1 Abstract

The turbo PrintServer 20 controller is a performance enhancement of the original PrintServer 20 system Controller. The turbo controller was developed to enable PostScript code to execute faster and thus improve page throughput for complex documents. The RETrACE analysis system was designed to analyze the performance of the original PrintServer 20 system and estimate expected performance of future systems. The turbo controller's processor and its three subsystems for memory, write buffer, and bit-map data transfer were selected based on the analysis results. Performance tests conducted on both the original and the turbo PrintServer 20 indicate the enhanced processing performance of the turbo controller. [The SERVER paper starts here.]

In 1988 the turbo controller project was conceived as a means of extending the life of the PrintServer 20 platform by introducing a performance-enhanced system controller. The system controller in the PrintServer 20 is housed within and powered by the printer or "print engine"; it is a concise implementation of a single-board computer containing a CPU, a memory subsystem, an Ethernet interface, and a printer interface. It supplies an environment in which a multitasking software system manages communications with remote clients and with the print engine, performs data conversion from the page description language (PostScript) to bit-map images, and provides management of physical print engine resources.

The original controller provided a maximum print speed of 20 pages per minute, but this performance could not be maintained when the document included complex text, graphics, or images. To improve page throughput for complex documents, a controller was needed on which PostScript code could execute faster. To enhance performance, the competition was moving toward controllers based on new industry-standard reduced instruction set computer (RISC) processors. Therefore, to be competitive, Digital's new controller was required to improve performance by five to eight

times that of the original controller, which had been based on the rtVAX microprocessor.

As challenging as the performance improvement would be to achieve, budgetary pressures forced restrictions on the implementation strategy. We were to use existing, qualified chips wherever possible in order to avoid new part qualification costs and application-specific integrated circuit (ASIC) development costs.

Early investigations indicated that the performance target was indeed achievable with existing inexpensive RISC processors, as well as a high-speed Digital proprietary VAX processor. A RISC processor would require porting a 2.5-megabyte (MB) software system, which was far beyond the scope of the project. The highest performance VAX processor and the associated support chips, which would not cause a problem with the software system, were far too expensive to be considered. Alternatives were therefore limited to less expensive, lower speed VAX processors: the low-risk, 60-nanosecond (ns) CMOS VAX or CVAX processor was proven, and the higher speed and more cost-effective "system on a chip" or SOC processor was under development. Either choice would have a minimal impact on the software system and would provide a cost-effective solution.

The original performance estimates for the CVAX and the SOC processors in general-purpose processing environments were below the lower bound of the performance target. The design team was also uncertain of the actual execution characteristics of the PrintServer software. For these reasons, it was decided to begin the project with a performance analysis of the original controller to determine the expected performance of a design based on either processor.

This paper discusses the problems encountered during our analysis and the solutions devised by the Hardcopy Systems Engineering Group to overcome them. The RETrACE tool suite, a performance analysis system, is described and analysis results

are provided. The paper then discusses the hardware architecture of the turbo controller and ends with a presentation of the performance results obtained for standard PostScript benchmarks.

2 Performance Analysis of the Original Controller

The PrintServer 20 software system consists of a VAXELN operating system, an Adobe Systems, Inc. PostScript interpreter, and a substantial amount of software to manage communications and resources. The task of analyzing its performance was complicated by two additional factors. First, the software system's behavior depended on the characteristics of the user's PostScript document. PostScript is an interpreted programming language. Thus, like any computer program, low-level machine performance can be dramatically affected by the program being executed. Second, and more painful, the proprietary nature of the PostScript interpreter prohibited us from obtaining code sources, and discussing its internal architecture with engineers from Adobe Systems.

While the characterization of a complex, partially proprietary, real-time software system is difficult, it is not impossible. Programmer counter address (PC) traces have offered many systems designers very detailed insight into the execution performance and characteristics of systems. PC traces provide a means to observe a system at a macroscopic level, allowing a view of the complex interactions between the hardware and software systems. System designers can use captured address traces from current machine performance to extrapolate expected performance of future systems and help them make architectural trade-offs.

The RETrACE Analysis System

The RETrACE tool suite was created to provide a nonintrusive means of capturing real-time PC traces and analyzing the captured addresses. The tool suite consists of both hardware and software components.

In order to keep expenses at a minimum, existing hardware was used wherever possible. Only one small module had to be developed to complete the RETrACE hardware platform.

The RETrACE hardware consists of the following:

- Two interconnect boards boot and operate a system controller on a table top. Developed as part of the original PrintServer 20, the boards connect the controller to a print engine and an Ethernet.

- The PrintServer 20 server controller was modified for use as an intelligent trace buffer system.

- The PrintServer 20 server controller's memory capacity (12MB) was extended using the standard 4MB memory module used on the Kanji version of the PrintServer 20.

- The RETrACE mother board was developed specifically for this tool suite. It contains a 32-bit wide, first-in, first-out (FIFO) buffer and two loosely coupled state machines.

- A standard PrintServer 20 system controller and print engine were used as the "system under observation."

- The console terminal was selected from the standard VT series of terminals.

A diagram of the RETrACE hardware system is shown in Figure 1.

The RETrACE mother board performed the data capture, using the modified controller's memory as a large buffer. The board monitored the processor bus of the system under observation by copying all addresses and communications between the rtVAX processor and its external floating-point unit. This copied data was placed into a FIFO buffer that in turn was written into the memory of the modified controller using a direct memory access (DMA) device. Since a standard PrintServer 20 controller and its optional memory expansion provide 16MB of storage, approximately 3 seconds of real-time execution address traces could be captured. The data capture continued until the trace buffer

memory was exhausted, at which point the data was uploaded over a network connection to a VAX VMS computer for analysis.

Due to the design of the original PrintServer 20 system, many large data areas and code sections were mapped into different explicit memory spaces. This subdivision provided a means of determining which code function was executing in any given segment of the address trace. With a simple statistical study it was possible to generate software execution histograms and to determine many of the characteristics of the system, including translation buffer, floating point, instruction stream (I-stream) versus data stream (D-stream), read versus write, and interrupt performance. Hit rates for fully associative caches of separate I-stream and D-stream, as well as a combined I- and D-stream cache, were also provided. These hit rates were determined for first-level write-through caches from 128 bytes up to 256 kilobytes (KB). Thus an upper bound for an optimum-performance cached memory system was determined.

Both processors under consideration possessed the ability to access a memory subsystem at speeds greater than that achievable with existing low-cost dynamic random-access memory (DRAM) technology. The performance numbers predicted by the processor groups indicated that cached memory subsystems were required. Because these subsystems can be expensive and their performance is subject to the peculiarities of the software that executes on them, a multilevel memory simulator was developed to allow accurate studies to be performed on proposed cache architectures.

The simulator was configured at run-time to simulate an arbitrary hierarchical memory system that was N levels deep, with an arbitrary size, associativity, performance, and behavior at each level. The memory level nearest the processor was defined as the first level, and the last as main memory. The simulator processed a trace file by walking each address in the file through the memory hierarchy starting nearest the processor at the first level. If a copy of the address was found at a given memory level, then a hit was signaled and the next address was processed. If that address was not found, then a miss was signaled

and the simulator would proceed to the next level of memory in the hierarchy.

Whenever a hit occurred at a given level, it was logged and all levels of memory in the hierarchy above it would allocate entries based on their defined allocation rules. While this procedure indicated the memory system performance for a proposed architecture, the overall system performance was still unknown. Using a simple rule based on the average execution time per address for the existing controller, and scaling that time based on the clock speed increase of proposed processors, an overall performance number was estimated for a system based on either processor with any arbitrary memory architecture.

Benchmark Selection

The RETrACE tools suite provided the components required to study the execution characteristics of the PrintServer system without changing the characteristics of its normal operation. The only difficulty was to narrow the focus of the benchmark list to provide a representative sample of PostScript documents to print. Due to time constraints, the list was limited to five benchmarks.

BM1 The BM1 benchmark stresses those aspects of the system that convert the mathematical representations of characters to bit-map representations, which comprise the form that is printed. This benchmark uses several fonts in standard character orientations, stressing both very large and small character sizes.

BM2 Of the same type as BM1, this benchmark stresses the transforms from mathematical to bit-mapped character representations; however, the characters printed are at arbitrary orientations with sizes ranging from typical to very small.

BM3 The BM3 benchmark is one of the standard benchmarks for PostScript performance qualification. It is a simple 41-page document that contains several different fonts. The benchmark is designed to characterize the standard text-handling performance of a printer. This benchmark is printed twice to ensure that all characters to be printed have been converted from mathematical outlines to bit-map representations of the characters. Thus the focus of the benchmark is to move the text data through the system, to copy the character bit maps to the 1MB region in memory that contains the image to be printed, and to print the image. It should be noted that this is the only benchmark that printed at engine speed on the PrintServer 20 system controller powered by the rtVAX system.

HOUSE A binary image file, the HOUSE benchmark was used to stress the communications aspects of the PrintServer system.

SCHEM The SCHEM benchmark was a vector representation of a logic schematic. This benchmark was used to stress the PostScript interpreter's ability to interpret nonnative PostScript code and to exhibit the characteristics of drawing vectors.

Analysis Results

The thrust of the analysis was to provide credible evidence to support architectural and implementation trade-offs. The major areas of focus were

- Memory system organization
- Printer interface performance
- Main memory bandwidth
- Overall system performance

Memory System Organization The statistical analysis of the trace information provided many clues to direct our investigation toward the optimum memory system architecture. The overall read-to-write ratio for the observed benchmarks ranged from as low as 4.3:1 up to 5.5:1, which means for a write-through cache system with a theoretical 100 percent read hit rate, the writes would degrade the overall hit rate to approximately 81 to 84 percent. As the analysis of the data progressed, it was understood that the write data must be studied very closely since it could have a dramatic impact on the overall cache miss rate. During the cache model simulations, the hit rates of the I-stream were between 85 to 90 percent. However, the D-stream hit rates were between 35 to 45 percent, with writes accounting for 60 to 90 percent of the total D-stream misses. To achieve the greatest positive effect on the hit rate of the system, enhancement of write-miss performance was the most advantageous. The two options to improve this performance were either to implement a write-back cache or to add a write buffer to the system. Further cache simulations showed that a write buffer would provide an 8 to 16 percent overall system performance improvement, which was equal to that of a write-back cache. The write buffer, however, was the more straightforward solution to implement.

Cache analysis revealed that the processors required different memory architectures. The CVAX had an internal 1KB, two-way set associative cache. This was to be configured as a mixed I- and D-stream cache. An additional 32KB to 64KB, two-cycle write-through cache was to be added externally. This would also be configured as a mixed I- and D-stream cache. A single-longword, two-cycle write buffer would provide enough buffering to reduce the dramatic impact of write misses. The SOC was proposed to have an internal write-back cache between 5KB and 8KB, with each 1KB region making up a single set. Cache simulations indicated that with a minimum internal mixed I- and D-stream cache of 5KB, five-way set associative, an external data cache would have to be over 64KB to have even a negligible effect on overall system performance. Therefore no external cache was recommended. To

mitigate the write-miss penalty, a two-cycle write buffer of 4 to 6 longwords was recommended.

As an acceleration technique, the original PrintServer 20 controller contained a memory access capability that allowed data written to memory to be logically ORed with data that was already stored. This technique was particularly useful when the software system was writing the image that was ultimately printed. As part of the process of generating an image to print, the individual characters appearing on a page must be copied from a region of memory called the font cache to another region called the frame buffer. The frame buffer contains the actual data that is sent to the print engine. To complicate things, the data written to the frame buffer must be able to overlay data that may already be there, thus requiring a logical OR function.

When a document was printing at or near the maximum engine speed of 20 pages per minute, analysis showed this low-level copying function consumed approximately 20 percent of the total system time allotted to generate and print one page. Thus a logical OR function in the memory system would reduce the number of memory data cycles from "2 reads 1 write" to "1 read 1 write," and reduce the impact from a second read occupying a useful cache location. Without this capability, the degradation would be between 5 and 10 percent of overall system performance when printing at or near 20 pages per minute. Therefore memory capability with a logical OR function was recommended.

Printer Interface Performance When a PrintServer 20 is printing, every page that exits the printer requires the 1MB frame buffer to be copied from memory to the print engine interface. Changing a program-controlled printer interface to one driven by a DMA device provided two significant advantages. The first was to reduce the real-time requirements on the PrintServer software system, and the second was to allow for a limited degree of parallelism on the controller. The parallelism was due to the ability of the processor to continue to execute from its cache memory system while the DMA device accessed memory. The processor only stops executing when a cache miss occurs.

Main Memory Bandwidth With a CVAX processor configured as recommended in the section Memory System Organization, the main memory system bandwidth requirement of the processor was 60 percent. For the SOC, it was 70 percent when an existing DRAM controller was used. A DMA-driven printer interface required 15 percent, and an Ethernet interface required nominally 4 percent with bursts up to 20 percent. Each subsystem was scrutinized to reduce its required memory bandwidth. The resulting recommendation was to add a 32-bit bus to the memory subsystem to provide a dedicated channel for all data being sent to the printer interface. This provision would reduce required memory bandwidth for the printer interface from 15 percent to about 7 percent. The system would then have a nominal memory bandwidth requirement of 71 percent for a CVAX system and 81 percent for an SOC.

Overall System Performance The execution characteristics of the original
PrintServer 20 provided some interesting surprises. Most floating-point calculations were performed in double precision; and even more interesting, for each floating-point operation, there was a floating-point conversion from single to double precision, and then back again. Since the precise operations were not required, a simple compiler switch removed the conversions and provided a 3 percent overall system performance improvement for floating-point-intensive PostScript documents. A second surprise came from the results of the BM3 benchmark, which indicated a translation buffer hit rate of 85 percent. At the time of the discovery, the PrintServer 20 was configured with a standard MicroVAX processor; however, by substituting an rtVAX, which uses one less memory access to reference its page tables, an 11 percent system performance improvement was achieved. With this improvement, the rtVAX processor provided enough power to allow the original PrintServer 20 to ship with its 20-page-per-minute designation. This information led the turbo controller designers to determine that the translation buffer of the SOC would be large enough for all the entries required.

Results

The final analysis revealed that the expected performance of a CVAX or SOC processor would place either design on the low side of the performance requirement. Therefore close attention to detail would be required during the implementation phase of the project as every ounce of performance mattered. The expectation was to have a choice between an SOC processor with a 40-ns cycle time and a CVAX processor with a 60-ns cycle time. The performance improvements of the two processors are compared in Table 1.

As the project schedule progressed, the risk associated with the new SOC processor decreased. As this risk window collapsed, it was understood that a turbo controller based on the SOC processor would not only perform better, but would also cost less as it would not require an external cache.

Table 1: Performance Improvement Relative to Original PrintServer 20 Controller

Benchmark	SOC Procesor	CVAX Processor
BM1	4.7	3.7
BM2	4.9	4.0
BM3	4.3	3.3
HOUSE	4.9	4.2
SCHEM	4.7	3.7

3 Turbo Controller Hardware Design

The turbo controller was destined for a relatively high-end printer. Therefore the hardware architecture had to provide maximum performance, even though this implementation would increase costs. Based on the results obtained during RETrACE analysis, the hardware design had the following implementation goals:

- The SOC would provide the CPU, the floating-point accelerator (FPA), and the cache subsystem. No second-level cache would be implemented.

- A four- to six-entry write buffer would be implemented.

- The transfer of bit-map data to the print engine would require a 32-bit DMA subsystem with scan-erase capability.

- The memory subsystem would support OR-mode memory access by the CPU and scan-erase access by the DMA controller.

Although both the SOC and rtVAX chips comply with the VAX architecture standard and both are conceptually very similar, they have significant differences in the bus interface. For example, the SOC uses a quadword cycle (one 32-bit address followed by two 32-bit data reads) to fill one internal cache block, while the rtVAX processor, which does not support caching, does not use this type of cycle. Also, the clocking system on the SOC was enhanced, and the timing relationships between signals were modified to improve performance.

The changes to the SOC bus interface, plus the required functional changes revealed by RETrACE analysis, meant that very little of the original PrintServer 20 controller design could be applied to the new controller. One of the first questions to be answered before the design of the turbo controller could begin, was whether or not one or more ASICs would be required for the design. This question had to be answered for three subsystems:

Main memory
Write buffer

In each case existing chips satisfied some of the requirements for the subsystem. In the end these chips met all our requirements, but only because they were used in ways not originally intended by the chip designers.

Main Memory

Since the SOC has a bus interface that is compatible with the CVAX chip, the most obvious chip to use as a main memory controller was the CVAX memory controller (CMCTL) chip.[1] It responds to all bus cycles generated by the SOC, and since it was already used on a number of platforms supported by the VAXELN operating system, its use greatly simplified porting VAXELN to the turbo controller. However, the turbo controller requires two special memory modes that are not provided directly by the CMCTL, namely OR mode and scan-erase mode. It was essential to devise a way to include these two modes if the CMCTL were to be used.

OR-mode memory is a technique used to improve performance during the writing of the page bit map into memory (scan conversion). During normal memory operation (called replace mode), the destination operand in memory is replaced by the source operand. During an OR-mode write cycle, the destination operand is modified as follows:

- For each logical zero in the source data being written, the corresponding destination bit in memory remains unchanged.
- For each logical one in the source data being written, the corresponding destination bit in memory is written with the corresponding bit in the pattern register.
- The pattern register is a 32-bit register which determines the "color" pattern of the "ink" being written on the page.

Figure 2 shows a portion of the logic between the CMCTL and the memory array that implements the OR-mode function in hardware. The OR-mode operation is accomplished by inverting the source data and connecting it to 32 independent write enables of the memory array. When a zero is written, it is inverted and the write cycle for that bit becomes a read cycle, thus preventing any change to the memory contents. When a one is written, it is inverted and the write is allowed to occur, but the data actually written depends on the value previously written into the pattern register.

Two features of the CMCTL chip make it possible to implement OR-mode memory. First, its 64MB address space is divided into 4 arrays of 4 banks (16 banks total). Second, the CMCTL chip can selectively disable parity checking on an array.

The large address space of the CMCTL allows the use of 2 arrays for replace mode and 2 arrays for OR mode, since the turbo controller supports up to 32MB of memory. The control signals of the two sets of arrays are combined such that OR mode and replace mode access the same physical memory, though in different ways. Parity error detection is disabled on the OR-mode arrays; thus a read-through OR-mode address space cannot cause a parity error. This is necessary because OR-mode write cycles may corrupt parity. Normally any bit map created using OR-mode write cycles is read using OR-mode read cycles.

The other special mode required for the main memory system is called scan-erase mode. It is an operating mode designed to improve bus utilization during the transfer of the bit map from main memory to a FIFO buffer connected to the printer data lines. This mode is made possible by a side effect of the error-correcting code (ECC)/parity generation logic in the CMCTL. Any time a masked write occurs (any write other than an aligned longword, such as a byte write), the destination longword must first be read by the CMCTL, then combined with the bytes to be written in order to generate the parity or ECC check bits for that longword.

Three operations occur during a single scan-erase cycle. Refer to the circuit drawing in Figure 3.

1. The bus master asserts the signal's "bit-map load" and "bit-map erase" and requests a masked write. The CMCTL performs a read, and the bit map is read onto the memory data bus.

2. Bit-map data is automatically transferred from the memory data bus into the FIFO buffer.

3. The CMCTL performs a write. However, since the bit-map erase signal has disabled the data path and the pull-down resistors have set the data-in lines to all zeros, the write cycle, which was intended by the designers of the chip as a masked write, has in fact become a memory clear operation.

Write Buffer

The LR3220 chip was chosen as the base for the write buffer subsystem. It provides a six-entry FIFO buffer for address, data, and byte mask and detects whether the processor has requested a read at a memory location for which a write is still pending. It also supports two operating modes: LR3000 mode and Harvard mode.

If it were not for the Harvard-mode feature, it would have been more difficult to include the LR3220 chip into the turbo controller. The LR3000 processor, for which this chip was designed, has staggered address timing. Some of the address and byte-mask bits are asserted on the falling edge of the clock, and the remaining bits are asserted on the rising edge of the clock. When the LR3220 chip is configured in LR3000 mode, the processor subsystem must meet these timing requirements. However, when the LR3220 chip is configured in Harvard mode, all address, data, and byte-mask information is read at the same rising clock edge.

The basic strategy for including the write buffer into the turbo controller was to insert the write buffer between the SOC and the rest of the system as shown in Figure 4. The SOC would issue read and write requests to the write buffer, and the write buffer would issue read and write requests to the rest of

the system. During CPU cycles the SOC and the write buffer have a master-slave relationship in which the SOC is the master. The relationship between the write buffer and the rest of the system is also a master-slave relationship; however, the write buffer is the master. In fact, the write-buffer output interface must look almost identical to the SOC.

The structure of the write-buffer subsystem is shown in Figure 5. The bus interface unit responds to read or write requests from the SOC. During write cycles, the bus interface writes the data into the LR3220 chip and immediately alerts the SOC to terminate the cycle quickly. Whenever one or more entries in the LR3220 chip have data, the bus cycle generator (BCG) removes the next entry and issues a write request to the appropriate subsystem.

The write-buffer subsystem allows the SOC to "read around" the write buffer, provided the address being read does not have a pending write in the LR3220. To handle this, the BCG includes an arbitration circuit. When the SOC requests a read cycle, the bus interface unit of the write buffer passes the request to the BCG. The BCG responds once it has completed any write cycle currently in progress, provided that the address to be read does not have a pending write in the write buffer. When the slave device being read acknowledges the BCG, the acknowledgment is passed back to the bus interface and finally to the SOC to terminate the cycle. The BCG then resumes its task of removing entries from the LR3220 chip and issuing writes to the rest of the system.

In order to maintain data coherency, the write-buffer subsystem enforces some additional protocols.

- All writes to any location other than main memory require a write-flush cycle; that is, the bus interface must wait until the LR3220 chip is empty before writing the data to it. Furthermore, the bus interface must wait until the BCG has finished the cycle before it acknowledges the SOC and allows it to perform the next cycle.

- All reads to any location other than main memory require a read flush, which has the same restrictions as a write flush. These restrictions are required to avoid the possibility of reading around a pending I/O space write, which often has side effects to other addresses.

- The write-buffer subsystem must pass all DMA bus transactions to the SOC to ensure that all cached memory locations that are modified by DMA cycles have their corresponding cache entry invalidated.

Bit-map Transfer Subsystem

The bit-map transfer subsystem transfers bit-map data, created by the PostScript interpreter, to the print engine. It is composed of the 32-bit DMA controller, a FIFO subsystem, and scan-erase logic in main memory as described in the section Main Memory.

The main requirements for the 32-bit DMA controller were

- 32MB address range

- Ability to transfer 32 bits at a time

- Ability to transfer the frame buffer forward (incrementing the source address) or backward (decrementing the source address)

None of the available DMA controller chips met all our requirements, but the AMD 9516 universal DMA controller (UDC) met some of them. The UDC is a 16-bit DMA controller with a 16MB address range and the ability to increment or decrement the source address. There were two drawbacks to the use of this chip. The software would have to ensure that the frame buffer was always within the lower 16MB of memory, and the UDC would use twice as much bus bandwidth since it could transfer only 16 bits at a time.

It was proposed that the UDC could be used as a full 32-bit DMA controller if it was connected to the bus "incorrectly" by shifting the data/address lines to the left by one bit. That is, data/address line 0 on the UDC would be connected to data /address line 1 on the bus; data/address line 1 of the UDC would be connected to data/address line 2 on the bus; etc. This type of connection doubles the address range of the chip and causes

the source address on the bus to increment by 4 bytes (32 bits) instead of 2 bytes (16 bits).

This decision had a few implementation impacts. For example, the register definitions were now incorrect, since all the bits in all the registers were shifted one bit to the left. However, once the software was modified to compensate for this, the UDC functioned properly as a 32-bit DMA controller. When combined with the scan-erase feature of main memory, it allowed us to achieve our bit-map transfer goal of reading 32 bits from memory, loading it into the FIFO subsystem, and clearing the memory location, all in a single DMA cycle.

4 Performance

In this section, the performance of the original PrintServer 20 is compared to the enhanced performance of the turbo PrintServer 20.

Except for performance, the original PrintServer 20 and the turbo PrintServer 20 have identical functional capabilities. Table 2 lists the five functional subsets that were characterized for performance on both printers. The first four functional subsets were rated using the PostScript real-time operator; they measure the elapsed CPU time needed to complete a test. The last functional subset was rated according to the rate of pages exiting the printer. The term "DECnet/DPS" refers to the DECnet job (a job is one of several multiprocessing tasks running on the controller) and the "distributed PrintServer software" job. The term "printer system" refers to the complete printer system, including the PostScript job and the printing overhead jobs. The printer system was rated according to the rate of pages exiting the printer.

Table_2:__Functional_Subsets_of_the_Printers

Functional Subsets	Characterization
PostScript job	Math operations per second
PostScript job	Text: characters per second
PostScript job	Graphics: vector inches per second
DECnet/DPS jobs	DECnet/DPS: kilobytes per second
Printer system excluding DECnet/DP	Image printing: square inches per second

Table 3 reports the general attributes of the five files that were run with the RETrACE system and characterized for performance.

Table_3:__Benchmark_File_Attributes

File Name	General Attributes of File
BM1.PS	Contains 39 pages of text with 13 fonts of various sizes. Some text strings are at varying angles.
BM2.PS	Contains 1 page of spiral text of various point sizes.
BM3.PS	Contains 41 pages of text with 5 fonts.
HOUSE.PS	Contains a 1-page bitonal image of 3000 blocks (DECnet limited).
SCHEM.PS	Contains a 65-page schematic of graphics (vectors) and text.

Math Operators Performance of the PostScript Job

Figure 6 illustrates the controllers' performance results in math operations per second. The test determines the time needed to perform 50,000 primitive math operators (e.g., adding two numbers 50,000 times) during a PostScript test document. The real-time operator reads the current time, and the repeat construct repeats the math operator. This test measures the performance of the CPU only.

Text Performance of the PostScript Job

Figure 7 compares the text performance of the PostScript job on the original controller and the turbo controller. The test determines how long it takes the PostScript job to compose 250,000 equally sized characters to the page buffer in memory, which eventually is sent to the print engine to be printed.

Graphics Performance of PostScript Job

An important means of characterizing graphics performance is in vector inches per second. Figure 8 shows the results obtained by running a PostScript vector program in which all vectors are at 45 degrees and vector lengths are from 0.1 inch to 3 inches.

Image Performance

The image test characterized the complete printer system, including the PostScript job and the printing overhead jobs, but excluding the DECnet/DPS time required to transfer an image file to a printer. Three one-square-inch bitonal images at device resolution were placed into the user dictionary and were used repeatedly during the performance measurement. The result of using these precached images was to eliminate the DECnet and DPS software time that would be required to transfer a full-page image from a host to the printer. Performance was measured by printing 10 pages of 80 square inches of image per page.

The pages were printed landscape and portrait to measure the image performance both on axis and off axis. (On axis means that the printer sequentially prints all bits of a word from the image on a single scan line. Off axis by 90 degrees means that the printer prints one bit from each word and does not print the next bit in the word until it is at the same position on the next scan line.) Figure 9 shows the results of the image performance test in square inches per second.

DECnet/DPS Jobs Performance

DECnet/DPS transfer rates can be ignored for text and graphics files, but these rates can consume most of the time needed to print large image files. For example, a single, letter-size page of image contains more than 1MB of image data, but the corresponding PostScript file contains more than 2MB. Because the image data is represented in American standard code for information interchange (ASCII) hexadecimal characters in PostScript, 8 bits of the PostScript file are needed to represent 4 bits of image data.

To measure DECnet/DPS, a PostScript file of 1MB of comments was sent to the printer. The clock was started when the beginning of the file was received by the PostScript interpreter and stopped when the end of the file was received. The assumption of this test method was that the PostScript interpreter can parse comment lines much faster than DECnet/DPS can transfer them.

The DECnet/DPS transfer rate is basically proportional to the slower of the host and printer processors. Figure 10 shows the DECnet/DPS results.

RETrACE Benchmark Files

The benchmark files listed in Table 4 are characterized both by the elapsed time from file arrival to file printed and by the amount of CPU time used to print the job. For example, in the BM3 benchmark, the speed is limited by the 20-page-per-minute print engine, but the CPU time needed to print the file can be used as a performance measurement.

Table 4: Benchmark Files Characterized by Elapsed Time and CPU Time (Seconds)

Benchmark File	Original CPU	Turbo CPU	Original Elapsed	Turbo Elapsed	delta CPU	delta Elapsed
BM1	7585	1707	7735	2050	4.4	3.8
BM2	238	51	241	51	4.7	4.7
BM3	56	15	128	120	3.7	1.1*
HOUSE	67	15	106	31	4.5	3.4
SCHEM	2802	625	3073	675	4.5	4.6

* Limited by engine.

5 Summary

The turbo controller enhanced the performance of the PrintServer 20 printer system. Its design was prompted by the need to maintain print speed performance for complex documents containing text, graphics, and images. The RETrACE system was designed to analyze the PrintServer 20 system to determine which architectural changes would provide the greatest improvement in PostScript performance. By optimizing hardware only in areas where it was truly worthwhile, we were able to use existing chips and reduce development costs. The subsystems of the turbo controller hardware that were optimized as a result of this analysis were the processor (SOC which provided CPU, floating-point accelerator, and cache subsystem), a memory subsystem with OR-mode and scan-erase access, a write-buffer subsystem, and a 32-bit DMA subsystem. Results of the performance tests for five

benchmarks, including PostScript jobs, indicate the levels of enhanced performance.

6 Acknowledgements

Chris Mayer designed and implemented the RETrACE multilevel cache simulator. He developed a ticketing algorithm that simplified the management of delayed behavior memory constructs such as write buffers.

7 Reference

1. D.K. Morgan, "The CVAX CMCTL-A CMOS Memory Controller Chip," Digital Technical Journal, vol. 1, no. 7 (August 1988):139-143.

Guido R. Simone Guido Simone is a principal engineer in the Print Systems Engineering Group and was the project leader and architect for the turbo PrintServer 20 controller. He is currently working on the development of a new print system architecture to be used with advanced printing technologies. In previous work, Guido was the project leader and architect for an rtVAX 78R32 CPU chip-based laser printer controller. Before joining Digital in 1980, he received a B.S. in electrical engineering from Rensselaer Polytechnic Institute.

Jeffrey A. Metzger Presently a senior engineer, Jeff came to Digital as a co-op student in 1983, working first in the Semiconductor Engineering Group and then in Hardcopy Engineering. He became a full-time employee after graduating from Cornell University in 1985. He introduced Hardcopy to system-level logic simulation, contributed to the hardware, software, and firmware development of the PrintServer 20, and developed RETrACE, which is used to characterize the execution behavior of PrintServer systems. Jeff is currently working in the Entry Systems Business Group on a next-generation processor product.

Gary P. Vaillette Senior hardware engineer Gary Vaillette has been involved in the design and implementation of printing system hardware since joining Digital in 1983. His current work includes performance characterization of PostScript printers and PrintServer products, and hardware implementation of CCITT decompression in the turbo PrintServer 20 product. Previously, Gary worked at Data General Corporation and helped to develop their token bus network product. He holds an A.A. (1974) in electrical engineering from Quinsigamond Community College and expects to receive a B.S.C.S. (May 1992) from Boston University.