The Technology of Near-Zero Energy Computing

VAINE

A Short History of Computing

The Semiconductor Industry has two significant pressures: the **pull** and the **push**. The pull continues to set the goal for faster computation, more storage, and better communication. Our demands for more data, more intelligently handled with ever-increasing computation efficiency, drives progress. The industry creates perceived cycles to have a planned and scheduled buying cadence. The push comes from advances in materials science, miniaturization, and architecture. These innovations are complex since we cannot plan advances in all fields. Moore's law, published in 1965, has provided the interface between the push-pull worlds by setting an economic observation that transistor density doubles every 24 months. This observation assumes we can reduce costs and improve performance through geometry shrinkage. The industry has relied on this manufacturing observation for the past 50 years.

The transistor was created by Bell Labs in 1947, made from germanium. We quickly progressed to silicon in 1954, which has driven most of the industry ever since. Unfortunately, more recently, our advancements have hit physical walls, stalling our economic assumptions; transistors at geometry levels of 5nm and below are increasingly more costly—in heat, energy, and manufacturing. Even if we can create the transistor density as advertised, we cannot handle the heat from all those switching transistors. We must carefully lay out transistors to avoid excessive heat areas or apply increasingly more sophisticated and expensive cooling methods. Heat and power limit the frequency and duration of high-performance activities, from a cellphone to a supercomputer. The sheer number of transistors means we cannot power all the transistors all the time, so there is not enough power to go around. This problem is known as **dark silicon**. It becomes necessary to carefully select which regions to enable on a case-by-case basis. We have heat and power issues from the transistor level, which is getting worse and more expensive. The pull pressure drives us closer and closer to the physical constraints of silicon. Significant advances are scarce, and the industry announces small increments as substantial achievements.

From the architecture side, scalar computing has not changed significantly since 2006, so a computer from 2006 runs roughly at the same scalar speed as a computer

today. Performance improvements have remained gradual with frequency increases. Geometry scaling was causing issues in power demand and heat dissipation. This problem is called the **end of Dennard's Scaling**, which occurred between 2005 and 2007. Architectures went towards parallelism as the solution. Mitigating the performance constraint by doing more in parallel and exploiting transistor density with duplication. We applied parallelism from data to systems level design using specialized operations (like Single Instruction Multiple Data) to multi/many-core designs (Symmetric Multi-Processors). This change also affected the governing law of performance, with the 1967 **Amdahl's law** appearing dominant. Amdahl's law states that the parts constraining parallel speedup are the ones we cannot parallelize, i.e., the serial portion. This law has historical roots, dating back to the mid-18th century **Law of Diminishing Returns**. More processors beyond a certain threshold does not mean linear performance gains.

We should briefly mention historical changes in data widths and types. The widths have changed due to capability and requirements; a small subset of the widths chosen are 4, 8, 14, 16, 24, 32, 36, 64, 80, 128, 256-bits, etc. These widths come with overheads in time, complexity, and storage requirements. The primary data types have also changed: integer, fixed-point, and floating-point. The types (and ranges) have changed due to cost and requirements. And potentially, in the future, more exotic types or vastly wider data widths may become necessary.

As Moore's law slowed, so did the rate at which general parallelism occurred, opening the door to more specialization. Specialization occurs at different levels: different processors with varying levels of efficiency (combining big and small cores), dedicated accelerators targeting specific applications, and lastly, moving the computation nearer to the data with **Near-Memory Computing**. Taking advantage of the simple rule, the nearer we move computation to the data, the faster and more efficient the process.

As the complexity of semiconductor chips has increased, there is a need to manage the design. Technologies such as chiplets have emerged to help integrate all the accelerators and compute elements by providing a general connection method using

what we call an interposer, opening up the ability to optimize specific functions. In 2000, IBM first discussed the possibility of using chiplets. By 2018, chiplets solutions started to appear. This change allows for much more variability by reducing the verification task and helping to satisfy the increasing pull pressure. Verification is a significant part of the overall development process, and by removing some of the duplication, we reduce the design time.

In summary, we have achieved impressive advances, but those advancements are struggling. Moore's law is ending, parallel computing is hitting boundaries, performance increases are slowing down, and heat is increasingly hindering progress. We are running out of tricks and relying more and more on incremental improvement. As we increase transistor density, we add heat and power issues. This pattern will continue to worsen, limiting our ability to improve actual performance and satisfy future demands. These fundamental limitations are caused by using concepts chosen decades ago. As we consider moving from 2D to 3D chip design, the problems become more acute, so what follows?

Adiabatic Reversible Computing

We need a new way to think about how computation occurs. As pointed out, traditional computation methods are coming to an end. Whether it is geometry shrinkage or architecture improvements, we struggle to maintain the improvement curves. Looking more deeply at the problem, we see two major issues: energy and heat. Both limit the rate of improvement. We need a computation technology that dramatically reduces heat and energy consumption. We can address this problem by introducing a technology called Adiabatic Reversible Computing.

Adiabatic Reversible Computing has theoretical origins from famous Physicists such as John von Neumann, Rolf Landauer, Tom Knight, Edward Fredkin, Tommaso Toffoli, Norman Margolus, Richard Feynman, and Charles Bennett, and some of these concepts ended up in Quantum Computing. In our case, we are applying these concepts to a broader area of computation and implementing them in standard CMOS processes.

The concept is to take the initial theoretical ideas and make them a reality in modern electronics. Dramatically improving energy efficiency and removing heat from the circuits. This change frees the designs, allowing us to increase the scale and performance from mobile phones to supercomputers, and allowing us to handle advanced workloads such as AI. The umbrella term (and visionary goal) we use for this objective is **Near-Zero Energy (NZE)** computing; a type of computing that is thermodynamically-aware of the implications of computation.

Why reversing is so important?

First, before diving into reverse, let us discuss forwards. Forward progression is essential. We run or walk to a location, computation is forward-focused, and time moves forward, not backward. Forwards allow us to reach goals, which is why we do things. But why is having a backward component so important? The simple answer is efficiency. We want energy cycles rather than energy losses. The loss occurs when energy is transferred to an environment. An efficient energy cycle requires a forward and backward phase that minimizes energy loss.

> Let's take a straightforward example: we throw a ball vertically into the air. That action requires energy (i.e., kinetic energy); as the ball slows down, it eventually succumbs to gravity (i.e., the energy transitions to potential energy). When the ball eventually descends, the total energy of the throw is near zero; there is some loss due to air friction, but the total energy goes towards zero. The energy used in the descend phase is nearly equal to that of the ascend phase

What exists near us with both a forward and a backward phase? Pendulums, internal combustion engines, regenerative brakes, flywheel, and reversible computing. In all cases, there is some loss in the system due to some form of friction.

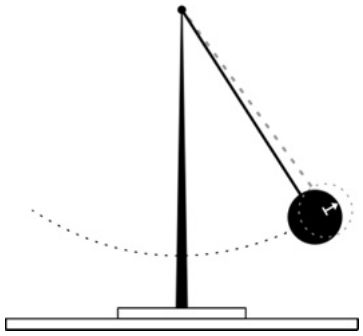

Figure 1: Pendulum

A pendulum slows down over a long period, so we add more energy to keep it going; an internal combustion engine requires more fuel; a regenerative brake captures some, but not all, of the energy created; a flywheel, once up to speed requires more energy to continue, and reversible computing requires an energy top-up. In all cases, energy efficiency and energy storage are the goals. But in all cases, efficiency is much better than leaving the world to be forward-driven. Backward adds a cycle into energy flow. By adding pseudo-reversibility, we immediately enhance the efficiency of any system. Pseudo-reversibility refers to a property of approximate reversibility—not adhering to a strict mathematical definition. With this concept, we can apply pseudo-reversibility to recycle energy for practical purposes. In other words, we reverse to create a cycle, and then we can optimize that cycle to achieve high efficiencies.

How?

Without digging too deeply into the details of the Physics, Near-Zero Energy computing is best described using two semiconductor chips, A and B. Both devices perform the same function and are built on the same CMOS process but are designed to see energy and heat flow differently.

1. Chip A works like the technology we see all around us; it is highly optimized toward a specific manufacturing geometry. There is no better design; the tools have optimized the layout and the circuitry. It uses the transistors and gates perfectly. Unfortunately, power and waste heat increase as the frequency increases. We call this **thermodynamically-unaware** technology.

2. By contrast, Chip B has been designed with the understanding of energy and heat flow. It is what is called **thermodynamically-aware**. It manages how energy and information flow through the circuits. When we provide energy and the input data to Chip B, most energy stays within the device. This way, it can produce results without generating heat—sipping energy while delivering the desired outcome. Note: there is always some friction in any system.

What is the difference? Remember, from the outside, both semiconductor chips behave the same from a digital perspective. Chip A requires a lot of support technology to handle the excess heat and significantly more energy to support the computation. Chip B requires near-zero energy to support the computation; by using a nominal amount of extra circuity to handle energy flow, it eliminates the need for dedicated cooling technology.

Chip B's characteristic comes from blending three critical technologies: an adiabatic energy flow system, a specialized oscillator called a resonator, and reversible computing. The adiabatic system creates thermodynamic awareness. The resonator creates the energy flow and capture. Finally, we select a form of reversible computing that enables the circuit to direct most of the energy back to the recycling resonator—creating an energy cycle within the logic circuit.

The Technology

The technology for Adiabatic Reversible Computing is different from traditional circuitry. We will first discuss traditional circuits and how they handle energy and computation and then discuss how Adiabatic Reversible Computing is different.

Traditional Circuitry: Traditional circuitry relies on a forward square or sinusoidal wave. We power the circuitry with a fixed voltage (Vdd), and the clock advances the computation as inputs to the logic (i.e., the clock is not the power supply). The wave acts as a clock, synchronizing the actions throughout a digital design. Any large design must accommodate clock skew as the wave passes through complex circuits. We also have to consider critical timing paths; it is paramount that we keep these paths within strict timing constraints. Each time a computation occurs in a traditional circuit, a

small amount of energy is released. The energy release is due to the erasure of information (i.e., by lowering the charge of individual bits). We see this energy release in the form of heat. As a digital system scales, so does the number of energy transitions—we see more circuits switching. Similarly, as we increase the computational rate (frequency of the clock), the amount of energy expelled by each switch increases—more transitions per second occur. In other words, we constantly provide energy to drive the circuitry, and that energy naturally produces waste heat. These characteristics act as a significant barrier to modern electronics. Designers have to adopt clever techniques to mitigate these characteristics—for instance, reducing high-frequency operating times, careful avoidance of heat concentration, and cooling if everything else fails. As previously introduced, we can label these circuits as being thermodynamically-unaware or simply open-loop circuits. This characteristic means that optimization, for the most part, is limited to the architectural level, i.e., standard logic cells remain the same.

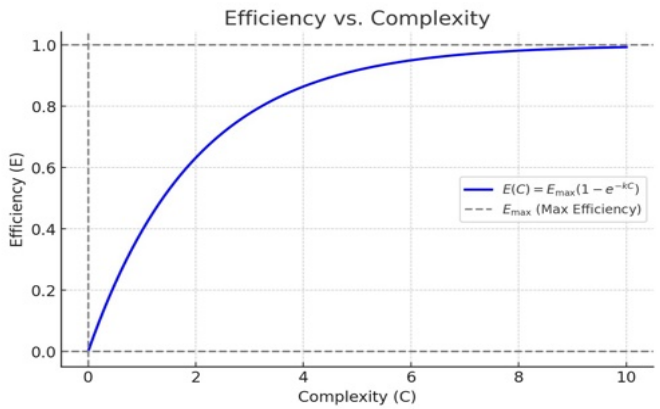

Figure 2: Optimizing Efficiency with just enough Complexity

Adiabatic Reversible Computing: We have described traditional circuit design, but how does Adiabatic Reversible Computing differ? Adiabatic Reversible Computing adds some level of complexity to the circuit. We introduce complexity to increase efficiency; see Figure 2. In other words, going to the simplest logic form does not necessarily mean we have the most optimal solution. A design should be as simple as possible, but not more simple than is required. With that said, the first significantly different piece is the driving clock.

To help with energy recycling, we drive the circuitry using a trapezoidal wave instead of a square wave. A trapezoidal wave is helpful in electrical energy recovery, mainly when switching power supplies and energy-efficient circuits are used. In our circuitry, the power supply is the clock. A trapezoid wave (compared with a straight up & down clock) has a slower rise and fall. The slower the rise and fall, the better the adiabatic operation. We want smooth transitions to reduce the amount of energy loss. The trapezoidal wave optimizes the transition between the energy storage and recovery phases. In power electronics, trapezoidal waveforms (instead of sinusoidal or abrupt transitions) allow smoother transitions in switching devices and reduce components' power dissipation and heat generation. This control helps with energy transfer required by recovery circuits (like resonant converters). Resonate converters rely on controlled current and voltage profiles. The trapezoidal waveform optimizes inductor and capacitor charging/discharging cycles for better energy transfer. The construction of a trapezoidal wave is non-trivial, especially if we want to embed the generators onto silicon. Constant research is required to produce ever more efficient generators.

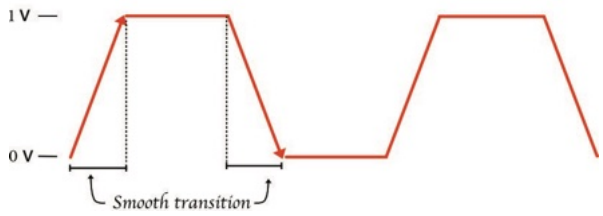

Figure 3: Trapezoidal wave

We have a different waveform driving the computation, so we now need a method to capture and recycle the energy within the circuit. This technology is called a resonator. Electrical resonators are circuits that store and transfer electrical energy efficiently by oscillating at a specific resonant frequency (or harmonic). They play a crucial role in energy harvesting. Traditional circuits convert energy to waste heat; with adiabatic circuits, the energy we would have lost to waste heat is recaptured and transferred as much as possible back into the circuit. Note that once energy is lost to heat, it cannot be recovered. There are different types of resonators, including circuit LC resonators and dielectric resonators. Again, these are good areas for optimization and miniaturization.

By carefully controlling the switching, balancing the parasitic (unwanted) capacitance and inductance of the circuit, and recycling the energy, we can carefully design highly efficient systems. However, we are still missing a critical piece. How do we create a cycle from the logic side? The cycle is complete with reversibility. We need to avoid unnecessary waste heat by reducing the immediate erasure of data. We must carefully manage the process so that the energy (and data) remains within the circuit as long as possible. We achieve this by keeping the digital inputs and outputs around long enough for energy reclamation. Traditional circuitry destroys this information immediately, creating heat. Efficiency is related to the amount of energy put into the system divided by the amount of energy not recycled. A traditional logic system has a recovery factor of 1 (one)[1], meaning a unit of energy put into the system gets to drive one computational operation before it is lost as heat. A fully Adiabatic Reversible Computing system can have a significantly higher value $\gg 1$, meaning a unit of energy can flow through the system many times – driving many computational operations – before it is lost as heat. The quality of our implementations determines this value– improvements in the resonator, trapezoidal wave generator, and the reversible computing logic directly affect the efficiency value. It becomes a function of engineering.

In summary, unlike traditional circuits, our adiabatic system waveform carefully controls the charge and discharge of energy (no spiking & no squelching). We have a resonator that can store and transfer energy. Finally, we use reversible computing logic to complete the circuit by allowing information to remain present. These components together create a thermodynamically-aware closed-loop energy circuit. By reducing heat loss, we improve efficiency. The computational results stay the same as those of a traditional digital system, but what happens under the logic compatibility layer is quite different. The complexity is higher; we move from suboptimal minimalism to a more optimal complexity level to achieve greater energy efficiency. Because we control the energy and information flow, the efficiency of Adiabatic Reversible Computing logic is potentially orders of magnitude greater than conventional logic.

1 The quantity is called Rf, the energy recovery factor: R_f = (logic energy)/(energy dissipated per cycle). We can relate this to energy recovery efficiency as $\eta = 1 - (1/R_f)$. For a traditional system with an R-factor of 1, efficiency is 0: no energy is recovered. The goal of Adiabatic Reversible Computing is to try to get η to approach 100%.

Summary

Near-Zero Energy computing can affect the direction of computation for the next 50 years. This change allows us to create entirely new forms of devices that we never thought possible. It is not magic, but it does rely on understanding the deep workings of energy and information flow through semiconductor chips.

With all technology, there are always trade-offs; nothing is utopian. We have to decide whether the challenges of a specific time overwhelm the trade-offs of a particular technology. We see the challenges of waste in heat and computation inefficiency in energy as the biggest hurdles for our time. The trade-off with NZE computing is slightly more silicon area at reduced upper frequencies. The bet we are placing is that today's requirements for excessive cooling overwhelm the trade-offs for thermodynamically-aware technology. NZE technology future-proofs scaling by trading a slight reduction in frequency for substantial 3D capability and opens the computational door to a volumetric computing future.

We believe this could dramatically affect everything from reducing the costs of learning and inference in Artificial Intelligence to significantly extending a mobile device's uptime on a single charge. By making electronic circuits more efficient and thermodynamically aware, we open up opportunities. We can effectively unstrap the power burden of cooling and focus our entire attention on computing.