Two Views of the Post PC World - Automata Processor and TOMI Celeste, Part 1

Russell Fish - March 18, 2014

David Patterson, Sophie Wilson, and Steve Jobs have torched the PC business.

Patterson along with Dave Ditzell co-authored the famous RISC paper documenting how simpler instruction sets could outperform complex ones.[1]

Sophie Wilson architected ARM, one of the first RISC processors based on these concepts.[2]

Steve Jobs proved that a RISC architecture powering toys, games, and cell phones could run a tablet at lower power and cost than a PC.[3]

It gets worse for the industry. Computer architecture in general seems to have bumped up against some hard limits. In particular Robert Denard's Scaling[4] and Gordon Moore's Law[5] have collided with Patterson's Walls[6].

Forty years ago Denard demonstrated that as MOSFET transistors were reduced in size, the speed improved. Furthermore, as the power supply voltage decreased, power consumption decreased by the square of the voltage.

A decade earlier Moore observed that the number of transistors on integrated circuits doubled approximately every two years.

Combining these two effects delivered three decades of progressively cheaper, faster, but hotter microprocessors until it all came to an end, just as David Patterson predicted.

Patterson (yes, the RISC guy) codified what most architects knew instinctively. All computers are limited by three elements:

1. How fast they access memory[7]
2. How hot they run[8]
3. The amount of parallelism in their pipeline[9]

The Walls explain why the Pentium 4 Extreme Edition introduced in 2004 ran 3.73GHz[10], but the top of the line Haswell i7 introduced September of last year is spec'd for 3.5GHz and 3.9GHz "turbo"[11].

In short, legacy computer architecture is at a dead end.

SO NOW WHAT?
Intel is not going out of business, but as the PC dies they will leave the microprocessor business they pioneered just like they left the DRAM business they also pioneered[12]. Lenovo might be a good acquirer.

We will discuss the dominant trends in future computing in more detail in Part III, but in short future computer functionality will be broadly split into:

- Very large massively parallel systems
- Connected by high speed network to
- Very small, person or thing related systems.

Both systems are currently constrained by:

- Performance
- Power consumption
- Cost

Google, Amazon, and Facebook infrastructure are good examples of massively parallel systems networked to small systems.

Smart phones, tablets, and of course Google Glass are current examples of very small systems.

In light of the PC debacle and Patterson's Walls, the future of computing until the end of silicon probably consists of some combination of the following:

1. ARM variations in everything from greeting cards to supercomputers,
2. Non-von Neumann architectures such as Micron’s Automata Processor, and
3. Multi-core CPUs in DRAM such as Venray’s TOMI Celeste.

In a future series we'll investigate #1.

This series discusses #2 and #3.

Micron's Automata Processor [13] [14]
Automata Processor (AP) is Micron's pattern matching engine it has been developing for seven years. AP is a data accelerator with applications ranging from deep packet inspection to graph analytics.

Micron builds AP in its DRAM fab which gives it several advantages compared to logic fabs such as TSMC:

1. DRAM transistors have several thousand times less current leakage than logic transistors.
2. DRAM transistors have much more consistent temperature performance than logic transistors.
3. DRAM transistors are really cheap to make compared to logic transistors.

The greatest disadvantage is the lack of metal layer interconnect. Most DRAM processes have three metal layers. Logic processes require up to 16.

AP is a good match for a DRAM process due to the simplicity of its logic and the row and column structure necessary for its parallel compares. **Micron's Automata Processor**

Automata Processor is a State Machine on Steroids
A state machine is a means of representing a system consisting of several states and the events that cause a transition from one state to another.

AP is a specialized programmable state machine with some important parallel enhancements.

An elevator controller is an example of a simple state machine.

Figure 1. Elevator State Machine

The system will remain in one of the states until one of the following events:

- The UP button is pressed
- The DOWN button is pressed
- The elevator reaches a floor

Figure 2. Elevator Truth Table

The above truth table can be implemented using a 32 x 2 bit memory and a latch as shown in Figure 3 State Machine Implementation.

Figure 3. State Machine Implementation

State machines have mostly been replaced by embedded processors, but one of the most important designs of the early PC business was implemented with exactly such a design. Using a state machine made from a PROM, self-taught Steve Wozniak reduced 45 ICs in the traditional design to 5 for the Apple II disc controller.[15]

Micron's AP retains the simplicity of Woz's state machine but adds two elements that enable dramatic levels of parallelism.

Figure 4. Apple II Disk Controller

AP Element One
The first element came from Ricardo A. Beaze-Yates's 1989 dissertation for the University of Waterloo.[16] In this document Ricardo proposed two methods to accelerate search. No one remembers the first, but the second is pretty important:

"In our second solution, we trade storage for time. We obtain a solution which works in time independent of the size of the answer. For this, we include in each internal node all the positions (in order) that match with that node (leaves of the subtree rooted by that node)."

Micron has implemented this structure as a 256 x 1 bit memory as shown in Figure 5 Micron's

Parallel Comparator[17].

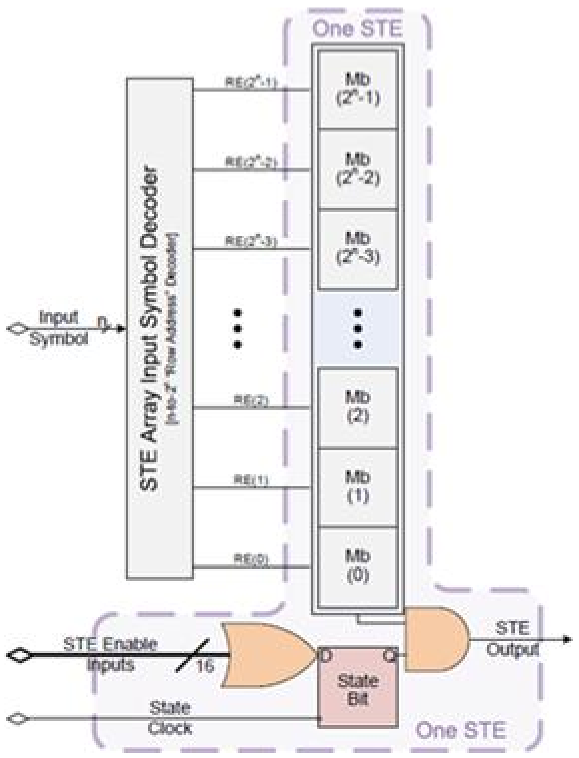

Figure 5. Micron's Parallel Comparator

The 256 bits correspond to all the possible 8-bit characters that could be matched. For example, if Automata Processor is searching an input stream for the letter "A", it is represented in ASCII as Hex 41. Location Hex 41 of the 256 bit RAM would be programmed with a "1". All other locations would contain "0".

The character to compare is presented to the address decoder which selects 1 of the 256 memory rows. The column line will contain the result of the match. If Hex 41 is presented to the address decoder, the column will contain a "1", indicating a match. Any other character will produce a "0".

Furthermore, multiple comparisons can be performed in parallel. For instance, text reading software looks for a variety of terminators indicating the end of a word. Space, comma, period, and carriage return can all be searched in a single cycle by setting the corresponding bits in the 256-bit memory.

AP Element Two

Unlike a simple state machine that allows a transition to a single state, Automata Processor achieves additional parallelism by allowing multiple transitions on a single event.

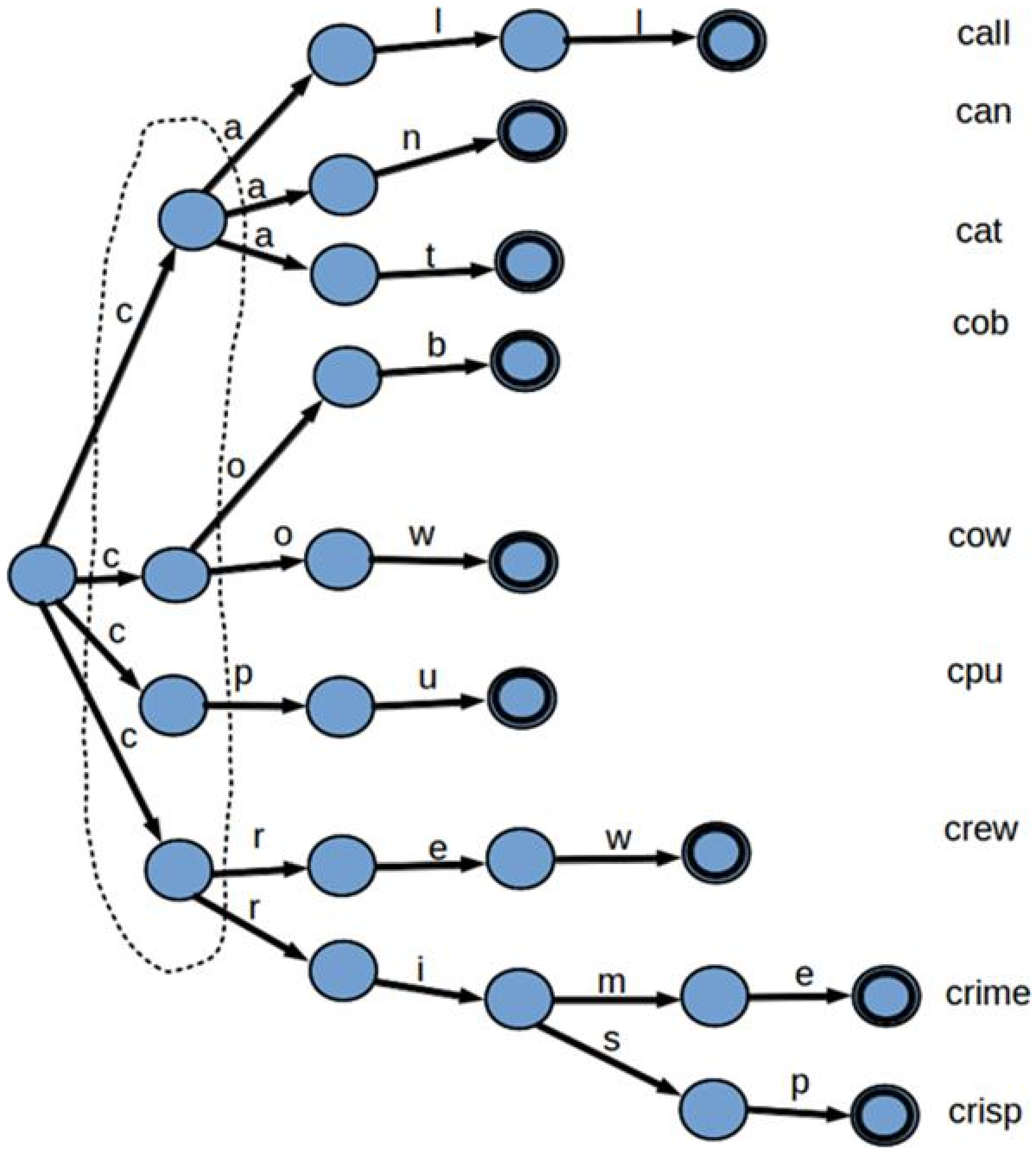

Figure 6. Example of Parallel States

The diagram above might be found in a text to speech program that looks for key words and then pronounces them. The actual program would contain a path for every word that could be recognized and probably require a dozen or more Automata chips.

The list of possible words is on the right side. Notice that after the letter "c" is found, there are 4 parallel comparisons being performed simultaneously. If an "a" is found, there are 3 parallel comparisons being performed. If a comparison fails, that state terminates.

As a result, any word no matter how large the dictionary, can be detected in the same number of cycles as it has letters.

Due to the parallelism of its compare and parallelism of its states, Automata Processor is probably several hundred times faster at decoding patterns than any existing computer architecture. Furthermore, as the size of the comparison dictionary increases, the AP speed advantage increases.

This performance will enable entirely new classes of products which we'll discuss in Part III of this series. **Venray's TOMI Celeste**

Venray's TOMI Celeste

Celeste is Venray's third generation of merged CPU and DRAM architecture. A merged CPU and DRAM constructs the CPU on the same die as the main DRAM memory using an unmodified DRAM process.

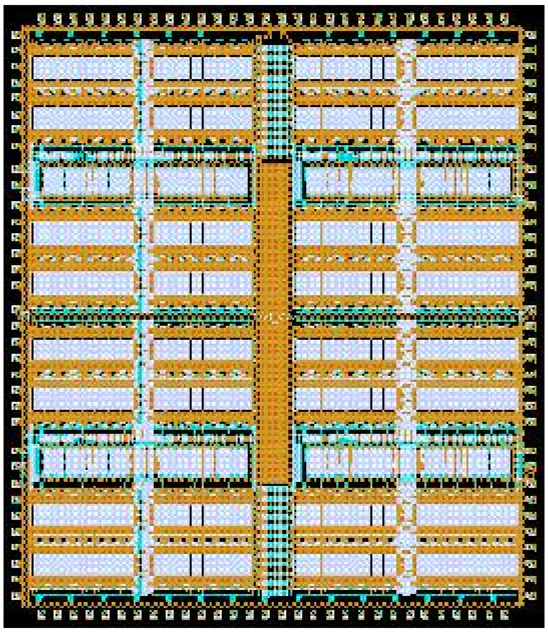

Figure 7. TOMI Aurora Layout

Embedding CPU into DRAM architecture was first proposed in 1989. From the patent:

"One solution to the bus bandwidth/bus path problem is to integrate a CPU directly onto the memory chips, giving every memory a direct bus to the CPU."[18]

Subsequent work showed the additional benefit of up to 90% reduction in power consumption compared to similar legacy computer architectures.

In the late 20th and early 21st century several projects embedded DRAM into CPUs and demonstrated the performance and power benefits but at great cost. The best known effort was David Patterson's iRAM, the same Patterson noted twice previously.[19]

As described in the original patent, the key to economically merging CPU and DRAM is integrating the CPU into the memory rather than embedding memory into the CPU.

The tradeoffs between the two techniques are as follows:

Venray chose to integrate CPU(s) into DRAM. Venray solved the 3 layer metal and space constraint by developing a CPU optimized for two of the most economically important techniques of the post-PC computer world; MapReduce and Graph Analytics. [20][21] Both techniques were pioneered by Google but are now widely distributed by just about every computer company.

As a result of the optimization, a 64-bit TOMI Celeste core is just over 44,000 transistors (less cache). Venray Director of Engineering Steve Krzentz said, "This small number of transistors is easily hand routed with 3 layers of metal using a bit-slice technique."

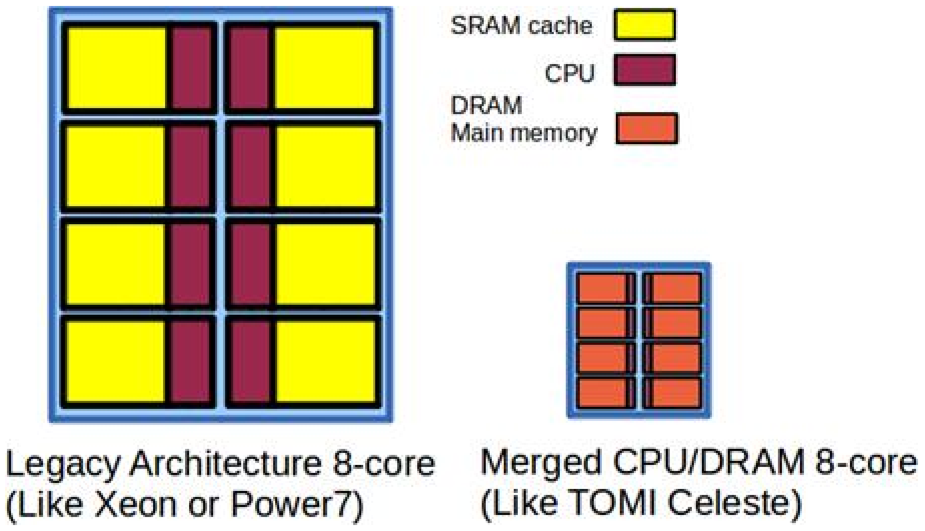
SRAM cache
CPU
DRAM Main memory
Legacy Architecture 8-core (Like Xeon or Power7)
Merged CPU/DRAM 8-core (Like TOMI Celeste)

Figure 8. Comparing Legacy and Merged CPU DRAM Architectures

Figure 8 shows the general relationship of an 8-core TOMI Celeste to a legacy 8-core architecture such as Xeon or Power7. Even though Celeste is built on a 46nm DRAM process and Power7 is on 32nm, Celeste is a much smaller die. Of course the Celeste die includes the DRAM main memory and the Xeon or Power die includes no main memory.

The SRAM cache that consumes up to two thirds the area and power of a legacy CPU is replaced by the ultra-dense DRAM main memory on Celeste.

Celeste Architecture
The TOMI RISC instruction set was designed to easily scale to any size instruction width while maintaining the same register architecture. The first two TOMI generations (TOMI Aurora and TOMI Borealis) were 32-bit designs. Celeste was expanded to 64-bits to allow addressing of really large data sets. To support really large systems, the Celeste interconnect bus connects directly to a parallel 3D Torus similar to that of IBM Blue Gene[22].

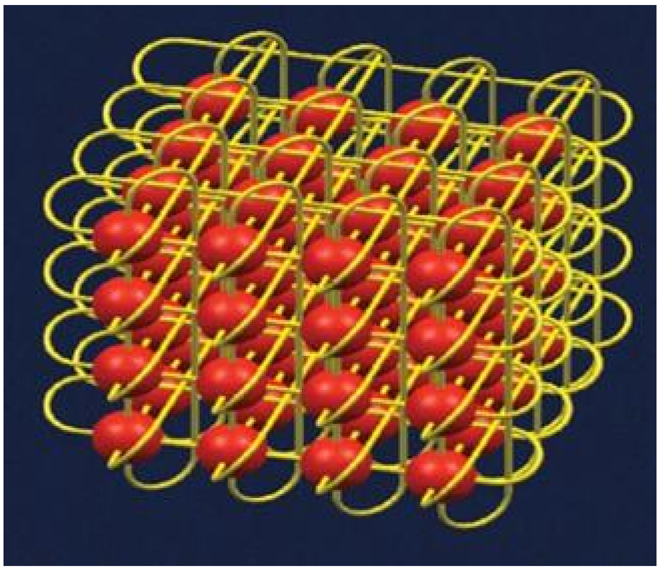
Figure 9. 3-D Torus

Smaller Really is Better
Smaller Really is Better
Celeste has a significant physical advantage over legacy CPUs in networking massively parallel systems.

The photo shows a portion of the fiber optic cabling required for the IBM Blue Gene 3D Torus. Due to the size of the server chassis, cable runs can exceed 1 meter.

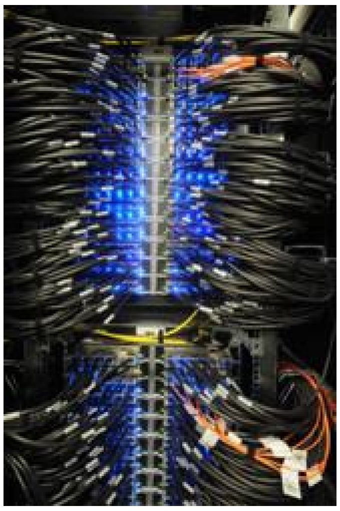
Figure 10. IBM Blue Gene Interconnect

A TOMI Celeste DIMM of 128 cores and 8GByte is 2.6" long and the DIMMs can be spaced ½” apart on the motherboard. Due to the proximity of DIMM to DIMM hops, the Torus network connection can be implemented with a parallel bus using existing on-chip DDR3 drivers and receivers. The parallel bus is simpler to implement, lower power, and significantly lower cost than an equivalent speed Infiniband optical connection.

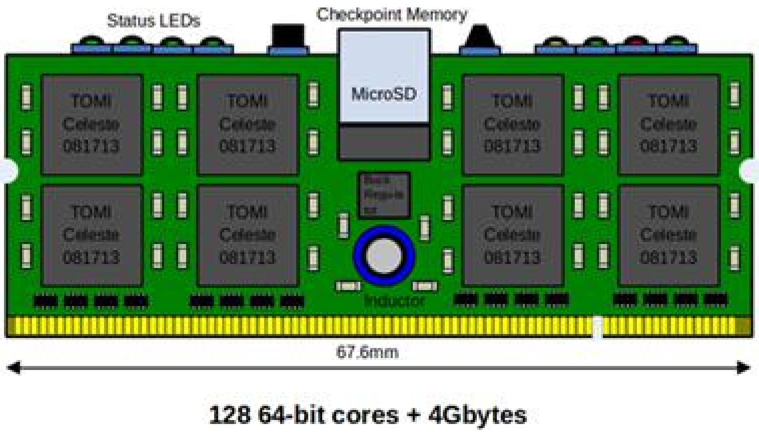

Figure 11. TOMI Celeste DIMM

Several Unexpected Discoveries

TOMI Celeste was optimized using Sandia Labs' MapReduce MPI[23] and the standard Big Data analytics benchmark, Graph500[24].

Through extensive running of these benchmarks on TOMI Celeste as well as legacy architectures Venray made several discoveries for parallel Big Data applications:

1. Increasing main memory size for a single or multi-core legacy CPU did not necessarily lead to improved performance, particularly for very large amounts of data.
2. Increasing cores per legacy CPU beyond 4 actually reduced performance. [This is consistent with Sandia Labs' findings[25].]
3. However, increasing cores per gigabyte almost linearly increased performance for TOMI Celeste, but not for legacy architectures.

Result 3 is most easily understood by examining MapReduce. MapReduce is an application that searches and sorts very large amounts of data. MapReduce is "embarrassingly parallel". This means that the work can be divided across many independent cores as long as each core has direct access to a portion of main memory. From EDN's "Multi-core and the Memory Wall"[26]:

> "Imagine the 1,000 page New York City phone book. Your task is to find a specific number in those thousand pages. The phone book is ordered by name, but the phone numbers are essentially random. You would begin a linear search at page 1 and continue until you found the desired number, probably several days later.

On the other hand, if you have a big Facebook community, you can give a page to each of your 1,000 friends and ask all of them to search at the same time. You will find the number approximately 1,000 times faster than if you searched by yourself."

Think of the phone book as main memory.

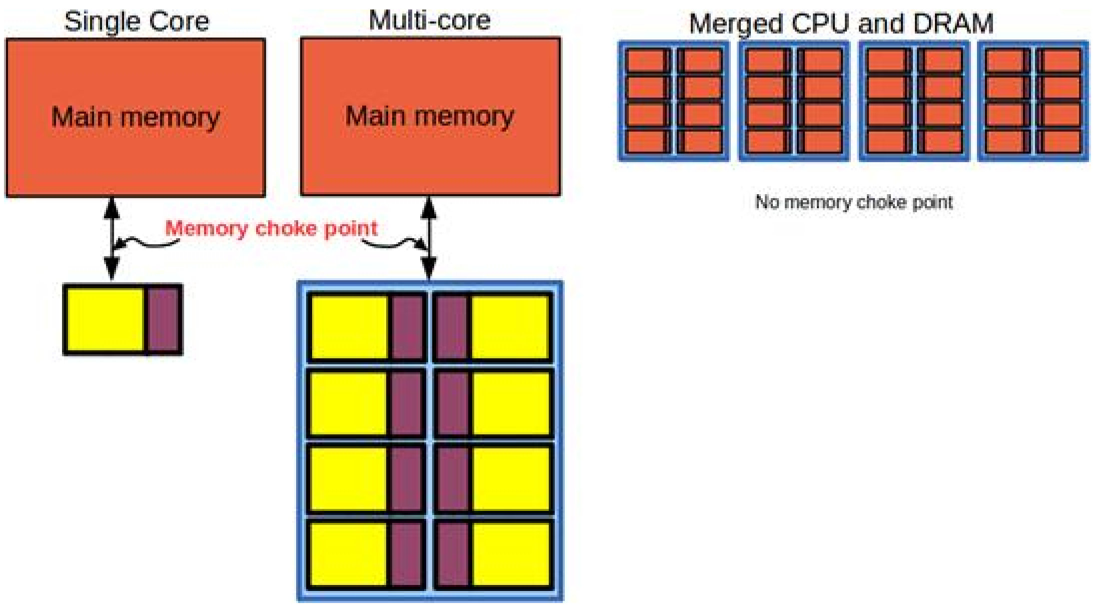

Figure 12. Comparing Big Data Implementations

The figure above shows a single CPU accessing the phone book in main memory. The memory choke point is clearly visible.

In the multi-core example, the memory choke point is an even greater impediment because more cores are attempting to read the phone book at the same time.

The merged CPU and DRAM graphic shows how the memory choke point is eliminated by integrating the CPU cores into main memory DRAM.

Adding more on-chip SRAM cache to a legacy architecture does not mitigate the problem because before any work can be performed, the cache must be first loaded from main memory across the choke point.

In essence:

For Big Data applications, the most important performance determinant is cores per gigabyte.

Similarities and Differences of the Two Approaches
Both Automata Processor and Celeste are fabricated in DRAM fabs, so both benefit from the very low transistor leakage and very cheap transistors.

Both make extensive use of parallelism to increase performance. AP is a specialized pattern matcher and as such will significantly outperform Celeste or any other stored program machine on such applications. Furthermore Automata's state diagram as shown in Figure 5 Example of Parallel States is in essence a directed graph such as that found in Graph500. AP could therefore be considered a small data graph analytics tool. Celeste is a general purpose CPU architecture and as such relies heavily on tools such as the gcc compiler and Linux toolchain.

Both share the performance and power consumption advantages of integrating logic with memory. AP uses its memory to store the patterns it seeks to match. The data stream must come from either a real time source, or another computer's memory. The Celeste DRAM appears as a cache of the entire main memory without the penalty power and cost penalty of large SRAM cache.

Both will almost certainly figure prominently in the future of computing.

Part II of this series investigates applications development for the two approaches.

About the author

Russell Fish's three-decade career dates from the birth of the microprocessor. One or more of his designs are licensed into most computers, cell phones, and video games manufactured today. Russell and Chuck Moore created the Sh-Boom Processor which was included in the IEEE's "25 Microchips That Shook The World". He has a BSEE from Georgia Tech and an MSEE from Arizona State.

References

1. D. A. Patterson and D. R. Ditzel, "The Case for the Reduced Instruction Set Computer," ACM SIGARCH Computer Architecture News, 8: 6, 25-33, Oct. 1980.
2. http://en.wikipedia.org/wiki/Sophie_Wilson
3. http://www.dailyfinance.com/2010/01/27/with-ipad-sounding-like-a-femine-hygiene-product-will-the-jokes/
4. http://en.wikipedia.org/wiki/Robert_H._Dennard
5. http://en.wikipedia.org/wiki/Moore%27s_law
6. http://www.slidefinder.net/f/future_computer_architecture_david_patterson/6912680
7. http://www.edn.com/design/systems-design/4368705/The-future-of-computers--Part-1-Multicore-and-the-Memory-Wall
8. http://www.edn.com/design/systems-design/4368858/Future-of-computers--Part-2-The-Power-Wall
9. http://www.edn.com/design/systems-design/4368983/Future-of-computing--Part-3-The-ILP-Wall-and-pipelines
10. http://en.wikipedia.org/wiki/Pentium_4
11. http://ark.intel.com/products/family/75023
12. http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&ved=0CDMQFjAC&url=http%3A%2F%2Fhighered.mcgraw-hill.com%2Fsites%2Fdl%2Ffree%2F0073122653%2F280053%2FCase_1_1_IntelDRAM.doc&ei=PPb7Uq7XHqj62gWQ-IDABw&usg=AFQjCNG0u0--iHfwPkzbmo7rxLFJhfh7GQ&bvm=bv.61190604,d.b2I
13. http://www.micron.com/~/media/Documents/Products/Other%20Documents/automata_processing_technical_paper.pdf
14. http://www.micron.com/~/media/Documents/Products/Other%20Documents/automata_processing_technical_paper_supplementary_material.pdf
15. http://ip.com/patfam/en/25419128
16. R. Baeza-Yates. Efficient Text Searching. PhD thesis, Dept. of Computer Science, Univ. of Waterloo, May 1989 https://cs.uwaterloo.ca/research/tr/1989/CS-89-17.pdf
17. https://www.google.com/patents/US5440749
18. http://www.micron.com/~/media/Documents/Products/Other%20Documents/automata_processing_technical_paper.pdf
19. US Patent 5,440,749, filed Aug, 3, 1989
20. http://en.wikipedia.org/wiki/Berkeley_IRAM_project
21. http://en.wikipedia.org/wiki/MapReduce
22. http://en.wikipedia.org/wiki/Graph_theory
23. http://en.wikipedia.org/wiki/Blue_Gene
24. http://www.graph500.org
25. https://share.sandia.gov/news/resources/news_releases/more-chip-cores-can-mean-s-ower-supercomputing-sandia-simulation-shows/

26. http://www.edn.com/design/systems-design/4368705/The-future-of-computers--Part-1-Multicore-and-the-Memory-Wall