# Taming the Beast: Memory Efficiency in an AI/Crypto World

By Bill Gervasi, Principal Memory Solutions Architect

The planet is facing a crisis in energy demand versus supply, and datacenters are at the center of this dilemma due to the increasing demand from new data-intensive applications. This article will explore the causes of datacenter inefficiency and speculate on methods to improve that efficiency. It will also acknowledge the US Department of Energy's analysis on energy efficiency, which provides a basis for this work. (1)

## Energy Demand and Where It's Being Used

The announcement that Three Mile Island nuclear reactor was being recommissioned to power an artificial intelligence (AI) datacenter might have been shocking news to some, but it is no secret in the industry that the exploding demand for energy is outpacing our ability to deliver power to datacenters. For the first time, power efficiency is now a higher priority to datacenter architects than performance of the individual components.

The Semiconductor Research Corporation modeled this increase in energy demand in the context of the planet's projected energy generation capacity, which includes the assumption that more nuclear power plants will be deployed. (2) Figure 1 shows a daunting projection, and the potential for the lines of supply and demand to intersect around the year 2055 has the electronics industry rethinking its choices in how datacenters can be designed.

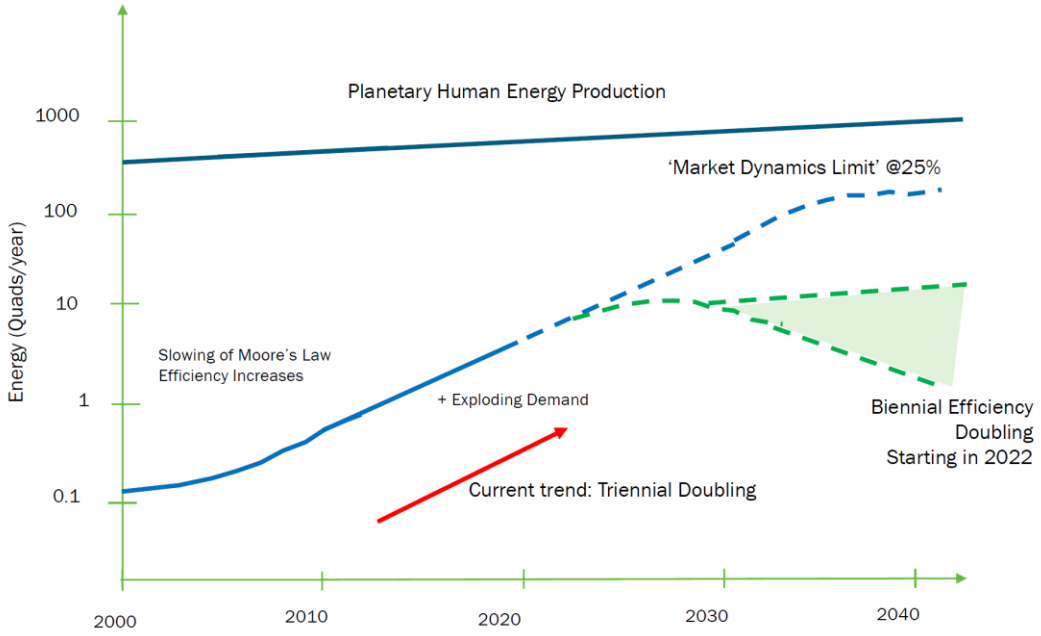


**Figure 1: Worldwide Energy Consumption Trends** (2)

Shankar at Stanford University broke down the places where we are spending that energy. (3) In addition to AI, another culprit in energy demand is cryptocurrency. When combined, AI and crypto are consuming over 1.5% of the planet's energy already. Some projections estimate that that their data consumption will increase to 3% by 2030 and 4.4% by 2035 (see Figure 2). (1) Note the scaling for the Y-axis in Figure 2: Applications such as cryptocoin mining require 18 orders of magnitude more energy than the base instructions on which the computers operate.

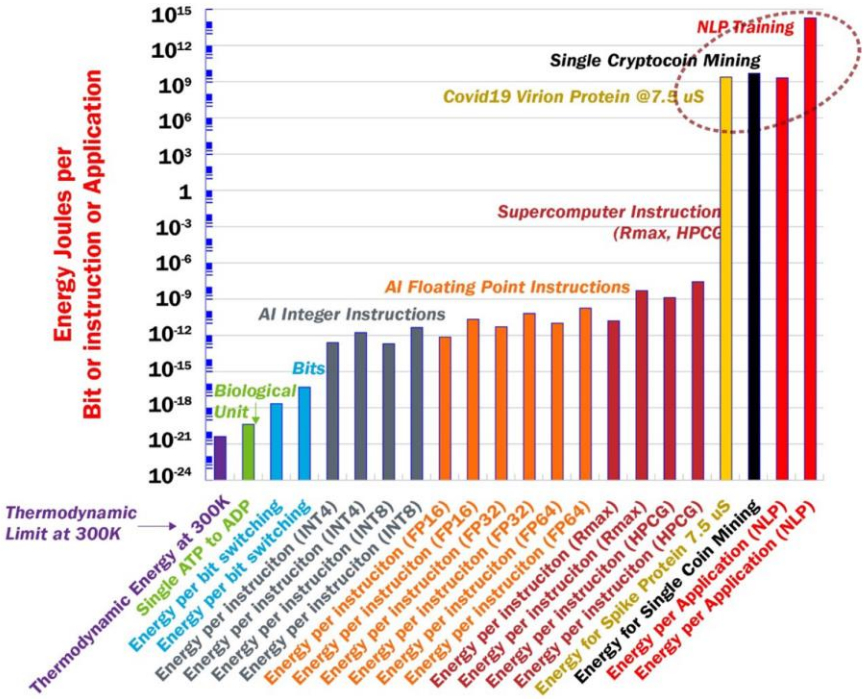


**Figure 2: Energy Consumption by Operation** (1)

With this in mind, it makes sense to determine the efficiency of a datacenter by measuring the work accomplished for each watt that is spent. Figure 3 breaks down the power consumption per operation. (1) It is critical to note that almost every operation in the top two-thirds of the table refer to moving data around, while the bottom third of the table represents data processing.

| Operation | Energy per bit |
|---|---|
| Wireless data | 10 – 30µJ |
| Internet: access | 40 – 80nJ |
| Internet: routing | 20nJ |
| Internet: optical WDM links | 3nJ |
| Reading DRAM | 5pJ |
| Communicating off chip | 1 – 20 pJ |
| Data link multiplexing and timing circuits | ~ 2 pJ |
| Communicating across chip | 600 fJ |
| Floating point operation | 100fJ |
| Energy in DRAM cell | 10fJ |
| Switching CMOS gate | ~50aJ – 3fJ |
| 1 electron at 1V, or 1 photon @1eV | 0.16aJ (160zJ) |

**Figure 3: Energy Use by System Function** (4)

The memory, storage, and communications hierarchy is commonly shown as a pyramid, with processor registers at the top, various levels of cache followed by DRAM, then storage and communications at the bottom (see Figure 5). This article will use this simplistic model later. The pyramid's biggest issue is that it does not highlight how each resource is on a separate bus. In addition, moving information from one resource to another typically involves multiple movements on many buses, each of which consumes power and generates heat.

Figure 4 shows an example in which an application is read from the disk though the CPU across one channel (e.g. a PCIe) to be written to the memory over another channel (e.g. a DDR), only to be read back to the CPU one cache line at a time to execute the application and store the temporary results back to the memory. The application may read content across a communications channel, such as PCIe to a wide area network, then crunch that data to be written back to the disk. Even in this simple example, it is obvious that the data processing is an exceptionally minor outcome and that data movement is dominant. The percentage of data operated upon rather than moved around is so close to zero as to be unmeasurable.

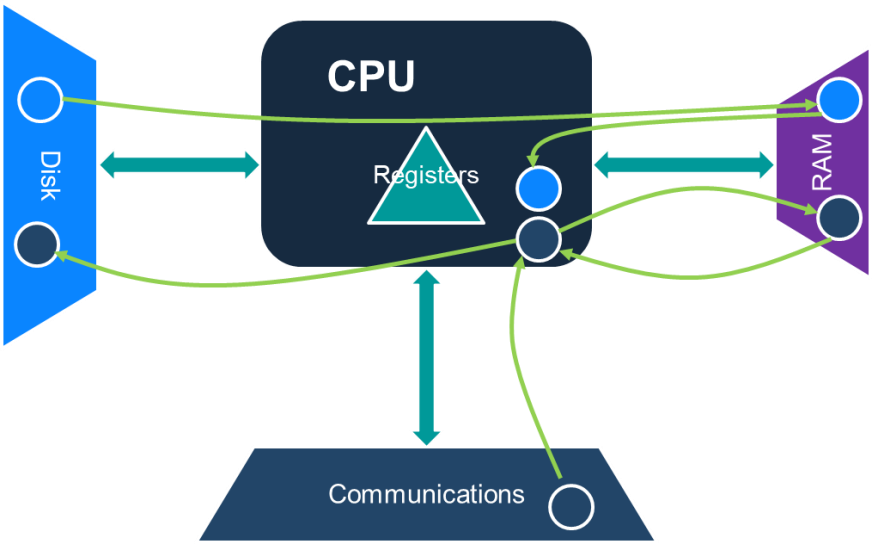


**Figure 4: Simple Application Data Movement** (5)

### Why Focus on Memory?

Memory utilization is a focus area because there is a high potential to make substantial improvements in energy efficiency. Memory consumes as much power as many CPUs, at about 22% of server power. The increasing number of tiers of memory creates both the best and worst of trends. The good news is that more power-efficient memories are being added closer to the processor. The bad news is that these near-memory tiers have limited capacity and require additional larger capacity, higher power memories to keep filling the data sets into the local memory. The power consumption of each tier adds to the total power footprint.

High bandwidth memory, for example, offers an interface around 1.5pJ/bit, which compares favorably to a double data rate memory module at 15pJ/bit (see Figure 5). Unfortunately, these memories still burn significant power (e.g. 75W or 100W per HBM stack) and they are co-located with the high-power processor on the same substrate. This makes cooling extremely challenging compared to DDR modules, which are around 15W each but located farther from the processor in areas that may be air-cooled.

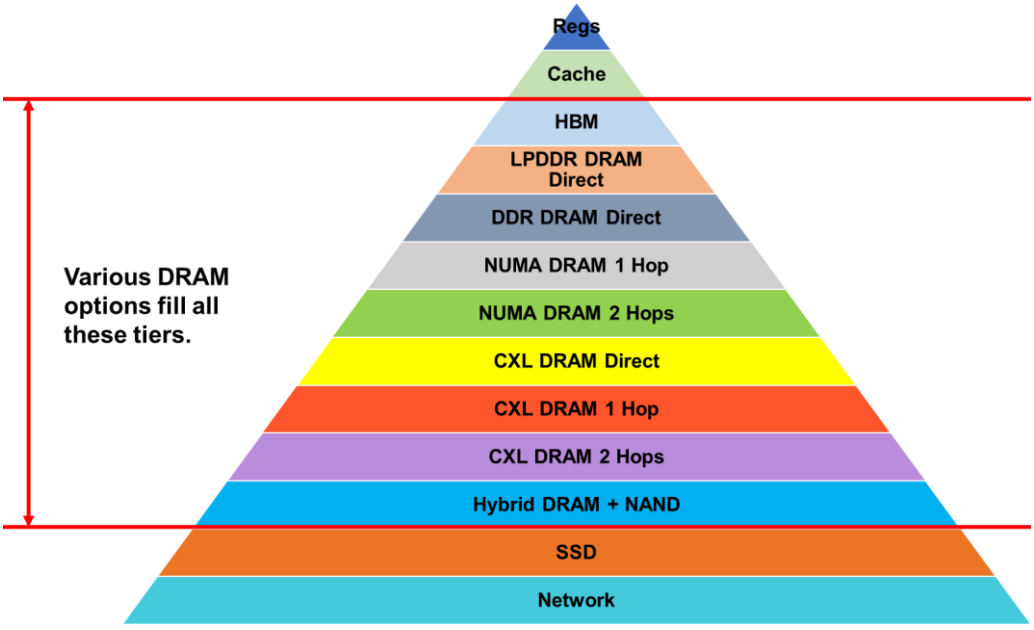


**Figure 5: Memory, Storage, and Network Tiers**

### Efficiency by Tier

Speculation can improve system performance tremendously, but speculation always implies waste as well — even processor registers have implied waste. A system variable with a 32-bit integer that never assumes a value outside the range 1 to 10 has an implied waste factor of 87.5%. Processor caches have very high hit rates of 95% and higher, so one could invert that number to imply a 5% waste. DRAM access efficiency drops the further the memory is from the processor, with direct attached DDR memory at 27% waste and CXL-attached DDR at over 40% waste.

These numbers may not sound bad until one considers the activity inside each DRAM that allows cache line hit rates. The majority of processors operate with a 64-byte cache line. Consider how 64 bytes map to the internal structure of a DRAM. Each DRAM has an internal page buffer of 1kB, and DRAMs are typically combined into ranks for 10 DRAMs energized per access (see Figure 6). To fulfill a single cache line, a DRAM module is “activated” to read 1kB from each DRAM into its sense amplifiers, or 10kB across the width of the module. 64 bytes are read and sent to the processor. DRAM activation is destructive — the cells of the memory core are wiped out by the activation — so the cells must be rewritten from the sense amplifiers back into the core. The math for a single random access is 20kB moved for 64 bytes of work, or 99.7% waste.

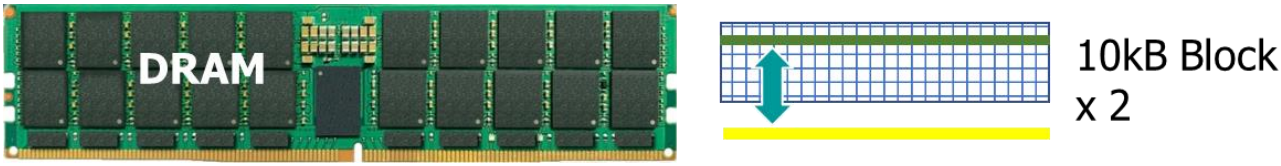


**Figure 6: Server DRAM Module and Page Buffer Activation Overhead**

This factor of 0.3% efficiency is only against that movement of a 64-byte cache line. If that DRAM tier is operating at a 60% hit rate, efficiency drops to 0.18%. If only 1 byte from that cache line was actually needed, the waste factor increases to 99.98%. As you can see in this simple example, datacenter efficiency is rapidly approaching zero.

Another form of speculation that improves system performance is execution and access speculation, where a processor may pre-load code on both sides of a branch condition in case the branch is taken. Many SSDs do the same, pre-loading pages that may be accessed. These forms of speculation have

100% waste if the branch is not taken or the access is never made.

### Total Cost of Ownership (TCO)

With electricity access becoming a bottleneck for datacenter expansion, architects are finally acknowledging that total cost of ownership (TCO) is a primary factor driving system design. While processor vendors focus strictly on performance, their customers are forced to determine whether they can power these machines and cool them. By some estimates, cooling a datacenter is currently consuming 43% of the cost of operating a datacenter, which is equivalent to the 43% required to run the machines themselves. (6)

This expenditure is driving architects to measure efficiency not only as petaFLOPS/second but also petaFLOPS/watt-hour.

### Improving Memory Energy Efficiency

Improving the accuracy of speculative accesses is an obvious key to taming memory subsystem power consumption. Similar to telling a doctor “It hurts when I do this,” system architects should ask the question, “Is this speculative access successful often enough to pay for the energy consumed?” For example, if a CXL memory module is in a memory pool and shared by multiple processors, what is the hit rate on any particular bank of DRAM? Should a page be left open (delaying precharge in case of another hit on that row of memory) or be closed (issuing the pre-charge immediately under the assumption it will not be accessed)?

Non-uniform memory access (NUMA) has been in server architectures for years to allow tightly coupled processors to share memory resources as demand shifts. However, multiple hops for each memory access can more than triple the power consumed, whereas moving the task to a processor closer to the memory resource can significantly reduce power (see Figure 7). Computational storage is a variation of task relocation that has had some success, though this success is limited by standards for the tasks executed on the devices.

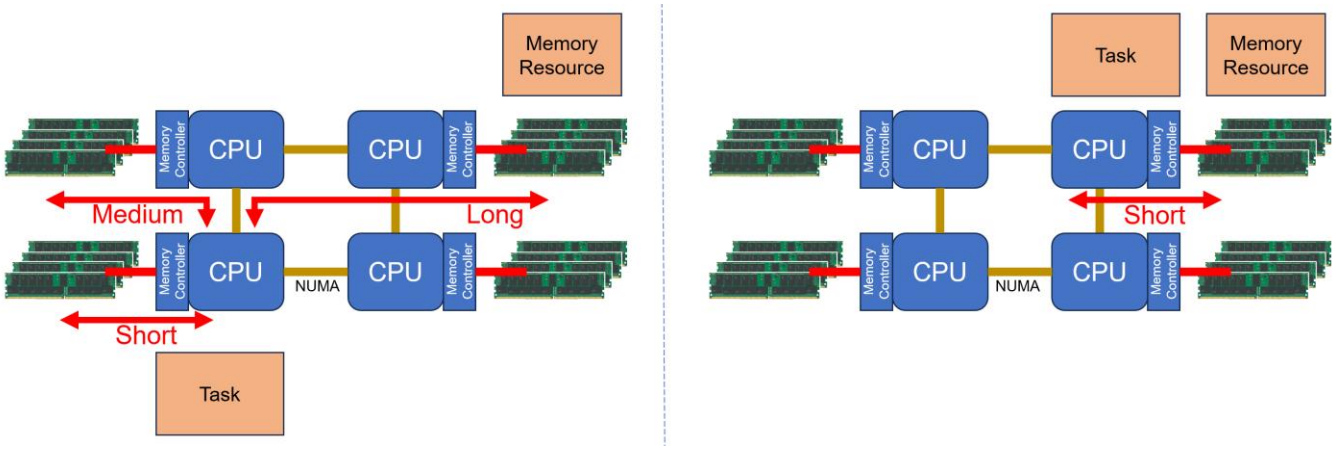


**Figure 7: Server DRAM Module and Page Buffer Activation Overhead**

Similarly, placing data in the appropriate tier of memory can have a significant impact on energy consumption. Figure 8 shows the temperature of the data, where hot data is accessed often, and cold data is accessed less often.

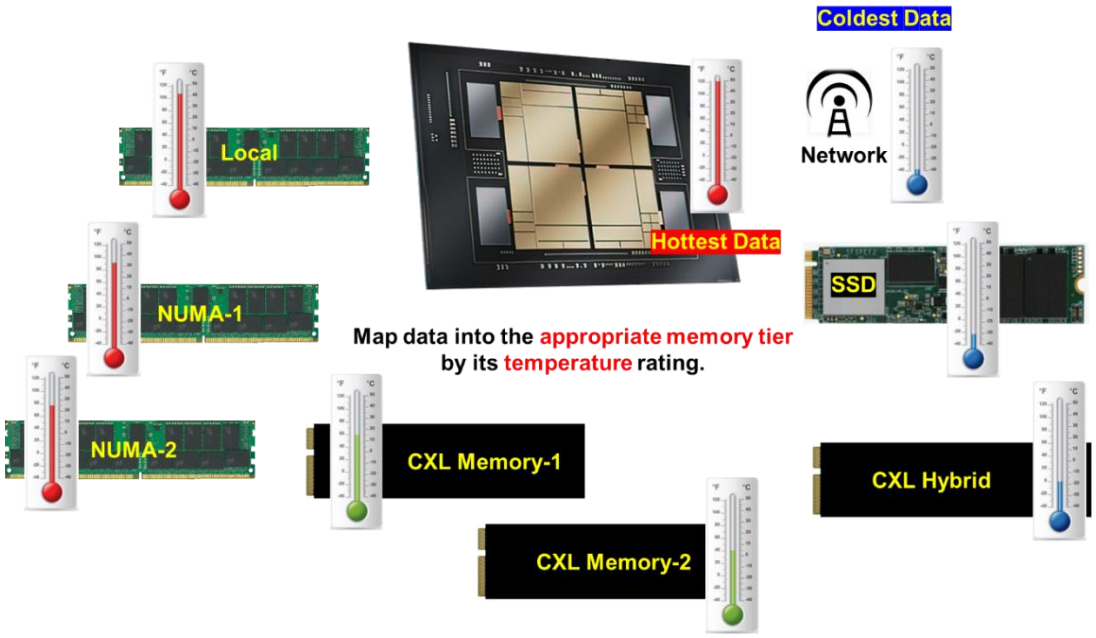


**Figure 8: Tiering Memory by Temperature**

Persistent memory is a system option that can be exploited for data reliability. Persistent memory is either based on a memory technology that does not lose its contents if the power fails (e.g. MRAM) or uses an energy source to maintain data integrity by saving DRAM contents in a non-volatile memory (NVM), such as a flash-on power failure. Persistent memory can also be thought of as a significant way to reduce system power by eliminating the need for “checkpointing,” or saving intermediate results (see Figure 9). In many systems, checkpointing is responsible for 7% to 8% of the system traffic and therefore power.

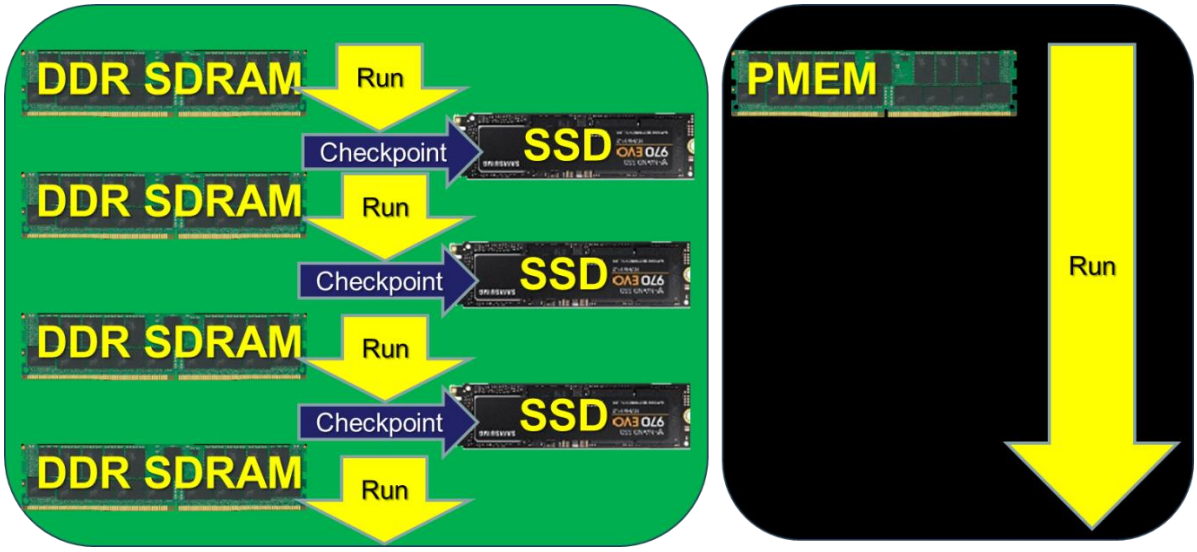


**Figure 9: Persistent Memory Can Reduce Checkpointing**

Hybrid memory modules that combine storage and direct access memory on the same module are available to minimize system traffic as well. For example, flash memory mounted as an SSD can be coupled with DRAM, which is directly accessed a cache line at a time. The efficiency of hybrid modules comes from the statistic that of the typical 4kB block moved from SSD to system memory, only 100 bytes on average are used, which results in an efficiency of only 2.5%.

## Software Has a Huge Impact on Efficiency

Hardware cannot fix every challenge; software plays a significant role in taming this beast, too. Zooming in on the power consumed by data type, orders of magnitude more power are used for complex and large data types such as floating point, whereas integer math consumes far less power (see Figure 10). This may be as simple as programmers considering the range of values needed by variables in their software. For example, “for (i=0; i<10; i++)” does not need for i to use a 32-bit counter value.

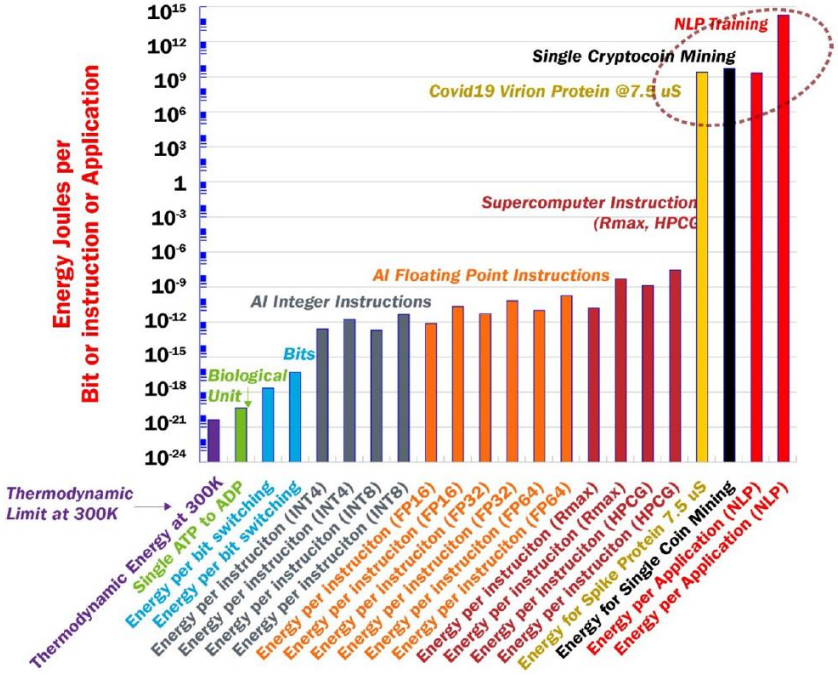


**Figure 10: Zoom-In on Energy Use by Processor Operation** (1)

The choice of variable types is sometimes the result of using the wrong programming language for the task (see Figure 11). Not all programming languages allow much flexibility in choosing the data types for variables, and these impacts are magnified tremendously by the matrix math employed by languages such as Python, a common tool for AI applications. Python has another energy-consuming characteristic: the programmer source is compiled to bytecode then interpreted by a virtual machine as opposed to C programming, which compiles to processor native codes.

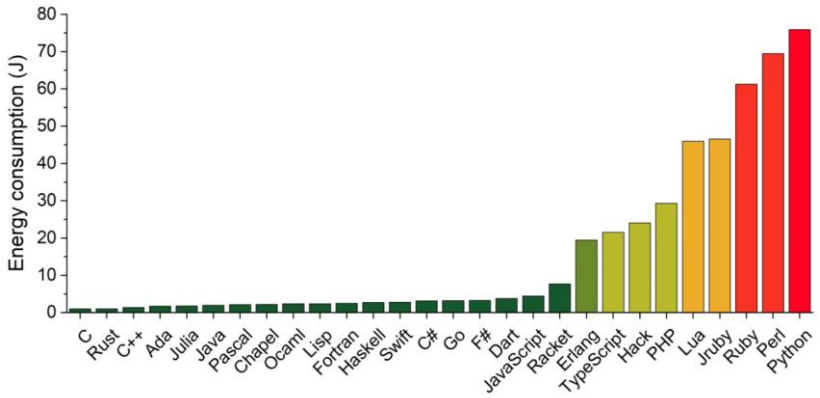


**Figure 11: Relative Efficiency of Programming Languages** (7)

## You Can't Fix What You Can't Measure

Measuring runtime power is a key to tuning efficiency. MPS's voltage regulators for memory modules, such as the MPQ8894, MPQ8895, and MPQ8896, are power management integrated circuits (PMICs) with an integrated system management interface to $I^2C$, $I^3C$, or SidebandBus. This system management interface allows the host system to interrogate the PMIC while the system is running. The current used by each voltage rail can be read from the PMIC to calculate the total power for the memory module while running test and measurement programs, or even while customer applications are running.

Triggers may be configured into the PMICs, and these devices can keep logs of any conditions that exceed the expected maximums. The host system may respond to the triggers by reading the telemetry registers then acting on those conditions, such as by throttling applications that exceed system-imposed limits.

Choosing an MPS PMIC is a power-saving measure. With improved 4% power regulation efficiency when compared to competing solutions, this results in a total datacenter power reduction of 2%. For a typical 300 megawatt-hour installation, this would reduce power by 6MWh and $CO_2$ emissions by roughly 4 metric tons per year. (8)

## Conclusions

Datacenters are projected to keep increasing power demands until they become physically or financially impossible to expand. The total cost of ownership has become a focus for all datacenter architects as they balance the needs for performance from their customers with the reality of providing those services in a cost-effective manner.

Datacenter efficiency, as measured by the data processed vs. data moved around, is embarrassingly low. However there are a number of ways to adjust efficiency, from cache management parameters to speculation priorities. Resource and job allocation over fabrics such as NUMA and CXL enable new classes of optimization.

The careful selection of energy efficient components such as MPS voltage regulators can play a significant role in reducing the energy use of a data center. Every percentage of efficiency improvement leads to major reductions in $CO_2$ emissions, a leading cause of pollution. MPS regulators take a holistic view of the system solution, providing high efficiency coupled with methods for measuring and fine tuning the solution to achieve optimal power savings.

Software plays a huge role in efficiency as well, from the low-level allocation of data types to the choice of programming languages for each task. In addition, measuring system efficiency at runtime helps datacenter operators monitor the health of the system and give insight into ways to improve or limit power as needed. MPS telemetry information helps system software understand where energy is being used.

Most importantly, TCO analysis requires a change in mindset from operations per second to operations per watt-hour, a major shift forced on the industry by skyrocketing power demand. The use of MPS's high efficiency voltage regulators helps reduce datacenter energy usage, which lowers the cost of providing data services.

**Notes:**


1) Energy Efficiency Scaling for Two Decades Research and Development Roadmap. (2024, August). U.S. Department of Energy. Retrieved November 7, 2025, from https://www.energy.gov/sites/default/files/2024-08/Draft_EES2_Roadmap_AMMTO_August29_2024.pdf
2) Kaarsberg, T. (2023, March 16). Microelectronics' Energy Efficiency Scaling for 2 Decades (EES2) Pledge and WG Day 2 Closing. Stanford. Retrieved November 7, 2025, from https://ees2.slac.stanford.edu/sites/default/files/2023-11/Kaarsberg%20Closing%20EES2%20Mar%2016.pdf
3) Shankar, S. (2024, November 13). Energy of Computing: Unsustainable Trends and Potential Solutions. Stanford Energy Seminar. https://events.stanford.edu/event/the-stanford-energy-seminar-Sadasivan
4) Gervasi, B. (n.d.). How CXL Lets Us Do Controller Memory Buffers the Right Way. Wolley Inc. Retrieved November 7, 2025, from https://files.futurememorystorage.com/proceedings/2024/20240806_DCTR-102-1_Gervasi.pdf
5) Gervasi, B. (2024, July 17). Averting a System and Software Induced Energy Crisis. https://life.ieee.org/event/systems-and-software-for-averting-an-energy-crisis-ieee-occs-oc-acm-mtg/

6) Gervasi, B. (2022). A Persistent CXL Memory Module with DRAM Performance. https://conferenceconcepts.app.box.com/s/yweter1sxdsqt4ynlu1go9cta36luulz
7) Kumar, A. (2024, August 3). Top 5 most energy efficient coding languages - WireUnwired Research. WireUnwired. https://wireunwired.com/top-5-most-energy-efficient-coding-languages/
8) Greenhouse Gas Equivalencies Calculator | US EPA. (2025, February 24). US EPA. https://www.epa.gov/energy/greenhouse-gas-equivalencies-calculator#results