Optoelectronic-cache memory system architecture

Donald M. Chiarulli and Steven P. Levitan

We present an investigation of the architecture of an optoelectronic cache that can integrate terabit optical memories with the electronic caches associated with high-performance uniprocessors and multiprocessors. The use of optoelectronic-cache memories enables these terabit technologies to provide transparently low-latency secondary memory with frame sizes comparable with disk pages but with latencies that approach those of electronic secondary-cache memories. This enables the implementation of terabit memories with effective access times comparable with the cycle times of current microprocessors. The cache design is based on the use of a smart-pixel array and combines parallel free-space optical input–output to-and-from optical memory with conventional electronic communication to the processor caches. This cache and the optical memory system to which it will interface provide a large random-access memory space that has a lower overall latency than that of magnetic disks and disk arrays. In addition, as a consequence of the high-bandwidth parallel input–output capabilities of optical memories, fault service times for the optoelectronic cache are substantially less than those currently achievable with any rotational media. © 1996 Optical Society of America

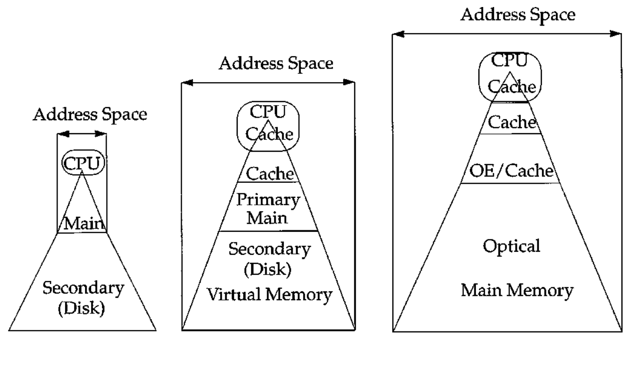
1. Introduction

Hierarchical memory architectures for computing systems are based on two fundamental paradigms of computer architecture: a hardware paradigm that states that smaller is faster and a software paradigm that programs access memory in patterns that exhibit spatial and temporal locality. Thus the latency inherent in the access to a large memory can be hidden in a pyramid with small and fast memory modules at the top, closest to the processor, and larger, slower memories at the bottom. Optical and optoelectronic (OE) memory devices offer the potential for building very large memories at the lowest level of the hierarchy. Unlike magnetic disks, optical memory provides random access throughout the address space, as well as high bandwidth and highly parallel data transfers. Recent developments in the integration of silicon and OE technology such as field-effect-transistor–self-electro-optic-effect devices or vertical-cavity surface-emitting lasers[1–5] have provided the devices necessary to integrate optical memories into a hierarchical OE memory system. Key to the successful design of such a system is the resolution of architectural issues such as the address translation mechanism, frame size at each level, write policy, replacement algorithms, and coherency support mechanism.[6] By utilizing an OE-cache memory, we can resolve both these architectural issues, as well as provide the necessary technology interface, and thus provide a seamless OE memory hierarchy that is compatible with modern uniprocessor and multiprocessor computing systems.

As shown in Fig. 1, in the conventional description of a memory hierarchy, a distinction is made between secondary memory, primary memory, and each level of cache memory. This distinction was originally based on the visibility of the memory relative to a machine language instruction. In this historical context, shown in Fig. 1(a), a primary, or main, memory was defined by the program address space (e.g., 16-bit addresses) and secondary memory, or backing store, was associated with input and output. Cache memories, to the extent they existed, were invisible and were first implemented as buffers between the processor and the primary memory. In modern systems, shown in Fig. 1(b), caches are implemented routinely and typically exist in multiple levels, with the first-level cache integrated into the processor itself. The distinction between pri-

The authors are with the University of Pittsburgh, Pittsburgh, Pennsylvania 15260; D. M. Chiarulli is with the Department of Computer Science and S. P. Levitan is with the Department of Electrical Engineering.

Received 22 September 1995; revised manuscript received 16 January 1996.
0003-6935/96/142449-08$10.00/0



(a) Historical (b) Modern Systems (c) Proposed Architecture

Fig. 1. Memory hierarchy evolution.

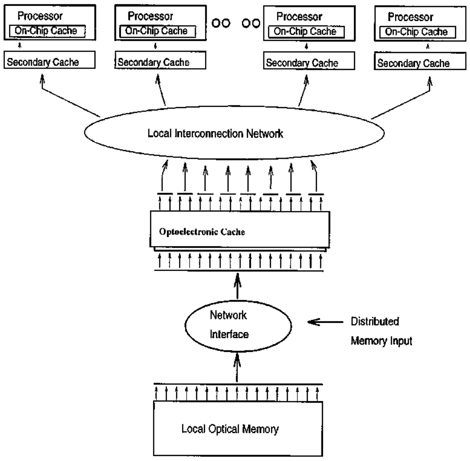


Fig. 3. Multiprocessor OE memory hierarchy.

mary and secondary memory has been significantly blurred by address segmentation and virtual memory systems. Typically, secondary memory now supports a much larger program address space, parts of which are swapped on demand into a semiconductor random access memory primary-memory level. In the discussion below, we dispense with the notion of distinct primary and secondary memories. As shown in Fig. 1(c), we merge these levels into a single OE memory at the lowest level of the hierarchy. The processor address space is directly supported in the optical memory. All levels between this optical memory and the processor are transparent to the processor and therefore are referred to as cache levels.

Figure 2 shows a block diagram of a physical realization of an OE memory system for a uniprocessor, and Fig. 3 shows a realization for a multiprocessor application. Reflected in these designs is the fact that most state-of-the-art processors use a two-level cache at the top of the memory hierarchy with a cache controller for these levels integrated on the processor chip. The top level, or primary cache, is a small on-chip memory. The secondary cache is somewhat larger and is off chip. These caches typically have access times of the order of the processor clock period, and data transfers between them are in the range of 10 to 100 words. At the lowest level, the optical memory provides high-capacity data storage. Between these levels, the OE-cache level links the secondary cache with the optical memory level. The OE cache is a dual-ported electronic memory with an optical port that connects to the optical memory and an electronic port that connects to the levels above.

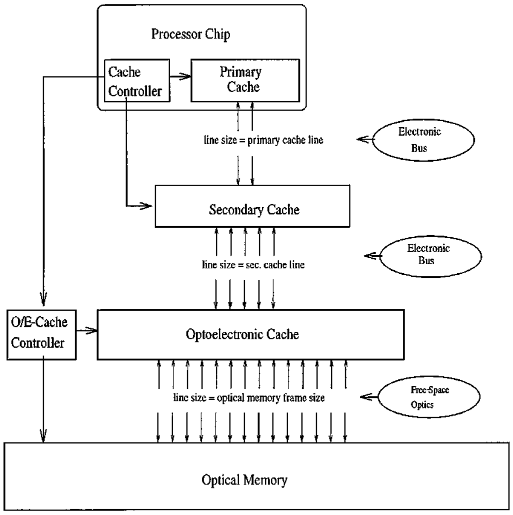


Fig. 2. Uniprocessor OE memory hierarchy.

In the shared-memory multiprocessor application shown in Fig. 3, the OE cache serves the same functions. However, in this design, it is also necessary that the cache be either multiported or banked to provide multiple access points on the electronic interface. Further, an interconnection network is necessary between the OE cache and the processor secondary caches. Alternative designs may eliminate this interconnection network by duplicating the OE cache at each processor and providing an interconnect at the optical memory level. Similarly, both local- and distributed-memory models might be supported because it is possible to implement the optical memory such that both local and networked banks of memory exist. Thus the OE cache is also an enabling technology for future multiprocessors that use large shared optical memory systems.

In this paper we present an investigation into the design of a new OE-cache level that will interface to a terabit optical memory to the electronic caches associated with one or more processors. The cache level is based on the use of smart-pixel array technology and combines parallel free-space optical input–output (I/O) to an optical memory with conventional electronic communication to the processor caches. The OE cache and the optical memory system to which it will interface provide a large random-access memory space that will have a lower overall latency than magnetic disks or disk arrays.

In Section 2 we briefly present the context of current or proposed optical memory systems and present our model for the OE interface to these memories. Next we outline a specific design for an

OE cache and cache controller. We then present a preliminary performance analysis based on analytical results and simulation data. We conclude with a discussion of the ramifications of our results.

2. Background

There are a number of competing optical memory technologies currently being investigated. We focus on nonrotational read–write media. This is because the access time of all rotational-media-based systems precludes their use as operating system transparent memories. The latency of these devices would necessitate a process context switch in the case of a fault; that is, in a multiprocessing environment, a different program would be run while waiting for the disk operation to complete.

Three-dimensional optical memory systems, on the other hand, have the potential of both fast access times and large capacities.[7] Typical examples of these systems are

- Spectral hole burning for memories at low temperatures[8,9] and the possibility of room-temperature devices.[10]
- Photorefractive materials for holographic storage.[11]
- Two-photon systems.[12]

All have the common characteristics of high access bandwidth supported largely by parallel access. Specifically, each reference returns a frame of data, in which the term frame refers to a large collection of bits typically related by membership in a specific data structure such as an image bitmap. In this discussion we select a less restrictive and technology-independent model for the optical memory. The model assumes only that it is a high-capacity memory with access parallelism modeled as a long word length. As with a conventional memory hierarchy, the access time is assumed to be significantly longer (2–3 orders of magnitude) than the clock period of the processors. I/O ports for the optical memory are assumed to be a free-space optical interconnect, with the number of channels corresponding to the number of bits in the memory word. However, given current or near-term technology limits, it may be necessary to multiplex the optical system in order to accommodate limitations on the density of OE device integration.

3. Optoelectronic-Cache Architecture

In this section we present a realization of the OE-cache level in the OE memory hierarchy. As shown in Fig. 1(c), the cache is in the same position as the primary memory in a conventional hierarchy. However, unlike primary memory, it is transparent to both the processor and the operating system. This level is the interface between the optical memory backing store and the secondary cache associated with the processor. Another distinguishing feature of the OE cache is its significantly larger line size than is typical for primary memory. A memory line (also commonly known as a cache line) is the amount of data transferred between levels of the hierarchy when a memory fault (or, equivalently, a cache miss) occurs. Thus the size of a line at a particular level is a trade-off between the locality supported within the memory traffic and the efficiency to which the cache is utilized. A large cache line more loosely constrains memory access locality. However, large cache lines will also bring fragments of unused memory into the cache; this effect is called internal fragmentation. In a conventional memory, the cost associated with internal fragmentation can be significant because the fault service time is typically linearly related to the line size. However, in the OE cache, the (much larger) line size is determined by the width of the optical memory word. The parallel access characteristics of an optical memory make it possible to transfer cache lines to and from the optical memory in a single access time. This is substantially faster than the equivalent transfer from a magnetic disk, which must allow for both rotational latency and serial transfers; this is a significant advantage. However, it has an effect on the organization of the cache itself and also has an impact on the mechanism for address translation and, in multiprocessor systems, coherency issues.

Figure 4 shows a block diagram of a design for the OE cache. In this diagram, optical I/O is transmitted and received by an array of self-electro-optic-

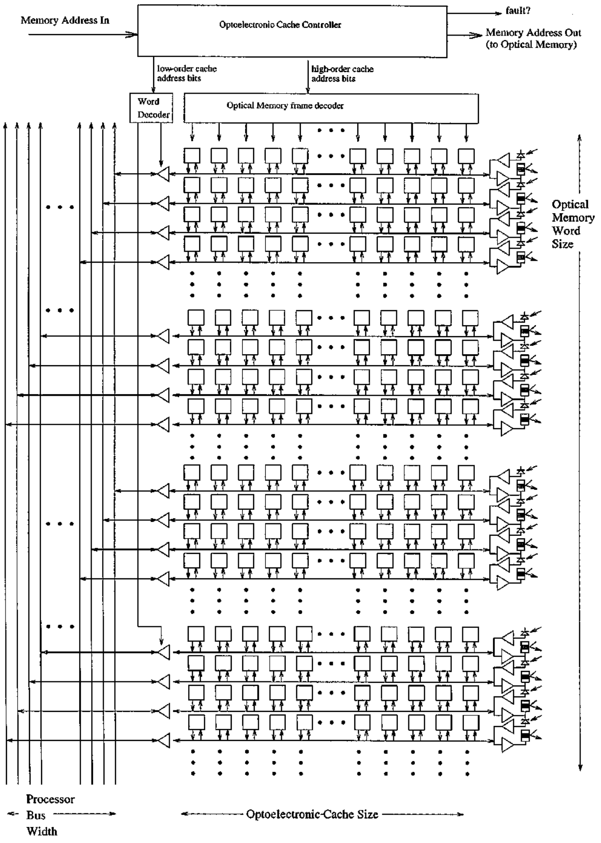


Fig. 4. OE-cache block diagram.

effect devices shown on the right. The electronic bus, drawn vertically on the left, connects the OE cache to the electronic secondary-cache level. The cache itself is modeled as a two-dimensional array of bits. Each column holds one cache line that corresponds to one word (frame) from the optical memory. Each column is subdivided into words, each of which is the width of the processor memory bus. Each of these words is in turn connected to the electronic I/O bus.

When a fault occurs in the secondary cache, the OE-cache controller processes the address to determine if there is a cache hit in the OE cache; in other words, if the requested location is present in the OE cache. If a hit occurs, the controller translates the address of the requested location from its location in the processor address space to an address within the OE cache. This address is partitioned as shown in Fig. 5. Once the address is translated, a pair of decoders handles the cache address. One decoder reads the high-order address bits and selects all the bits in a single column. The other decodes the low-order bits and selects one electronic memory word within the selected column. Thus, when the memory is accessed from the optical memory side, an entire cache line is read or written simultaneously. On the electronic side, a single word is selected when both the corresponding cache line (column) and the corresponding word offset are enabled onto the electronic bus.

We assume that, although Fig. 4 shows the OE cache as a monolithic implementation, it is likely that an actual implementation may partition the memory both along the word width and the memory depth. Also, given the relative bandwidth of the optical interconnect to the two memories, it is possible that the optical system may be multiplexed to reduce the device count. Error detection or correction mechanisms can also be built into the OE cache by the connection of this hardware to the horizontal bus that is carrying the optical memory word.

4. Controller Architecture

The discussion of address decoding in Section 3 assumed a cache hit. In other words, the cache controller shown at the top of Fig. 4 interpreted an incoming memory address, determined that the desired location was in the cache, and translated the memory address into a cache address. In this section we briefly describe how such a translation might take place in the OE-cache controller.

Consider an n-bit memory address that corresponds to a location in the data memory space of a processor. If it is assumed that the word size in the processor memory space and the word size of the optical memory are powers of 2, this location is at some specific offset within a larger optical memory word. Thus the n-bit address is partitioned into fields that correspond to a location in optical memory and the offset of the word. This organization is identical to the one shown in Fig. 5, except that the high-order bits now identify an optical memory word within the address space of the processor. In fact, it is identical to the organization of addresses at any level of the memory hierarchy at which the high-order bits select a specific memory line and the low-order bits select the offset within the line.

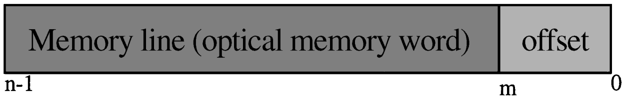


Fig. 5. OE-cache address consisting of an optical memory word and the offset.

The number of bits in the high-order partition of these addresses is determined by the relative sizes of the memory address space and each of the cache levels. Address translation is the operation of mapping from a value for the high-order bits in the memory address space to the high-order bits (cache line number) of a cache address. There are a number of methods for accomplishing this translation, which are well documented in the literature on memory systems.[13] They include low-latency solutions that use direct and fixed mappings. More complex methods use associative memory lookups and others use hierarchical tables. Each has different characteristics for latency, implementation efficiency, and cache utilization. In general, address-translation mechanisms with higher latencies tend to use the cache more efficiently and tend to lower fault rates. Thus, if the cost of a fault is high, such as is the case for swapping to and from a disk, then a designer is willing to tolerate a higher latency in address translation (for either a hit or a miss) in order to minimize the frequency of faults. For example, in a conventional memory hierarchy between primary memory and a swapping disk, fault costs can typically be of the order of milliseconds. Hence the dynamic address-translation algorithms used in a virtual memory system may add latency of 2 or 3 times the normal memory latency as overhead in order to implement nearly optimal replacement strategies. With an optical memory at the lowest level of the hierarchy, these fault costs are reduced to microseconds. Thus a significantly faster (but less optimal) address-translation mechanism can be utilized.

Throughout this discussion we have assumed that the optical memory replaces both the primary memory and the disk backing store of a conventional memory system. Thus the traditional notion of a virtual memory as a process-level address space is replaced by a single, large, processor-level address space. This design is consistent with the current trends in processor design in which 64-bit address spaces are quite common. When an optical memory technology is used to populate these huge address spaces, conventional mechanisms for memory management in operating systems will be obsolete. Both virtual memory mechanisms as well as file system organizations will be replaced by commonname-space object-oriented operating systems.[14,15] In the near term, however, it is still possible to integrate the proposed optical memory system into conventional virtual memory operating systems, which assume a unique address space for each process, by simply making an association between the upper bits of a optical memory address with the process-id of a running process.

5. Performance Analysis

In this section, we present an analysis of the relative performance of the OE memory system architecture compared with that of traditional memory hierarchies.

The average memory latency $\overline{L_x}$ at any level x of a memory hierarchy can be calculated as

$$\overline{L_x} = (1 - p_x)L_x + p_x L_{(x-1)},$$
$$L_0 = L_{\text{BackingStore}},$$

where p_x is the fault probability, $(1 - p_x)$ is the hit probability, and L_x is the memory access time at level x. L_0 is the latency associated with the memory at the lowest level of the hierarchy, commonly known as the backing store. In this expression we approximate the miss penalty, at all but the lowest level, to the average latency of the next lower level. This approximation is accurate if we assume that memory banking, or other prefetching techniques, have been implemented between these levels. At the L_0 level, specifically when disk drives are used as the backing store, is it necessary to consider the transfer time of a memory line as part of the latency. In this case, if T_s is the average seek time, T_r is the average rotational latency, and T_x is the transfer rate of a disk-based backing store, the miss latency of a memory line of size n_m is

$$L_0^{\text{disk}} = (T_s + T_r + n_m T_x).$$

Alternatively, when an optical memory is used as the backing store and the entire cache line is transferred in parallel, only T_0, the access time of the optical memory, needs to be considered:

$$L_0^{\text{optical}} = T_0.$$

With only this difference taken into account, Fig. 6 shows a plot of the average latency versus the hit rate for two, single-level memory systems. One uses disk technology as the backing store, the other uses an optical memory as the backing store. For the plot of Fig. 6, L_1 is set to 10 ns, T_0 is set to 1 μs, and the average disk latency L_0^{disk} is assumed to sum to 1 ms. Latency on the y axis is plotted on a log scale, and hit rates are varied only in the 90%–100% range. Clearly, the large region between the lines represents the potential latency improvement with an optical backing store. However, this improvement assumes that a given application will fault at the same rate in both systems. This may not be true, because, in the OE memory system, the line size must be significantly larger in order to exploit the parallelism in the memory transfers. The corresponding line size in the electronic-memory–magnetic-disk-based memory system will tend to be smaller, primarily in order to reduce the transfer component of the miss latency.

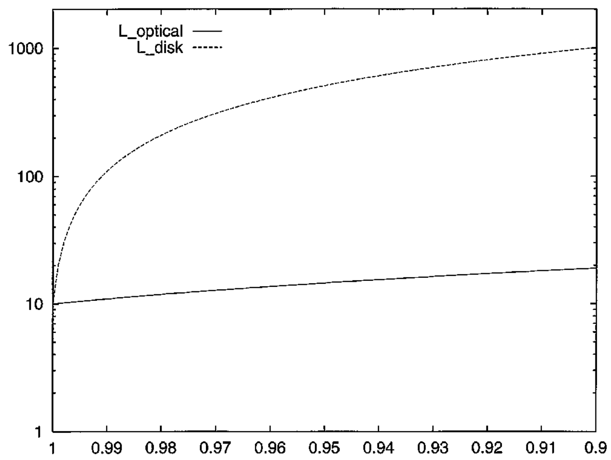


Fig. 6. Latency (nanoseconds) versus hit rate (percentage).

The factors that influence the choice of line size in a memory system are the locality behavior of the applications, the acceptable level of internal fragmentation, the size and the complexity of the tables used by the memory controller, and the miss penalty associated with loading the memory line into the cache. For hierarchies based on disk and disk arrays as backing stores, primary-memory line sizes (pages) are typically of the order of 128 to 4096 bytes. Assuming an optical memory frame size of 1 Mbit, the corresponding line size for an OE cache is 128 kbytes. Clearly the OE memory hierarchy has the ability to replace a much large cache line with a lower penalty. However, given that such a large cache line will have a correspondingly larger amount of internal fragmentation we can expect a corresponding increase in the fault rate. In Section 6 we show that, for several applications, these internal-fragmentation effects are minor compared with the performance gained because of reductions in the miss penalty.

6. Simulation Results

To investigate the relative fault rates of an OE memory hierachy versus a conventional electronic-memory–magnetic-disk-based hierachy, we implemented models of two memory systems. Each has a three-level hierarchy. The first version models the behavior of an electronic primary memory at level one with a magnetic disk as the backing store. The second version models the behavior of an OE cache at level one with optical memory as the backing store. The top two levels in both models are electronic primary- and secondary-cache memories with identical characteristics. The sizes and latency associated with each level are summarized in Tables 1

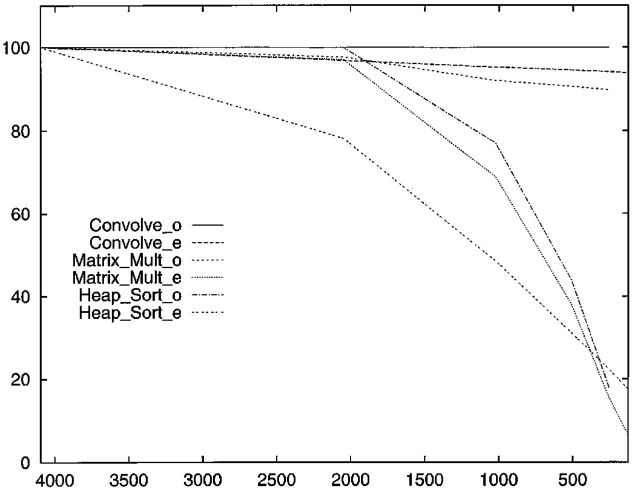


Fig. 7. Hit rate (percentage) versus memory size (kilobytes).

and 2, from which it can be seen that the only differences are in the line size of the main electronic memory versus the OE cache and the hit latency for the disk memory versus the optical memory.

Address translation at each level was modeled by the use of direct mapping in the primary cache, set-associative mapping in the secondary cache, and dynamic address translation (page tables) in the main memory/OE cache. Direct mapping in the primary cache was implemented as follows. For each address, the low-order 6 bits were treated as the offset within the 64-byte cache line. Of the remaining bits, the low-order 12 bits were used to select one of the 4096 cache lines in the primary cache, while the high-order bits were compared with a tag field attached to the selected cache line. The tag field contained the high-order bits of the addresses currently stored in the cache memory line. A match with the requested address constituted a cache hit. A miss required access to the secondary cache. Address translation in the secondary cache was set associative. As with direct mapping, the low-order 6 bits, the offset portion, of the address were removed. The remaining bits were partitioned such that the low-order 10 bits were used to select one of 1024 sets of four cache lines. In this case, a cache hit occurred if the tag field of any of the four cache lines in the selected set matched the requested address. Finally, in the case of the main memory/OE cache level, table-driven dynamic address translation was modeled. In an actual implementation this would mean that the line-number portion of the address would be used to index a table that contained the address of the corresponding cache line in the case of a hit or a flag in the case of a miss; to save memory in the simulator, this was implemented as a linear search of the tag fields in the cache (with no time penalty). This model is also functionally equivalent to a full associative address translation.

On the basis of these two models, three applications (an image convolution, a heap sort, and a matrix multiplication) were coded in C and instrumented to provide memory address traces for all data memory read operations. Each application was sized to run in a 4-Mbyte address space. Two sets of runs were made for each application, one set for each memory model. Each set of runs consisted of running the application in successfully smaller main (or OE-cache) memory sizes. Sizes ranged from 64 Mbytes to 128 kbytes (256 kbytes for the optical runs) for each set. The first set assumed a line (page) size of 2048 bytes and a fault latency of 1 ms for seek + DMA (direct memory access) transfer time of (100 × 2048) ns. The second set assumed a 128-kbyte page size and a 1000-ns miss penalty. Figure 7 is a plot of the hit rate versus memory size, for primary memory (e) or OE cache (o) at level one.

The data in Fig. 7 are plotted for memory sizes ranging from 4 Mbytes, the size at which the entire application can be loaded, down to 128 kbytes, which corresponds to a single line in the OE cache. The results demonstrate, for these applications, that internal-fragmentation effects do not decrease the hit rate (increase the fault rate) except in the case of the heap sort application with a very small optoelectronic cache (less than 512 kbytes). In all other cases, the increases in fault rate that are due to internal fragmentation are entirely offset by a reduc-

Table 1. Electronic Memory Simulation Parameters

Cache Level	Size	Lines	Line Size (bytes)	Hit Latency (ns)	Miss Penalty
Primary	512 kbytes	4096	64	10	(Secondary access)
Secondary	1 Mbyte	4096	256	50	(Main memory access)
Main memory	64 Mbytes to 128 kbytes	32K to 64	2048	100	(Disk access)
Disk				Seek + Transfer (avg = 1 ms)	

Table 2. OE Memory Simulation Parameters

Cache Level	Size	Lines	Line Size	Hit Latency (ns)	Miss Penalty
Primary	512 kbytes	4096	64 bytes	10	(Secondary access)
Secondary	1 Mbyte	4096	256 bytes	50	(Main memory access)
Optocache	64 Mbytes to 256 kbytes	512 to 2	128 kbytes	100	(Disk access)
Optical memory				1000	

tion in faults caused by the greater amount of data transferred per fault.

Returning to the latency calculations, we can now determine how these effects combine and compare average latency of the memory systems as a whole. The average latency of the electronic–disk memory system can be recursively constructed as

$$\begin{aligned}\overline{L}_{\text{disk}} = {} & (1 - p_p)L_p \\ & + p_p\{(1 - p_s)L_s + p_s[(1 - p_m)L_m \\ & + p_m(T_s + T_r + n_m T_x)]\},\end{aligned}$$

where L_p, L_s, and L_m are the access latency of the primary cache, secondary cache, and primary memory, respectively, and p_p, p_s, and p_m are the fault rates for the primary cache, secondary cache, and primary memory, respectively. Similarly, for the OE memory system, the average latency can be written as

$$\begin{aligned}\overline{L}_{\text{optoelectronic}} = {} & (1 - p_p)L_p \\ & + p_p\{(1 - p_s)L_s \\ & + p_s[(1 - p_{\text{oc}})L_{\text{oc}} + p_{\text{oc}}T_0]\},\end{aligned}$$

where p_{oc} and L_{oc} are the hit probability and the latency of the OE cache, respectively. Using the specifications in Tables 1 and 2 and the fault rate data from the simulations, we computed and plotted the average latency of each memory system in Fig. 8 for the three applications tested. The results are plotted on a log scale for memory size and latency. The latency plotted is the average memory access time though all levels of the memory hierarchy.

7. Discussion

For the applications tested, the simulations show a 3–4 order of magnitude increase in the performance of the optoelectronic memory system (with 1-µs random access to 1-Mbit pages) versus that of a conventional memory hierarchy with a rotational magnetic backing store. Thus we have demonstrated that OE-cache memories can be used to interface effectively a low-latency optical backing store for an OE memory hierarchy. Although line sizes in the cache are typically larger than disk pages, average memory-access latency is not adversely affected by the additional internal fragmentation introduced. We are currently investigating the relations among various competing technologies for the optical memory and the smart-pixel array used in the cache. These technology choices must be considered in the context of architectural issues such as the address-translation mechanism, frame size at each level of the memory hierarchy, write policy, replacement algorithms, and coherency support mechanisms for multiprocessor implementations.

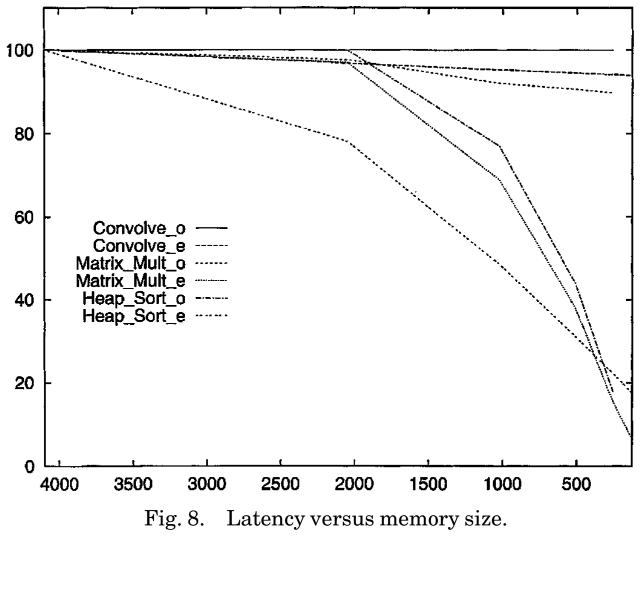


Fig. 8. Latency versus memory size.

This research has been supported, in part, by a grant from the U.S. Air Force Office of Scientific Research under contract F49620-93-1-0023DEF.

References

1. I. Ogura, K. Kurihara, S. Kawai, and M. Kajita, "A multiple wavelength vertical-cavity surface-emitting laser (VCSEL) array for optical interconnection," IEICE Trans. Electron. **E78-C(1),** 22–27 (1995).
2. J. Neff, "Optical interconnects based on two-dimensional VCSEL arrays," in *Proceedings of the First International Workshop on Massively Parallel Processing Using Optical Interconnections* (IEEE Computer Society, New York, 1994), pp. 202–212.
3. A. Dickinson, "An optical respite from the von Neuman bottleneck," in *Optical Computing,* Vol. 6 of 1991 OSA Technical Digest Series (Optical Society of America, Washington, D.C., 1991), pp. 239–242.
4. A. L. Lentine and D. A. B. Miller, "Evolution of the SEED technology: bistable logic gates to optoelectronic smart pixels," IEEE J. Quantum Electron. **29,** 655–669 (1993).
5. A. V. Krishnamoorthy, J. Ford, K. Goosen, J. A. Walker, A. L. Lentine, L. A. D'Asaro, S. P. Hui, B. Tseng, and D. A. B. Miller, "Implementation of a photonic page buffer based on GaAs MQW modulators bonded directory over active silicon VLSI circuits," *Optical Computing,* Vol. 10 of 1995 OSA Technical Digest Series (Optical Society of America, Washington, D.C., 1995).
6. S. Przybylski, M. Horowitz, and J. Hennessy, "Performance tradeoffs in cache design," in *Proceedings of the Fifteenth Annual Symposium on Computer Architecture,* H. J. Siegel, ed. (Association for Computing Machinery, New York, 1988), Vol. 16(2), pp. 290–298.
7. S. Esener, "3-D optical memories for high performance computing," in *Spatial Light Modulators and Applications III,* U. Efron, ed., Proc. SPIE **1150,** 113–119 (1989).
8. P. Wild, S. E. Bucher, and F. A. Burkhalter, "Hole burning, Stark effect, and data storage," Appl. Opt. **24,** 1526–1530 (1985).
9. C. D. Caro, A. Renn, and U. P. Wild, "Hole burning, Stark effect, and data storage: holographic recording and detection of spectral holes," Appl. Opt. **30,** 2890–2898 (1991).
10. S. Arnold, C. T. Liu, W. B. Whitten, and J. M. Ramsey, "Room-temperature microparticle-based persistent spectral hole burning memory," Opt. Lett. **16,** 420–422 (1991).
11. G. Burr, F. H. Mok, and D. Psaltis, "Storage of 10000 holograms in LiNbO$_3$:Fe," in *Conference on Lasers and Electro-Optics,* Vol. 8 of 1994 OSA Technical Digest Series (Optical Society of America, Washington, D.C., 1994), p. 9.

12. J. E. Ford, S. Hunter, R. Piyaket, and Y. Fainman, "3-D two photon memory materials and systems," in *Organic and Biological Optoelectronics,* P. M. Rentzepis, Proc. SPIE **1853,** 5–13 (1993).

13. W. Stallings, *Computer Organization and Architecture* (Macmillan, New York, 1987).

14. R. Trehan and K. Maeda, "A distributed object oriented language and operating system," in *Proceeding of the Twenty-Sixth Hawaii International Conference on System Sciences,* (IEEE, New York, 1993), pp. 70–79.

15. V. G. Antonov, "A regular architecture for operating systems," Operating Syst. Rev. **24**(3), 22–39 (1990).