



IEEE Cleveland Section CHAIRMAN'S COLUMN

by Allen Morinec

Happy New Year to you and your families! As I write this column, only three days from the new millennium, there appears to be little anxiety about Y2K glitches or massive failures. As you read this column, you already know the effects.

The IEEE Cleveland Section will have new challenges ahead of itself this year. Almost all sections, including the Cleveland Section, have switched to an electronic newsletter and a posting on our Web Site. With this changeover, we have lost contact with many of our members. Our officers and executive committee are dedicated to review this issue, learn the other needs of our members and serve their needs.

Some goals I have this year include:

- Learning the needs of our members, such as professional and career needs, benefits, and educational needs.
- For myself and Society Chapter Chairmen, to develop a personal relationship with our members.
- To deliver our newsletter to all interested parties.
- To grow our membership by getting the word out on our benefits and by working closing with electrical engineering students at the local universities.
- To grow our turnout at local meetings by offering more educational and interesting topics.

Our Cleveland Section is always searching for new volunteers, for interesting guest speakers and fun tours. Please call me at (216) 447-2505 or e-mail me at powerquality@ieee.org so we can discuss your ideas. Happy New Year to all! I hope to hear from or see you soon.

IEEE Cleveland Section Officers Election Results

The following officers have been elected for the IEEE Cleveland Section Year 2000 executive committee:

Chair	Allen G. Morinec
Vice Chair	Carl Dister
Secretary	Mike Branicky
Treasurer	Ted Lapponi

Congratulations to our new slate of officers! The Section looks forward to a successful year under your leadership.

Patent Reform Is Here

by Michael Garvey

After years of trying to go it alone, patent reform legislation was finally carried through Congress on the back of a budget bill. Patent Reform had been held up by a small but vocal group of opponents. On November 29, 1999, President Clinton signed the Inventor Protection Act, which was added to the Omnibus Spending Bill in the final days of the most recent Congressional session.

Throughout this year, this column will survey some of the changes resulting from the new law. Below is a tantalizing summary of the main provisions.

“Inventors’ Rights” -- Invention promoters must disclose past success rates, and the Patent Office will make complaints against the promoters publicly available. Anyone who has been scammed by one of these outfits can sue for damages without proving specific financial harm, and the damages can be trebled in some cases.

“Fee Changes” -- Certain patent fees are being reduced. Certain trademark fees are being decreased. For example, the patent application filing fee changed from \$760 to \$690 on December 29, 1999.

“First Inventor Defense” -- Also known as “prior user rights”, this permits infringement of a patent by someone who can prove he used a method of doing business before it was invented by the patent holder.

“Patent Term Guarantee” -- Patent applications that are delayed in the Patent Office will be granted patent terms longer than the current 20 years after the application filing date.

“Patent Application Publication” -- U.S. applications that have corresponding applications published abroad will be published by the U.S. Patent Office.

“Two Party Reexaminations” -- The involvement of a party challenging a patent through reexamination will be expanded.

“Patent Office Administration” -- The Patent Office is granted more authority over management and administrative matters. Public advisory committees will be created.

“Miscellaneous Provisions” -- A number of corrections and technical changes were made to the existing patent laws.

Some of these changes were effective November 29, 1999; others will take effect in the coming months, the latest change being effective on November 29, 2000. None of these changes is expected to have a dramatic effect on the patent system overall, but some of the changes could have significant effects for certain industries or certain types of inventors. Keep watching this column for reports on the changes and their effects.

Michael Garvey is a patent attorney with Pearne, Gordon, McCoy & Granger LLP.



CPU Performance Improvement 3: Cache Memory

by Steve Belovich

Computer engineers have long recognized that memory systems are one of the main bottlenecks of a von Neumann hardware architecture. Historically, CPUs have always been faster than RAMs, which made memory data access delays inevitable. What was needed was a memory that was both large and fast. Unfortunately, these requirements were diametrically opposed, since fast memories required a lot more hardware per stored bit; it could not be made as large as slower memories. Although the maximum practical size of both slow and fast RAM have increased greatly during the last 50 years, the inverse relationship between size and speed still exists and remains essentially independent of technology.

In 1946, A.W. Burks, H.H. Goldstine and J. von Neumann in Preliminary Discussion of the Logical Design of an Electronic Computing Instrument recognized that a hierarchical memory system could be built, with each level in the hierarchy trading size for speed. The smallest and fastest RAM would be located nearest to the CPU (in an electrical sense), while the slowest and largest RAM would be furthest away. The CPU would look first to the small, fast RAM for an item. If not found, it would next try the larger, slower RAM. This process would continue down through the memory hierarchy until the needed item was found. Clearly, if the operands were most often located within the faster RAM, then the effective memory access time would be very short, and the computer would run faster.

What Makes It Work?

The "Principle of locality" is the primary reason that hierarchical memory systems actually work. This principle states that all program favor a portion of their address space an any instant in time. The principle has two parts:

Temporal locality: If an item is accessed, it will tend to be accessed again soon.

Spatial locality: If an item is accessed, other items nearby in address space will tend to be accessed soon.

Colloquially speaking, this means that memory usage is very non-uniform, both in time and in space. If the "upper" (i.e., the faster) level of the memory hierarchy could be kept filled with items that are near an item that was most recently accessed, the hit ratio would be maximized. The entire memory system would perform at nearly the rate of the fastest portion at a fraction of the cost.

All memory hierarchy strategies map the addresses from the larger memory to the smaller memory. Thus, the contents of the i -th level memory is a proper subset of the contents of the $(i-1)$ st level memory, and so on.

Terminology

Adjacent levels within a memory hierarchy are generally managed independently from one another. The smallest amount of data that can be either present or absent in two adjacent levels of a memory hierarchy is called a block.

Blocks can be either fixed or variable in size. Each memory within the hierarchy is an integral multiple of the block size. Hardware considerations dictate that data is transferred between memory levels in units of blocks.

"Cache" is the name given to the highest level of a memory within a hierarchy. The term "line" is often used as a synonym for "block" when referring to units of data within a cache. Typical block (or line) sizes range from 4 to 128 bytes.

Finding an operand in the faster, smaller RAM is called a "hit," while not finding it is called a "miss". The "hit ratio" (or hit rate) is the fraction of items that are found in the "upper" (i.e., faster) level of a memory hierarchy. Note that a hit ratio can be defined and measured for each level within a memory hierarchy. The "miss rate" is equal to 1-hit rate. Typical miss rates are between 0.01 and 0.20 (1% to 20%).

"Hit time" is the time required to access the upper level of a memory hierarchy, including the time needed to decide whether the access is a hit or a miss. "Miss penalty" is the time needed to replace a block of memory in the upper level with one in the lower level.

Performance Calculations

Statistics govern the performance of a memory hierarchy, since they determine the values of the hit or miss rates, which are unitless quantities. Hardware speeds govern the hit time, miss penalty and other durations. The average (i.e., "effective") memory access time is given by: $\text{avg. access time} = \text{hit time} + \text{miss rate} * \text{miss penalty}$.

The miss penalty and miss rates are functions of the block size. Larger block sizes increase the miss penalty because they require more time to transfer the data items from the lower level memory to the higher level.

Increasing the block size will lower the miss rate until the reduced misses due to spatial locality (larger blocks) are overcome by the increased misses. This occurs because the larger blocks are displacing too many blocks that hold recently accessed data that will be needed again soon (temporal locality). Figure 1 shows the relationship between average access time and memory block size.

Memory Hierarchy Design Issue #1: Block Placement

Memory hierarchy design deals with four issues: (a) Block placement, (b) Block identification, (c) Block replacement and (d) Write strategy. Block placement can happen in three major ways. If each block (or line) can be put in only one specific place in the cache, the cache is called direct-mapped.

If each block can be placed anywhere within the cache, the cache is called fully associative. If a block can only be placed within a restricted set of locations, the cache is called set associative. A "set" is a group of two or more blocks within the cache. A block is first mapped to a specific set; then it can be placed anywhere within that set. Note that if

the set size is one, the cache becomes direct mapped. If the set size is equal to the entire size of the cache, then the cache becomes fully associative.

Direct-mapped caches are easiest to build from a hardware standpoint; however, they have a higher miss rate, since memory blocks cannot be placed arbitrarily within them. Fully associative caches have the lowest miss rates, but the entire cache must be searched for a desired data item, which requires a lot of hardware. The set associative cache is most common and represents an acceptable tradeoff between performance and hardware complexity.

Memory Hierarchy Design Issue #2: Block Identification

Cache data includes a "tag" which indicates origins of the data. These tags are checked by hardware to see if they match the block-frame address from the CPU. The search for matching block-frame addresses is done in parallel, usually by an associative memory. If this were not the case, the tag search time would totally negate any performance improvement that the cache might provide.

Not all areas of the cache contain valid data all the time. Most caches have a "valid bit" as part of the tag. This bit is set to indicate that the specific block is useful and up-to-date.

In a set-associative cache (the most common type), the tag field associated with each block is divided into three fields: (a) block-offset (selects the desired data from the block). (b) index (selects the specific set) and (c) tag (used for comparison). The block offset field is not used for comparison, since it merely points to the specific data item within the cache block. The index field is also not used for comparison, since it is used only to position the block within the cache.

Memory Hierarchy Design Issue #3: Block Replacement

Whenever a miss occurs, the memory block containing the desired data must be found and copied to the cache so that the CPU can access it. Note that the CPU does not bypass the cache and go directly to other memory levels lower in the Hierarchy. To provide this capability would greatly increase the hardware complexity and cost for very little gain. When a block of memory needs to be copied to the cache, it generally replaces another block that is already there.

Direct-mapped caches leave no choice for block replacement, since the physical address of a memory block completely determines its location within the cache. The block to be copied to the cache is put in the one unique place where it can go, and that's that!

Fully associative and set associative caches use two general strategies to choose blocks for replacement. "Random" replacement selects a block at random for replacement. This spreads allocation uniformly through the cache. "Least-recently used" (LRU) strategy replaces the "oldest" block within the cache. This principle is the corollary to temporal locality: the block least recently used is the one least likely to be needed in the future.

The random replacement strategy is easiest to implement in hardware, but has the poorest performance. The least-recently used scheme is the best performance-wise, but requires a lot of hardware to track the usage history of each block, particularly for large caches. In practice, the LRU

algorithm is approximated to reduce hardware complexity.

Memory Hierarchy Design Issue #4: Write Strategy

The majority of cache accesses are read operations. Write operations typically account for less than 10% of all memory traffic. Write operations, however, are complex and time-consuming enough that they may significantly degrade performance. The CPU processor word size determines the maximum portion of a block that may be written in one operation, typically between four and eight bytes. This means that a block needs to be read, the proper portion modified and then the entire block written to the cache (because the minimum data transfer in a memory hierarchy is units of blocks).

The two main cache write strategies for a write "hit" are:

Write-through--The data is written to both the cache and the corresponding block in the next lowest level of the memory hierarchy. The CPU stalls during the time it takes to write the block back to the lower level memory.

Write-back--The data is written only to the block in the cache. The modified cache block, which must be written to main RAM eventually, is usually written back only when it is replaced. These cache blocks which have been modified are called "dirty" and have to be written to a lower level memory when they are replaced. A "dirty" bit within the tag field keeps track of the dirty cache blocks.

Write-back cache designs are faster because writes occur near the hardware speed of the cache. Multiple writes to the same block only require one write back to the lower level memory. Memory bandwidth is conserved; hence, this design is helpful for multi-processor machines.

Write-through caches ensure that main memory always has the latest copy of all data which is better for power-failure handling, disaster recovery and context switching. Read misses do not waste time by forcing a write back for dirty blocks. Multi-processors also can make use of this design, since the cache and lower levels of RAM are always consistent.

A write miss results in two options:

Write allocate--The required block is loaded into the cache; then it is treated as a write hit.

No write allocate (also called write around)--The block is modified in the lower level memory directly and is not loaded into the cache.

Write allocate is generally used with write-back cache designs in the hope that subsequent writes to the block will be "caught" by the cache. No write allocate is generally used with write-through cache designs, since subsequent writes to the cache will still have to go to the lower level memory.

Software Implications

Like pipelining, the ideal hierarchical memory system requires no software intervention. All operating system and application software functions independently of the memory system and cannot tell if such a system is even in place. The net result is that the effective memory access is faster, so that the overall program executions speed is improved, relative to a computer system having a non-hierarchical memory.

Happy New Year from Your Editor

The editors of the Cleveland "Contact" send a hearty "Congratulations" to our Section's newly elected officers, headed by Chairman Allen Morinec. Allen's efforts to expand our newsletter coverage of Section events have been extraordinary. We look forward to working with him at the start of the new millennium.

And a most happy, healthy and prosperous New Year to all our readership! E-mail us your articles, opinions and suggestions for topics you'd like to see explored in the "Contact." You can reach us at mgreene828@aol.com. But remember, in order for the "Contact" to reach you by the beginning of each month, assuring that you are aware of all upcoming events in time, your articles must reach us by the 20th of the preceding month.

IEEE Cleveland Section 2000 Officers

ELECTED SECTION OFFICERS

Section Chairman	Allen G. Morinec	216-447-2505
Vice Chair	Carl Dister	440-255-2977
Secretary	Mike Branicky	216-368-6430
Treasurer	Ted Lapponi	216-771-2060

APPOINTED SECTION OFFICERS

Area Rep/Ad Manager	Lee Bernasek	440-446-1985
Awards Chairman	Gerald Lucak	440-255-2977
CTSC Representative	open	
Database Manager	James Sens	216-642-1230
Education Chairman	open	
Engineer Week Rep.	Dick Carlson	440-951-1595
Newsletter Editor	M. Greene	216-292-2686
PACE Representative	open	
Consultants Network	Dragan Dugandzic	440-257-2151

TECHNICAL SOCIETY CHAPTER CHAIR

Computer	Steve Belovich	216-267-1881
Control Systems	Wei Lin	216-368-4493
Industrial Applic.	Carl Dister	440-255-2977
*MIREA	Vincent Lalli	216-433-2354
**MLE	Masud Tabib-Azar	216-368-6431
Power Engineering	Allen Morinec	216-447-2505
Systems/Men/Cybernetics	Ben Malakooti	216-368-4462
Vehicular Technology	open	

STUDENT BRANCH OFFICERS

CSU Advisor	George Kramerich	216-687-2405
CSU Chair		
CWRU Advisor	Robert Edwards	216-368-2833
CWRU Chair	John Bennardo	216-754-1972

IEEE Internet

Webmaster

Website: <http://www.ewh.ieee.org/r2/cleveland/>

News Group Mgr.	Richard Bloss	216-464-0405
FAX		216-464-0490
e-mail:	aa974@cleveland.freenet.edu	

JOB LISTING SERVICE

Send blank e-mail to: info.ieeeusa.jobs.r02@ieee.org

*For member address changes, please call the IEEE Hotline, (800) 678-IEEE.
Do not call the newsletter editor. For advertising rates, contact Lee Bernasek.*

***MIREA:** Engineering in Medicine & Biology/
Instrumentation & Measurement/ Reliability/ Industrial
Electronics/Aerospace & Electronic Systems

****MLE:** Microwave Theory & Techniques/Lasers &
Optoelectronics/Electron Devices

**Please note: if you are trying to contact the IEEE
Cleveland Section by FAX, dial 440-473-6557.
The line is open twenty-four hours per day.**



Cleveland Section

Editor:
M. Greene

Art Director:
Jeff Greene

Advertising Manager:
Lee Bernasek

IEEE Chairman:
Allen G. Morinec

The "Contact"
is published
eight times per year:
September, October,
November, January,
February, March,
April and May.
The deadline for
electronic Newsletter
articles
is the 20th of the
previous month.