ARRAY

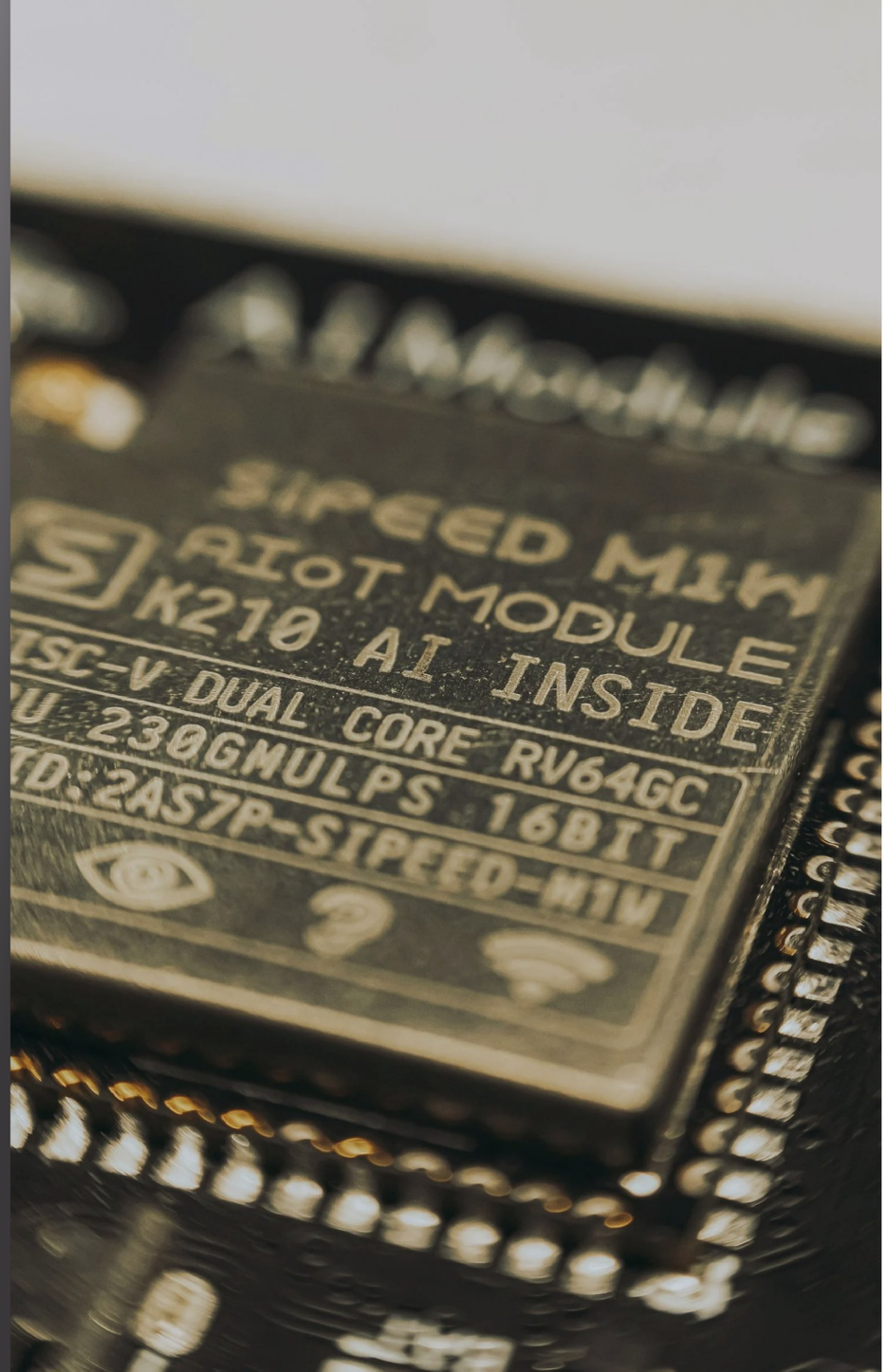
TECHNICAL OVERVIEW

A Compute-In-Memory Arithmetic core with +85% efficiency and throughput gains using standard cmos technology

www.fastarithmeticunits-dataroom.com

[ARRAY ARCHITECTURE, CORP.]

We describe optimal representations and codings of numbers and mathematical structures, and operations at the most fundamental level. **While others are trying to solve the Von Neumann Bottleneck in the lab with new materials or with complicated single-function designs, we've solved it mathematically.** Our Fast Arithmetic Unit enables Compute-In-Memory arithmetic for varying applications, using standard CMOS cells and manufacturing processes. In a separate document, we have outlined a 16-Month Development Plan for the fully functional HASH Engine for next generation Bitcoin Mining ASICs.



Processor Architecture

Searching for the Perfect Balance

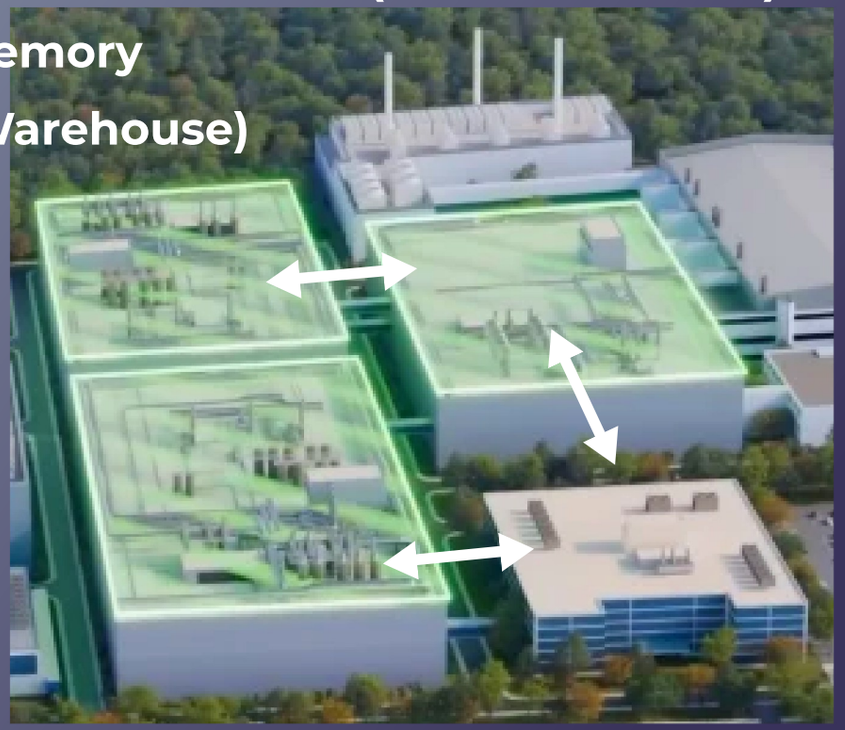
Design and Manufacture

Implementing a new processor architecture is challenging. **Various factors have to be optimized in a balanced way.** These factors include design costs, production processes, cost-effectiveness, scalability, energy and time efficiency, performance, and materials.

The End of Dennard Scaling has made power a first-class design constraint. **Energy is the primary constraint in scaling computing performance, rather than transistor count or clock speed.** Power density is increasing, limiting further performance gains through increased clock frequency.

Arithmetic Logic Unit
(Production Line)

Memory
(Warehouse)



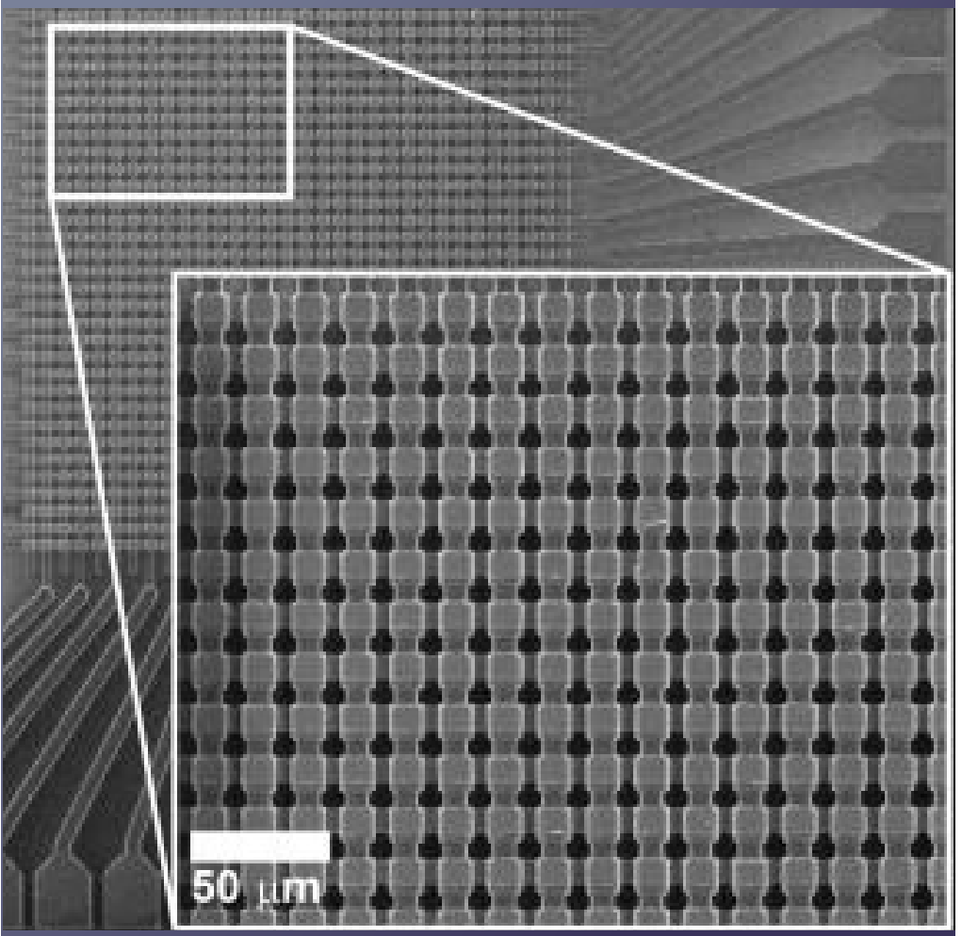
Control Unit
(Corporate Office)

Von Neumann Architecture

The most common architecture for computer systems is the Von Neumann Architecture which separates a processor into three essential subunits that perform all its tasks:

1) Memory (Registers; On-Chip SRAM, DRAM, MRAM, etc.)
2) Arithmetic Logic Unit (ALU)
3) Control Unit (CU)

A processor can be compared to a manufacturing company. The Memory acts as Warehouse, storing the raw material and unfinished/finished goods. **The ALU is the Production Line,** and the Control Unit is the Corporate Office that schedules and sends instructions between the other two sites.

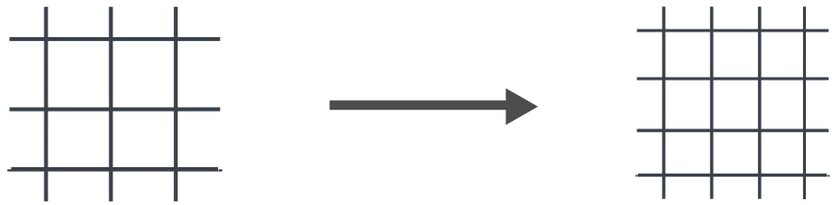


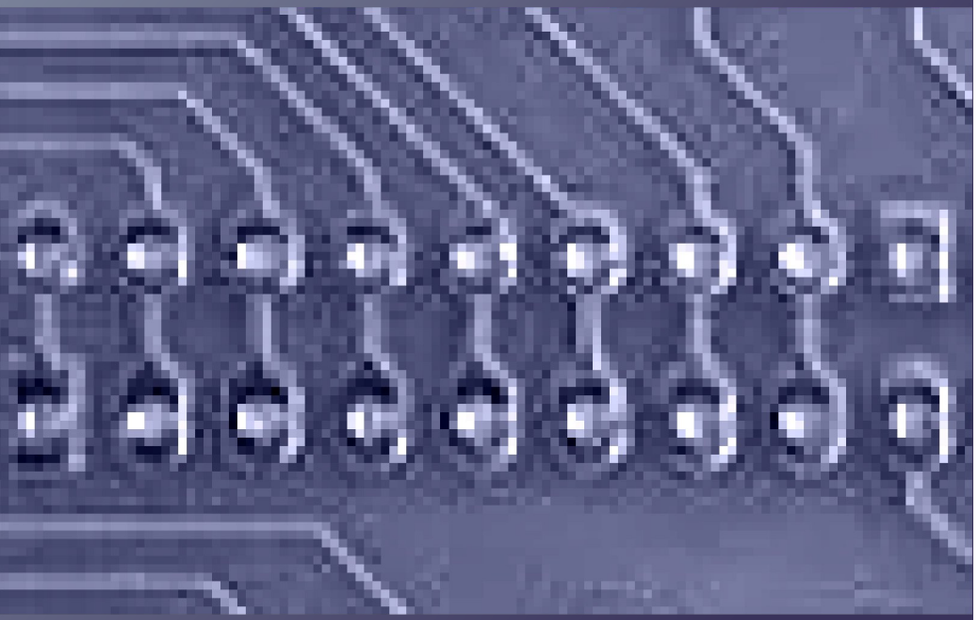
Memory

Following our analogy, the warehouse **(Memory) is a very simple building consisting of streets and avenues (rows and columns)** perfectly aligned in a rectangular grid.

This Manhattan-style grid is easy to scale for additional capacity. You simply add more streets and avenues for more bits or inputs.

Scalability



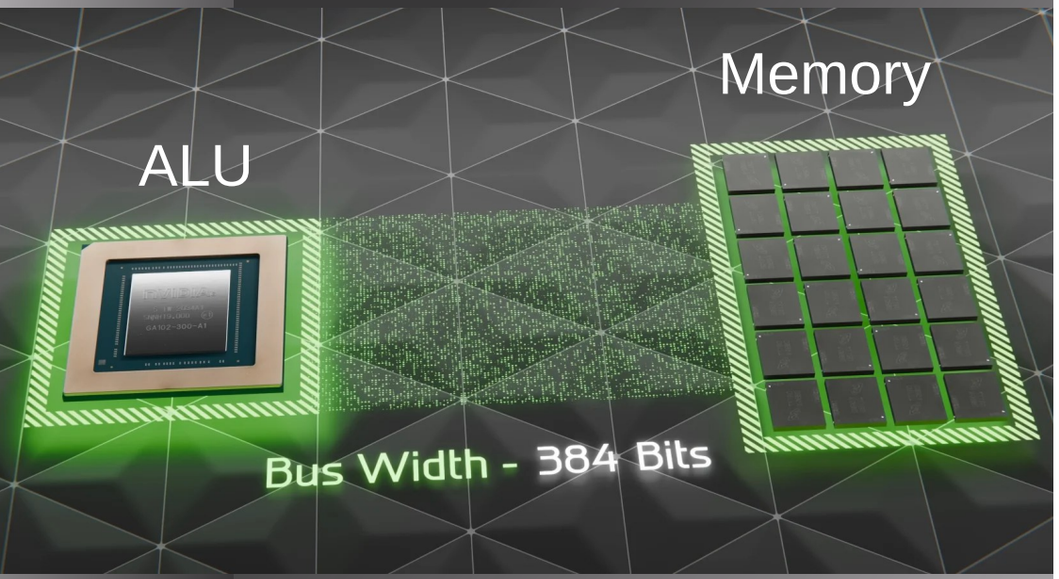


Arithmetic Logic Unit

The Production Line is where the manufacturing takes place. Likewise, in a processor, **the ALU is where all the mathematical and logical operations are executed.** In essence, it is the processor. For this reason, the ALU is prioritized for optimization. **First-Generation Adders had a linear and scalable design based on the traditional "carry-over" algorithm.** These ALUs are like a train: a number of box-cars are connected in series. Each box-car computes a significant bit of the result and sends the "carry flag" to the next bit. However, these adders are **very slow because they act as a production line that only works on one stage of the product at a time,** instead of working on several stages of the product at a time. The algorithm itself requires the addition to be done one bit at a time, right to left.

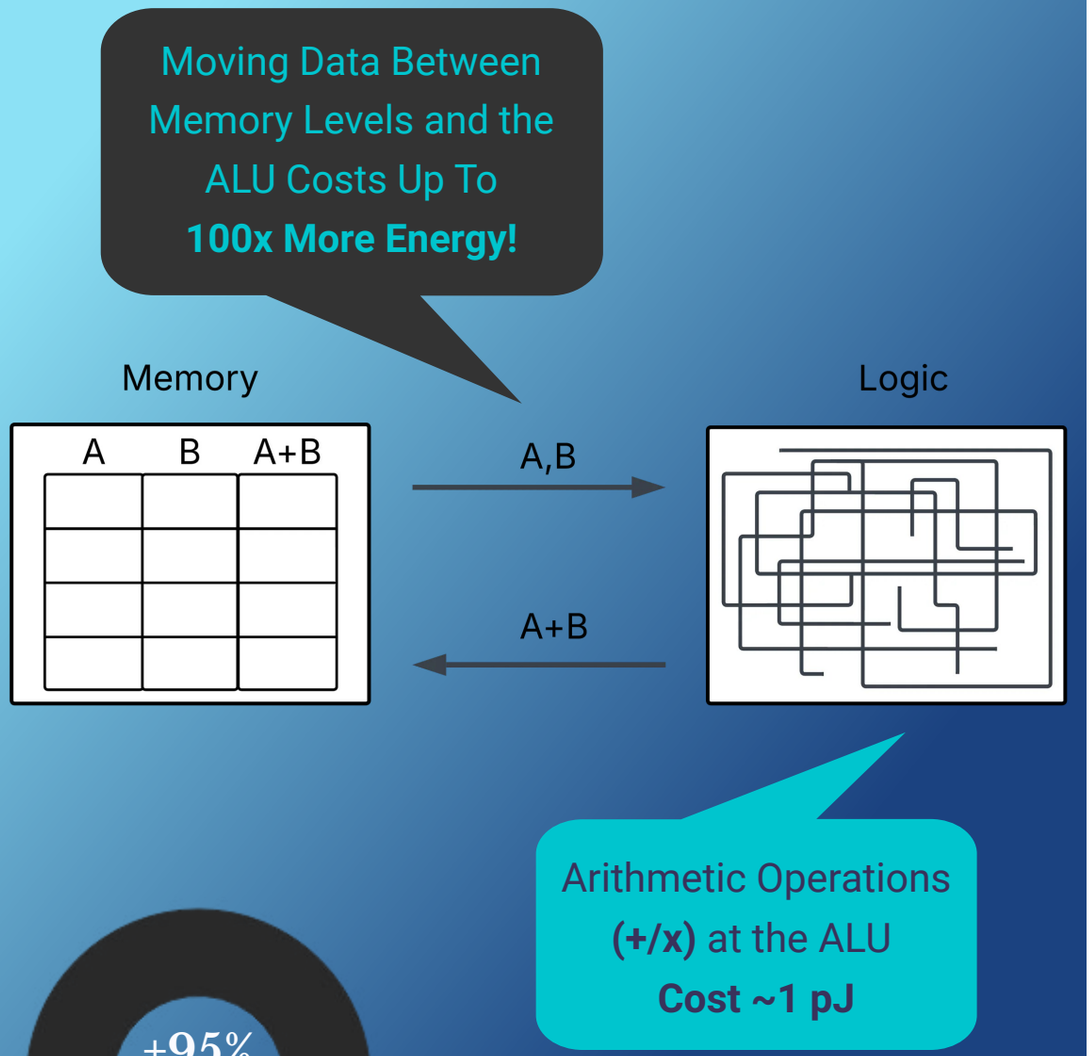
State-of-the-art ALUs perform operations "in parallel" using modern versions and modifications of the "carry-over" algorithm. However, these circuits are still operating within the carry-over paradigm of addition and they come with their own efficiency and performance tradeoffs.

Just as the floor layout of an efficient production line is more complicated than a warehouse, the ALU is also a more complicated circuit than the Memory. It is not a regular grid. It is an **exceedingly complex and irregular layout,** and complexity increases exponentially with bit-width. **This lack of scalability is one of the most pressing difficulties in designing new processors.**



Von Neumann Bottleneck and CIM

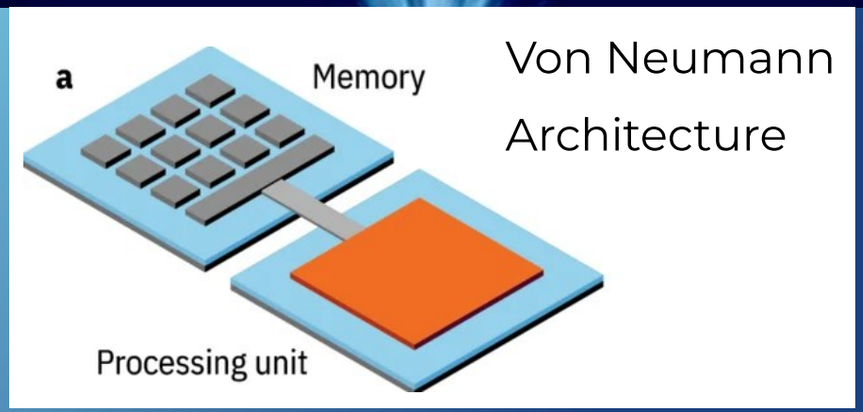
Re Imagine Computing



Von Neumann Bottleneck

A fundamental problem of computer architecture that sets important limitations on computing throughput and efficiency is the *Von Neumann Bottleneck.* **Separating the Memory and the ALU results in a bandwidth problem, and an associated time and energy cost.** Information is constantly relayed back and forth, **and processors spend most of their time and energy on this data migration alone.**

The bus connecting the Memory and the ALU is like a road connecting the warehouse and the production line. **If we increase the production line and warehouse's capacity too much, the road is going to suffer traffic jams** and schedules will inevitably collapse.



Von Neumann Architecture

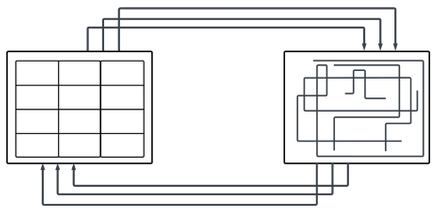
Von Neumann Bottleneck

To execute a single mathematical operation information data is commuted several times. To multiply 3 and 4 the processor adds 3, four times to itself. First the memory sends a pair of 3s to be added at the ALU. The result, 6, is sent back to the memory. Then, the memory sends 6 and 3 to be added at the ALU. This is repeated until the final result is obtained. **To execute the 64 round SHA-256 compression function, data will travel back and forth between registers and Hash Engines at least that number of times.**

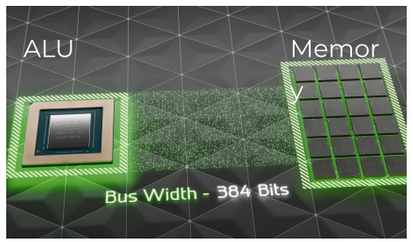
Causes for Von Neumann Bottleneck

Solving the Von Neumann Bottleneck is crucial for curving energy demands and performance limitations in modern processors because it makes up for **60-90% of the time latency and energy consumption,** due to four main reasons:

The peripheral distance travelled between units is huge compared to the distances travelled inside the subunits.



We have a **bandwidth problem** when trying to increase bit capacity too much.

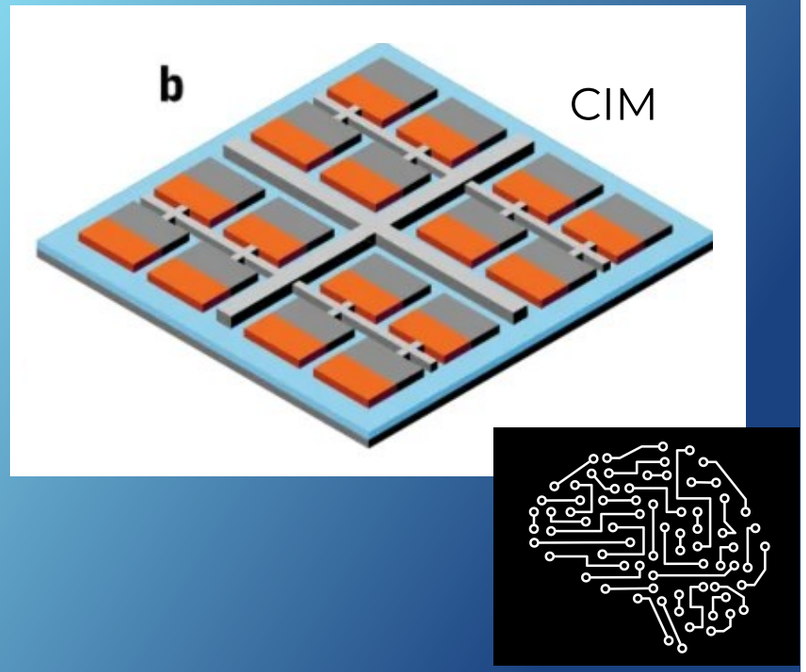


Energy costs are disproportionate.

~1 pJ to Add/Multiply
vs.
~100 pJ to Move Data

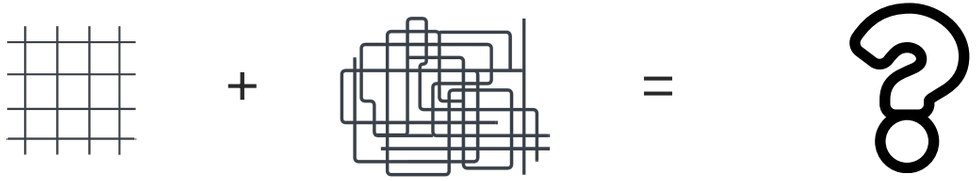
A single operation requires information to migrate between the memory and the ALU **back and forth several times.**

Compute-In-Memory



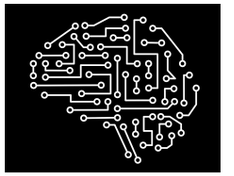
The biological brain is a source of inspiration for overcoming the Von Neumann Bottleneck because memory and logic are co-located. **CIM is an attempt to bypass the Von Neumann Bottleneck by computing In-Memory, providing unparalleled time and energy efficiency.** This is equivalent to having the production line in the same building as the warehouse, eliminating traffic jams between the two.

However, implementation is not a straightforward task. **Significant resources have been spent searching for new transistor and memory types.** As mentioned earlier, the floor layout of the Memory is a regular rectangular grid, while traditional ALU circuits are very complicated and irregular. **These two completely different circuits are very difficult to place and operate in the same area.**

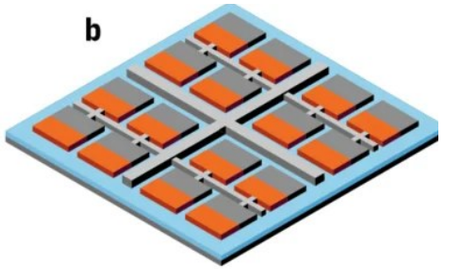


Compute-In-Memory

- The biological brain is able to **store and process information "locally".**

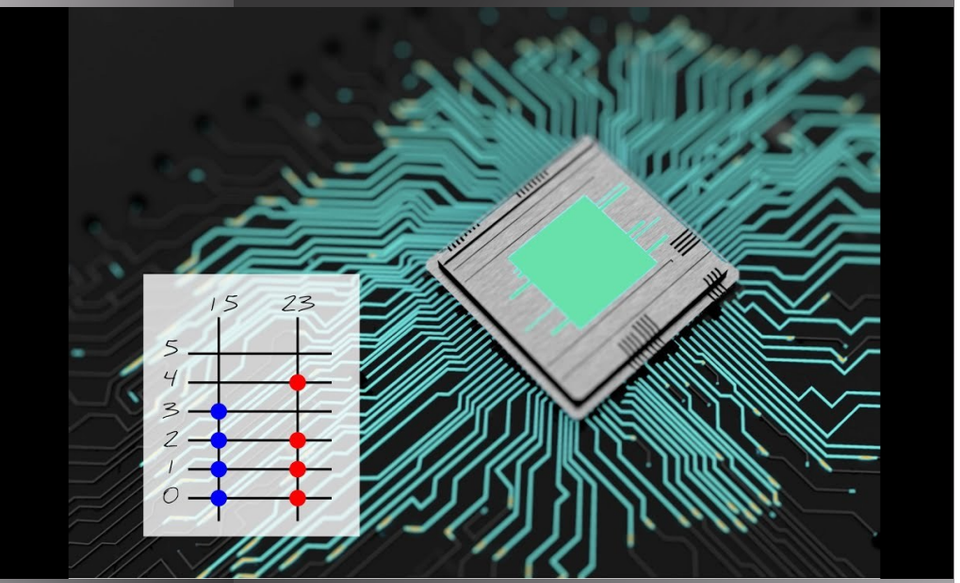


- Compute-In-Memory is proposed as an attempt to eliminate the Von Neumann Bottleneck. **Performing computations directly within the memory, offering up to 90% Time & Energy efficiency.**



- Implementing **CIM requires expensive R&D into new transistor and memory types,** demanding new materials, fabs and processes, etc.

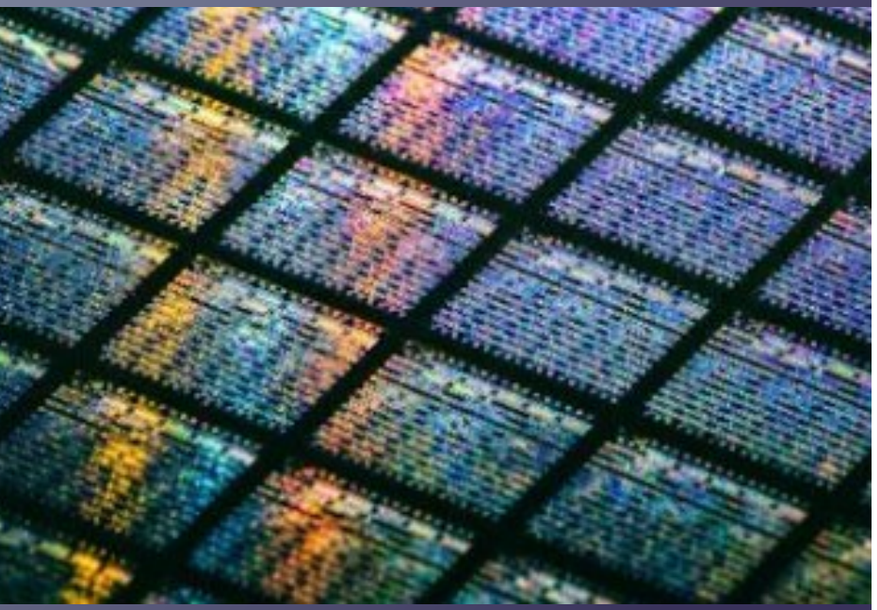
- One of the basic difficulties in implementing CIM is the fact that **Memory and ALU are topologically, and materially, very different.**



Our IP Technology: Fast Arithmetic Unit

Re Imagine Computing

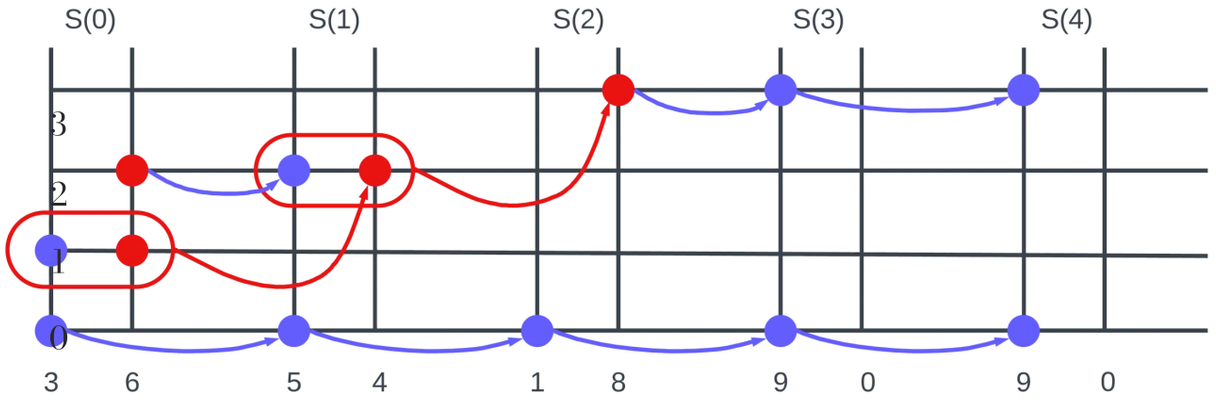
We aim to implement a **widely applicable, low-cost, and scalable arithmetic core.** This Fast Arithmetic Unit (FAU) will enable **efficient, high-perfomance processors CIM** without the huge R&D costs of finding new transistor and memory types. **It is a new arithmetic architecture based on existing CMOS technology.**

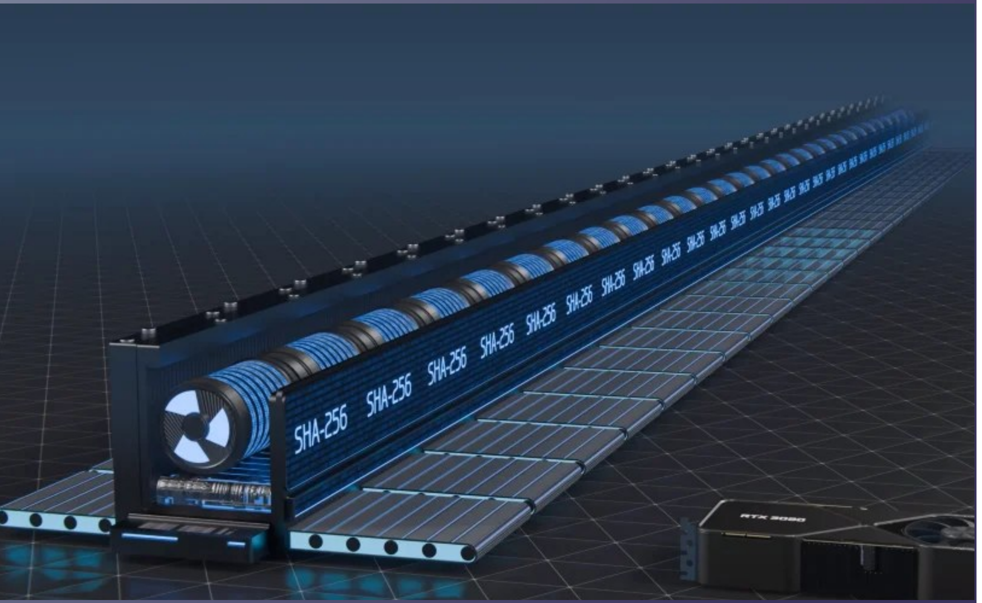


Simple and Linear Fast Adder

Non Von Neumann architectures usually solve the memory-wall problem materially, either through experimental transistors, or hardwiring for very specific workloads. We addressed the problem at the mathematical level, simultaneously solving bit-scaling, and the memory-wall.

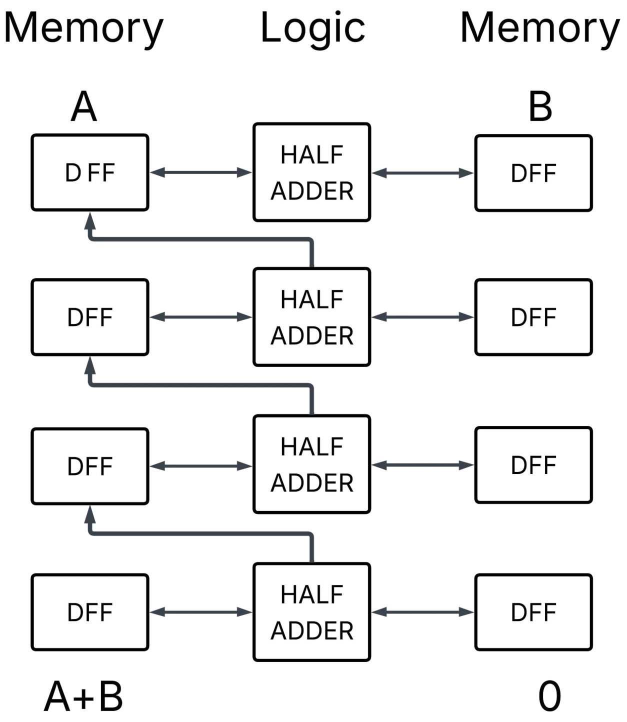
The core IP, the Simple and Linear Fast Adder, is a **scalable circuit that cuts design and production costs, by co-locating memory and arithmetic logic** in a Manhattan-style grid of streets and avenues, **offering the unparalleled efficiency and performance of CIM,** using the industries most mature memory element (Flip Flop).

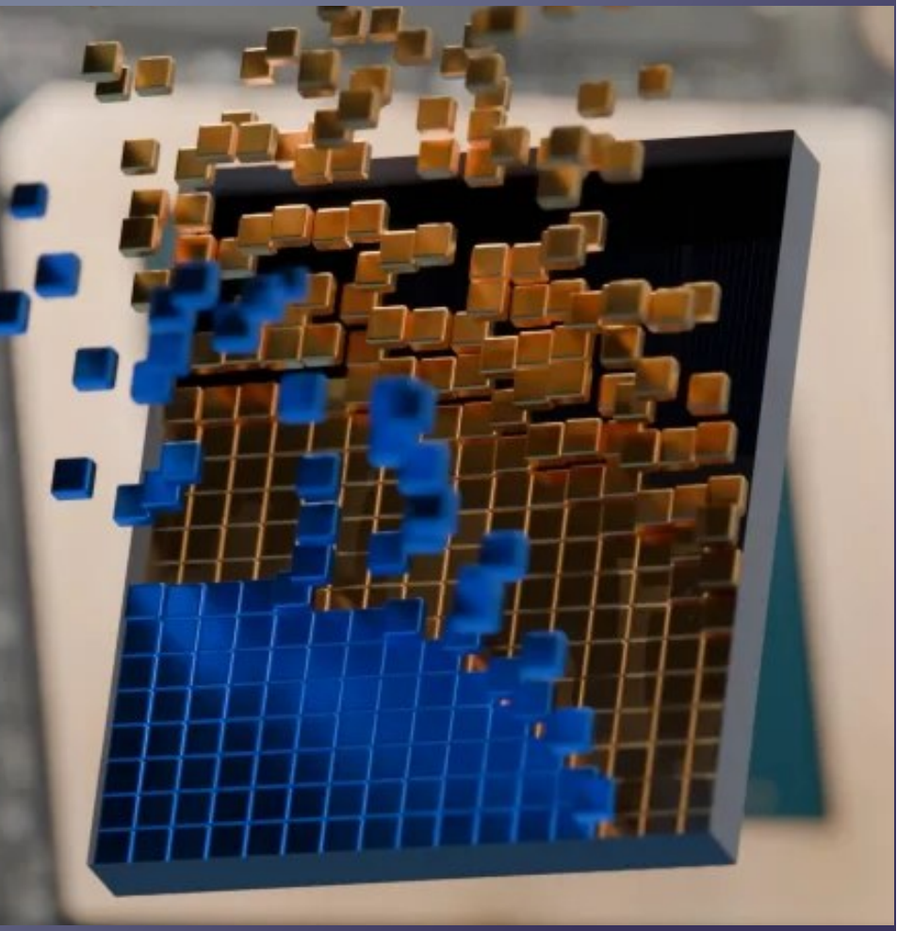




Simple and Linear Fast Adder

A series of subunits are connected in sequence to act as a Finite State Machine that executes a novel algorithm of addition. **An *n*-bit adder consists of *n* copies of identical subunits connected in series.** Each of these subunits consists of two (2) double edge-triggered memory registers, one (1) 'XOR' and one (1) 'AND' gate. The registers of each subunit are connected to their corresponding logic gates ('XOR' and 'AND' gates) in that significant bit. **Registers are updated every clock cycle, all bits in parallel, until the machine reaches stable state in logarithmic-time.**





Fast Arithmetic Units

The SLFA can be generalized into a rectangular grid routed in a Manhattan-style array of streets and avenues that executes addition of multiple inputs. **Memory and transistors for bit-wise addition are co-located in a one-to-one correspondence, eliminating migration time and energy altogether.** The design is scalable with minimum area for Matrix Multiplication (Multiply Accumulate Circuits), and Hash Cores, among others.

- **Low-Power, Low-Cost** Design
- Linearly **Scalable Bit-Width**
- **In-Situ Processing** (Super-Fast and Energy-Efficient)
- Based on **Standard CMOS** Memory and Transistor Technology
- **Universal Applicability**

Comparison Table

	Traditional Adder Architecture		Non Von Neumann Architectures			
	Ripple Carry-Over (1st Gen)	**Parallel** (Carry-Look Ahead, Carry-Save, Kogge-Stone, etc.)	**Near-Memory Computing**	**Compute-In-Memory Transistor Arrays** (FeFETs, MRAM, ReRAM, etc.)	**ASIC** (On-Chip Pipelines)	**Simple and Linear Fast Adder** (Bit-Level Embedded Logic & Memory)
Competitors	Low-Power Systems Texas Instruments	Traditional Processors Intel, Nvidia, IBM, AMD, etc.	NeuroBlade, Nvidia, MythicAI	IBM, Intel, Samsung, SK Hynix, MythicAI	BitMain, Etched, Nvidia, Cerebras, NeuroBlade	**Our IP**
Energy (Efficiency)	◐	○	◕	◕	◕	●
Time (Latency)	○	◐	◕	◕	◕	●
Manufacturing (Costs and Requirements)	◕	◐	◐	○	◐	●
Technology (R&D)	◕	◕	◐	○	◐	●
Applicability (Use Cases)	◐	◕	◐	◐	◐	●
Scalability (Bit Length)	◕	◐	◐	◐	◐	●

Traction

2020

Theoretical background.

Mathematical Proofs.

2022

International (PCT) patent application filed in key global jurisdictions.

2024

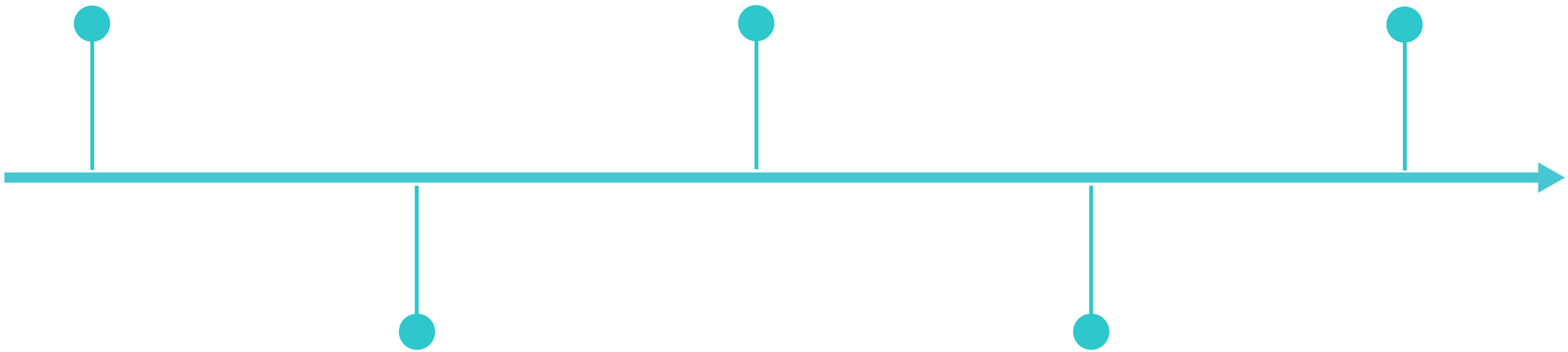
- IEEE International Conference, Publication, and Peer-Reivew
- Independent IP Valuation at $8 M USD
- Third-Party Market Report

2025

Logical Synthesis and Analysis for In-Memory Matrix Multiplication (mathematical operation behind AI and Computer Graphics) and Bitcoin Mining Engine.

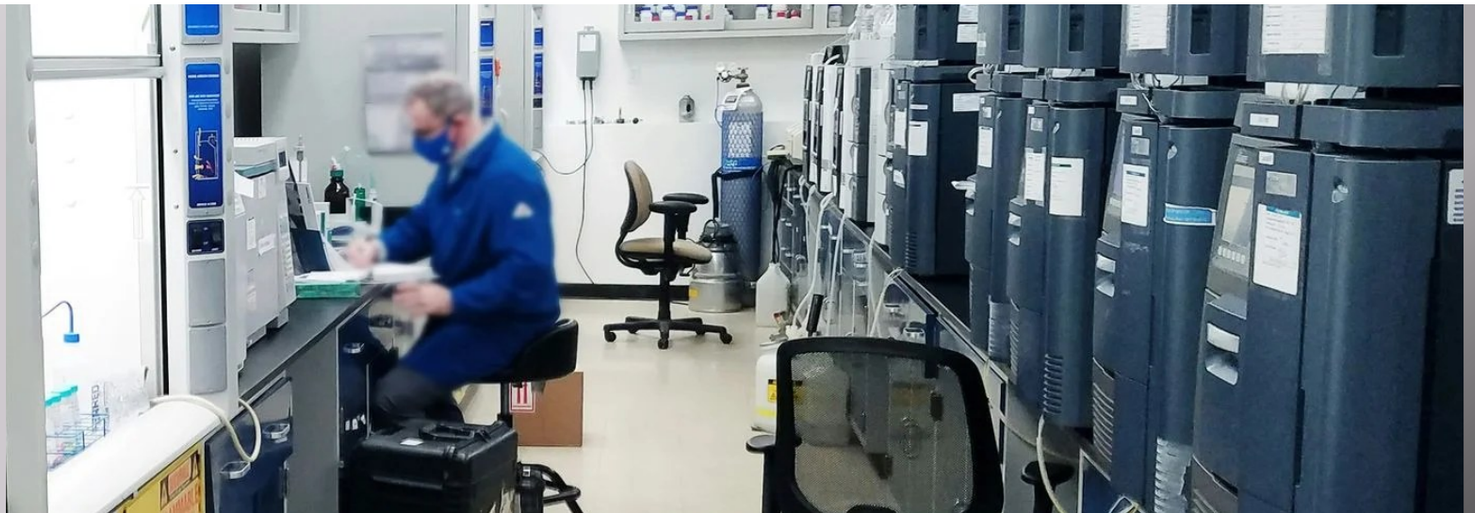
2026

Ready for Silicon Prototyping.



Where We Are

The theoretical foundations, mathematical algorithms, circuit logic, and RTL design have been detailed in numerous international conferences and peer-reviewed publications. We studied and published basic results regarding the FAU's capabilities. Our next goals include partnering and building a world-class team for validating the FAU's performance through emulations, benchmarking and prototyping to lay the groundwork for subsequent licensing, and joint development.



THEORETICAL FRAMEWORK

Our approach addresses the core challenges of numerical representations and computational complexity. By revising mathematical foundations, we enable fast, low-powered calculations at both hardware and software levels. This revolutionary framework supports the processor industry's transformation.

SUPPORTING RESEARCH

- “Simple and Linear Fast Adder Based on a Simple Representation of Natural and Real Numbers”. Computer Science Special Session, 55 Mexican Congress of Mathematics, Guadalajara, Jalisco, 2022.
- "An Algorithm for Fast Multiplication and Addition of Multiple Inputs and It's Implementation for In-Memory-Computing". Computer Science Special Sessions, 56 Mexican Congress of Mathematics, San Luis Potosí, 2023.
- "Simple and Linear Fast Adder of Multiple Inputs and It's Implementation for a Compute-In-Memory Architecture”. International Conference on Artificial Intelligence, Computer, Data Sciences and Applications, 1-2 February 2024, Victoria-Seychelles.
- "Simple and Linear Fast Adder of Multiple Inputs and Its Implementation in a Compute-In-Memory Architecture," *IEEE Xplore,* 2024 International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA), Victoria, Seychelles, 2024, pp. 1-11.
- International Application Published Under the Patent Cooperation Treaty (PCT).

MATHEMATICAL PAPERS

- Ramírez, Juan Pablo. 2019. "A New Set Theory for Analysis" Axioms 8, no. 1: 31. https://doi.org/10.3390/axioms8010031.
- “The Nature of Numbers”. Logic and Foundations Special Session, 52 Mexican Congress of Mathematics, Monterrey, Nuevo León, 2019.
- Lovyagin, Y.N.; Lovyagin, N.Y. Finite Arithmetic Axiomatization for the Basis of Hyperrational Non-Standard Analysis. Axioms 2021, 10, 263.
- “Simple Representation of Natural and Real Numbers”. Logic and Foundations Special Sessions, 55 Mexican Congress of Mathematics, Guadalajara, Jalisco, 2022.
- Juan Pablo Ramirez. (2023). Canonical Set Theory with Applications from Parallel Addition of Multiple Inputs to Matrix Multiplication and Data Structures. www.binaryprojx.com.

Application Theory

[COMPUTER SCIENCE]

Foundational Study

[CORE MATHEMATICS]

THANK **YOU!**

www.fastarithmeticunits-dataroom.com

Juan Pablo Ramirez

Architect, Project Manager & CEO

jramirez@binaryprojx.com

Pablo César Vázquez Estrella

Chief Financial Officer

pablo.vazquez@binaryprojx.com

www.fastarithmeticunits-dataroom.com