Scalable processors in the billion-transistor era: IRAM

Orfeas Liossatos

1 Scalable processors in the billion-transistor era: IRAM

1.1 Paper description

In September 1997, Kozyrakis et al. claimed that the worst constraining factor of computer performance in the coming years would be memory latency [1]. Indeed, since 1987, the yearly memory latency improvement rate had been 7%, lagging behind the 55% improvement rate of microprocessor performance [2]. Many predicted that clock speed improvements would soon yield exponentially diminishing returns, so beating the Memory Wall was of central importance [3]. In this context, the Berkeley Intelligent RAM (IRAM) Project was born [4]. The IRAM approach allocates a large fraction of on-chip real estate for DRAM to act as main memory instead of less-dense SRAM caches. They hypothesized that this approach would offer high bandwidth from the very wide busless memory interface, little memory latency due to the short distance from processor to main memory, low energy consumption by avoiding off-chip memory accesses through high capacitance buses, and simpler circuitry permitting higher clock-speeds. Finally, they proposed that IRAM would be best coupled with a pairing of vector and scalar processors, which they claimed had mature compiler technology at the time, vector processors having been in use for CRAY-series supercomputers for 30 years, and would be well suited for multimedia data-hungry applications which were rising in prevalence [5].

1.2 Results

The main result of the paper is centered around a potential billion-transistor vector IRAM (VIRAM) system. Assuming a minimum feature size of 0.13 mm, die area of 400mm^2, and a 1-GHz clock, the vector unit would feature two load, one store, and two arithmetic units, with eight 64-bit pipelines each, yielding peak performance of 16 GFLOPS (for 64-bit floats). The on-chip memory would be organized into 32 sections of 16 1.5-Mbit banks for a total of 96 Mbytes, and would serve the vector unit 192 Gbytes per second with 20-ns latency. The scalar core can issue two instructions at once, and uses fast L1 caches. No concrete predictions are made about the energy consumption of this specific architecture, but a related paper by the same authors claims that IRAM consumes 22% of the energy consumed by a standard cache memory hierarchy [6]. No comparison is made between this architecture and others of a similar transistor count within this paper, but a few of related papers from Berkeley that paint partial pictures of VIRAM system performance will be discussed.

1.3 Discussion

The Berkeley IRAM project, although highly influential, had seemingly paradoxical results. As of February 2022, Google Scholar returns 1026 unique papers citing the project's seminal 1997 paper 'A Case for Intelligent RAM' [7], yet the precise combination of vector-plus-scalar processors and on-chip DRAM only saw the light of day once with the project's very own 2002 VIRAM1 chip [8]. However, this does not mean their efforts were fruitless. In fact, the pairing of vector and scalar processors appeared in the world's most powerful computer from 2001 to 2004, Earth Simulator, an NEC SX 6 with an off-chip SDRAM shared memory architecture [9]. Furthermore, IRAM has since gained the name "embedded DRAM" (eDRAM) and has in recent years been studied as a candidate memory architecture for neural network accelerators for its high memory density [10][11][12].

So why don't we see widespread adoption of VIRAM-like architectures in PCs today? With regard to *"Scalable processors in the billion-transistor era: IRAM"*, if the proposed architecture were to succeed, two major assumptions about industry readiness and compiler technology had to hold.

1.3.1 Industry

First is the problem of industry adoption of eDRAM. Processors and DRAM memory are produced by entirely different industries, with the principle manufacturers being Qualcomm, and Intel, and Nvidia for processors, and Samsung, SK Hynix, and Micron for DRAM [14][15]. Integrating both components into a single chip requires widespread industry acceptance for such a collaboration to emerge, and VIRAM's performance may not have been as compelling as other solutions, especially as the evidence of its performance is not strong. Some such results follow.

1. Simulations from Nguyen (1999) suggest that VIRAM systems perform the motion estimation algorithm from H.263 video encoding faster than some dedicated digital signal processors (DSPs) [16].

2. Herman et al. (2000) claim that VIRAM1 performs about as well as VLIW systems, with the main bottleneck being the scalar core [17].

3. Gebis et. al (2004) claim that VIRAM1 achieved 61.7 on the EEMBC's TeleBench benchmark for DSP algorithms, beating a "variety of high-performance embedded processors", although the TI C6416 achieved 379.1 just the next year, in 2003 [8][18].

Another obstacle to industry adoption may have been that integrating main memory onto the chip is an inflexible design: Extending the memory is impossible unless an external connection was provided, so the system architect must know the precise memory requirements beforehand. This makes memory into a specialized part instead of a commodity, which drives up the cost [20].

It turned out that the greatest driver of performance gains was clock speed, permitted by smaller transistors and deeper pipelines, where innovation could very easily translate into industry production, as most exemplified in 2004 by Intel's 3.8-GHz, 31-stage pipeline Pentium IV Prescott processor [19]. Therefore, due to the complexity of industrial coordination, the lack of compelling evidence for VIRAM's performance, and the existence of a promising alternative at the time, it was difficult to leverage industry forces in a push for VIRAM technology.

1.3.2 Compilers

Second is the inadequacy of automatic vectorization. Vector processors require special assembly instructions to take advantage of data parallelism and offset the loss of chip area from the larger decoder [5]. Programmers can either manually vectorize their code, or turn to a compiler to perform automatic vectorization. The difficulty of this task has been exacerbated over time, with the rise in popularity of languages that implement pointers and references such as C and Python [21]. Therefore, although many well known compilers such as GCC, LLVM, and ICC implement vectorization, it still remains an active topic of research, with some researchers even looking into AI code automatic vectorization in order to tackle so-called 'unvectorizable' code [22][23].

Despite these drawbacks, vector processors were in mainstream consumer PC use within Nvidia GPU warps, which consisted of small vector (named 'stream') processors, up until the release of the GeForce 8800 series of processors in 2006, which fit an equivalent number of scalar thread processors into the same area, eliminating vector register management and vector operation support for good [27].

1.3.3 Conclusion

For the above reasons, as well as an apparent lack of proper consideration towards the trade-off between low-latency deep cache memory hierachies and high capacity DRAM and the space of combinations inbetween, and two mistaken citations about a potential increase to memory bandwidth (50-100×, not 50-200×) and the state of the processor-memory gap (55%, not 60%), this paper was not successful in predicting the absence of embedded DRAM and vector processors in the billion-transistor era.

2 A Single-Chip Multiprocessor

2.1 Paper description

In 1997, Nayfeh et al. from Stanford University presented the case for a multiprocessor on a single chip as the best candidate for a billion-transistor architecture [26]. They began by noticing that processors mostly only exploit instruction-level parallelism in a single sequential program using out-of-order execution and dynamic branch prediction. Therefore, enlarged processors would lead to higher instruction-per-cycle rates of execution within a single thread. However, they also require quadratically more die area, siphon more energy due to the many crossbars and multiplexers, have longer design and verification times, and incur larger penalties from branch mis-predictions due to deep pipelines. By noticing emerging trends such as multimedia computing with significant loop-level parallelism, the authors advocated for microarchitectures that expose multiple threads of control. They compared two alternatives: simultaneous multithreading (SMT), and chip multiprocessors (CMP). SMT processors inherit many of the scaling problems of superscalar processors, whereas the CMP architecture can stay small and fast. Similar to Kozyrakis et al., the authors believed that wire delay would be a crucial driver of innovation, and that it would push microarchitectures into highly partitioned designs like the CMP. They believed loop-level parallelism could be converted into thread-level parallelism, but this would need software innovation in terms of language primitives and compiler support.

2.2 Results

The paper's core result stems from a simulation of three different billion-transistor architectures: superscalar, SMT, and CMP. Both the superscalar and SMT processors had 12-issue logic, with 128-Kbyte instruction and data caches connected to a level-two 8-Mbyte cache that feeds into off-chip DRAM. For the same transistor count, the CMP had 8 dual-issue CPUs, each with its own tiny 16-Kbyte instruction and data caches.

Performance evaluated on four diverse benchmarks revealed that the absence of thread-level parallelism reduced SMT to the performance of a superscalar processor and CMP to a single one of its cores. On the other hand, when thread-level parallelism was abundant, CMP maintained a slight lead over SMT due having 4 more issue slots in total. Benchmarks with high memory bandwidth demands resulted in frequent cache misses that stalled the superscalar processor, whereas SMT was able to reschedule its operation while the cache was waiting for data. Again, the CMP took the lead over SMT, thanks to each of its CPUs being able to access its own cache independently of the others. SMT's core limitation, besides wire latency, was its unified data cache architecture, which caused bandwidth issues. The quantitative results of the benchmark suggest that SMT is 6-7 times better than a baseline uniprocessor in the presence of thread-level parallelism, and CMP is 7-8 times better.

2.3 Discussion

This paper turned out to be highly predictive of the future. Nowadays, essentially every processor is multi-core, with the top 50 most popular processors of all time listed on pc-builds.com offering four cores or more [28]. Although the authors realize that a CMP processor may run more threads than SMT for the same transistor count, the authors do not push the concept to the extreme, as eventually happened in GPUs, with the first billion-transistor NVIDIA GeForce GTX 260 exposing parallel 216 stream processors [40]. Multiprocessing was not a new problem for computer architects, as they were already familiar with the difficulties of shared-memory multiprocessing (SMP), where multiple CPUs maintain cache coherency by snooping a common bus. An example SMP system was DECs VAX 8820/8840/8860 from 1988 [41].

It takes only minor insight to see that shrinking the size of each CPU could allow them to live on the same chip and pay rent in terms of low-latency communication and low power usage. But it takes major insight to predict that the industry was ready for such a move. As stated above, the primary driver for innovation in computer chips soon after 1997 was clock speed. In 2004, Intel's 3.8-GHz Pentium 4 Prescott actually ran at a higher rate than today's Apple M1 3.2-GHz high-performance cores [19][35]. Therefore, the authors must have predicted that when clock speed improvement would slow down, the industry would make a move towards multicore processors, where they correctly identify that the repetitive, simpler design allows for less time spent in the hardware verification stage [42].

Although the authors did in fact succeed in predicting which microprocessor architecture would become ubiquitous, there also existed friction in the adoption of multi-chip processors that remained unstated in the paper, and which should have pushed their 10 year prior estimate backwards by a couple of years. First, an additional element of risk must enter the probability industry adoption. If a faulty core is grouped with a set of functioning cores, the entire chip must be discarded, which lowers total chip yield. And second, the upgrade from single to multi-threaded programming is nontrivial, and time must be invested by programmers to learn parallel primitives and refactor their code. Unfortunately, data dependencies and out-of-order execution can result in any number of complex software effects like deadlocks that are difficult to debug. We do in fact see that the first commercially available 8-core, 1.2-billion transistor computer, IBM's POWER7 32M L3, was released in 2010 [43]. As a side note, the POWER7 also used eDRAM as an L3 cache.

2.3.1 Conclusion

Overall, this paper provides a balanced and quantitative comparison between the two multithreading alternatives, along many axes of analysis ranging from low-level wiring considerations all the way to the number of engineers needed to validate the designs. Its evidence-backed forecasting is therefore highly accurate.

3 Forecast

According to Kahneman (2003), a prediction may be overly optimistic due to a lack of consideration towards distributional information like risk [29]. Therefore, in an attempt to avoid taking the "inside view" with specific timelines about microprocessors that I personally find exciting, I will take a look at some material, architecture, and software trends in order to paint a picture of the future.

First, various forms of scaling will fast be approaching their end, with Moore's law predicted to falter by 2025 [30]. If there is still room at the bottom, the transistor that will constitute the microprocessors of 2030 either already exists in a lab somewhere, given that it took the GAA-FET transistor 10 years after its sub 5 nm demonstration in 2006 to be used in a commercial processor [31][32], or there is currently no scalable replacement for conventional transistors beyond searching for exotic, novel degrees of transistor freedom from electric charge, with spin and valley technology [33][34]. If the latter case turns out to be correct, then scaling must be sought elsewhere, beyond today's massively parallel systems.

A major trend that is still underway is the push towards extreme hardware specialization and co-packaged heterogeneous accelerators. Apple's M1 processor combines high-performance cores, energy-efficient cores, a GPU, and a 16-core neural engine all in one [35]. Computers which aren't equivalent to Turing Machines can lower theoretical limits of certain extremely common algorithms such as graph search and integer factorization, which can be solved efficiently on neuromorphic and quantum computers, respectively [36][37]. These special processors may also be packaged on-chip along with any number of fantastical processors.

However, testing and offsetting production errors from such a complicated combination incurs a cost which manufacturers will try to mitigate by pivoting their factories towards an increasingly modular approach to chip production: Chiplets [38]. The programmer must also be able to interact freely with each of these computing units, which will require activating new programming environments similar to GPU Instancer within Unity [39]. Finally, on the more speculative side of hyper-specialization, it is plausible that certain processors will drop programmability in favour of extreme performance and security. Perhaps this will be the case for brain-computer interfaces with read-write capabilities, where the effect of an arbitrary code injection attack would be disastrous.

References

[1] Kozyrakis, Christoforos E., et al. *Scalable processors in the billion-transistor era: IRAM.* Computer 30.9 (1997): 75-78.

[2] J.L. Hennessy and D.A. Patterson. *Computer Architecture: A Quantitative Approach*, 2nd ed., Morgan Kaufmann, San Mateo, Calif., 1996.

[3] Wulf, Wm A., and Sally A. McKee. *Hitting the memory wall: Implications of the obvious.* ACM SIGARCH computer architecture news 23.1 (1995): 20-24.

[4] Berkeley Intelligent RAM Project. `http://iram.cs.berkeley.edu/index.html`. Retrieved 04-02-2022.

[5] Russell, Richard M. *The CRAY-1 computer system.* Communications of the ACM 21.1 (1978): 63-72.

[6] Fromm, Richard, et al. *The energy efficiency of IRAM architectures.* ACM SIGARCH Computer Architecture News 25.2 (1997): 327-337.

[7] Patterson, David, et al. *A case for intelligent RAM.* IEEE micro 17.2 (1997): 34-44.

[8] Gebis, Joseph, et al. *Viram1: A media-oriented vector processor with embedded dram.* DAC04 (2004): 7-11.

[9] Oliker, Leonid, et al. *Performance evaluation of the SX-6 vector architecture for scientific computations.* Concurrency and Computation: Practice and Experience 17.1 (2005): 69-93.

[10] Judd, Patrick, et al. *Stripes: Bit-serial deep neural network computing.* 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). IEEE, 2016.

[11] Tu, Fengbin, et al. *RANA: Towards efficient neural acceleration with refresh-optimized embedded DRAM.* 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA). IEEE, 2018.

[12] Yoo, Taegeun, et al. *A logic compatible 4T dual embedded DRAM array for in-memory computation of deep neural networks.* 2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED). IEEE, 2019.

[13] 7-cpu. `https://www.7-cpu.com/cpu/P6.html`. Retrieved 04-02-2022.

[14] Kim, Jae-Kyung, Jon-Mo Yoon, and Bong-Soo Lee. *Assessment of Competitive Edge of Major Global Semiconductor Vendors for Self-Driving Solutions (Level 3 and Above)-Evaluation of Qualcomm, Intel, and Nvidia.* Intel, and Nvidia, Asia-pacific Journal of Convergent Research Interchange 6.10 (2020): 165-180.

[15] Bruce Jacob, David Wang, and Spencer Ng . *Memory Systems: Cache, DRAM, Disk.* Morgan Kaufmann. (2010) ISBN 978-0-08-055384-9.

[16] Nguyen, Thinh PQ, Avideh Zakhor, and Kathy Yelick. *Performance Analysis of an H. 263 video encoder for VIRAM.* Departament of Electrical Engineering and Computer Ciences (University of California at Berkeley) (1999).

[17] Herman, Jeffrey, John Loo, and Xiaoyi Tang. *A Comparison of the VIRAM-1 and VLIW Architectures for use on Singular Value Decomposition.* University of California, Berkeley 94720 (1776).

[18] EEBMC TeleBench. `https://www.eembc.org/benchmark/index_old.php`. Retrieved 04-02-2022.

[19] Burger, Doug, and James R. Goodman. *Billion-transistor architectures: There and back again.* Computer 37.3 (2004): 22-28.

[20] Keitel-Schulz, Doris, and Norbert Wehn. *Embedded DRAM development: Technology, physical design, and application issues.* IEEE Design & Test of Computers 18.3 (2001): 7-15.

[21] Auto-vectorization in GCC. `https://gcc.gnu.org/projects/tree-ssa/vectorization.html`. Retrieved 04-02-2022.

[22] Feng, Jing Ge, Ye Ping He, and Qiu Ming Tao. *Evaluation of Compilers' Capability of Automatic Vectorization Based on Source Code Analysis.* Scientific Programming 2021 (2021).

[23] Stock, Kevin, Louis-Noël Pouchet, and P. Sadayappan. *Using machine learning to improve automatic vectorization.* ACM Transactions on Architecture and Code Optimization (TACO) 8.4 (2012): 1-23.

[24] Hou, Kaixi, Hao Wang, and Wu-chun Feng. *A framework for the automatic vectorization of parallel sort on x86-based processors.* IEEE Transactions on Parallel and Distributed Systems 29.5 (2018): 958-972.

[25] Francis, Rhys S., and Ian D. Mathieson. *A benchmark parallel sort for shared memory multiprocessors.* IEEE Transactions on Computers 37.12 (1988): 1619-1626.

[26] Nayfeh, B. A., and K. Olukotun. *A single-chip multiprocessor.* Computer 30.9 (1997): 79-85.

[27] Nickolls, John, and William J. Dally. *The GPU computing era.* IEEE micro 30.2 (2010): 56-69.

[28] PC builds. *Most popular Processors / CPUs.* All time. `https://pc-builds.com/most-popular/processors/`. Retrieved 04-02-2022.

[29] Lovallo, D; Kahneman, D (2003). *Delusions of success. How optimism undermines executives' decisions.* Harvard Business Review. 81 (7): 56–63. PMID 12858711.

[30] Shalf, John. *The future of computing beyond Moore's law.* Philosophical Transactions of the Royal Society A 378.2166 (2020): 20190061.

[31] Lee, Hyunjin, et al. *Sub-5nm all-around gate FinFET for ultimate scaling.* 2006 Symposium on VLSI Technology, 2006. Digest of Technical Papers.. IEEE, 2006.

[32] Sebastian, Anthony. *IBM unveils world's first 5nm chip.* Ars Technica. (2017) Retrieved 04-02-2022.

[33] Sugahara, Satoshi, and Junsaku Nitta. *Spin-transistor electronics: An overview and outlook*. Proceedings of the IEEE 98.12 (2010): 2124-2154.

[34] Li, Lingfei, et al. *Room-temperature valleytronic transistor*. Nature nanotechnology 15.9 (2020): 743-749.

[35] Apple. *Apple unleashes M1.* `https://www.apple.com/newsroom/2020/11/apple-unleashes-m1/`. Retrieved 04-02-2022.

[36] Davies, Mike, et al. *Advancing neuromorphic computing with loihi: A survey of results and outlook*. Proceedings of the IEEE 109.5 (2021): 911-934.

[37] Pirandola, Stefano, et al. *Advances in quantum cryptography*. Advances in optics and photonics 12.4 (2020): 1012-1236.

[38] Mounce, Gabriel, et al. *Chiplet based approach for heterogeneous processing and packaging architectures.* 2016 IEEE Aerospace Conference. IEEE, 2016.

[39] Gurbu. Introducing GPU Instancer (2018). `https://www.gurbu.com/`. Retrieved 04-02-2022.

[40] btarunr. *GeForce GTX 260 with 216 Stream Processors Pictured, Benchmarked.* Tech Power Up. (2008) `https://www.techpowerup.com/71319/geforce-gtx-260-with-216-stream-processors-pictured-benchmarked`. Retrieved 04-02-2022.

[41] DEC. *VAX 8820/8840/8860 System Hardware User's Guide.* (1988) `http://www.bitsavers.org/pdf/dec/vax/8800/EK-8840H-UG-001_88xx_System_Hardware_Users_Guide_Mar88.pdf`. Retrieved 04-02-2022.

[42] Foster, Harry D. *Trends in functional verification: A 2014 industry study.* Proceedings of the 52nd Annual Design Automation Conference. 2015.

[43] Kanter, David. *New Information on POWER7.* (2009) (`https://www.realworldtech.com/hot-chips-2009-preview/2/` Retrieved 04-02-2022.