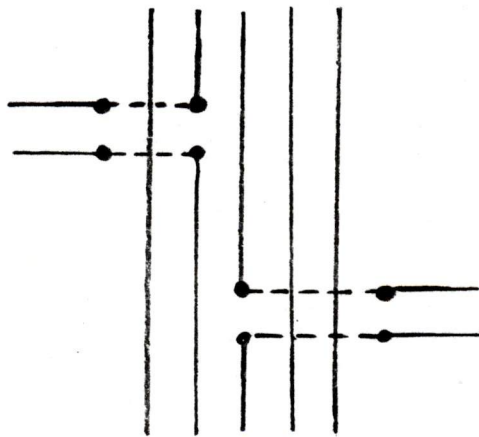


Figure 4

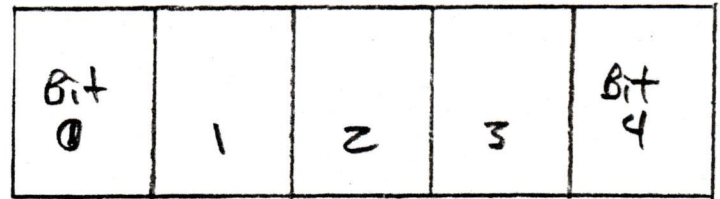
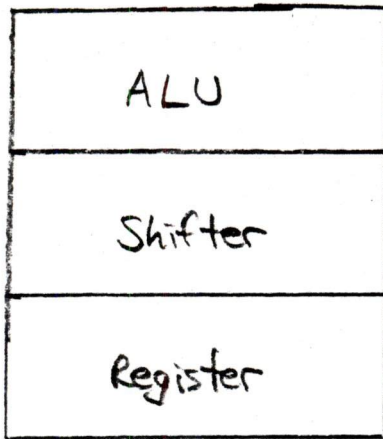


Simple  
Routing



Complex  
Routing

Figure 5



Level 1

Level 2 of Register

Figure 6

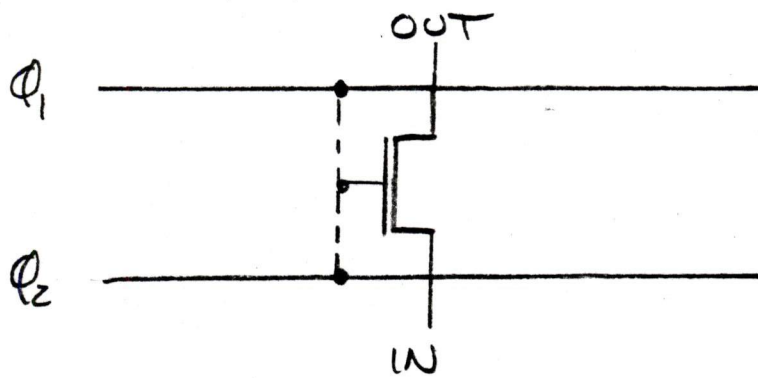


Figure 7

```
FOR i := 1 STEP 1 UNTIL shifterwidth DO
  drawshift(leftnum, rightnum);
```

$$\text{power} := (\beta / Z_{pu}) \cdot V_{DD} \cdot V_{TH0} \cdot \text{regwidth};$$

$$\text{delay} := 90 \cdot C_{LOAD} / Z_{pu};$$

Figure 8

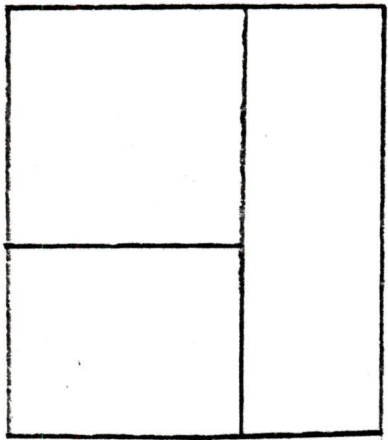
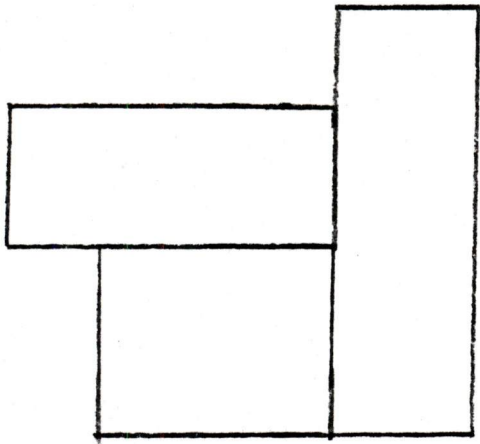


Figure 9

+-----+  
|disital|  
+-----+

INTEROFFICE MEMORANDUM

TO: Distribution

DATE: August 24, 1979

CC:

FROM: Hank Walker

DEPT: Project Scorpio

EXT: 223-5465

LOC: ML4-3/T34

*Hank Walker  
(DBT)*

SUBJECT: Design Methodology  
-----

COMPANY CONFIDENTIAL DRAFT

## INTRODUCTION

This memo represents my thoughts on the design methodology and CAD tools that should be used to implement a microVAX chip set. I will also comment on how these tools should be used and on some possible design styles for microVAX. By design methodology, I mean the design principles that the design team uses while performing circuit and layout design. Today the typical procedure is for the logic/circuit designer to work with the architect to develop a transistor-level description of each chip. The circuit designer then works with the layout designer to generate a layout for each chip. This architecture to layout process usually takes 2 to 3 years, including several iterations.

The chips of the microVAX will be up to an order of magnitude more complex than the FONZ-11 MMU. A large increase in productivity will be required to design a complete chip set in several years. Our goal will be to shorten the design cycle for a chip set from several years to only a few months. This will allow the designers to make many rapid iterations and to explore many different design alternatives. This will hopefully result in an improved final design. This goal will require a new design process.

Rapid design iterations can only be accomplished if the design process is highly automated. The amount of hand design, especially at the layout stage of the process, will have to be greatly reduced. The obvious solution is the use of CAD tools.

The plan that I will outline here recommends the type of design tools that should be built and how they will support our design methodology. Most current design aids only help during layout and circuit simulation. Our tools must integrate both the circuit and layout design. This must include all phases of the design from the chip floor plan to the complete layout. The CAD tools should take care of all of the bookkeeping associated with a design. This will allow minor circuit or layout modifications to be automatically reflected in the entire chip design.

In order to achieve these goals, our design methodology must be both structured and hierarchical. Hierarchical design can be thought of as tree-structured design. A design consists of cells (macros) which are logical groupings of transistors. The lowest level cells would be RAM cells or gates. These cells would be combined to form register cells, which in turn would form registers. Registers, ALUs, RAMs, and other high-level cells would be combined to form a chip. Rapid design iterations can be accomplished if a change in a low-level cell is reflected in higher-level cells in both the circuit design and the

layout.

In order to make the circuit design and layout easier the principles of structured design, which include hierarchical design will be applied. All random logic will be implemented with PLAs (except possibly clock logic), and a regular bus structure will be used as much as possible. The most obvious features of a regular design are many repeated cells, a hierarchy of cells, and a regular layout, which usually means lots of straight wires.

Our design tools will be quite different than existing CAD tools. The reason for this is that we think of our design as having some intelligence, changing to meet the situation, rather than being something static. This idea can be seen by considering the cell descriptions of a design. A cell will be self-documenting. In addition to containing a description of the circuit and layout, it will have information concerning its power consumption, speed, and other parameters. None of these parameters is necessarily constant. For example, a PLA cell could take inputs describing the logic equations which it is to implement. The cell could then change its circuit and layout to produce the required PLA. These cells can be thought of as little programs rather than circuit descriptions. I am aware that many designers are not programmers. For this reason, the design tools will make the creation of cells as easy as possible, avoiding a lot of programming. In addition, the design team will include several programmers.

#### BASIC ASSUMPTIONS

I have been given some basic assumptions to use in developing the design methodology for microVAX, and the plan for implementing it. In addition, I have added several assumptions of my own. The assumptions are as follows:

1. Chip Set Architecture -- A specification of the chip set is complete by the time circuit design and layout begin. This includes partitioning, pinouts, and the internal structure of each chip, such as buses, adders, RAM, etc. Note that structured design principles have to be used by the architect if there is to be any chance of producing a structured layout. Note also that the architecture specification allows the designer the freedom to choose how to implement each function. For example, a comparator could be either an adder or a full comparator. The architect will usually have some ideas about this and can work with the logic/circuit designer to make a particular choice.
2. Fabrication Process -- The fabrication process that will be used to build the chip set is known at the beginning of the program. This includes design rules and process models. The maximum allowable circuit density and die sizes must be available to the architect when the chip partitioning is done. The design rules of the process are required by the tools for generating layouts. The process models are required for circuit simulation. I am assuming that the 2 micron process proposed by Microproducts will be used to fabricate microVAX. Preliminary design rules and process models should be available in time for tool development.
3. Schedule -- The initial schedule that my suggestions are

based upon is 1 year for design tool development and 2 years for chip set design and fabrication. This last item includes 6 months to complete the first pass. Chip set architecture design may be concurrent with tool development.

4. Budget and Manpower -- The budget that I am assuming is as follows: (incomplete)

Tool Development	\$300K	(2 programmers for 3 years)
Chip Set Design	\$400K	(4 designers for 2 years)
	-----	
3 Year Total	\$700K	

5. The Designer -- Perhaps the key assumption that I am making is that the designer will have a broad base of knowledge covering circuit/logic design, architecture, and programming. Knowledge of architecture is not essential, but it would help in the conversion of the architecture to a circuit design. I think that a knowledge of programming, including the principles of structured design, will be very helpful when using the design tools.

## STRUCTURED HIERARCHICAL DESIGN METHODOLOGY

### Introduction

Before I discuss the tools which are required to implement a microVAX, I must describe the methodology that will be used in designing the chips. To me, structured design, like structured programming, means the elimination of randomness. This means that a chip should look more like a RAM rather than a bowl of spaghetti. This implies that the layout is structured, but it should be obvious that this requires that the circuit design be structured too.

Many designers think that a structured design is one in which most of the random logic is implemented with PLAs. This is true, but does not go far enough. A design using PLAs can result in an unstructured layout if an attempt is made to eliminate the holes, especially in a sparse PLA. It is possible to lay out the PLA in a regular manner without wasting much area. Another potential problem is that the busing between PLAs and the function blocks that they are controlling often results in much wasted space. An example of this is the MC68000, which uses a large number of PLAs, but only five percent of the chip area is transistor.

### Wiring

The way to eliminate random wiring is to plan power and bus routing at the chip floor plan stage of the design. Space must be allocated for buses at this stage in order to prevent routing problems when detailed design is done. An alternative to allocating separate wiring channels is to embed the buses in the function blocks as shown in Figure 1. This results in bigger blocks while simplifying the floor plan.

Communications theory and gate-array experience has shown that as a design becomes more complex, the percentage of chip area required by the wiring increases. The only way to control this growth is to reduce the amount of global wiring. The obvious ways to do this are to keep processing and control local. Multiplexing buses also helps. A simple example of local processing is a data bus. A bus could carry D and D' to

all function blocks, or it could only carry  $D$  and generate  $D'$  locally as needed as in Figure 2. This results in only half the number of bus wires, but it costs more logic and a gate delay. The size of the bus drivers can be increased to maintain the same performance. When the added logic is considered, the result will be less power consumption and area. The reason for this is simply that in VLSI transistors are free while wires are very expensive. As dimensions continue to shrink, local processing will become even more attractive. So one of the principles of our methodology will be to minimize global wiring at the expense of more local processing.

Another way to reduce wiring area is to match the pitch of connecting function blocks so that their interconnects are very simple as shown in Figure 3. One measure of the cost of the wiring on the left is that a 90 degree turn of an  $N$ -wide bus takes as much area as  $N$ -squared transistors. If the pitches of the blocks can be varied while maintaining constant areas, then a substantial area savings will occur. So another guiding principle of our methodology will be to match the pitch of connecting function blocks as closely as possible.

No matter how structured a design, there will still be occasions where a bus must be routed from one function block to another. The best example of this is routing to the pad drivers. Rather than trying to contort the blocks to fit the buses to the pad locations, it will be much easier to allocate wiring channels at the periphery of the chip for pad routing as in Figure 4. This means that we will require a wire router. This could be a simple router which involves no crossovers, or it could be a general one such as a gate-array autorouter as shown in Figure 5. So another principle will be to use autorouting for the pads, and wherever a separate space must be allocated for bus routing.

### Hierarchical Design

Our primary goal is a short design cycle. In order to achieve this, the design effort, the amount of thinking that a designer does, must be reduced. One way to do this is to reduce the number of function blocks in a design. This will also reduce the computational requirements of the CAD tools. How is it possible to get by with only 10 or 20 blocks? Most designs have several types of pad drivers, numerous clock generators, and registers, not to mention the buses, which could be considered to be a block. The two methods that we will use are hierarchical design and computable cells.

A design hierarchy can be thought of as a tree structure made up of cells. Each cell is a function block which can be a layout or a circuit description, or any other logical entity. Consider the logical hierarchy of a chip. At the bottom level are cells which describe primitives such as gates, or RAM cells, or even very low-level geometry, such as a contact. These simple cells are combined to form the next level of the hierarchy with cells such as pad drivers, registers, or ALUs. These cells can in turn be combined to form still more complex cells, and the process continues to the top-level cell which is the chip.

Hierarchical design allows the designer to restrict his point-of-view to the level of detail that he desires, and to limit the number of function blocks that he must think about at any one time as shown in Figure 6. This will become even more important as chip complexity increases, not only for the designer, but for the tools. The amount of computation that the design tools must do generally increases



exponentially with the number of function blocks. By using hierarchical design, we can convert this growth to something more nearly linear. This is so because we can apply all of the design methods outlined here to each of the levels of the hierarchy independent of the other levels.

### Computable Cells

Another powerful concept is that of computable cells. Instead of viewing a cell as a static item containing a circuit or layout description, a cell can be viewed as a subroutine. A cell can then be modified to fit a particular situation by passing different input parameters to it. An example of this at Caltech is a clock driver that can generate 17 different clocks. This is done by changing the layout as in Figure 7. Another example that is easy to visualize is a PLA cell. Input parameters could be logic equations, input locations, and output locations. The cell would generate the required PLA. Note that in this case the size of the cell is unknown. At the block layout stage, the area could be estimated. A better solution is to ask the cell. Give it the input terms and it can calculate the area required. This can then be used by the designer or a design tool. Note also that the ability to specify the input and output locations is what is needed to match the pitches of function blocks.

Computable cells can do almost anything because they are really a small program. Layouts can be modified to fit a particular need. Buffers can be enlarged to increase their speed. But layout is only one possibility. Circuit descriptions, including resistances, capacitances, transistor sizes, and net lists can be defined in or computed by a cell as in Figure 8. This information can be passed to a circuit simulator, so in effect, the cell carries its behavioral description with it. Another possibility is that the cell could contain data on its power consumption. This could be used by a design tool to determine the width of the power buses.

All cells do not have to be computable. In many cases, especially primitive cells, an exact geometrical description will be sufficient. These cells can be combined with computable cells because of hierarchical design. A computable cell can have conditional calls to absolute cells. One way of using this is to adjust the pitch of a cell. This can be done by stretching a layout, or choosing a close fit from a library of cells. This allows the layout to vary while preventing wasted area.

### Hand vs Automatic Layout

It is usually the case that 20 percent of the chip area takes 80 percent of the time to lay out. This 20 percent consists of the interconnect between function blocks. The layout of low-level cells such as a RAM cell can easily be done by hand. The cells are small and easy to optimize. Their interfaces to other cells are simple and easy to define. At higher levels of the hierarchy, there are fewer function blocks, but their interconnect structure is quite complex, and hand layout becomes very tedious and error prone. I want our design tool to let humans do what they do best, which is hand design of low-level cells, and to let the computer do what it does best, which is the bookkeeping and rapid iteration which is required to perform high-level layout.

Our design principles already support this. Hand design of low-level cells can be done just like any other cell design. The high-level wirings can be taken care of by an autorouter. However this will not guarantee

that the resulting layout is optimized. The aspect ratios of the blocks in the floor plan that the designer chooses may not be optimum, and we do not want to force him to specify their exact sizes. It has been shown from state-array work that most of the savings in a layout come from optimizations of the floor plan rather than in low-level cells. Our tool should be able to take a rough floor plan from the designer and "flow" it to obtain the best layout. By this I mean that the exact sizes and aspect ratios of the blocks would be obtained by descending the hierarchy. These sizes would be used to obtain a starting floor plan. This layout will almost certainly have holes in it. The tool will vary the aspect ratios of the blocks to obtain a better layout. This can be done by telling the computable cells in the blocks to adjust their pitches. This process can iterate until an optimum layout is obtained as in Figure 9. The designer can look at this layout and possibly make major changes in the floor plan that might produce a better layout. I think of this whole process as similar to heating up blocks of wax and letting them adjust to fill gaps between them. Our goal is to produce a final layout that is no more than 10 to 15 percent larger in area than a completely hand-designed layout.

### Summary

The key points of the design methodology are as follows:

Minimize global wiring - Use local control and computation, even at a cost of more transistors, to reduce long-distance communication paths.

Match the pitch of function blocks - Adjust the aspect ratio of adjoining blocks in order to reduce the wiring area between them.

Embed buses in function blocks - This will allow the designer to use the area under buses for logic, and allows him more control over bus placement, as seen by the lower levels of the hierarchy.

Use autorouting to connect pad drivers and wherever wiring channels must be allocated - This will reduce the design time and introduce another degree of freedom.

Use hierarchical design - This is a basic principle which eases the task of both the designer and the design tools. It is a basic tenant of all CAD work.

Use computable cells - This will make cells much more powerful and reduces the design effort. It also increases design flexibility and makes many of the other design methodology goals possible.

Do only low-level cell design by hand - Let the tool do the high-level layout, which is what it does best. The designer only needs to provide rough size estimates for the function blocks in the chip floor plan.

A CAD tool for use with this design methodology must support each of the concepts that I have outlined above. I will now review current CAD tools and specify exactly what is needed in any tool that will support this methodology.

### TOOL DEVELOPMENT-THE PRESENT

## Introduction

The interactive tools available to DEC from Caltech, such as STICKS or GEOM are not acceptable for building a microVAX. These programs have less than a man-year of programming effort in them and are unuseable by anyone but the author. DEC's CALMA is a commercial quality interactive system, but it will not support our design methodology. The reason for this is that we want to use computable cells, while the CALMA will only allow the creation of static cells. It may be possible to use this feature for some cell definitions, but it will require translation from the CALMA database format to our format, which will probably be Simula. However we may be able to use the CALMOS autorouting software, or at least steal their algorithms.

DEC has quite a few design tools that have textual I/O. Examples are SLIC, PLATO, microcode assemblers, and ISP emulators. Our focus is on circuit design and layout, so simulators such as SLIC, and PLA generators such as PLATO are the only programs that we are interested in. Our tools should be able to generate input for simulators and accept the output of programs such as PLATO.

The experience of Caltech has been that mature design tools that have textual I/O are much easier to implement than interactive systems, requiring only a few man-months of effort before they can be used by designers. Two good examples are LAP, which was written in a week, and Bristle Blocks, which took about 6 man-months of programming to reach useable status. LAP has been used to successfully design 3 chips. Bristle Blocks has been used to design one chip which was not fabricated.

### LAP

LAP deals only with the layout stage of design. It converts a textual description of the layout consisting of rectangles, wires, symbols (cells), and other items to an intermediate layout language CIF2.0. This intermediate form can be the input to a plotting program CIF20P, which is used to check the layout, or to a pattern generation program PAT, which generates the PG tape for the maskmaker. LAP supports hierarchical design concepts, that is, the user describes symbols such as a resistor cell, or an inverter, which can then be used as a primitive in other symbols. In this manner the whole chip is constructed. Because LAP is written in a programming language, (in this case Simula), the user can include small programs inside his cells to modify them for different uses.

### Bristle Blocks

Bristle Blocks is a much more ambitious program. It is an integrated circuit design tool. The name refers to the design style which Bristle Blocks enforces. Users select or define low-level cells, or blocks, which are specified by their internal structure and their interface points, or bristles. Blocks are composed to make larger blocks, with the interconnect work done completely by the Bristle Blocks system.

Drawing an analogy between integrated circuit design and software design, Bristle Blocks is referred to as a silicon compiler. In Bristle Blocks, the designer specifies the blocks and their interconnect structure in a high-level language. Designing ICs by laying down individual wires and boxes corresponds to programming in machine code. A simple artwork language which supports macros (symbols) corresponds to using an assembler. Bristle Blocks qualifies as a compiler because the

blocks it manipulates are not just fixed structures, whose exact positions and interconnect are specified by the designer. Instead, the blocks are parameterized functions whose placement and interconnect are done by the Bristle Blocks system. The blocks are parameterized so that Bristle Blocks can adapt a single block to a variety of conditions, much like a subroutine. For example, blocks can be stretched to make the pitch of their bristles match up with those of adjacent blocks. Also, a block can be specified so that the ratios of its bus drivers change depending on the capacitance of the bus it is driving.

Bristle Blocks is rather restrictive in the type of designs that can be built with it. Specifically, only datapath machines with an external microcode store are allowed. There are other restrictions too. Another major problem with Bristle Blocks is that it is written in ICL, an unsupported Caltech language written in MACRO-10.

## Gass

A goal of Project Scorpio is the advanced development of design tools. One goal is to rewrite Bristle Blocks in Simula, using LAP as a base. This new program, called GASS (Gary's ASSEMBLER), will duplicate all of the functions of Bristle Blocks, including simple wire-routing, cell placement, control PLA and clock driver generation. GASS will try to be more general by being modular, and allowing more user intervention. In order to design chips other than datapaths, the modules of GASS can be rearranged, in effect producing a new compiler for each type of chip.

## TOOL DEVELOPMENT-THE FUTURE

### Introduction

Only one year is available for tool development. This is not much time at all, so we would like to use currently available tools if possible. There are no interactive tools available which suit our needs, so we must develop our own. Unfortunately, interactive graphics tools are notoriously difficult to write, requiring up to 20 man-years of programming effort to achieve commercial quality. Therefore we cannot develop a new interactive tool for microVAX. We will develop a non-interactive CAD tool, with textual I/O, because it will be relatively easy to write.

### Tool Requirements

Our design tool must support our design methodology as described above. Our methodology really requires a tool like GASS or Bristle Blocks, so we really want what I will call a chip assembler. The requirements of the assembler are as follows:

1. The assembler must be implemented in a high-level language. This will greatly reduce development time. It is also essential for the implementation of computable cells.
2. The assembler must support computable cells. This is the heart of the tool which will give it its power. Computable cells will allow the stretching and iteration that is required for rapid design.
3. The tool must support hierarchical design. Cells designed with the assembler must be able to call and parameters to

to and from other cells.

4. The assembler must contain a wire router. This is required for pad routing and occasional random buses.
5. The output for layouts will be in CIF2.0. The assembler must also be able to generate the source file for a circuit simulator and be able to read in CIF2.0 database files. The use of CIF2.0 will allow CIF20P to be used to plot the designs and for PAT to be used to generate the PG tapes.
6. The assembler must contain a box packer. By this I mean an algorithm that can "flow" the function blocks to eliminate wasted area.

These are just the basic requirements that will directly support the design principles of the methodology. This will form a base chip assembler that can be extended by adding modules on top of the base in order to optimize it for a particular type of chip.

The requirements of the assembler sound very similar to the capabilities of both Bristle Blocks and GASS. This is no coincidence. The authors of both of these tools agrees with my design philosophy and incorporated much of it into the tools. But the chip assembler that is desired for microVAX must be more general than this. For example, neither program can support a database of computable cells. It would be nice if the chip assembler could call cells in a database, for example, when it is adjusting the aspect ratio of a block. I have not listed this as a requirement because it would be very difficult to implement, in effect requiring the use of an extensible language. Our assembler will almost certainly be implemented in Simula as an extension of GASS. Simula is not an extensible language. In order to incorporate new cells into a design, it will be necessary to compile them with the rest of the assembler, or to use absolute cells written in CIF2.0, which can be interpreted at run time.

#### Development Plan

I am assuming that two programmers will have one year in order to develop a chip assembler. I am also assuming that they are familiar with the concepts of a chip assembler and are fluent in Simula. Since GASS and Bristle Blocks both took less than half of a man-year to implement, a sufficiently robust tool can be easily written in two man-years.

Rather than attempting to make a completely general tool, I think that the assembler should be modular and optimized to each type of chip that is designed. For example, there will probably be two basic chip types in the microVAX. One type will have mostly memory and some logic, such as a cache chip or a microcode control chip. The other type will be a datapath chip with lots of different function blocks and buses, with some control logic, such as the data chip, or the floating-point chip. I think that modifying the assembler to deal with each type of chip will be the best way to get efficient layouts without too much programming effort.

Architecture development will probably be taking place concurrently with the tool development effort. This will allow for easy interaction with the architects, circuit designers, and layout designers. They can provide sample designs to test the assembler and at the same time can

become familiar with its use. This should save several months at the beginning of the circuit design phase of the project.

## USING THE CHIP ASSEMBLER

### Introduction

All of the circuit and layout design effort will revolve around the chip assembler. The circuit designer and chip set architect first develop a rough circuit specification. The layout designer uses this to obtain an approximate floor plan. The circuit and layout designer then make more detailed estimates of area, speed, and power consumption. These numbers are used by simulators to expose any problems with the upper levels of the design. Finally detailed cell designs are laid out and programmed as computable cells. The assembler then iterates, varying the layout to obtain an optimum design. The designers watch this process and receive simulation results and statistics on the efficiency of the design. They then use this data to make new design decisions.

A major difference between this process and current practice is that exact answers are not needed until later in the process. Much of the layout and circuit design work can proceed using rough estimates or guesses about the design. The reasons for this are that the chip assembler will take care of the little details of the design, and that most of the area savings in the layout will come at the chip floor plan level. If the estimates of speed, area, or power for the lower level cells are accurate to within 20 or even 30 percent, then the chip assembler can take over and optimize the design. The designers will operate in the mode of exploring many designs accurate to within 20 percent. They will settle on this one, and then do detailed design. If the detailed results are not satisfactory, they can alter their high-level assumptions, using the new results, until design goals are met. Note that if an iteration is required, it is very likely that most of the computable cells defined in the previous cycle can be reused, reducing wasted effort.

### The Design Process

The starting point of the circuit design process is the chip specification. The circuit designer must decide what type of circuit he will use to implement each function. The chip set architect will act as a consultant during this phase of the design since he probably had some implementation in mind during the chip specification. The level of detail that the circuit designer will go to is basic schemes, such as implementing the ALU with lookahead carry or a multiplexer as a separate function, or with a tristate bus. He can then work with the layout designer on a first pass of the chip floor plan. The layout designer can usually make rough estimates of the area required for each function, and their overall shape based on past experience. This may be a problem area on the microVAX since it may be the first design fabricated using the 2 micron process. However the design rules should be available for at least several months before circuit design begins. This should provide sufficient time for the layout engineer to get some feel for the areas of different functions.

Using the size estimates of the function blocks, the layout designer can add the power and signal buses, and allocate space for pad routings. If the circuit is designed using the methodology prescribed above, then bus routings will not be very difficult. Instead most of the effort will be in lining up the function blocks and matching their

Pitches. But remember that we are seeking an answer correct within only 20 percent. The chip set architect will have already done this when making his design tradeoffs, so this stage should proceed smoothly, requiring no more than a few weeks.

The circuit designer can now descend the chip hierarchy, and develop the approximate circuit designs of the subcells. Now the circuit descriptions will become more accurate, such as including the actual topology of the circuit, but not the transistor sizes. The layout designer can make more accurate size estimates of these smaller cells.

Performance estimates can also be made for RAMs, PLAs, ALUs, and other blocks. These can be input to a circuit or logic simulator to expose any timing or speed problems in the design. These cell area estimates can be used to reduce the uncertainty in the area estimates of higher level cells. Perhaps a surprise will force the floor plan to be modified, and the design cycle repeated. As the area estimates become more accurate, the box packer algorithm can be used to optimize the floor plan at higher levels of the hierarchy. At each level of the hierarchy, the computable cells can be at least partially specified, such as including the code which processes information that is passed to and from lower level cells, such as power consumption. Recall that at this stage, the programmers will be available to help write the computable cells. This process continues, with backtracking when problems crop up, until the bottom-level cells are reached.

Detailed circuit design and layout must now be done. This does not necessarily mean the complete specification of all transistor sizes or the exact layout. If computable cells are used, this will not be known until the entire chip is assembled. The cells will usually have some default layout, which is close to the previously estimated values. When the assembler adjusts the aspect ratios of the higher levels of the design, it must tell each of the cells in the lower level to change its shape too, in order to get an exact size. New variations are tried until the area is minimized.

The design area is not the only parameter that the assembler can optimize. It is possible for it to meet specified speed or power goals too, or to make some compromise. The assembler allows the designer to watch this process and to intervene. This ability for the designer to provide feedback to the assembler greatly reduces the effort of the assembler and speeds the design task. A good example of where designer feedback can really help is in datapath chips such as those generated by Bristle Blocks. All of the cells in the datapath lie on a common bus. They adjust their pitch to match that of the fattest cell on the bus by stretching parts of the layout. This obviously wastes area. A simple form of feedback would be for each cell to calculate its unstretched and stretched areas. The sum of the difference in these areas is the area wasted in the design. The assembler can also point out which cell is the fattest. The designer can then develop a new layout that more nearly matches the pitch of the other cells.

I have a goal of 6 months for the first pass of the design. I estimate that the effort should be split up as follows. The first 2 or 3 weeks should be spent developing the chip floor plan. The next 3 months should be spent in descending the cell hierarchy and writing the computable cells. The last two months should be spent iterating the design with the assembler. At least one or more passes should be performed per day. One method might be to run the assembler overnight and analyze the results during the day and modify the design for a run the next night. If the modifications are small enough, and the program will

run in only a few hours, it may be possible to do two runs per day. This may be the case in the final phases of the design. So at least 40 or so iterations should be possible.

## DESIGN STYLES FOR MICROVAX

### Introduction

The chip set architecture of microVAX, as of this moment, consists of 5 custom chips. These are a virtual cache, or virtual tag store, two microcode controllers, a datapath, and a floating-point processor. I will give a rough description of each of the chip floor plans and how the chip should be designed using the assembler.

### The Cache

The cache chip consists primarily of static RAM for tag store for the cache and translation buffer, translation addresses, and possibly memory data. Aside from the logic associated directly with the RAM, such as sense amplifiers, and address decoders, the logic consists primarily of the comparators. The number of function blocks on this chip will be quite few, and there will only be several layers of hierarchy. The first level of hierarchy could include comparators, sense amplifiers, address decoders, memory arrays, pad drivers, and buses. Most of these blocks will only have one level of hierarchy below them, such as RAM cells. Note also that it may be best if most of this chip, which is RAM, were defined absolutely with no computable cells. The reason for this is that a RAM cell layout will probably already exist. This allows the designers to determine the areas of most of the function blocks exactly, with the prime unknown being the buses, and the small amount of control logic to handle misses and other exceptions.

Note that we cannot really make use of most of the power of the chip assembler, using computable cells, or complex iterations. The reason for this is that the cache is memory intensive. Memories are the most structured of all functions. They can be hand-designed rather easily, especially if our design methodology is used to eliminate the need to lay out the pad routing and other irregular parts of the design.

### The Microcode Controller

The two microcode controllers contain a large PLA/ROM, an instruction buffer, and a microsequencer. The only difference between the two is the contents of the PLA/ROM. Like the cache, it is easy to calculate the exact amount of area required for the microcode. In our case, we can use the PLA generator to transform logic equations into the layout. The only complex logic on the chip are the buffer and the sequencer. The microsequencer contains the microPC, the JUMP logic, and similar control. At the chip floor plan level, it can be treated as several blocks, because there are only a few layers of hierarchy. The instruction buffer is a regular structure, which only requires 2 or 3 levels of hierarchy, but should probably be shown as one block at the top level because the lower levels are tightly coupled. This will hide a lot of interconnect from the chip floor plan. Like the cache, the chip assembler cannot save much design time, except by laying out the PLA.

### The Datapath

The datapath is a very complex chip and is the first to really use the power of the chip assembler. The top level will probably contain 20



function blocks, such as register files, ALUs, multiplexers, shifters, and control PLAs. Some of the control logic can be imbedded in the function block that it is controlling. There are several floor plans that might be used. The first is one in which sizes are determined for each of the function blocks, and they are laid out using the autorouter to connect up the irregular buses between them. A second possibility is to use one common bus with all of the cells connected to it, such as in the OM or as Bristle Blocks uses. This would greatly simplify the layout and cell design, but it might result in a lot of wasted area. A compromise, which I favor, is to combine the two so that some of the chip is implemented using a common bus, while the rest are separate function blocks. An obvious candidate for a separate block is the register file, which could be attached to one end of the common bus. I think that this chip will involve much more effort than the cache or microcode chips. I think that the way to meet the 6 month schedule is to finish up the simple chips fast, and then those designers can help with the more complex chips. Another alternative is to use several designers on the difficult chips.

### The Floating-Point Processor

The floating-point unit will be the most complex of the chips. It has not been defined yet, but if it is close to the implementation of the VAX-11/780, it will have up to 50 function blocks at the level of multiplexers, ALUs, and registers. It also will contain many wide data buses. I think that the common databus approach is almost mandatory for short design time. The FPP can be broken up into several of these buses with simple communication between them. There will still be several separate function blocks, possibly such as an 8 by 32 multiplier array. Almost twice as much design effort may be required for this chip as the datapath chip, so it might be best to have two designers working on it full-time.

COMPANY CONFIDENTIAL

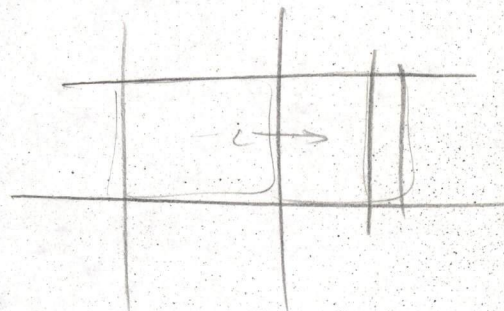
→ Gordon Bell

[4/7/79 DRAFT

-- Final draft due 4/11/79]

FY80 APOLLO 11 ADVANCED DEVELOPMENT  
PROPOSAL

Craig Mudge 4/7/79



## SUMMARY

Funding of \$500K is requested for this two-stage project, an aggressive VLSI design effort.

Stage I, to be completed 12/31/79, will deliver two things:

- (a) VLSI DESIGN TOOL PLAN The hardware and software plan to develop a set of tools for designing the next generation of high density (30K - 100K transistors per die) custom MOS chips at DEC.
- (b) VLSI-VAX-CHIP-SET ARCHITECTURE Accurate chip count showing all inter-chip partitioning; floor plan for each chip; rough performance specs. Supporting design data will go down to circuit and layout design for critical cells.

Stage II is the phaseover of results, ideas and designs to the product development team, which will gearing up for the detailed design/implementation effort in FY81.

The project builds upon seminal work done in FY79 (Jack Burness on a color graphics-based designer workstation; the VAX VLSI task teams led by Dileep Bhandarkar; Craig Mudge's work on architecture and structured chip design; and ideas and software from the DEC-sponsored Silicon Structures Project at Caltech).

The project is highly experimental and has many technical risks associated with it. Because much innovation is required, the group will be small and unstructured. Moreover, the group needs to be kept protected from demands to do fire-fighting for existing projects.

There is a slight chance that some of the results from Stage I may be negative results, e.g., "Defer the implementation until 1983 because we found that 2 micron technology is necessary for a viable MOS chip set implementation of VAX".

The \$500K is split roughly into \$ 350k for people, \$75K equipment, and \$75K contracted expense.

This proposal details the technical items of study, project staffing, budget, lab equipment, deliverables, and risks.

# I. THE DESIGN SPACE

Novel design techniques and architectures need to be developed if DEC is to properly match its low-end product needs with the high transistor densities that semiconductor fabrication technology makes possible. With VLSI, the major problem is the escalating design time, not so much the architecture. With transistor densities of 30K per die (which our Worcester HMOS process is capable of doing) and 100K within a couple of years, existing design techniques cannot cope -- they will yield per-chip design costs of 30 man-years and elapsed times of 6 years.

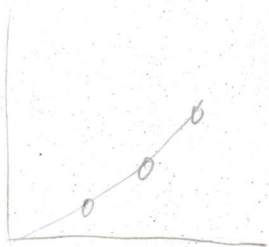
Clearly, new design techniques need to be developed. One promising technique is to describe cell layouts by parameterized program procedures (as distinct from passive data-sets of coordinates). Highly regular architectural blocks (which can be easily replicated across a chip) need to be found.

The design problem is one which is hurting design shops throughout the industry, not just DEC. Intel still cranks out unstructured designs using the traditional sheets-of-nylar approach (with some help from black-and-white graphics). Although IBM has had the gate-array approach mastered for several years in its products using bipolar technology, its capability in custom MOS appears to be behind. HP appears to be ahead of everybody.

At DEC, our design space is different from that at Intel. Since we are not competing at the chip level, chip cost is not our overwhelming concern. We can tolerate the lower chip manufacturing yield that comes with larger dice. Moreover, a computer manufacturer can develop special IC packages which provide better cooling and larger pin outs, albeit at a higher per package cost. We intend to use the extra silicon area and package advantage to give DEC a competitive advantage in several ways:

- a. testability features
- b. shorter time to market  
(structured designs consume more area, but take less time to design and verify)
- c. higher performance  
(by reducing the number of off-chip delays)
- d. higher performance  
(by dissipating more power per die)

*chips vs cache*



## 2. ITEMS OF STUDY

### 2.1 CHIP-SET ARCHITECTURE

Chip design will be carried to the point where an accurate chip count can be given. Based on the chip partitioning work to date, it is desirable that the design has between 5 and 7 chips.

We have have used several techniques to minimize inter-chip communication (the ratio of on-chip to off-chip delay is 1:10 in MOS). They include:

- a. functional partitioning (one chip for each of cache, translation buffer, I-box, E-box1, E-box2, and E-box3)
- b. distributed control (PLA/microcode sequencer and ROM on each chip; off-chip microcode only for less frequent operations)
- c. machine state, e.g., general registers, replicated in each chip
- d. asynchronous operation



## 2.2 A CHIP ASSEMBLER

We will build a prototype software system for hierarchical assembly of cell layouts. Cell layouts will be represented by program procedures (cell definitions which can instantiate themselves as drawings). Functional blocks (also described by procedures) are aggregates of these cells. The final chip is an assembly which links all these blocks together with blocks (from a library) of pad drivers, control line buffers, alignment markers, etc. and the power and ground distribution.

(The term "assembler" points up the strong analogy with software engineering. A long-term goal (not for this project, though) is to develop a silicon compiler. The system we will build is an assembler - mmacro-language translator plus linkage editor and loader).

Central to this approach is the notion that wiring strategy (routing of bus wires, control wires, and power and ground) be done BEFORE the detailed design of the functional blocks. Note that this is the reverse order from the way chips are designed today - blocks are designed first, and then wired together.

*like chip system design*

The main function of the chip assembler is BOOKKEEPING: it allows us to make a basic circuit change (e.g., when parasitic capacitances become better understood as chip design progresses) in a low level cell and then have its geometric impact automatically ripple through the whole chip layout. The rippling would occur in a large batch program run overnight instead of in the several weeks consumed by cutting and pasting of mylar and paper.

A key idea is to use stretchable cell geometries - a cell must stretch, for example, to match the wiring pitches of adjacent cells, or to accommodate a fat power bus (whose width is not known until final chip assembly time) running through it.

I believe that the ability to make many (rapid) iterations on the chip design is crucial. This is only possible if the layout (and other representations of the chip) are in machine-readable form right at the beginning of the design.

As a first step we will install the Bristleblocks system. It is built on top of ICL, an integrated circuit layout language at Caltech. Unfortunately, ICL is written in Macro-10.

The output from this study will be an evaluation of chip assemblers and the selection of an IC layout language.

I am designing a translation buffer chip (to be finished in May 79) with Bristleblocks and hence getting some idea of the strengths and limitations of the approach.

## 2.3 STICKS AND COLOR GRAPHICS

### a. Automatic layout generation and compaction

Given a sticks representation of a circuit (and the process design rules) it is possible to generate a layout. Jack Burness will be experimenting in this area. He hopes to develop algorithms which will be competitive with hand layouts. As a first step, he will install the toy system currently running at Caltech.

HP have had a six-person project on colored sticks (graphics processor, data base design, and software design) for three years now. They plan to deliver the first system this summer.

### b. Sticks as a notation

Sticks is a highly expressive notation. It captures both the function and

4

topology of a circuit, and hence is not only a powerful notation in which to think about design, but also good for communicating between designers. Le Nguyen is using colored sticks (pencil and paper version) for both purposes in the Pusart design in Microproducts. A black and white subset has been successfully used on T-11 according to Rich Olsen.

Given that the notation is central to our work, we need to have sticks editors on our color graphics stations. For example, a designer saves time if he can input his circuit to a circuit simulator as a sticks diagram instead of in some awfully cryptic language describing a net list. In March, Jack Burness entered one of my circuits (an 18 transistor cell of the translation buffer) into sticks and produced an MSINC input file. All I had to add was the code describing the input wave forms and initial voltage levels.

Sticks might also be used as a chip-planning tool (interconnecting higher-level functional blocks) but I don't understand this yet.

#### 2.4 PROCESS TECHNOLOGY - A DENSITY MODEL FOR CHIP PLANNING

There are several basic circuit configurations used (PLA, RAM, pad drivers, gates, etc.) and each has a different area and power requirement. We will select several possible processes (4 micron HMOS, 2.5 micron HMOS II, 2.5 micron CMOS-SOS, for example) and characterize (area, power, speed) each basic circuit configuration. The involvement of people from Worcester and Microproducts, e.g., Dick Spencer, is vital in this part of the study.

We will develop a model of chip area as a function of the basic circuit configurations and area for wirability and power and ground.

The design rules for each process will be simplified (along the lines of Carver Mead's dimensionless lambda approach) to make the model more tractable. Hank Walker has had recent experience with fitting the lambda model to Intel's HMOS. The wirability assumptions will be based on a study of F-11, T-11, 8085, and OM2 chips.

The model will be used to answer such questions as "What MOS channel length is required for us to put both a 32-entry translation buffer and a 256-byte cache on an I-box chip?" Given a technology progress function of process evolution, we can then say in which year such a chip could be built.

#### 2.5 HARDWARE FOR COLOR PLOTS AND DISPLAY

Colored representations are indispensable. By convention we use red for polysilicon, green for diffusion, blue for metal, yellow for ion implant, black for contact cuts. For more complex processes (two layers of poly or two layers of metal, for example), more colors are needed (and so more pencil colors and more bit planes in a display's frame store).

We make extensive use of color graphics displays (about 512 by 512 point resolution), simple plotters (such as the 8" by 17" HP four-color plotter), and large inkjet drum plotters. This hardware is used extensively for visual checks at all stages and levels in the design (from cell layouts, through chip floor plans, to complete chip plots) and checks just prior to mask generation.

Use of graphics for interactive entering and editing of geometry is a different matter. A recent experiment with black and white Applicon terminals at Intel showed that online layout design of a cell can now be done as fast as hand design. Based on this, and on the problems with the geometric editors at Caltech, I believe that it will be some time before designers will interactively enter geometries at a terminal.

However, the colored sticks work described above does use interactive graphics and is very promising.

This part of the Apollo 11 Advanced Development project will make recommendations on the specific displays and plotters to be used for production systems in FY81. The color stations being built in Tewksbury in FY79 should be adequate for the experimentation, but are unlikely to be the choice for production work.

## 2.6 EXISTING LSI CAD TOOLS AT DEC

The layout and circuit design tools discussed above are just one part of the total suite of necessary tools, which includes logic simulators and design rule verifiers, just to name two. Fortunately, most of the suite can be adaptations of existing tools. In this advanced development project, we are emphasizing the tools that are not yet available in DEC, namely those for handling the layout of very large MOS chips.

The complete suite is as follows.

- SAGE logic simulation
- SLIC circuit simulation
- DRC design rule checker
- microcode tools
- PLA code generators
- SCALD/SUDS
- IV interconnect verification
- ISPS simulation
- PG tape generation
- data base management system

Val Patel has committed a person from Microproducts to work on this part of the project, which will evaluate each of the tools. Can it handle the large chips? Is the necessary file conversion procedure available?

This part of the advanced development project will also ensure that NEW tools under development in Microproducts (e.g., the proposed FTA - fast turnaround for MOS) are integrated into the plan.

The project will produce a plan (\$ M of budget and schedule) to achieve the necessary upgrades or rewrites from scratch.

## 2.7 CALMOS AUTOROUTING

The results of the current experiment (funded by Microproducts) on autorouting of functional blocks in T-11 will be assessed.

## 2.8 TESTABILITY

[to be supplied]

## 2.9 AREAS NOT TO BE COVERED

### 2.9.1 Chip packaging technology

An assumption about chip package will be made early on in the project. Today, two packages (68 pins and 84 pins, each capable of dissipating 3 watts) seems to be the best guess. (Hence we will produce two versions of the chip set partitioning).

### 2.9.2 I/O subsystem

[Mike, Duane: how do we get this vital area covered? ]

### 3. PROJECT STAFFING

Duane 6

Craig Mudge	(Project Leader)
Jack Burness	Color graphics; compaction algorithms; stretchable sticks; data structures; SLIC input from sticks
(circuit designer)	Structured chip design techniques; VAX chip set spec
Gary Tarolli	(Caltech graduate spending summer at DEC) Installation of Caltech software; new color graphics software development
Hank Walker	(Caltech graduate spending summer at DEC) Chip-set architecture; density model of process technology
(LSI CAD person)	VLSI design tool plan; evaluation of existing DEC tools; data base
(logic designer)	Board-level-product design; VAX chip set spec

### 4. LAB. EQUIPMENT AND SPACE

Most of the software in this project uses a DECSYSTEM 10 or DECSYSTEM 20. In the long term, DEC will want to convert its VLSI CAD work to VAX because of the address space limitations of 10s and 20s. However, it is essential that we use 10s and 20s in our experimental work so that we avoid an extensive software conversion of Caltech software (all written for the 20).

#### 4.1 Color graphics designer work stations. (\$15K each)

Three of these stations are being constructed by Nat Parke's group in Tewksbury with FY79 funds from Microproducts. Each station consists of

- CRT color monitor (Hitachi)
- LSI-11
- HP 7221A four-color plotter
- VT52 (or equivalent)

#### 4.2 2020 computer system (2020-DC) (\$68K transfer cost)

KS10 CPU, 512 Mwords primary memory, two RM03 disks, Unibus adapter, and line printer.

Note: ownership of this system is not only 2 to 3 times cheaper than buying time from a central facility, but it provides the fast response time essential for interactive graphics.

This configuration is based on a workload study done by Gary Tarolli and Jack Burness in March 79.

#### 4.3 Other facilities

KL10 time for circuit simulation, logic and ISP simulation.  
Applicon plotter time.

#### 4.4 Space

Lab Space



2020 area; 3 color terminals; 3 standard lab benches; storage; 3  
terminal pool; library.  
**Office Space**  
7 offices each with terminal;  
secretarial space;  
small conference room.

7

5. BUDGET (FY80)

PROJECT STAFF

(secretary)	12 man months
Mudge	12
Burness	12
(circuit designer)	12
Tarolli	2
Walker	2
(LSI CAD person)	12
(logic designer)	6

Total 70 man months at \$5K per month \$350K

EQUIPMENT DEPRECIATION

40% of \$75K 30K

CONTRACTED EXPENSES

Mask sets (3 at \$10K each)	30K	
Wafer fab at WX	10K	
KL10 time	30K	70K
	---	

OTHER

Consultants (Dan Dabberpohl, Bill Roberts)	10K	
Travel not covered in o'head	5K	
1/3 of DEC resident at Caltech	25K	
Sponsorship of VLSI research at MIT, Stanford, CMU	5K	50K
	---	-----
		\$500K
		=====

Craig Mudge

4/7/79

[ \* Curt Rowley: could you help me firm up this figure. I need to include FA&T plus field service charges and supplies (disks etc) plus phone installation and charges ]

Mudge  
Nick  
Supnik  
Jud  
gib  
Will  
Rikep  
Bob Kinn  
Sam

+-----+  
| d i g i t a l |  
+-----+

interoffice memorandum

To: Distribution

Date: 31 January 79  
From: Dileep Bhandarkar  
Dept: VAX/PDP-11 Sys. Arch.  
DTN: 247-2021  
Loc/Mail Stop: TW/A08

Subj: VLSI VAX

This memo should serve as the framework for further discussion on VLSI VAX. The product should be available during FY82. At the board level the transfer cost for a single board computer (processor with 256KB) should be less than \$1000. At the system level product transfer costs under \$2,500 are needed for an entry level system (CPU, 256KB, 20MB mass storage, CRT, four serial lines). Tom Northrup and I will pursue product issues further with product lines.

To achieve the above cost goals requires the CPU costs to be under \$200. I feel that the number of chip designs required should be kept to a minimum (less than five). The total number of chips required to build the processor should be less than fifteen. The chip partitioning will be determined by the projected gate densities of the technology chosen. In order to be successful at the system level we need dual removable or fixed plus removable disks with 20MB at around \$1000 transfer cost.

The best in-house technology today is 4 HMOS. The Intel 8086 was designed with this technology. The 8086 has 29,000 transistors and required 184 man-months to build (24 for definition, 60 for design, 100 for layout) compared to 70 man-months for the 8080. Intel expects the next generation processor to require about 320 man-months in 1980. The 8086 was designed with 6,600 drawn devices replicated approximately; one of these (1 transistor ROM cell) was replicated over 9000 times. With device densities doubling about every one to two years, designs must be based on fewer drawn devices. We should be able to use the next generation HMOS (SMOS?) process for VLSI VAX. Dick Spencer's work should give us a better feel for the evolution of our process technology.

We must use graphics terminals for VLSI design. Caltech uses an LSI-11 based color graphics. MIT uses a color and a high resolution black and white graphics terminal. Jack Burness is planning on acquiring a couple of color terminals from Caltech. We need to structure our design more than in the past. I expect future VLSI designs to be based on a library of parts or supercells. These supercells will be of fairly high complexity (ALU's registers, etc). We need to explore ways of expressing these supercell designs so that they can be used over several generations of device geometries. We should explore the feasibility of defining rules for our current HMOS process so that we can express supercell designs that can be scaled down or adapted to projected geometries for the next generation. The other possibility is to use something like the STICKS notation and develop translators to convert them to layout. Supercells stored in a library can be recalled with desired parameters and placed with a graphics terminal. Bob Kusik and I will drive the identification of further refinement and our needs in the CAD area.

I view VLSI VAX as our first large VLSI system project. We should use it as a vehicle to put in place a VLSI design system that can be used to implement other custom MOS chips that could provide us great leverage in our business.

TO: Jack Burness  
Dave Morgan  
Dick Clayton

DATE: December 14, 1978  
FROM: Will Sherwood  
DEPT: LSI CAD  
EXT: 238-2393  
LOC/MAIL STOP: WZ-2

*weo*

CC: Le Nguyen ' Bob Yodlowski  
Spencer Hu Dick Spencer  
Dave Ressler John Maenpaa  
Steve Greenberg Joe Zeh  
Bob Kudik **Gordon Bell**

DEC 20 1978  
12-214

SUBJ: CAD FOR LSI VAX

After attending the first meeting (chaired by Jack Burness) on organizing CAD tools for MicroVAX, I feel we in the CAD community have an excellent opportunity for advancing the state of art CAD tools both in development (enhancement) of current tools at DEC (most likely there's funding) and by bringing in new CAD tools from academia and industry. Finally a project has come to us well in advance BEFORE its start to interact and discuss possibilities instead of coming to us midstream for firefighting aid and aides.

We have breathing time now to think clearly about what our CAD dreams are and what might be feasible for the project. As Ivan Sutherland and Peggy Wesley mentioned at the CAD Symposium, it is CAD tools that ALLOW projects to be feasible, not necessarily technology. Let's not let CAD be a restriction this time.

We're not too proud to bring in outside tools - we can't develop everything ourselves! There are many "free" programs around that could help us, but we shouldn't wait for them to come knocking on our doorstep. Do you realize that SUDS was designed in 1964? It took 12 years for it to finally gain widespread usage within DEC. I believe we can't wait this long for other tools. VLSI certainly can't wait.

For example, a SUDS extension SCALD (Structured Computer-Aided Logic Design-SCALD) was developed by Tom McWilliams and Kurt Widdoes for the Stanford-1 project. These extensions are written in PASCAL/370. The concept has been developed and proven: the SCALD system allowed two people to design and build architecture of 370 complexity in a short amount of time (man-years).

Conversion to PASCAL/10 or BLISS and usage certainly warrant investigation. (Maybe we should have them come and give an in-depth tutorial.)

Perhaps the biggest bottleneck in a custom LSI design at DEC is the layout phase. There's an infinite amount of room here for CAD tools. Perhaps STIX or a spin-off from the Caltech SSP project can be used. (I hear Jack is investigating.) Maybe even a variation of SCALD can be adapted.

Within DEC, there are several new projects underway: SAGE3, VOTE, TUMS, MICRO-ISPS, and others. These projects should be involved with the forthcoming needs and schedule of MicroVAX. Everyone benefits in this way. We are beginning to determine SAGE3's relationship with MicroVAX.

There is a plethora of tools that are at our disposal. I have touched only on a few. Attached is my view of the CAD chart that was presented at the CAD MicroVAX meeting along with applicable tools.

(Attach.)

WS:vg

## CAD TOOLS FOR MICROVAX TASKS

PRODUCT/ DESIGN PHASE	Currently in use at DEC	Extension to current DEC tool	New in-house tool	Proposed in-house tool	Out-house tool (free)	Out-house tool (\$\$)	New Interface needed	TOOL	COMMENTS
Architectural Concepts	x		x	x	x		x	ISPS/TUMS or MIMIC SCALD/SUDS SAGE3	Dev. & Used on Stanford-1 Multiproc.
System Design	x		x	x	x		x	ISPS/TUMS or MIMIC SCALD/SUDS SAGE3	
Microcode Dev.	x x x		x x	x x				ISPS/TUMS or MIMIC MICRO2 SAGE3 MICRO-ISPS PLATO	
Logic Design	x x x		x	x				SUDS SAGE2 SAGE3 PLATO	
Circuit Des.	x			x			x	SLIC SPLICE SUDS	
Layout Design				x x				STIX (SSP tool)	
Digitizing	(x)							(CALMA)	
Testing	(x) (x) x		x x	x x				VOTE SAGE2 SAGE3 MICRO2 TEKIO	
Process			x	x				MODFIT	
<i>Data Base</i>				x					

*Jack Duran*  
*Preliminary Report*



INTRODUCTION

This document is intended to be a combination guide and recommendation of the future graphics requirements for the design of integrated circuits at Digital Equipment Corporation. All suggestions, explanations, and recommendations are the author's as the author now views the situation.

This document is specifically aimed at the time period of five to ten years from today, without losing sight of Digital Equipment Corporation's immediate needs.

Since this is the first draft, the author would appreciate any and all feedback which the readers would care to provide.

WHERE WE ARE

At the current time Digital Equipment Corporation is essentially designing integrated circuits using two principal technologies: MOS and Bipolar. From a general point of overall complexity, a MOS device is in general more complex than a Bipolar Device of equivalent die area.

This additional complexity originates from the fact that certain geometric patterns (e.g. metal interconnections, transistors, etc.) are physically smaller in MOS, and thus for a given die size (integrated circuit chip area) a MOS device will therefore be able to contain a more complex circuit (or more numerous simple circuits) than a Bipolar chip, resulting in more polygons. A polygon (e.g. square, rectangle, etc.) count is a good basic method for determining one aspect of the complexity of an integrated circuit. This is because the construction of the various regions on the various layers of an integrated circuit is always specified by polygons. Thus the final result of any current integrated circuit design, regardless of the CAD tools utilized, will be the generation of polygons.

As an example of complexity, a COMET chip typically contains 35,000 to 40,000 simple polygons involved in just the interconnection portion, while a MOS chip such as the 313 contains approximately 167,000 simple polygons. A simple polygon is defined as being one of the basic shapes (i.e. a rectangle or square). In reality, a chip is layed out using complex polygons. However, from a human placement and routing point of view, a "bent" figure requires much more time than a simple polygon. Thus by giving the number of simple polygons, a more qualitative measure is obtained for determining layout effort. In reality, the 313 contains a total of about 144,000 polygons. Thus approximately an extra 23,000 polygons were produced by breaking the more complex polygons into simple polygons.

Since the physical process to producing integrated circuit chips involves the use of polygons, all specifications at any level of systems design must eventually be converted to polygons. Thus when one talks of "laying out" a MOS chip, one really means getting a to