A DUAL PROCESSOR VAX* 11/780

George H. Goble and Michael H. Marsh

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

ABSTRACT

This paper describes the design of a dual processor VAX 11/780 built at the Purdue University Electrical Engineering School. It covers the conversion of a standard single processor VAX 11/780 into a dual processor system. A detailed description of hardware modifications performed and a parts list are included.

The dual processor VAX is currently running a modified version of the UNIX** (Fourth Berkeley Distribution) operating system. Because of licensing restrictions, operating system modifications will only be covered in general. Finally some performance evaluation will be discussed along with the problems encountered.

1. Introduction

This paper describes the design of a dual processor VAX 11/780 built at the Purdue University Electrical Engineering School.

The primary reasons for this project were:

* A dual processor VAX is much more cost effective than two single processor VAXes.
* Inconveniences to users are less for a dual processor system than two separate single processor systems since users have to maintain only one login account and one set of files.
* The dual processor VAX requires less hardware maintenance since there is less hardware to fail than in two single processor systems.
* The dual processor VAX requires much less software (system administration) maintenance than two single processor systems.

Digital Equipment Corporation's (DEC) VAX-11/780 computer system is one of the more cost effective computers on the market today. Additionally, there are a wealth of second source vendors supplying memory, disks, and other peripherals for VAXes at substantially reduced prices or increased quality. A large "ALL-DEC" VAX-11/780 system will cost between $500,000 and $600,000. If one does self maintenance and makes wise decisions on purchasing "NON-DEC" peripherals, the cost drops to around $250,000 to $300,000 for the same performance. We have noticed unused memory and disk bandwidth (indicating "compute" bound operation) in the VAX configurations running the UNIX operating system at the Schools of Engineering at Purdue University. For about $55,000 (university cost), we have almost doubled the performance (by a factor of 1.9) of a $250,000, self maintenance, large VAX-11/780 computer system by the addition of a second processor. The second processor uses much of the wasted disk I/O and memory bandwidth and almost doubles system performance for only a small fraction of the cost of a second complete computer system.

The Purdue Electrical Engineering School dual processor VAX does not need expensive multiported peripherals or controllers. The standard VAX 11/780 (see figure 1) consists of a 32 bit *Synchronous Backplane Interconnect* (SBI) to which sixteen *NEXI* may be connected. A *NEXUS* (singular for NEXI) may be a processor, UNIBUS Adapter (UBA), memory controller (and memory), or a high speed channel for disks/tapes (MASSBUS Adapter or MBA). Each NEXUS is assigned a *TR* number from one to sixteen that addresses that NEXUS. The UNIBUS Adapter NEXUS facilitates data transfers between a DEC PDP-11 UNIBUS and the SBI. Although slightly slower than a PDP-11 UNIBUS, most PDP-11 UNIBUS peripherals may be connected to a VAX UBA. The MASSBUS Adapter provides a DEC MASSBUS to which DEC disks and tape drives may be connected. Unlike the UBA, only one device on each MBA may have a data transfer in progress, while other devices are doing positioning operations, etc. However, the MASSBUS is more of a burst mode multiplexor for high speed, DMA devices where the UNIBUS is more for the slower, interrupt per word/byte transfer devices. System Industries disks connect directly to the SBI, thereby eliminating the need for the MASSBUS hardware. The SBI is limited in length to three meters or less and is terminated on both ends, one end by the processor and on the other end by a terminator board. *A Purdue dual processor VAX is constructed by replacing the SBI terminator board with another processor.* Since both processors are on the same SBI, all peripherals and memory are shared

This work was supported by the School of Electrical Engineering, Purdue University, West Lafayette, Indiana, 47907.

* VAX, DEC, MASSBUS, DECtape, UNIBUS, SBI, LSI-11, and PDP are trademarks of Digital Equipment Corporation.

** UNIX is a trademark of Bell Laboratories.

0149-7111/82/0000/0291$00.75 © 1982 IEEE

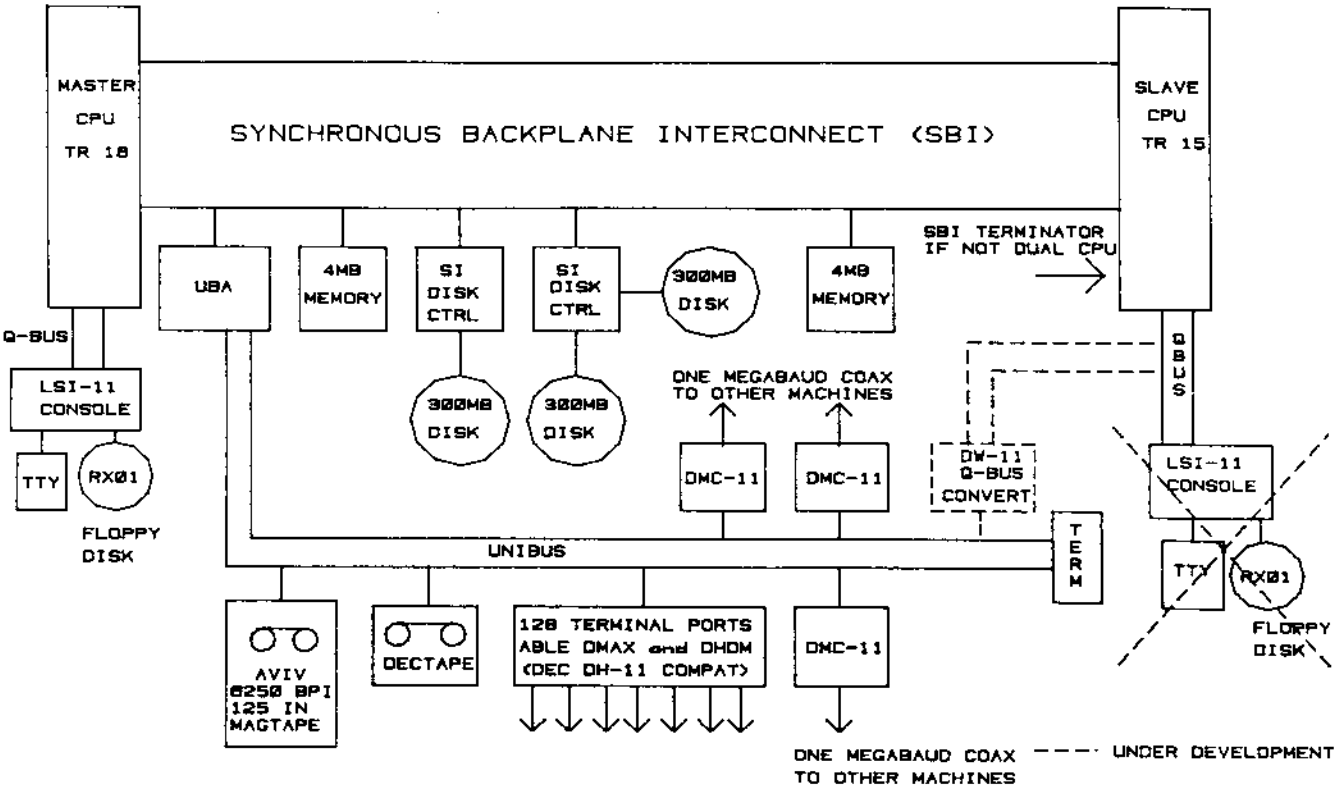

Figure 1 Purdue EE School Dual Processor VAX-11/780.

by both processors, resulting in their higher utilization.

Indirect advantages of the Purdue dual processor VAX include the saving of inconvenience to the users of the system. Compared to two separate systems, users now need only one login account and have only one set of files to worry about. There is much less hardware for a dual processor VAX than with two separate systems, therefore there is less hardware that can fail. From the system software maintenance/administration point of view, only one system need be maintained instead of two.

The rest of this paper discusses the building of the hardware, modifications to the single VAX system required, and problems encountered. In addition, general operating system changes are covered along with results from performance benchmarks and operating experience. It is assumed that readers of this paper are familiar with the *1980-81 VAX Hardware Handbook* from Digital Equipment Corp (DEC) [1].

2. Initial Configuration

The system configuration Purdue started with consisted of a VAX 11/780 processor, eight megabytes of memory (mostly Trendata) in two DEC MS780 memory controllers, two System Industries (SI) SBI disk controllers and three CDC 9766 300 megabyte disk drives. The SI disk systems look like three DEC RM05 disks on two MBAs to the software. One UBA is present. Attached to the UBA are a Telex 6253 magtape drive (800/1600/6250 BPI, 125 inch/second) with an AVIV controller, and eight now Able Computer Technology (ACT) DMAX/16s (software compatible with DEC DH-11's). Three DEC DMC-11 (1 megabaud) high speed serial interfaces connected to other machines, and a DECtape compatible system made by Computer Operations are also connected to the UBA.

3. Hardware Modifications

On standard VAX 11/780 systems, the processor is assigned to be the *TR 16* NEXUS on the SBI. Since one cannot have two NEXI assigned to the same TR number, one of the CPUs TR's had to be reassigned. We chose to reassign the add-on processor (which will now be referred to as the *slave* processor) to TR 15. The slave processor is connected to the SBI by removing the SBI terminator and connecting the CPU to the end of the SBI in place of that terminator. A special set of SBI bus cables is required with signal and ground lines reversed. The reverse is needed since a processor has only an *output* SBI connector whereas the terminator has only an *input* SBI connector with signal and grounds reversed on the input connector. These can be made by reversing signal and ground in a DEC supplied set, or a reversed cable set can be ordered from AMP (see Appendix A).

Two processors are able to operate on one SBI because bus arbitration is distributed among all NEXI

connected to the SBI. In other words, there is no single arbitration unit in the processors as on the UNIBUS PDP-11 systems. The clock signals for the SBI may be generated by either processor, but not both. The clock jumpers on one processor must be removed (we did it on the TR 15 slave). They are jumpers w15 to w22 located on module M8232 (CLK). Removing the clock jumpers also removes the termination on the six SBI clock lines. One must build a clock terminator on the clock board (as we did) or install jumpers on the backplane connecting the clock signals on the CLK board to the clock terminating network on the M8237 (TRS) board. (Note: The clock lines from the SBI go to the CLK board and do not connect to the terminators on the TRS board as the engineering drawings show [2].) We wired these terminating resistors onto the slave's clock card, although it would be preferable to build a card that would fit over the SBI backplane pins, especially if one has DEC maintenance.

To test out the dual processor ideas, we used a new 11/780. Currently, each 11/780 processor has its own console terminal, floppy disk, and LSI-11 computer. This is wasteful, unless the goal is to achieve maximum redundancy. In non-maintenance mode, the only purpose of the slave processor console subsystem, is to load the slave processor's writable control store (WCS) on power up and to start the VAX slave. We plan to eliminate the slave's console subsystem altogether. The 11/780 processor is just a Q-bus device to the LSI-11 console system. It is planned to purchase a DEC DW-11 UNIBUS to Q-bus converter and use it to connect the slave processor's Q-bus to the UNIBUS, thereby eliminating any need for the second LSI-11, floppy, and console terminal. For maintenance, it is planned to make the first LSI-11 switchable between the two Q-buses. Dashed lines on figure 1 show the planned changes to eliminate the second console system.

With the slave processor replacing the SBI terminator, all SBI memory and peripherals are equally accessible to each processor. No expensive dual-ported memory or disk controllers are needed. No problems occur with stale data in cache (data that has been changed in main memory but not in cache) since each processor's cache monitors the SBI address lines and invalidates addresses written into by other SBI devices or processors. Care must be exercised when altering the page table mapping for system address space and switching a process from one processor to another; the "save process context" and "load process context" instructions do not flush the address translation buffer for system address space. The address translation buffer must be flushed manually via a MTPR (move to internal processor register) to the TBIA (translation buffer invalidate all) internal register after a process switch.

4. Software

The operating system used is 4.1BSD UNIX from U C Berkeley [3]. Licensing restrictions prohibit a detailed discussion of UNIX modifications for a dual processor here. A master-slave relationship is used by the operating system. The slave CPU (with respect to the operating system software) does not have to be the TR 15 "slave" CPU. Which ever processor is booted first becomes the UNIX master. Because the master (first booted) processor is started in a normal fashion, it is not necessary to bring up the second processor, therefore allowing single processor mode operation if desired. The slave processor is started by simply issuing a console INIT, depositing the starting address of the slave startup code into the program counter and issuing a console CONTINUE.

The slave processor startup code searches the list of runnable processes (RUNQ) and tries to find the best priority process that is not doing a system call and runs it. Only the UNIX master processor is allowed to process system calls and field device interrupts from SBI devices. Only *user mode* processing is done by the UNIX slave processor. If the process running on the slave processor makes a system call or gets a trap, its context is saved, it is placed back on the RUNQ, and flagged so that only the master processor can run it. It is a difficult task to modify the UNIX kernel to provide correct interlocking in order to allowing multiple processors to execute the same kernel code simultaneously and this is not being considered at this time.

Adding another processor (user mode only) will increase the number of total system calls and traps per unit of time. Since only the master processor handles traps and system calls, the percentage of time in kernel mode will increase for the master processor. Interactive processing (which on UNIX tends to be mostly system calls) will suffer badly if the master processor gets saturated with system calls. Master scheduling was changed such that after processing a system call, the master processor looks for any other processes with system calls pending and services them before returning to user mode execution. This increases the number of system calls that can be processed before saturating the master. UNIX system calls tend to require fixed short bursts of time before completing or blocking (typically .1 seconds or less each), so processing them in a FIFO manner maintains reasonably good interactive response.

Each processor has an internal *Interrupt Priority Level* (IPL) register that may contain a decimal value from 0 to 31. Any NEXUS or other device that can generate an interrupt is assigned an interrupt level between 1 and 31. An interrupt can be serviced by a processor only if the assigned level for the device requesting an interrupt is greater than the current IPL of that processor. Consider the handling of interrupts with two processors on one SBI bus, in particular, what happens to interrupts generated by SBI devices (disks, tapes, etc). Both processors will take each interrupt if the IPLs are below that of the interrupting device. If processor A raises its IPL to greater than or equal that of an interrupting device and processor B *services* the interrupt (removing its cause), then processor A will never see that interrupt, even if its IPL is lowered to zero at a later time. Dual UNIX tries to

execute user mode processes on the slave processor with IPL at 17 whenever possible. This locks out SBI device interrupts from bothering the slave processor, but allows its interval clock to interrupt on IPL 18. Any program can execute an REI (return from interrupt) [4] instruction in user mode and set the processor IPL to zero. When this happens, the slave processor takes the next interrupt (or pending interrupt) which comes along. The user mode PSL (processor status longword) is modified to restore the processor IPL to 17 when the REI is executed. To test the effects of double interrupts, production UNIX was run on the master processor for over three hours with 30 to 50 users. An *idle loop* running at IPL 0 was run on the slave processor. All device interrupts were then fielded by both processors. The slave processor merely counted each interrupt (did not remove its cause) and executed the REI instruction No interrupts were lost by the master processor during this time. Interrupts generated by the interval clock are local to each processor and do not go onto the SBI.

5. Performance

Dual processor performance is affected by several factors. There was enough main memory so results were not overly influenced by paging and swapping to disk, however, the effects of running dual UNIX in a memory starved environment have yet to be determined. Also, there were no major disk I/O bottlenecks. The Purdue EE UNIX kernel currently has 1000 1024 byte disk buffers in main memory to form a *disk cache*. Currently there are two SBI disk controllers that may be transferring data simultaneously. It was observed that a heavily loaded single processor system with 100 disk buffers (more or less the number of disk buffers on a standard UNIX system) typically generated 30 to 40 disk I/O operations per second on each of the disk controllers (mostly 1K bytes/operation). This is beginning to approach the limits of the disk hardware (CDC 9766's) with the limiting factors being disk rotational latency (3600 RPM) and seek time. Increasing the number of disk buffers to 1000 dropped the number of I/O operations to a typical five to ten per second. Adding a second processor is going to generate more disk I/O. A larger disk cache coupled with the 1024 byte disk blocks used by 4BSD (Fourth Berkeley Software Distribution) UNIX (Bell UNIX uses 512 bytes) make the use of Berkeley's UNIX a necessity [5]. The net effect of 1024 byte disk blocks verses 512 byte blocks is an almost doubling in the effective disk I/O bandwidth.

Two processors sharing a single memory are going to generate memory contention and thus reduce efficiency. Each processor has its own cache memory and instruction prefetch buffer that greatly reduce access to main memory. If two memory controllers are present on a VAX 11/780 with equal amounts of memory in each, then they can be interleaved to allow overlapping operations, thereby generating an effective increase in memory bandwidth. The number of SBI transactions (almost all are to or from main memory) has been observed to lie between 350K to 450K per second (running multiuser production) for a single processor system with one memory controller and processor cache in operation. Turning off the cache resulted in the SBI transactions soaring to 1 to 1.1 million per second and a large degradation in system performance. Since it is known the SBI itself can handle about 3 million transactions per second, it is assumed memory became saturated (everything transfers to or from memory) when the 1 million transactions were reached.

Dual processors running with one memory controller (caches enabled) typically showed memory contention nearly 25 percent of the time for multiuser production. The system slowdown from contention was only about 5 percent because of instruction buffer prefetching. Running two memory controllers, two-way interleaved, resulted in almost no contention (around 2 percent). One can write a program that does continuous memory writes, thereby losing all the effect of the processor cache, and causing memory contention of almost 100 percent in a single memory controller dual processor system. Using the MOVC5 (move characters and fill memory) instruction [6] to fill large blocks of memory is a worst case test. If both processors are executing MOVC5 memory fill programs to cause 100 percent memory contention, then throughput is reduced to that of a single processor system. For a dual processor, single memory controller system, the throughput reduction from memory contention is inversely proportional to the processor cache hit rates. Two memory controllers (interleaved) still reduce memory contention to near zero, even when running a MOVC5 fill program on each processor, thereby giving an almost two times increase in throughput over a single processor system.

For most applications, it does not make sense to purchase a second MS-780 memory controller to increase the memory bandwidth. However, there is currently a limit of 4 megabytes of memory that can be installed into a MS-780 controller. To add more total memory beyond 4 megabytes, a second MS-780 memory controller is required. The addition of a second memory controller is expensive, about $28,000, not counting the memory to fill it. Work is underway here to modify an existing MS-780 memory controller to allow it to handle eight megabytes, thus eliminating the need for two controllers.

A benchmark consisting of eleven simultaneously running processes comprising a typical Purdue EE job mix was run under dual and single processor versions. The processes consisted of two Fortran compiles of a 600 line program, one C compile of a 1400 line program, six C compiles of a 200 line program, and two text formatting (nroff) processes of a 1400 line file. Running a second processor produced a speedup of a little over 90 percent (100 percent means a doubling of throughput) in the above job mix with two memory controllers. Running only one memory controller showed between 85 and 90 percent speedup. Variations in the choice of benchmarks, tuning strategies of the scheduling algorithms, and especially the

number of system calls executed by processes can easily move these figures plus or minus 10 percent or so.

6. Programming Considerations

Unless someone (system manager) pays careful attention to the interactive job mix, certain programming styles may greatly reduce system throughput. Heavily used applications programs have been known to make one or two orders of magnitude more system calls than are really needed. If the whole system is full of processes doing nothing but system calls (e.g. doing 1 byte reads from a disk), adding a second processor will only make things worse. A few hours spent by an experienced programmer on modifying heavily used applications programs to consolidate system calls has resulted in an increase of real throughput (real work accomplished) by factors of two to five at our installation. At Purdue EE, a constant monitoring of the job mix for heavily used inefficient (in misuse of system calls) programs with feedback to the people who maintain them has resulted in more throughput than would be obtained without monitoring by two more complete VAX systems. A heavy scheduling penalty is now assessed against processes that do more or less continuous system calls, thereby rewarding those programs that are efficient. Without the feedback, most programmers and students do not realize the system impact generated by moving one byte of data with each system call. After all, a system call just adds one more line to their program from their point of view. With some guidance on how to buffer data transfers and otherwise reduce the number of system calls, a programmer can reduce the system overhead and the impact on performance of programs in general. I/O buffering (reduces system calls) is now automatically used by almost all new software, however, there exists a wealth of older software that does not make use of buffered I/O.

7. Problems encountered

Under some conditions, the PROBER (probe memory read accessibility) instruction [7] induces problems that cause the failure of later (prefetched) instructions. This is even true for normal single processor VAXes doing heavy I/O. If a PROBER instruction resides in memory near the end of a page, and its execution results in an access not valid condition, then during later instruction processing (an instruction fetch from the first byte on the next page, sometimes within three or four instructions of the PROBER) will result in an access not valid condition(page fault) even though the access is valid. The current theory on the cause of the trap is that while a PROBER is executing, the instruction buffer is prefetching across a page boundary and is slowed down slightly because of memory references from other processors or I/O devices, causing the prefetched instruction to get tagged as residing at a bad address. The current fix is to insert seven or eight NOP instructions before the PROBER, forcing the PROBER to reside in the next memory page.

The REI instruction causes a reserved operand fault if one is in kernel mode at IPL 18 or higher and attempts to return to user mode at IPL 17 with the PDP-11 compatibility mode bit set in the user mode PSL. REI works properly if the PDP-11 compatibility bit is not set. Our solution is to do the REI back to user (PDP-11) mode with an IPL of zero or move this process back to the master processor at the earliest chance.

Periodically (about once a week) each of the processors would get a *write timeouts*. The theory was that the peak SBI load into memory was too high and the processor finally timed out during a memory write. To test this hypothesis, an attempt was made to produce the timeouts by placing the system under a load by having both processors executing MOVC5's and both disk and tapes doing 100 block reads. The system performance was severely degraded, but it kept working under the maximum processor and I/O loads we could generate, thereby, proving the theory wrong. The timeouts are now suspected to be the result of a bug in the microcode, but the failure has not been reproduced reliably enough to determine its cause.

Certain long running instructions, like MOVC5s, can be interrupted and resumed later. An interrupted instruction sets the FPD (first part done) bit in the PSL and leaves the current state in the general registers so a later REI will resume execution of the interrupted instruction. In [8] it is implied that a process that has been stopped with the FPD bit set in the PSL cannot be resumed on a processor with a different engineering change level. There is no software to handle this problem yet.

The LSI-11 console processor connected to the hardware slave 11/780 processor gets TRAP-4 timeouts sometimes because the slave processor has no control over the SBI clock. Console commands like $>>>INIT$ when issued on the master's console cause the master's (and SBI) clock to stop momentarily. This causes read timeouts if the slave's LSI-11 is executing a function that requires the clock running. Also both LSI-11 consoles cannot be booted simultaneously since the loading of the microcode requires the clock to be stepped.

The effect of heavy SBI/memory utilization resulting from a dual processor VAX 11/780 on DEC MASSBUS disks is not known. SI disk controllers that have four sectors' buffering internally do not have any DATA-LATE problem.

8. Conclusions

Providing one does ones own hardware maintenance instead of purchasing a DEC field service maintenance contract, it is cost effective to build a dual processor VAX 11/780. The parts cost to add an additional processor to an existing VAX 11/780 is about $65,000 (list price, not counting any discounts). All parts are available from DEC except the reversed SBI cables. See appendix A for parts list.

Adding more than one extra processor on the same SBI is not reasonable, because the length of the SBI would have to exceed its design length of three meters. Also there is insufficient memory bandwidth remaining and extensive hardware wiring modifications would have to be performed.

9. Acknowledgments

We would like to thank Bill Joy, UC Berkeley, for all the work he has done improving the UNIX operating system. In particular, this project would have not been possible without the speeding up of the kernel system call code. PDP-11 UNIX and Bell UNIX/32V typically spent over 70 percent of the processor time processing system calls and interrupts on heavily loaded systems with many users. Bill's improvements have reduced the time spent in the kernel to 50 percent or less under the same job mixes, thus making the dual processor concept feasible. Additionally, he has improved I/O efficiency by switching UNIX from 512 byte disk blocks to 1024 bytes. A heavily loaded 512 byte UNIX would not have been able to support the increased disk load presented by a second processor in our opinion.

We would like to thank the rest of the Purdue Electrical Engineering School hardware staff, Joe Rogers, Curt Freeland, Peter Hallenbeck, and Tom Tengdin for their efforts and help in making this project a success. Dr. Clarence "Ben" Coates (head of the EE School), H. J. and Leah Siegel (EE faculty members), and Bill Simmons (manager of EE School Digital Services) provided administrative and moral support. Thanks to Armando Stettner of Digital Equipment Corporation for reviewing this paper. We would also like to thank all the users of our UNIX system who put up with all the down time and crashes while this project was being debugged.

10. References

[1] Digital Equipment Corp, *VAX Hardware Handbook 1980-81,* Digital Equipment Corp, Maynard, MA 01754, 1981.

[2] Digital Equipment Corp, *KA-11/780 Engineering Drawings,* Digital Equipment Corp, Maynard, MA 01754, 1981.

[3] Bell Telephone Labs and Computer Science Department, University of California, Berkeley, *Unix Programmer's Manual, Seventh Edition, Virtual VAX-11 Version, November, 1980,* Computer Science Division, University of California, Berkeley, CA 94720, Nov. 1980.

[4] Digital Equipment Corp, *1981 VAX Architecture Handbook,* pp. 161-162, Digital Equipment Corp, Maynard, MA 01754, 1981.

[5] Bill Joy, *Changes in the VAX System in the Fourth Berkeley Distribution, November, 1980, Unix Programmer's Manual, Seventh Edition, Vol 2C, Virtual VAX-11 Version, November, 1980,* Computer Science Division, University of California, Berkeley, CA 94720, Nov. 1980.

[6] Digital Equipment Corp, *1981 VAX Architecture Handbook,* pp. 288-289, Digital Equipment Corp, Maynard, MA 01754, 1981.

[7] Digital Equipment Corp, *1981 VAX Architecture Handbook,* p. 160, Digital Equipment Corp, Maynard, MA 01754, 1981.

[8] Digital Equipment Corp, *VAX Hardware Handbook 1980-81,* p. 54, Digital Equipment Corp, Maynard, MA 01754, 1981.

11. Appendix A - parts list to upgrade to a dual processor 11/780

Disclaimer:

A dual processor VAX has not yet been constructed from the following parts (they are all on order). To save time, a new VAX 11/780 was used to provide the add-on processor for the system described in this paper. It is possible something has been overlooked and this parts list should not be construed as absolute and correct. All prices are list, spring 1981, with no discounts applied.

DEC part or Module	cost	Description
M8218	1,845	SBL SBI interface low bits
M8219	1,700	SBH SBI interface high bits
M8220	3,405	CAM cache address matrix
M8221	3,870	CDM cache data matrix
M8222	1,975	TBM translation buffer matrix
M8223	1,470	IDP instruction data path
M8224	1,810	IRC instruction decode
M8225	1,755	DBP data aligner
M8226	1,625	DEP data path bits 31-16
M8227	1,870	DDP data path bits 15-8
M8228	1,705	DCP data path bits 7-0
M8229	1,525	DAP data path control
M8230	1,670	CEH condition codes and exceptions
M8231	1,615	ICL interrupt control
M8232	1,515	CLK clock
M8233	11,770	WCS writable control store
M8234	4,750	PCS prom control store
M8235	1,430	USC microsequencer control
M8236	1,920	CIB console interface board
M8237	1,225	TRS SBI terminator and silo
7013628	1,500	CPU backplane
H7100A	2,916	+5V CPU power supply
7014956	3,706	CPU power supply with -5v
H9602HA	4,460	CPU expansion cabinet
M8217	620	DW-11 UNIBUS to Q bus converter
	$63,652	

```
Alternative method
A-4A-t1780   33,545   VAX 11/780 spares kit
M8221         3,870   CDM cache data matrix
M8224         1,810   IRC instruction decode
M8233        11,770   WCS writable control store
M8217           620   DW-11 UNIBUS to Q bus converter
7013628       1,500   VAX 11/780 CPU backplane
H7100A        2,916   +5V CPU power supply
7014956       3,706   CPU power supply with -5V
H9602HA       4,460   CPU expansion cabinet
              -----
             $64,197

To each of these orders add:

quantity 6, AMP 226475-6 coax ribbon cable,
36 inch @36.54 = $199.24

(order from AMP Special Industries,
 1050 Morse, Elk Grove, Ill 60007).
```

Appendix B: Recent Developments

During the middle of December, 1981, our first dual 780 was dismantled. It was constructed in July 1981 using the processor from a second complete VAX system, just to test the idea. The add-on processor has been equipped with peripherals and memory and is now a separate system. Another VAX processor (in a cabinet by itself) has been assembled from 'spare parts' and is now connected in place of the first add-on processor to form the second dual 780. It is now operational after 3 days of solid debugging to find an intermittant backplane jumper wire. At least two more VAX 11/780 systems will be made dual processor during the next few months.

The LSI-11 console subsystem for the add-on processor has now been replaced by the DW-11 UNIBUS to Q-BUS converter as shown in Figure 1. A user mode C program has been written to provide elementary console support for the add-on processor during UNIX operation. Additionally, the DW-11 has eliminated the problem of one LSI-11 stopping the VAX cpu clock (causing the other LSI-11 to timeout).

The intermittant write timeout errors which occurred on the first dual 780 system were finally traced to a bad SBI cable when the second dual was constructed.

The recently announced (Feb. 1982) DEC dual processor VAX 11/782 consists of two separate VAX 11/780 systems interconnected via shared memory controllers (MA-780). A DEC dual processor VAX with comparable memory (8 megabytes) to the Purdue dual VAX requires four MA-780 memory controllers ($39,500 each). These are not required by the Purdue machine.

Following is a more complete list of parts needed in addition to those specified in Appendix A. Prices are shown if known.

Description	DEC part #	Price
CPU card cage	70-13713-00	est. $50
card guides(60)	12-12405-00	$6.00
stud plate	74-18902-00	
bus bar cover	74-19483-00	
support plate	74-18904-00	
'U' bar (3)	74-18903-00	
H7111 TOD P.S.(optional)		$897.00
backplane cover	74-19458-00	
ac lo cables(3)	70-14212-2M	
SCP cable	70-14243-21	
ov temp cabl(2)	70-14213-0K	
SCP cable	70-11411-0J	$20.00
TODC cable	70-14248-3M	
-5V cable	70-15073-00	
power cable(7)	70-14249-1A	
power cable(7)	70-14530-1A	
sys cntrl panel	54-12932-00	$289.00

Power supply cables - #8 wire, available locally.

Following are a few maintenance and construction tips which were learned during five months of operation of the prototype dual VAX and the construction of the second dual VAX.

Assembling a dual processor VAX

1. Upon receiving our spare board set and the other boards necessary to make a complete cpu, we installed them all in a normal VAX, ran diagnostics, and then Unix for a few days to check them out.

2. Remove jumpers w15 through w22 on the new clock board so it cannot drive the clock signals onto the bus.

3. Install terminating resistors on the clockboard from which you just removed the jumpers a' la' page TRSC of the M8237 board (178 and 121 ohms to -5.2V) OR install jumpers on the backplane of the cpu which is going to be the Unix slave between slot 16(clock board) and slot 1(Terminator board) between EF1-EF1, EF2-EF2, EH2-EH2, EJ1-EJ1, EJ2-EJ2, and EK2-EK2. This connects the SBI clock lines which go normally to the clock card to the terminator instead, since when you remove the jumpers they are no longer terminated.

4. Install the System control panel in the new SBI expansion cabinet with the switch in the LOCAL position and cable power to it and a 10 conductor ribbon cable as in a normal System Control Panel. The key switch would normally turn on the power controller, but we left this unconnected on the add-on System Control Panel and instead took the wire that would normally connect from the SCP to the power controller and ran it from the power controller to the delayed power control bus of the end cabinet on the original system so that turning on the original system will also bring up the add-on processor.

5. Set the TR level of one of the cpus to 15, and put jumpers in the add-on backplane for the standard WCS, PCS options, and SYSID register (near front of KA780 printset).

6. Connect +5 and -5 power supplies and aclo/dclo cables as in the original processor. The fan should be powered from an unswitched outlet so the air sail switch will not kill the system for lack of airflow on power-up. The power supplies should be powered from switched outlets of course.

7. The Time Of Day Clock needs a power supply which can be purchased or you can just parallel the lines from the other cpu's supply.

8. If you are running the add-on as the slave processor, plug in the cables from the DW-11 to the backplane where the LSI cables normally go and plug in the special reversed SBI cables (after checking them for shorts and continuity) and away you go.

8a. If you are running the add-on as the master processor because you have a floating point accelerator, you will need to make a few changes. You will have to put the 'active' clock board in the add-on cpu. You will have to run the real console LSI cables to the add-on cpu and the DW-11 cables to the original cpu. You will also have to run the wires from the floppy control relay over to the add-on cpu and plug them in where they were on the other cpu. If you have the money, you can buy another floating point accelerator and power supply and install it in the add-on unit, although the power supply will have to fit in the other cabinet as there is not enough room in the add-on.

Diagnostics

To run cpu diagnostics, several things are required:

1. The cpu you are running diagnostics on must have the 'active' clock board. Simply swap the boards if you have built the terminator on the board. If not, you will have to swap the backplane jumper wires also.

2. The cpu you are running diagnostics on must be at TR 16.

3. The cpu you are running diagnostics on must have the LSI console. The floppy drive can be plugged in directly instead of through the relay for diagnostics.

4. The cpu you are not running diagnostics on must be disabled so it cannot assert the FAULT line during a certain microdiagnostic test. The easiest way to do this is to pull out the M8218 and M8219 boards. Don't pull out the M8237 (terminator) board.

By changing the above mentioned items, you can run diagnostics on either cpu, and all the peripheral diagnostics will run on either.
GOOD LUCK

George Goble
Mike Marsh