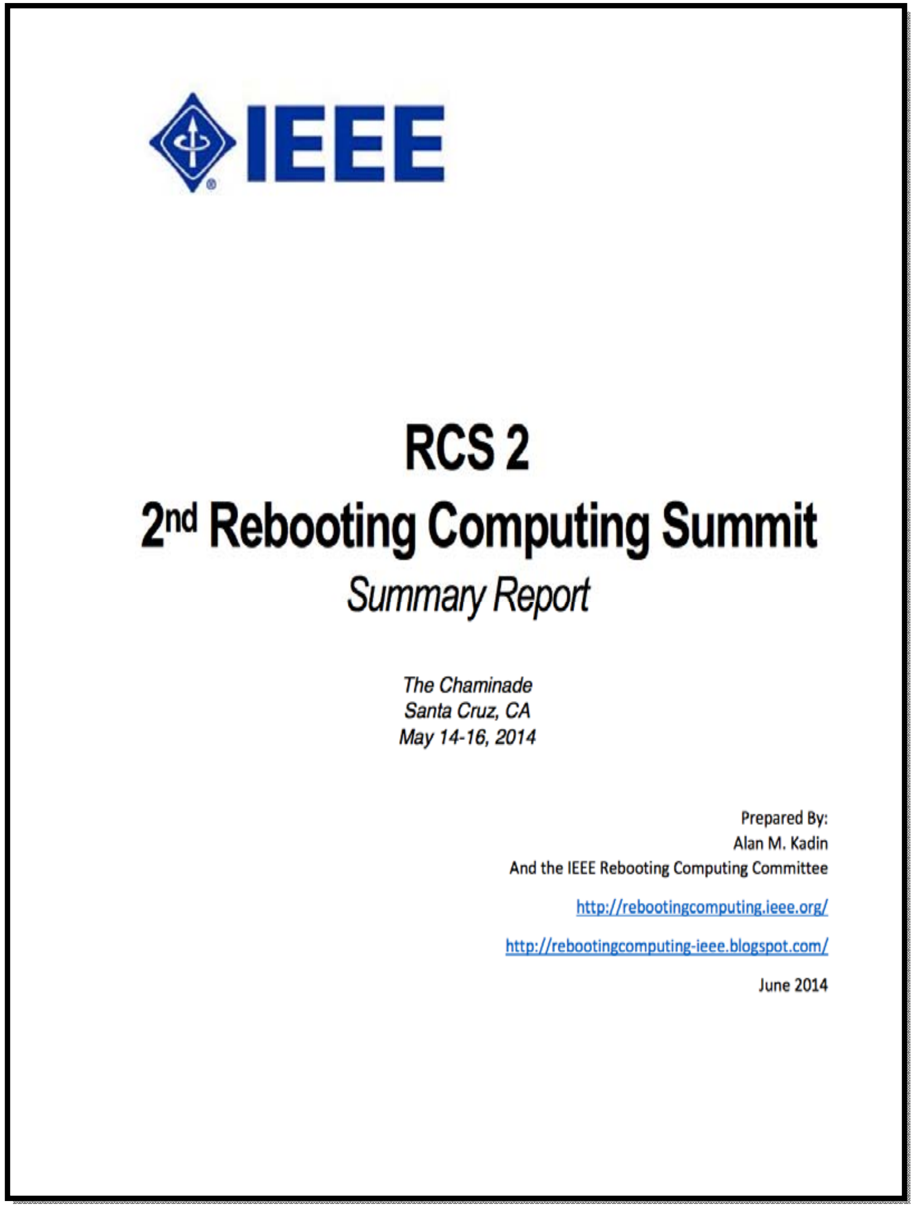
Some thoughts on beyond exascale (post-moore ish)

Rick Stevens

Argonne National Laboratory



IEEE

RCS 2

2nd Rebooting Computing Summit

Summary Report

The Chaminade
Santa Cruz, CA
May 14-16, 2014

Prepared By:
Alan M. Kadin
And the IEEE Rebooting Computing Committee

http://rebootingcomputing.ieee.org/

http://rebootingcomputing-ieee.blogspot.com/

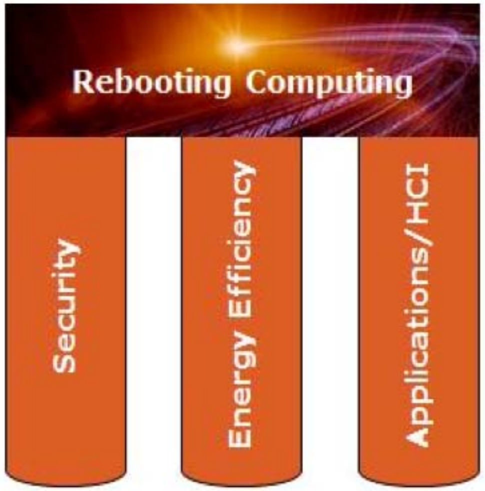
June 2014

Augmenting CMOS

Neuromorphic Computing

Approximate Computing

Adiabatic/Reversible



Augmenting CMOS

- Silicon CMOS circuits have been the central technology of the digital revolution for 40 years, and the performance of CMOS devices and systems have been following Moore's law (doubling in performance every year or two) for the past several decades, together with device scaling to smaller dimensions and integration to larger scales. CMOS appears to be reaching physical limits, including size and power density, but there is presently no technology available that can take its place. How should CMOS be augmented with integration of new materials, devices, logic, and system design, in order to extend enhancement of computer performance for the next decade or more? This approach strongly overlaps with the semiconductor industry roadmap (ITRS), so RCS 2 coordinated this topic with ITRS.

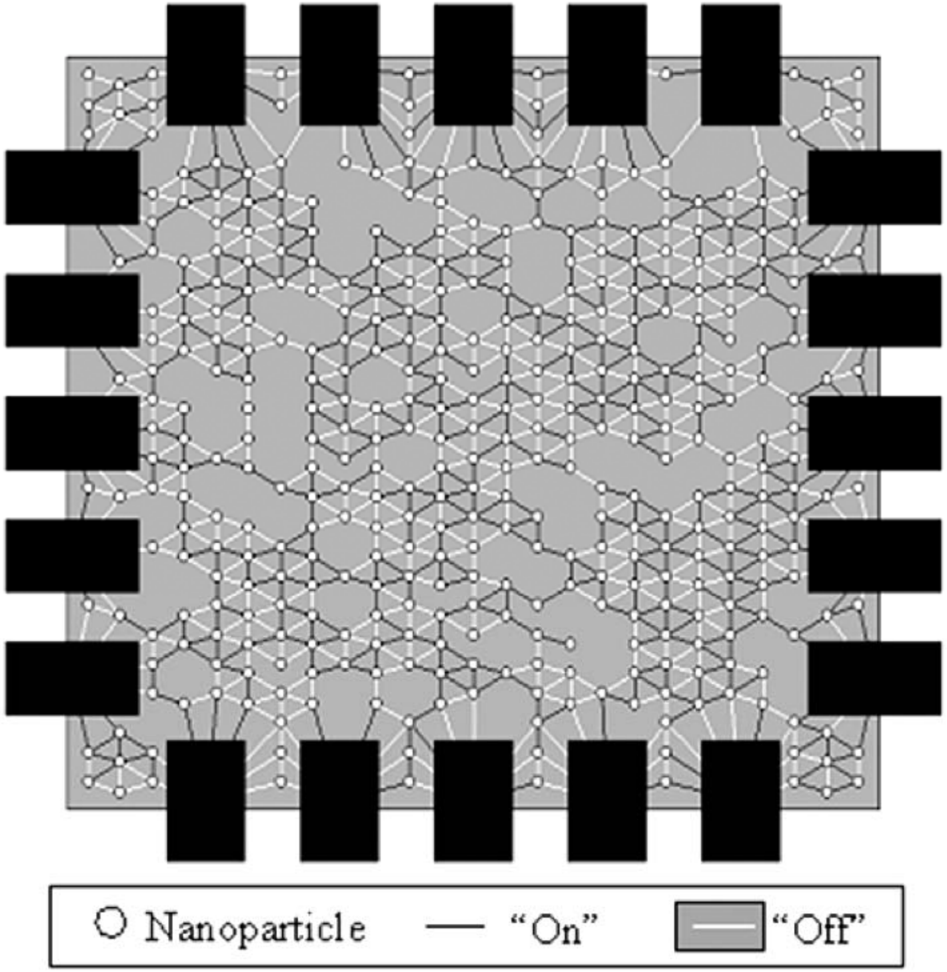


Fig. 1. Simulated self-assembled nanocell is depicted. The black rectangles at the edges are the I/O leads. The entire cell, excluding the outer portions of the contact pads, would be approximately 1 $\mu\mathrm{m}^2$.

IEEE TRANSACTIONS ON NANOTECHNOLOGY, VOL. 1, NO. 2, JUNE 2002

Nano-cell Tiles

- Network is static per cell
- Nanocell is trained post-fabrication by changing the states of molecular switches
- Mortal Programming
    1. finding switch states such that the given nanocell functions as the target logic device and
    2. finding a series of voltage pulses (applied to the I/O pins) that give rise to these desired switch states.

1 bit adder

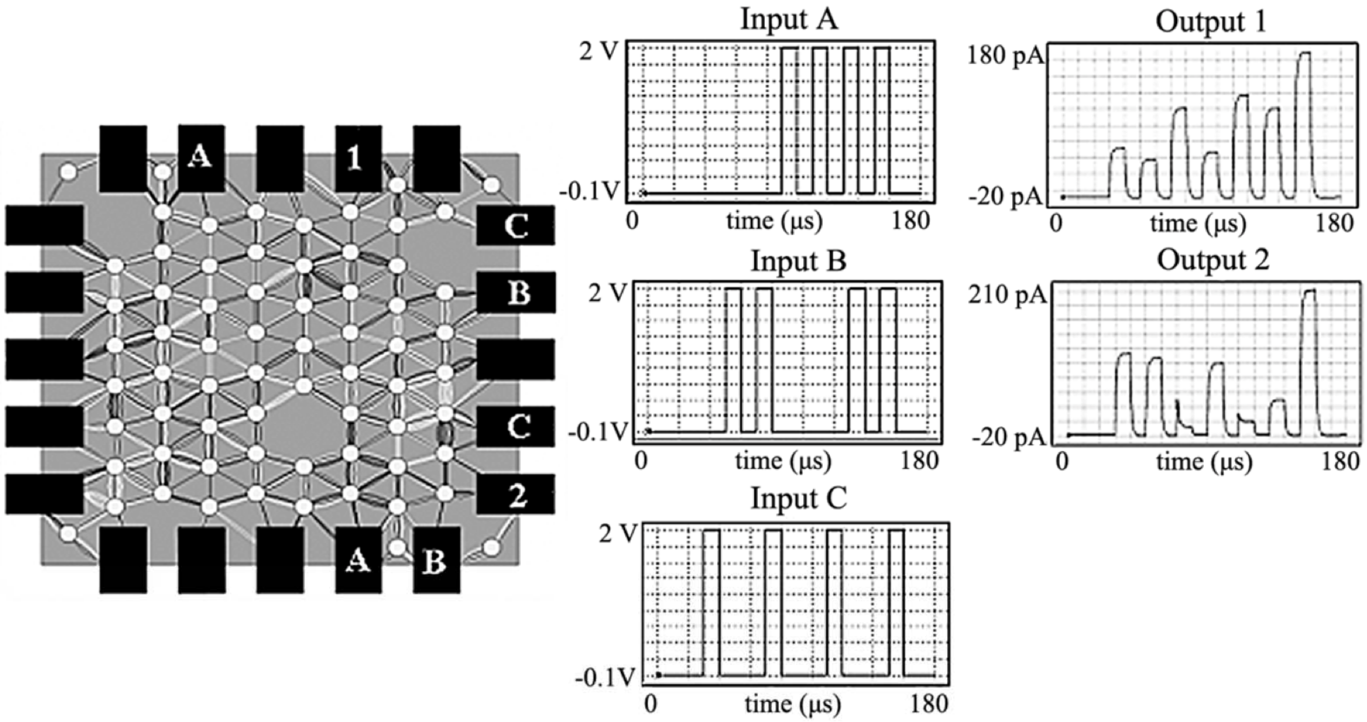
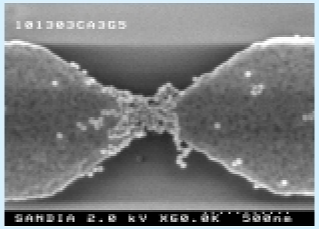
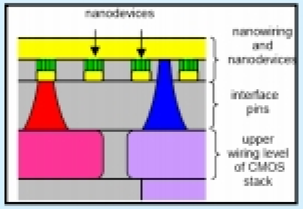
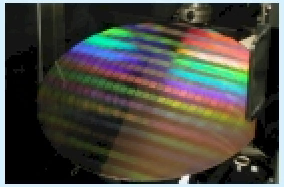
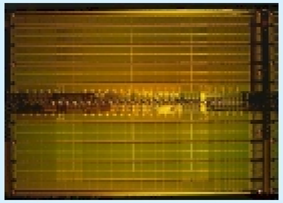
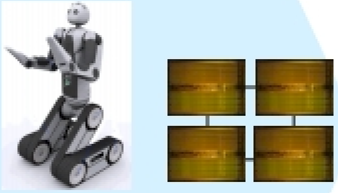
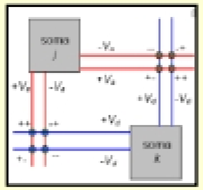
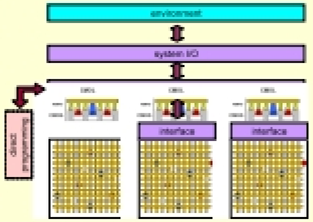
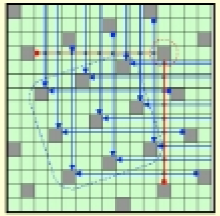
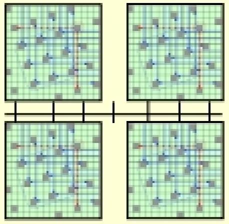
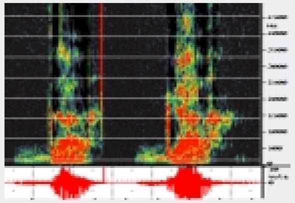
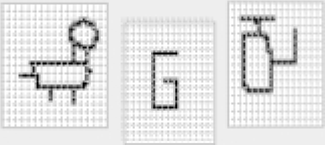
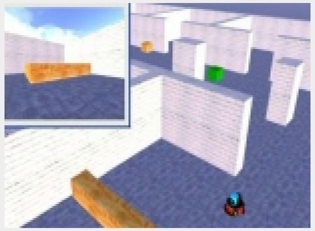
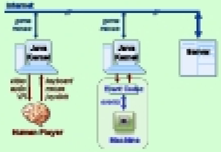


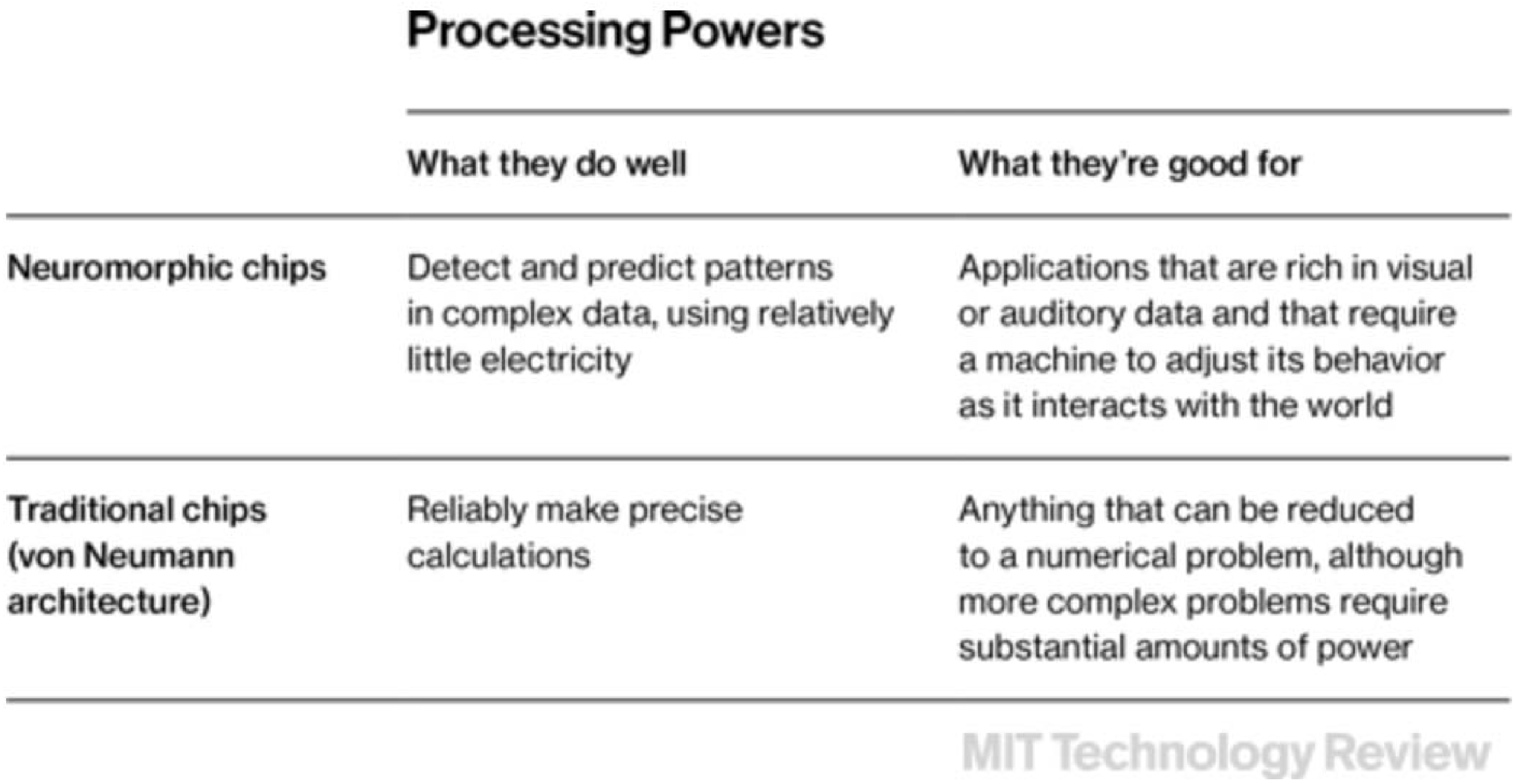
Fig. 6. A 1-bit adder is demonstrated on a randomly assembled nanocell using the SPICE interface model. The plots show the $V(t)$ for the inputs, $I(t)$ for the outputs and the $I(V)$ curve used for the molecules in the ON state. The OFF state is the same as in Fig. 3. The truth table for a 1-bit adder is displayed, as well.

Neuromorphic Computing

- A brain is constructed from slow, non-uniform, unreliable devices on the 10 um scale, yet its computational performance exceeds that of the best supercomputers in many respects, with much lower power dissipation. What can this teach us about the next generation of electronic computers? Neuromorphic computing studies the organization of the brain (neurons, connecting synapses, hierarchies and levels of abstraction, etc.) to identify those features (massive device parallelism, adaptive circuitry, content addressable distributed memory) that may be emulated in electronic circuits. The goal is to construct a new class of computers that combines the best features of both electronics and brains.

SyNAPSE Program Plan

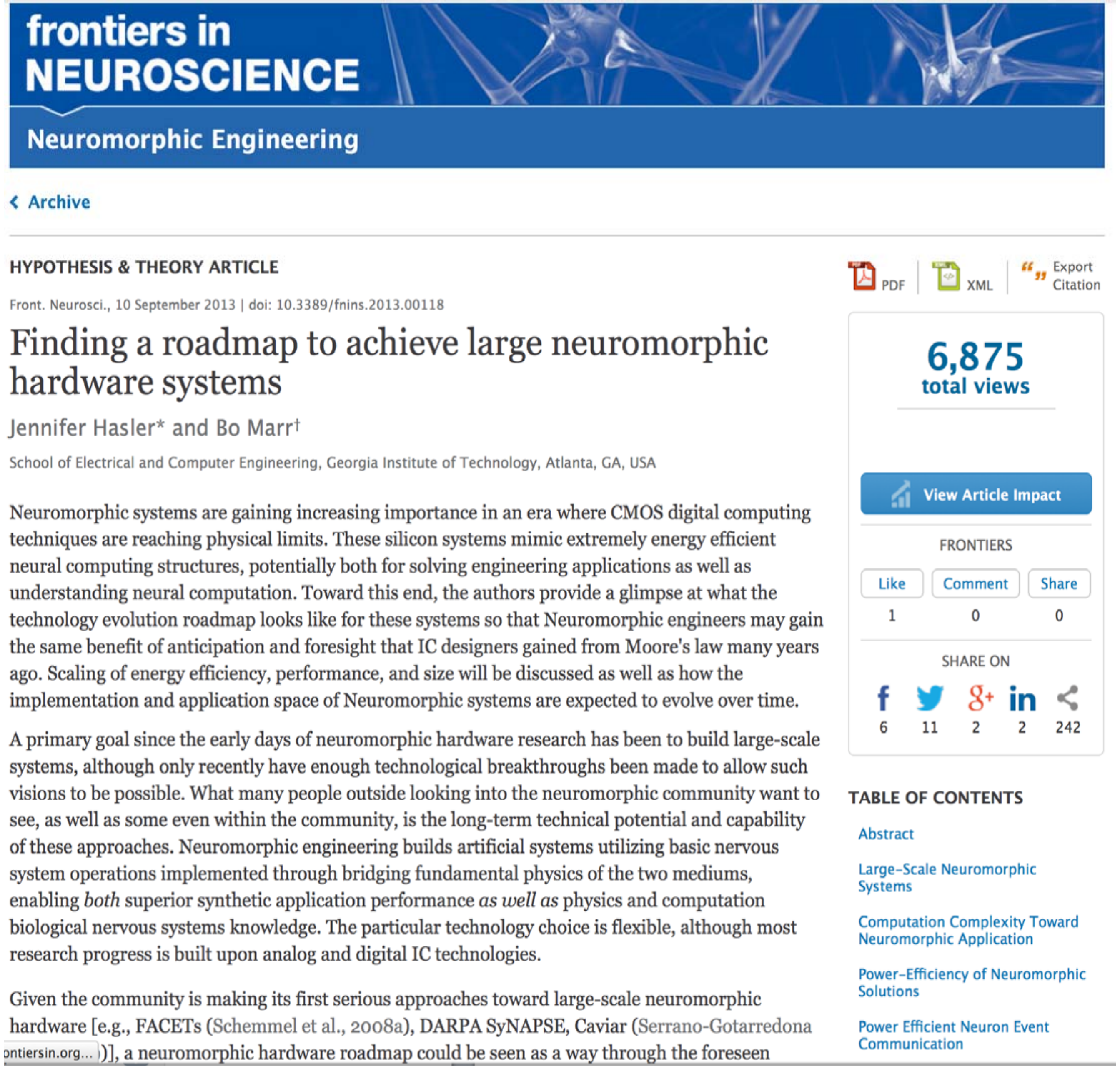
	Phase 0	Phase 1	Phase 2	Phase 3	Phase 4
Hardware	 Component Synapse Development	 nanodevices nanowiring and nanodevices interface pins upper wiring level of CMOS stack Process and Component Circuit Development	 CMOS Process Integration	 ~10^6 neuron single chip implementation	 ~10^8 neuron multi-chip robot
Architecture & Tools	 Microcircuit Architecture Development	 environment system I/O interface interface direct programming System Level Architecture Development	 10^6 Neuron Design for Simulation and Hardware Layout	 10^8 neuron design for simulation and hardware layout	Comprehensive Design Capability
Emulation & Simulation		 Simulate Large Neural Subsystem Dynamics	 ~10^6 neuron level Benchmark	 ~10^8 neuron level Benchmark	"Human" level Design (~10^{10} neuron)
Environment		 Internet Java Kernel Java Kernel Server Human Player Machine Build	Expand & Refine	Expand & Sustain	Sustain



Processing Powers

	What they do well	What they're good for
Neuromorphic chips	Detect and predict patterns in complex data, using relatively little electricity	Applications that are rich in visual or auditory data and that require a machine to adjust its behavior as it interacts with the world
Traditional chips (von Neumann architecture)	Reliably make precise calculations	Anything that can be reduced to a numerical problem, although more complex problems require substantial amounts of power

MIT Technology Review



HYPOTHESIS & THEORY ARTICLE

Front. Neurosci., 10 September 2013 | doi: 10.3389/fnins.2013.00118

Finding a roadmap to achieve large neuromorphic hardware systems

Jennifer Hasler* and Bo Marr†

School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

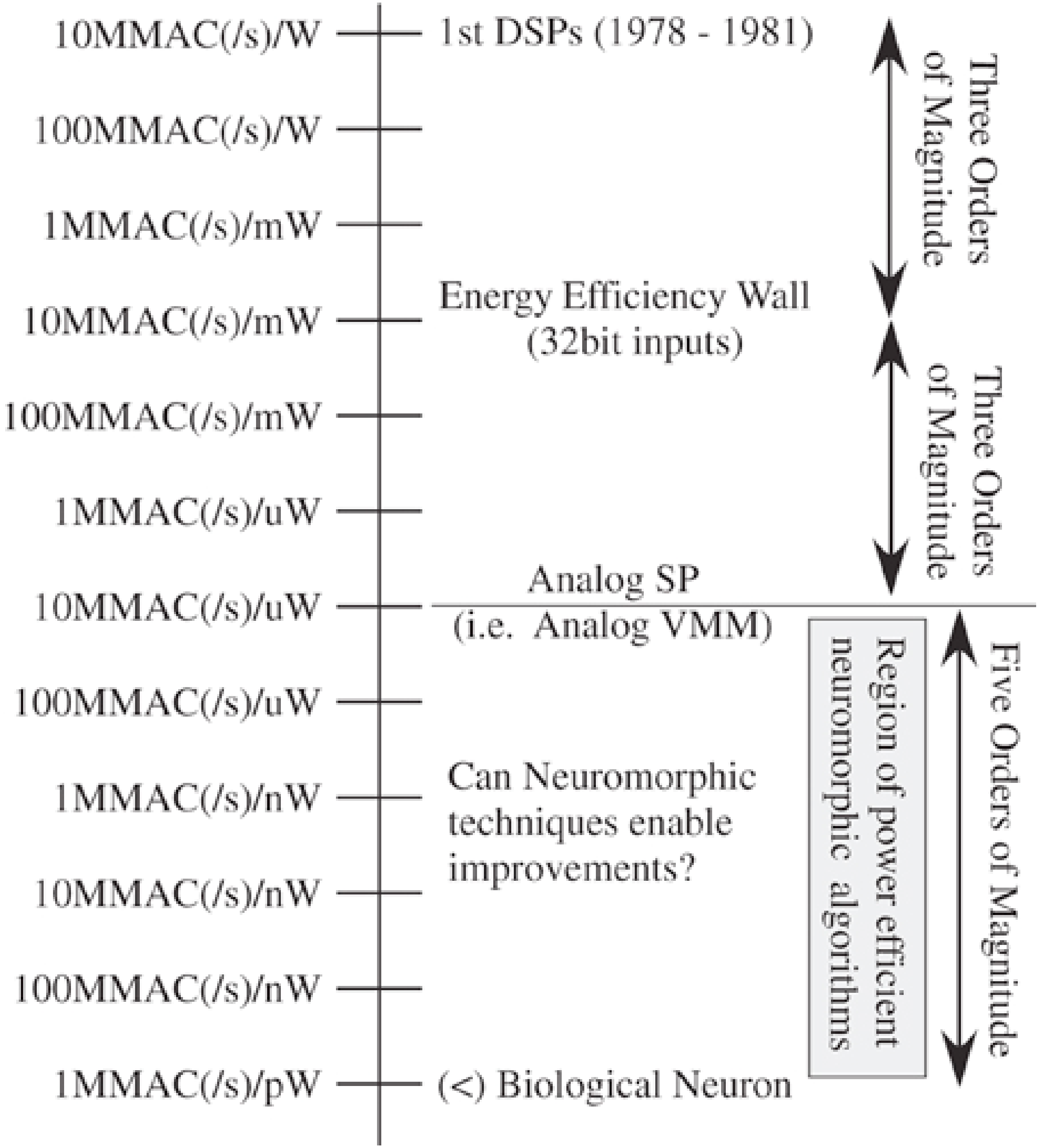
Neuromorphic systems are gaining increasing importance in an era where CMOS digital computing techniques are reaching physical limits. These silicon systems mimic extremely energy efficient neural computing structures, potentially both for solving engineering applications as well as understanding neural computation. Toward this end, the authors provide a glimpse at what the technology evolution roadmap looks like for these systems so that Neuromorphic engineers may gain the same benefit of anticipation and foresight that IC designers gained from Moore's law many years ago. Scaling of energy efficiency, performance, and size will be discussed as well as how the implementation and application space of Neuromorphic systems are expected to evolve over time.

A primary goal since the early days of neuromorphic hardware research has been to build large-scale systems, although only recently have enough technological breakthroughs been made to allow such visions to be possible. What many people outside looking into the neuromorphic community want to see, as well as some even within the community, is the long-term technical potential and capability of these approaches. Neuromorphic engineering builds artificial systems utilizing basic nervous system operations implemented through bridging fundamental physics of the two mediums, enabling *both* superior synthetic application performance *as well as* physics and computation biological nervous systems knowledge. The particular technology choice is flexible, although most research progress is built upon analog and digital IC technologies.

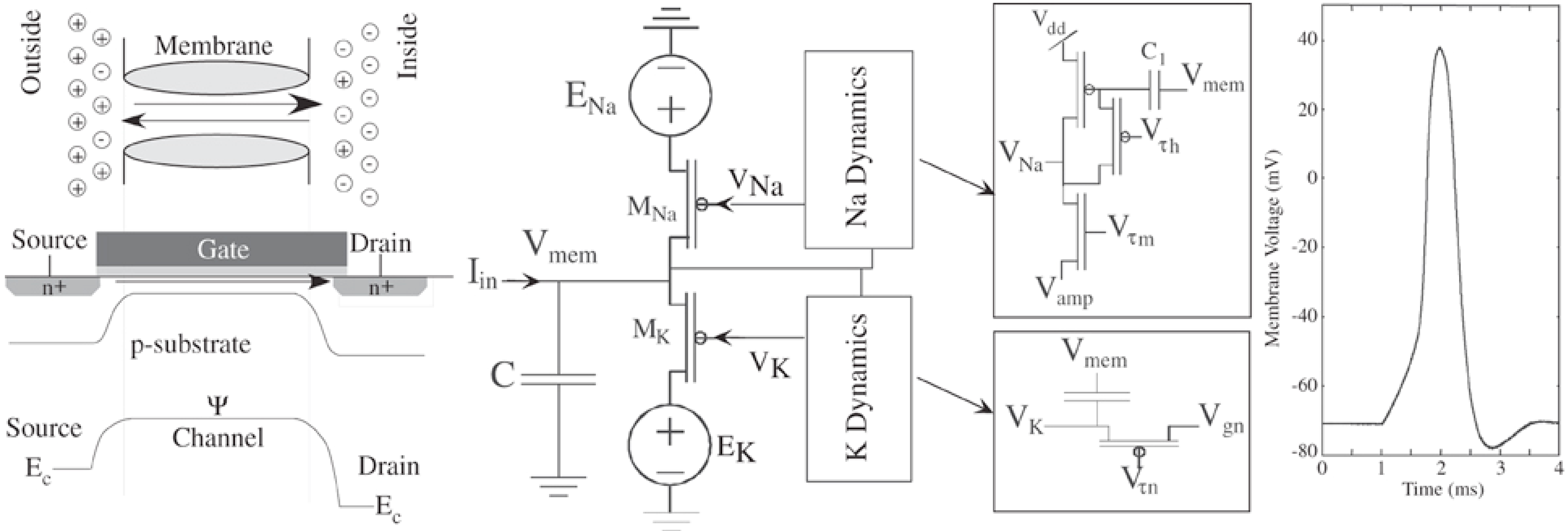
Given the community is making its first serious approaches toward large-scale neuromorphic hardware [e.g., FACETs (Schemmel et al., 2008a), DARPA SyNAPSE, Caviar (Serrano-Gotarredona)], a neuromorphic hardware roadmap could be seen as a way through the foreseen

TABLE OF CONTENTS

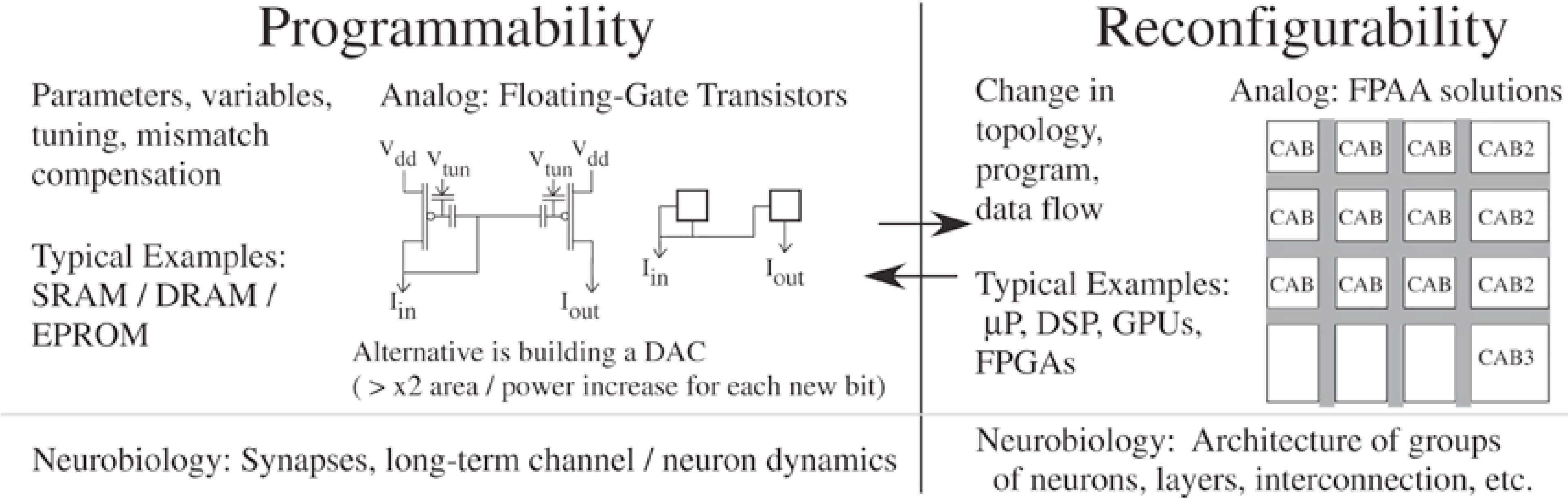
Power Efficiency Scaling



MOSFET Channel Modeling of Biological Channels



Field Programmable Analog Arrays



Programmability

Parameters, variables, tuning, mismatch compensation

Typical Examples: SRAM / DRAM / EPROM

Analog: Floating-Gate Transistors

Alternative is building a DAC
(> x2 area / power increase for each new bit)

Neurobiology: Synapses, long-term channel / neuron dynamics

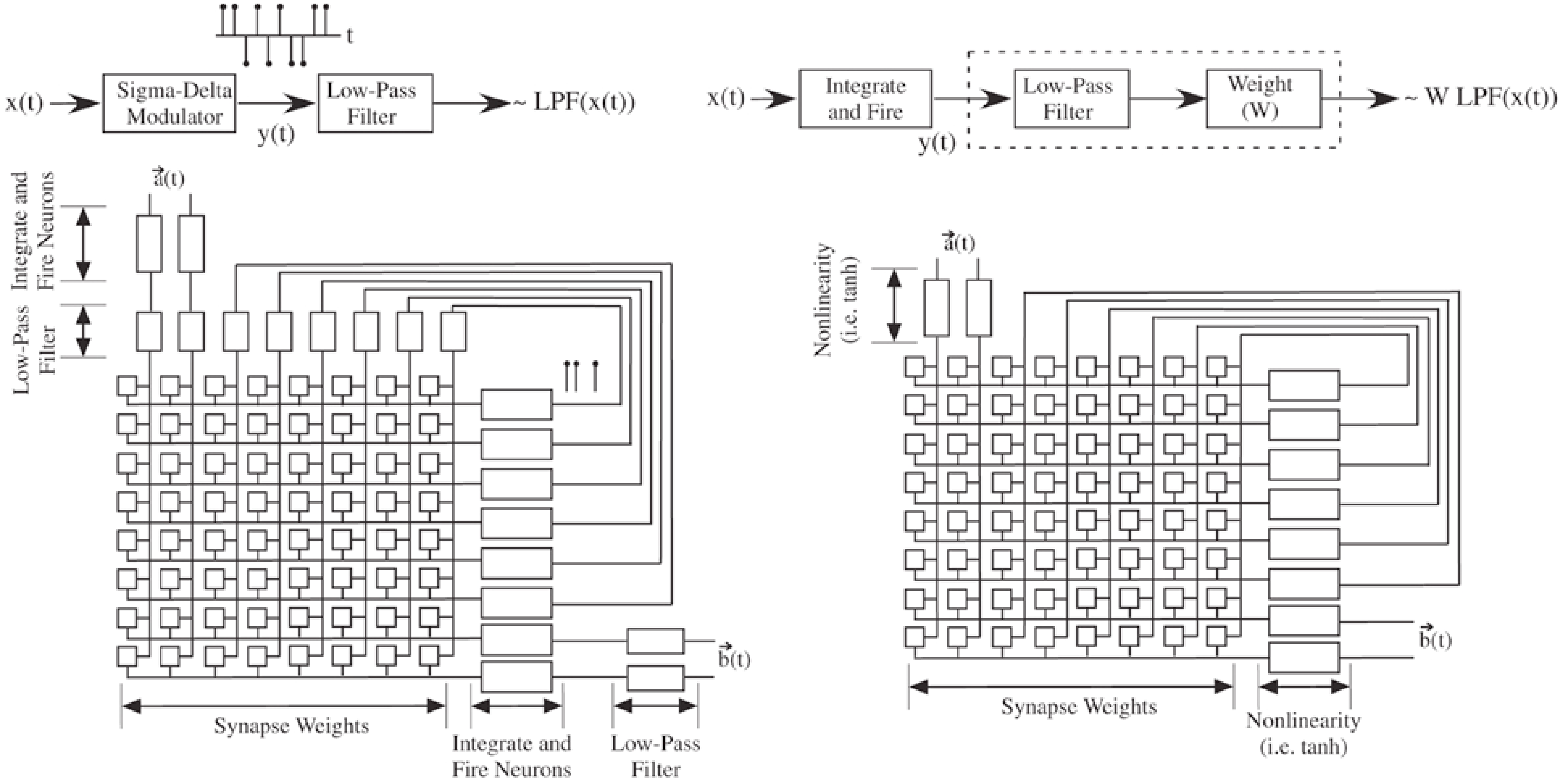
Reconfigurability

Change in topology, program, data flow

Typical Examples: µP, DSP, GPUs, FPGAs

Analog: FPAA solutions

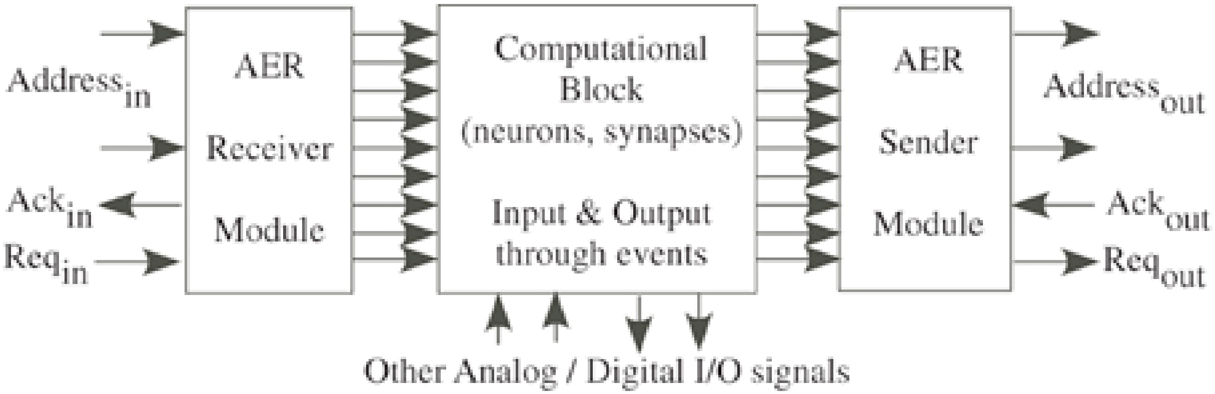
Neurobiology: Architecture of groups of neurons, layers, interconnection, etc.



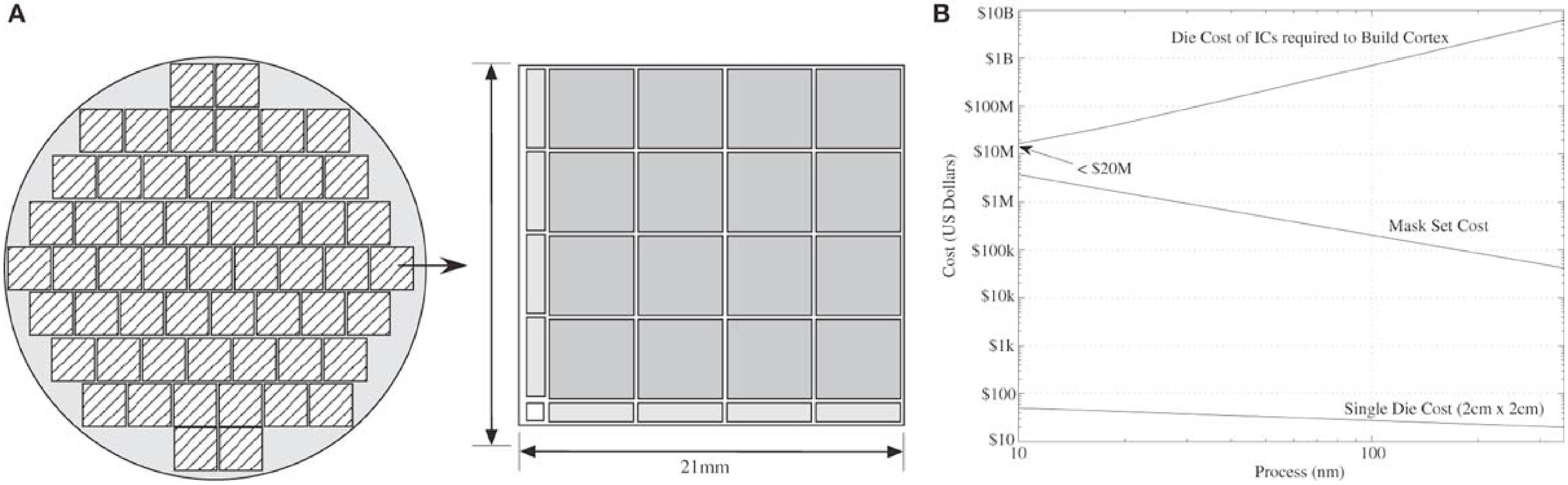
Chip built	Process node (nm)	Die area (mm^2)	No of synapses	Synapse area (μm^2)	Syn density	Synapse storage resolution and complexity
GT neuron1d (Brink et al., 2012)	350	25	30,000	133	**1088**	>10 bit, STDP
FACETs chip (Schemmel et al., 2006, 2008b)	180	25	98,304	108	3338	4 bit register
Stanford STDP	250	10.2	21,504	238	3810	STDP, no storage
INI chip (Indiveri et al., 2006)	800	1.6	256	4495	7023	1 bit w/learning dynam
ISS + INI chip (Camilleri et al., 2007)	350	68.9	16,384	3200	26,122	2.5 w/learning dynam

Bold value indicates synapse density as the synapse area normalized by the square of the process node.

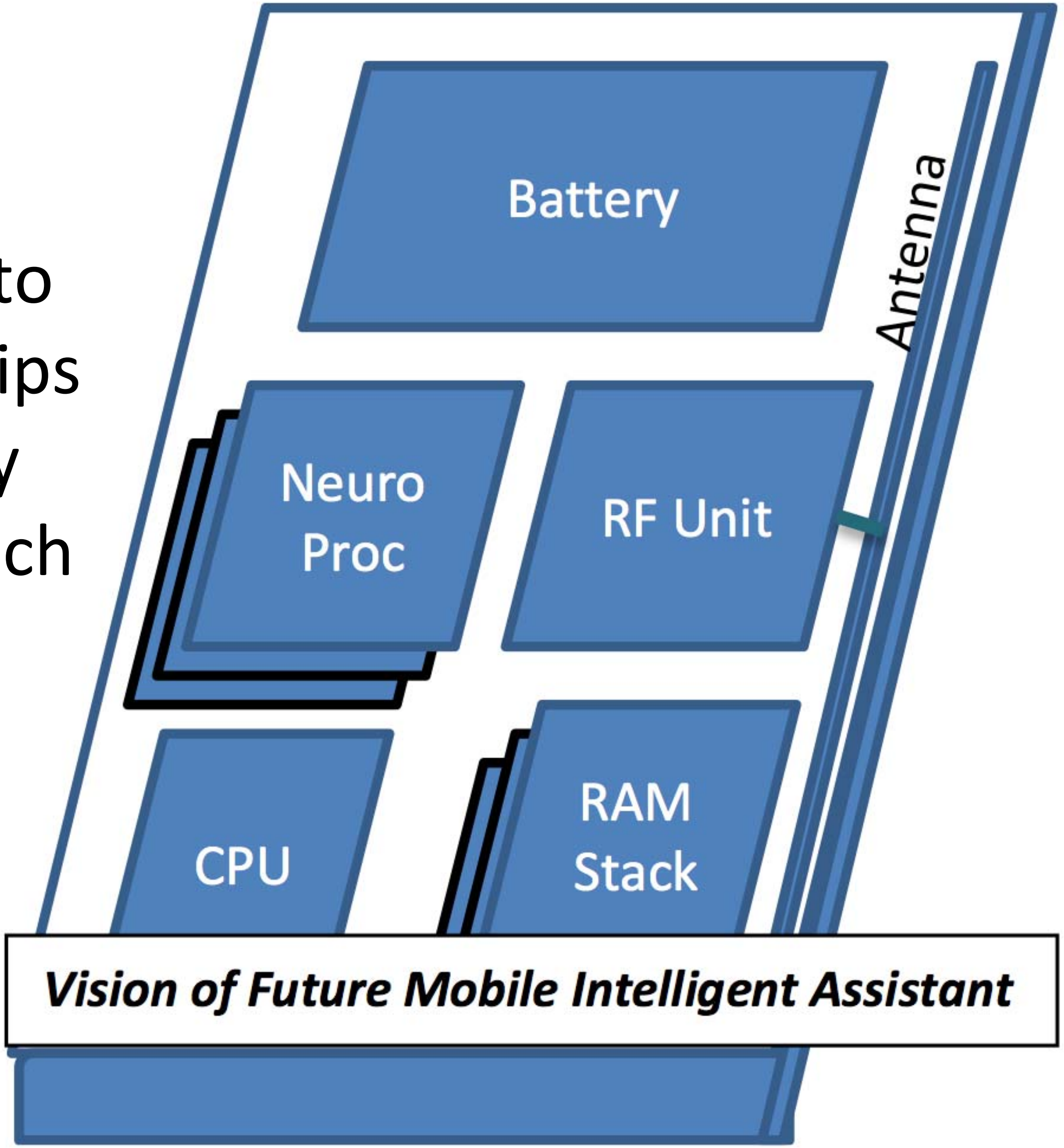
Address Event Representation



$20M dollar Cortex in 2020?

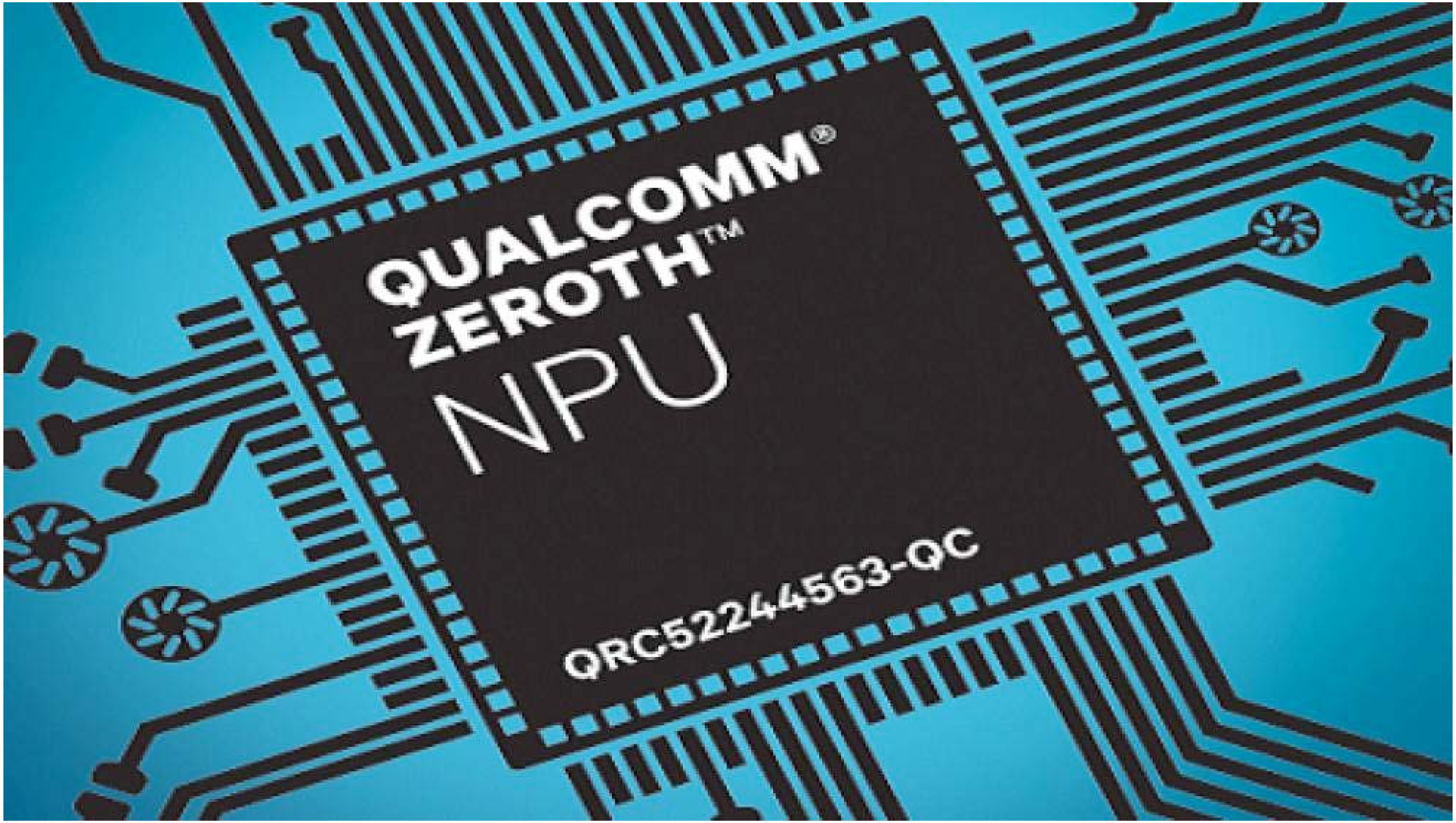


Qualcomm could add a "neural processing unit" to mobile-phone chips to handle sensory data and tasks such as image recognition.



Vision of Future Mobile Intelligent Assistant

Qualcomm Neuromorphic Chip



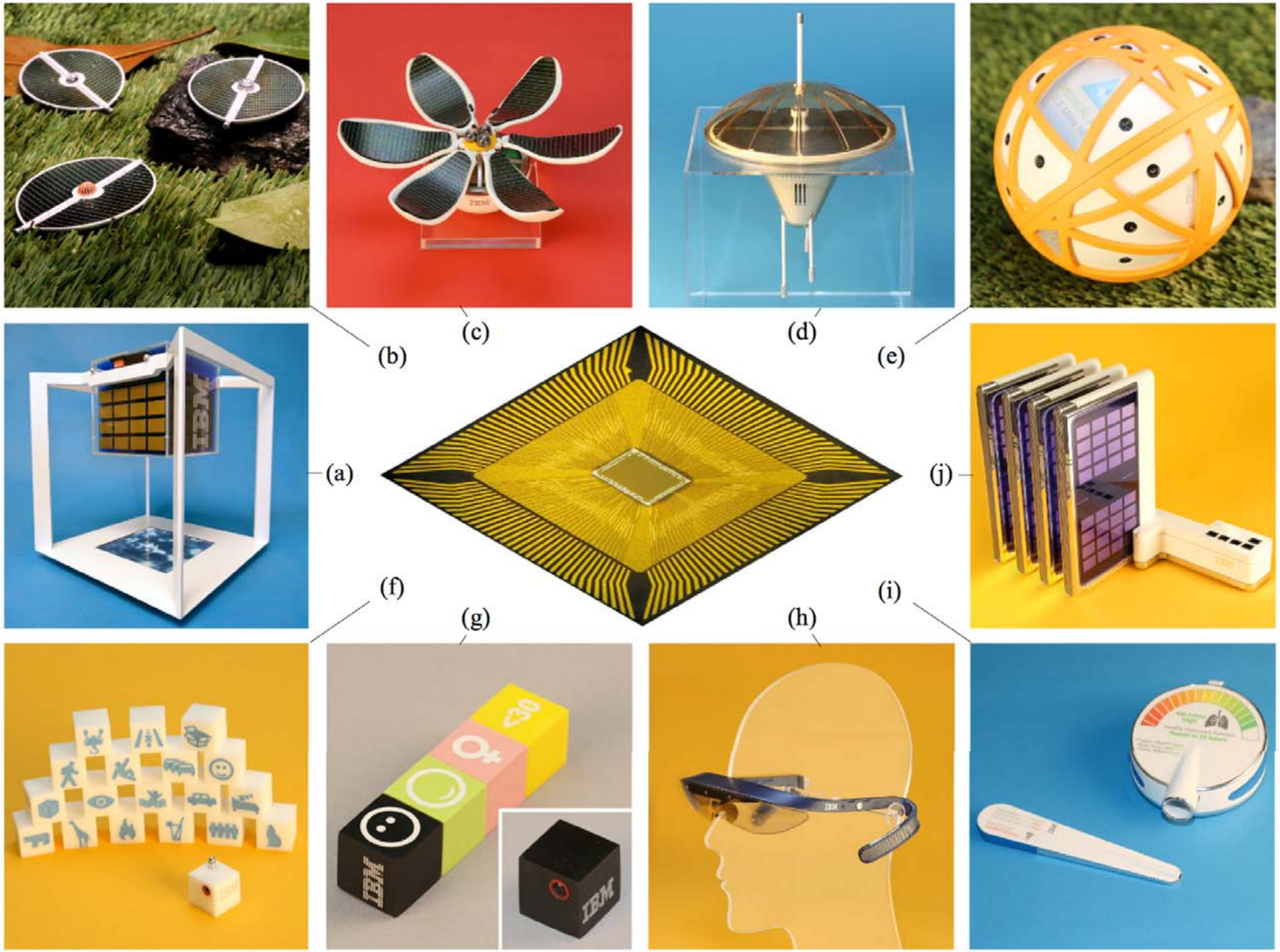


Fig. 1 Concept Models: (a) BrainCube, (b) Sensor Leaves, (c) Conversation Flower, (d) Jellyfish, (e) Tumbleweed, (f) Vision Cubes, (g) Composable Cubes, (h) Vision Assistive, (i) Home Health Wand and Pulmonary Monitor, (j) Build-a-Brain

Approximate Computing

- Historically computing hardware and software were designed for numerical calculations requiring a high degree of precision, such as 32 bits. But many present applications (such as image processing and data mining) do not require an exact answer; they just need a sufficiently good answer quickly. Furthermore, conventional logic circuits are highly sensitive to bit errors, which are to be avoided at all cost. But as devices get smaller and their counts get larger, the likelihood of random errors increases. Approximate computing represents a variety of software and hardware approaches that seek to trade off accuracy for speed, efficiency, and error-tolerance.

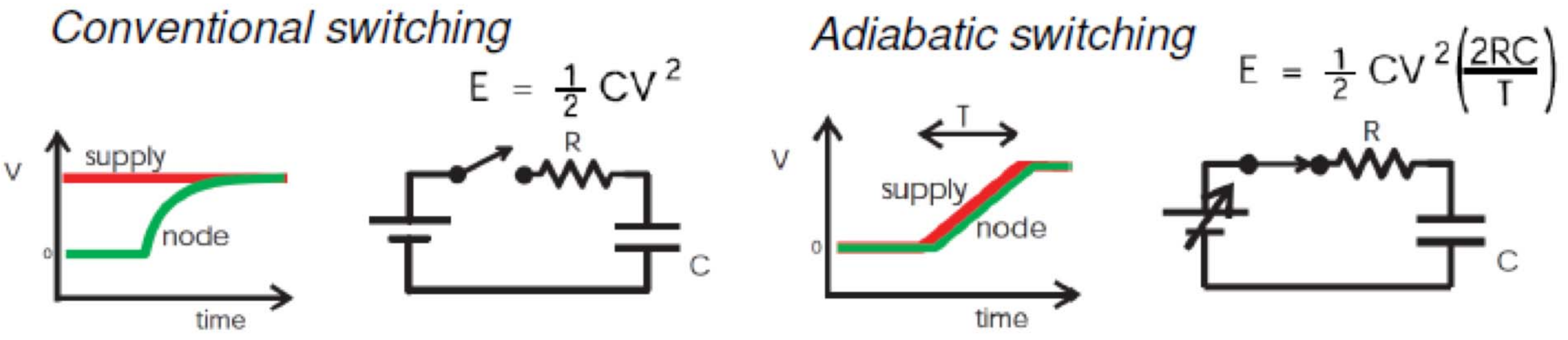
Adiabatic/Reversible Computing

- One of the primary sources of power dissipation in digital circuits is associated with switching of transistors and other elements. The basic binary switching energy is typically far larger than the fundamental limit ~kT, and much of the energy is effectively wasted. Adiabatic and reversible computing describe a class of approaches to reducing power dissipation on the circuit level by minimizing and reusing switching energy, and applying supply voltages only when necessary.

Adiabatic and Reversable Computing

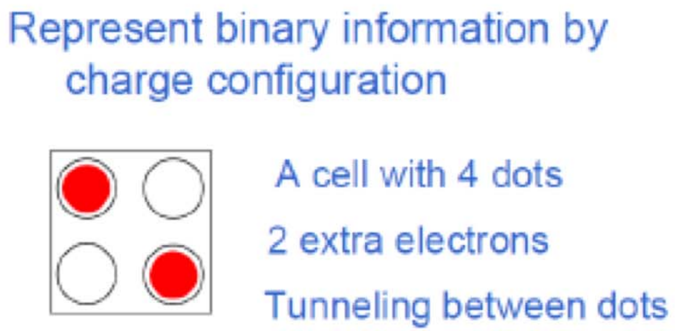
CMOS implementation would require 27x circuit overhead

Milestone targets 64-bit adder and/or 1 Gflops processors using 1% of current power



QCA

nSQUID

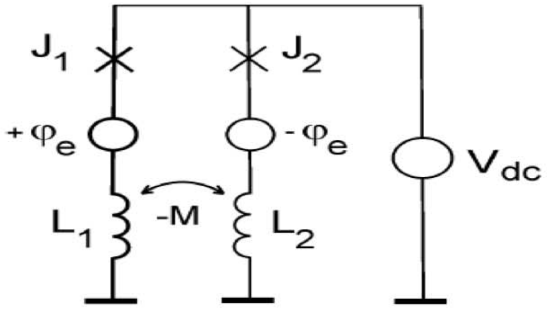


Represent binary information by charge configuration

A cell with 4 dots

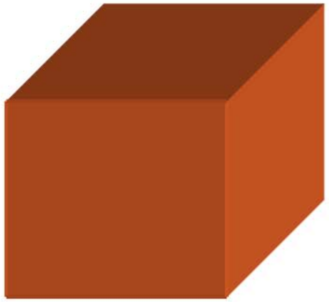
2 extra electrons

Tunneling between dots

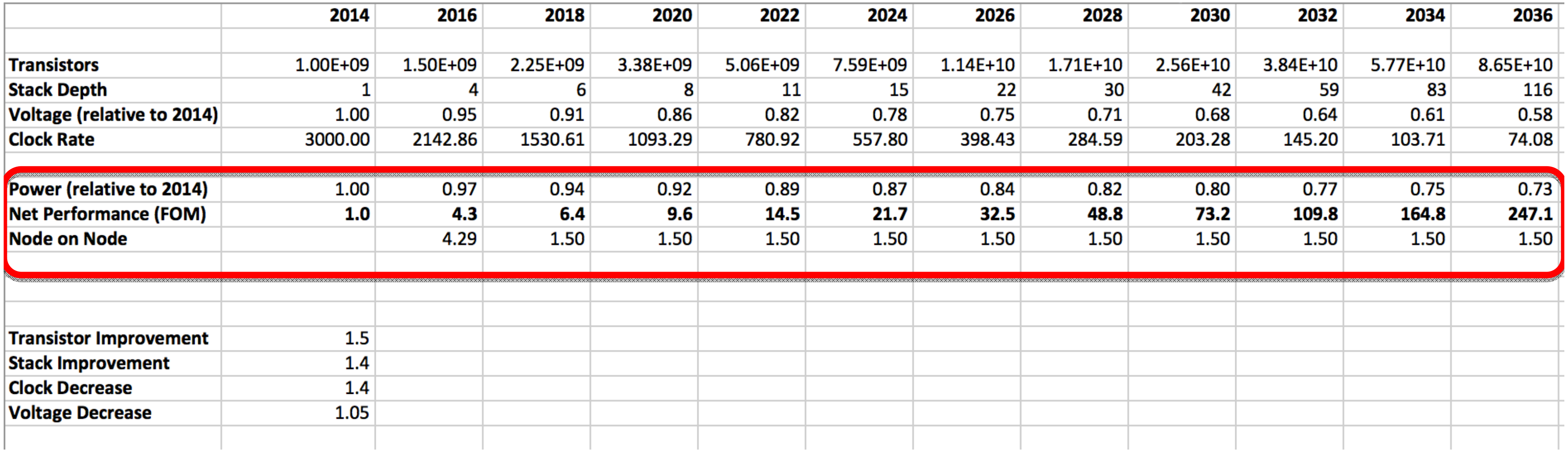


Ultra-Low Power Circuit Design

Combination of ideas.. Adiabatic for low power.. Low Clock for Low Power ..
3D stacking (either from packaging, or novel fab for high-density)
1 Mhz x 10^{15} Transistors == 3,000x improvement over current 2D design point

Timeframe	**Today**	**Changes**	**Tomorrow**
Integration scale	10^8 logic transistors	×10^7	10^{15} logic transistors
Clock speed	3 GHz	3000× slower	1 MHz
Performance	Chip is 2D comprised of 100 nm^2 gates. 	3000× reduction in joules/op OR 3000× increase in energy efficiency	Chip is 3D comprised of 100 nm^3 gates. 

Scaling Clock Down and Layers Up



	2014	2016	2018	2020	2022	2024	2026	2028	2030	2032	2034	2036
Transistors	1.00E+09	1.50E+09	2.25E+09	3.38E+09	5.06E+09	7.59E+09	1.14E+10	1.71E+10	2.56E+10	3.84E+10	5.77E+10	8.65E+10
Stack Depth	1	4	6	8	11	15	22	30	42	59	83	116
Voltage (relative to 2014)	1.00	0.95	0.91	0.86	0.82	0.78	0.75	0.71	0.68	0.64	0.61	0.58
Clock Rate	3000.00	2142.86	1530.61	1093.29	780.92	557.80	398.43	284.59	203.28	145.20	103.71	74.08
Power (relative to 2014)	1.00	0.97	0.94	0.92	0.89	0.87	0.84	0.82	0.80	0.77	0.75	0.73
Net Performance (FOM)	**1.0**	**4.3**	**6.4**	**9.6**	**14.5**	**21.7**	**32.5**	**48.8**	**73.2**	**109.8**	**164.8**	**247.1**
Node on Node		4.29	1.50	1.50	1.50	1.50	1.50	1.50	1.50	1.50	1.50	1.50
Transistor Improvement	1.5											
Stack Improvement	1.4											
Clock Decrease	1.4											
Voltage Decrease	1.05											

Early Efforts in Reversible Computing

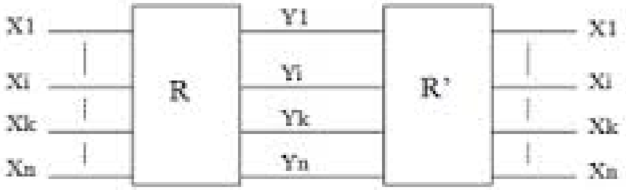
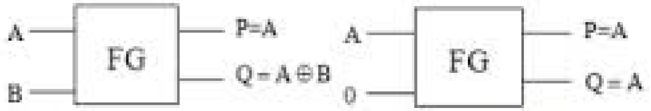
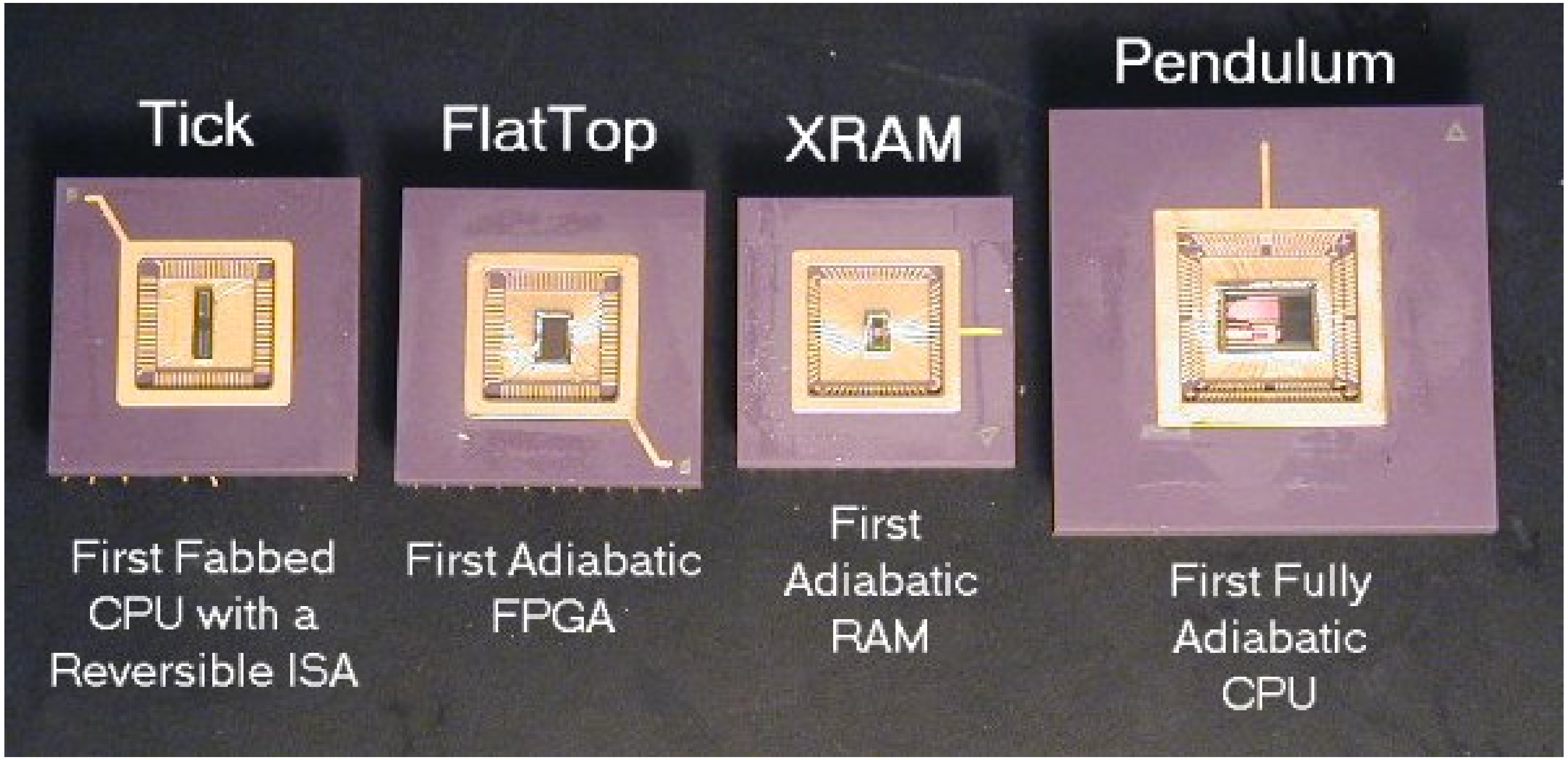


Figure 1. Cascading of a reversible gate R with its inverse R'



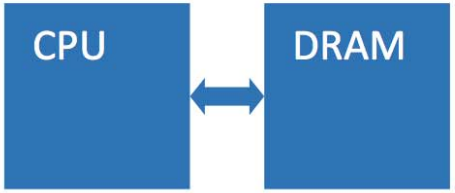
(a) Feynman gate (b) Feynman gate for copying


Tick
FlatTop
XRAM
Pendulum
First Fabbed CPU with a Reversible ISA
First Adiabatic FPGA
First Adiabatic RAM
First Fully Adiabatic CPU

RCS2 Trending…

Current computer technology:

Hardware:

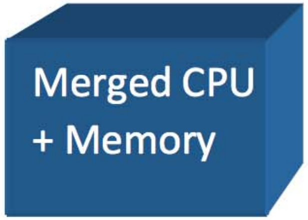


Software for von Neumann architecture:

- FORTRAN, C, Java
- SQL
- HTML
- etc.

Emerging vision of rebooted computer technology:

Hardware:



- Continued exponential increase in devices using third dimension
- Improved power efficiency

Software modes:

- von Neumann-class (FORTRAN, C, Java, SQL, HTML, etc.)
- Highly-parallel (GPU code like CUDA)
- Neuromorphic
- Approximate
- etc.

On the Way to the Forum

- Simple but complete abstractions to test new computing substrates
- Ultra RISC is one such approach
  - One Instruction Set Computer (OISC)
- Universal computers
  - Transport Triggered Architecture Machines
  - Bit Manipulating Machines
  - Arithmetic Based Turing-Complete Machines*

SUBLEQ

Subtract and branch if less than or equal to zero

The subleq instruction ("SUbtract and Branch if Less than or EQual to zero") subtracts the contents at address *a* from the contents at address *b*, stores the result at address *b*, and then, *if the result is not positive*, transfers control to address *c* (if the result is positive, execution proceeds to the next instruction in sequence).

```
subleq a, b, c        ; Mem[b] = Mem[b] - Mem[a]
                          ; if (Mem[b] ≤ 0) goto c
```

28 Subleq on an FPGA

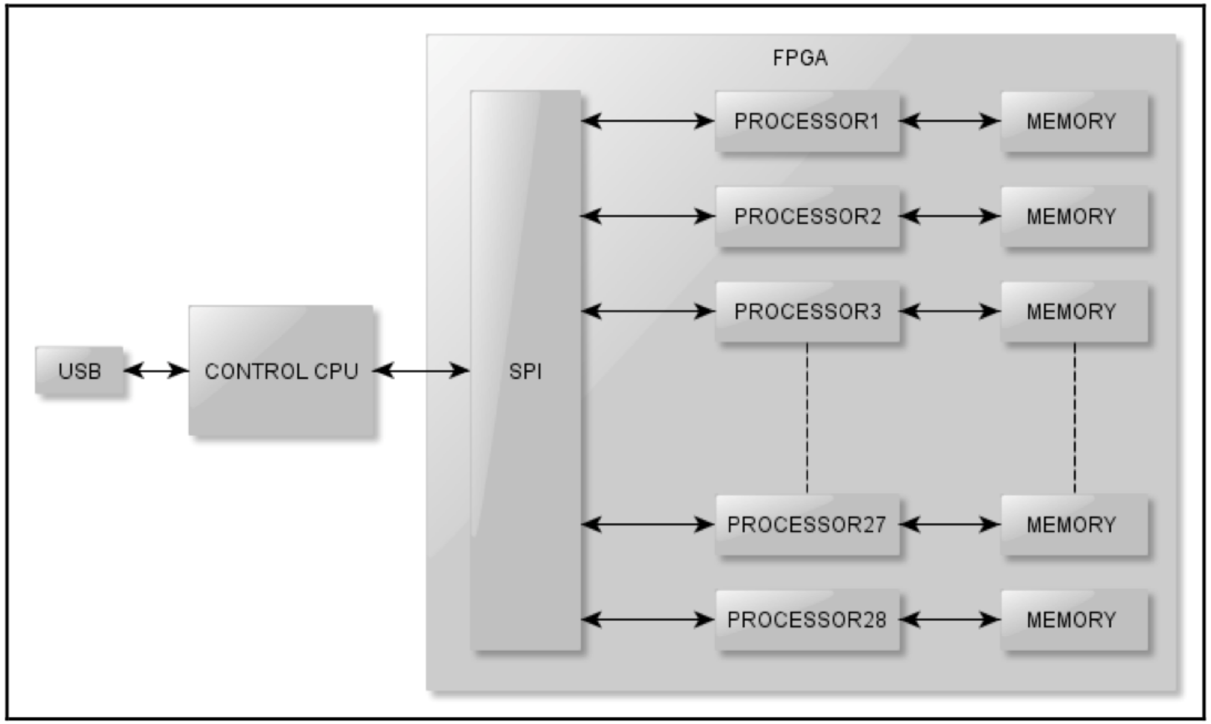


Figure 2 Block-diagram of the board

28 Subleq on an FPGA

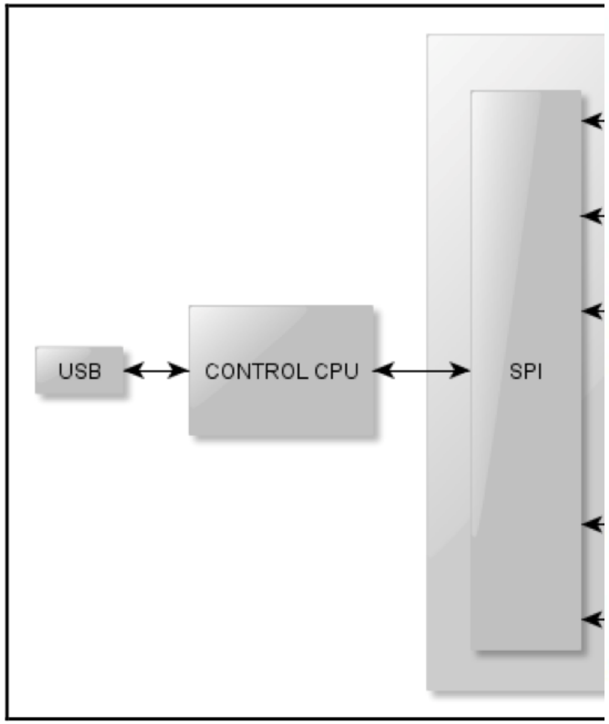


Figure 2 Block-di

Figure 4 FPGA board, 28 Subleq processors with allocated 2 Kb per processor

Carbon Nanotube Computer
Stanford in Nature September 2013

LETTER

doi:10.1038/nature12502

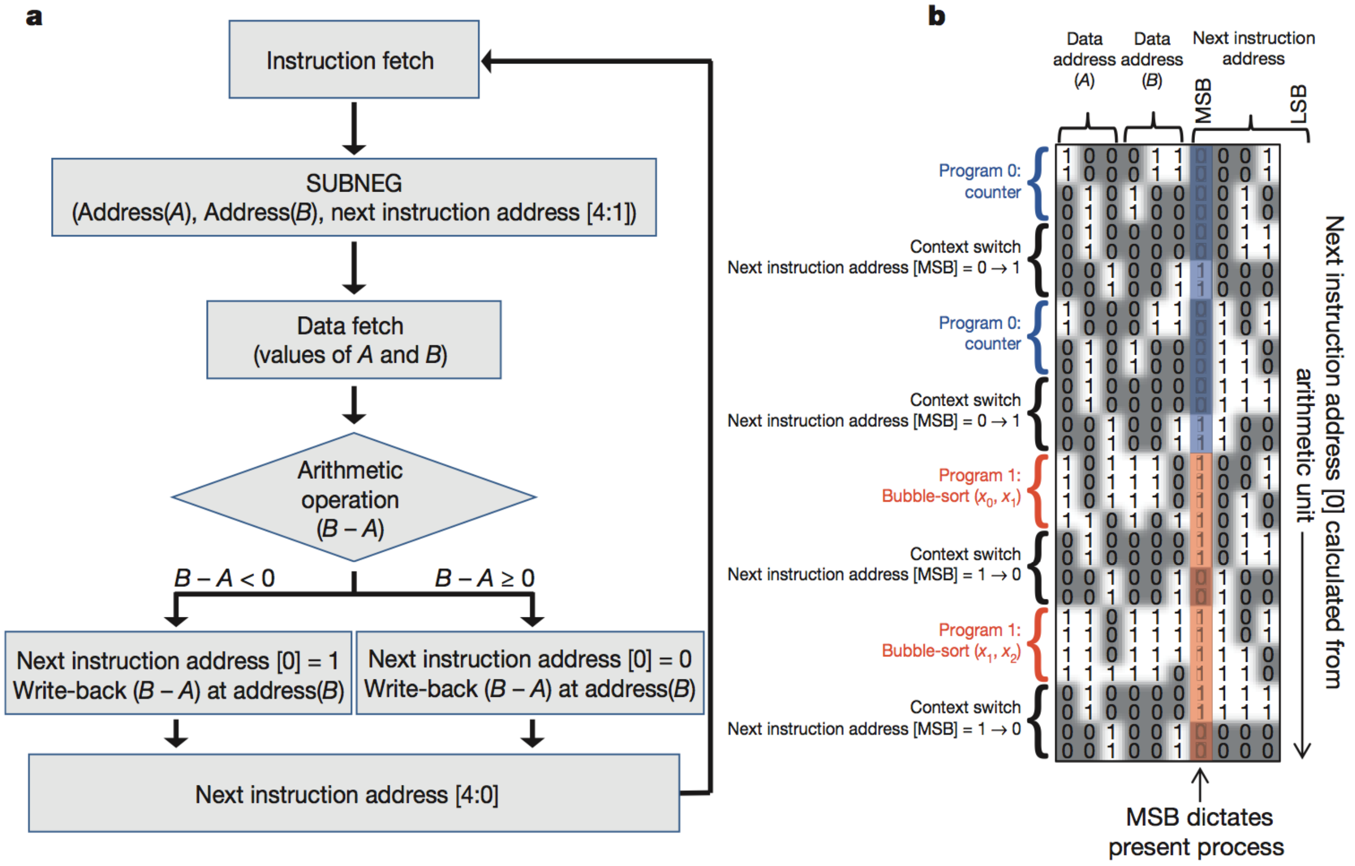
Carbon nanotube computer

Max M. Shulaker[1], Gage Hills[2], Nishant Patil[3], Hai Wei[4], Hong-Yu Chen[5], H.-S. Philip Wong[6] & Subhasish Mitra[7]

The miniaturization of electronic devices has been the principal driving force behind the semiconductor industry, and has brought about major improvements in computational power and energy efficiency. Although advances with silicon-based electronics continue to be made, alternative technologies are being explored. Digital circuits based on transistors fabricated from carbon nanotubes (CNTs)

to incorrect logic functionality, whereas metallic CNTs have little or no bandgap, resulting in high leakage currents and incorrect logic functionality[20]. The imperfection-immune design methodology, which combines circuit design techniques with CNT processing solutions, overcomes these problems[20,21]. It enables us to demonstrate, for the first time, a complete CNT computer, realized entirely using CNFETs.

MIPS on top of SUBNEG on top of CNT

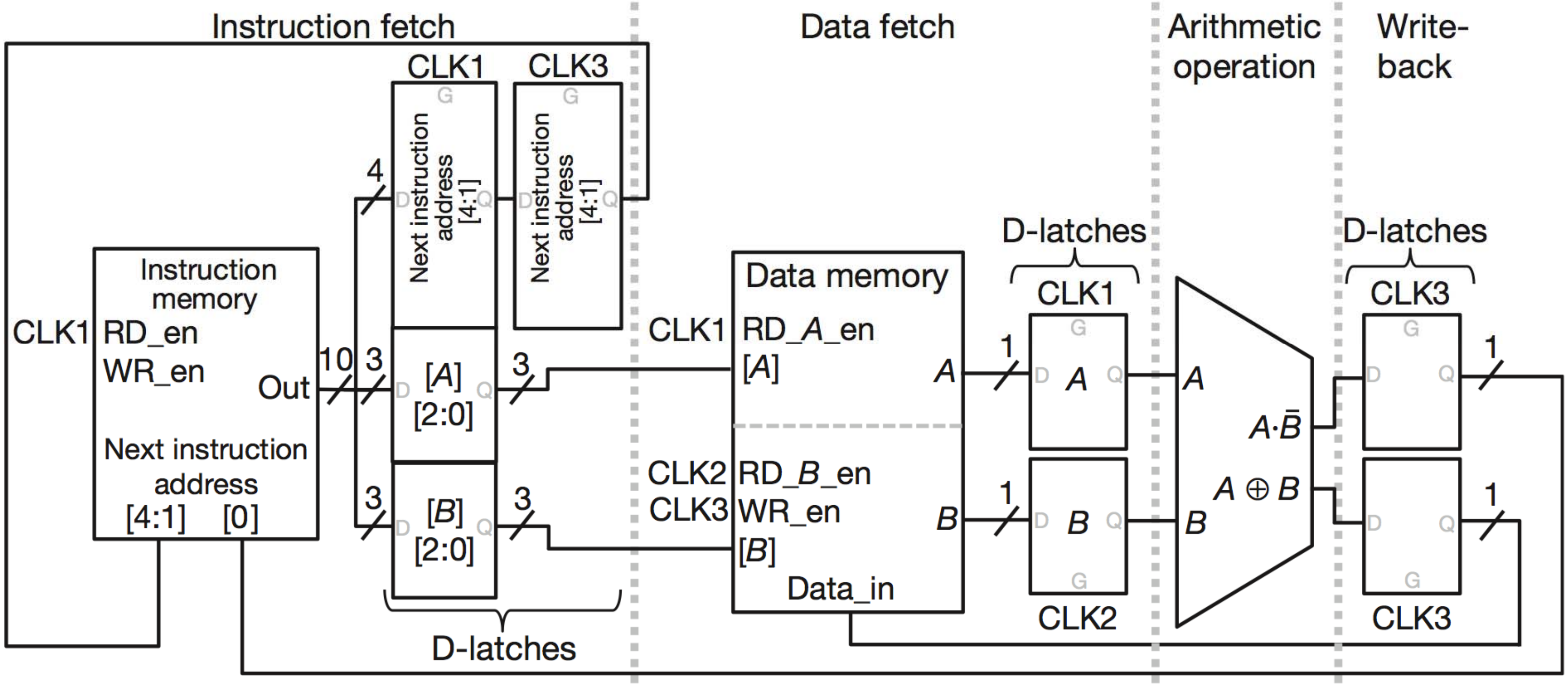

a
Instruction fetch
SUBNEG
(Address(A), Address(B), next instruction address [4:1])
Data fetch
(values of A and B)
Arithmetic operation (B – A)
B – A < 0
B – A ≥ 0
Next instruction address [0] = 1
Write-back (B – A) at address(B)
Next instruction address [0] = 0
Write-back (B – A) at address(B)
Next instruction address [4:0]
b
Data address (A)
Data address (B)
Next instruction address
MSB
LSB
Program 0: counter
Context switch
Next instruction address [MSB] = 0 → 1
Program 0: counter
Context switch
Next instruction address [MSB] = 0 → 1
Program 1: Bubble-sort (x_0, x_1)
Context switch
Next instruction address [MSB] = 1 → 0
Program 1: Bubble-sort (x_1, x_2)
Context switch
Next instruction address [MSB] = 1 → 0
Next instruction address [0] calculated from arithmetic unit
MSB dictates present process

	Data address (A)			Data address (B)			Next instruction address			
							MSB			LSB
Program 0: counter	1	0	0	0	1	1	0	0	0	1
	1	0	0	0	1	1	0	0	0	1
	0	1	0	1	0	0	0	0	1	0
	0	1	0	1	0	0	0	0	1	0
Context switch Next instruction address [MSB] = 0 → 1	0	1	0	0	0	0	0	0	1	1
	0	1	0	0	0	0	0	0	1	1
	0	0	1	0	0	1	1	0	0	0
	0	0	1	0	0	1	1	0	0	0
Program 0: counter	1	0	0	0	1	1	0	1	0	1
	1	0	0	0	1	1	0	1	0	1
	0	1	0	1	0	0	0	1	1	0
	0	1	0	1	0	0	0	1	1	0
Context switch Next instruction address [MSB] = 0 → 1	0	1	0	0	0	0	0	1	1	1
	0	1	0	0	0	0	0	1	1	1
	0	0	1	0	0	1	1	1	0	0
	0	0	1	0	0	1	1	1	0	0
Program 1: Bubble-sort (x_0, x_1)	1	0	1	1	1	0	1	0	0	1
	1	0	1	1	1	0	1	0	0	1
	1	0	1	1	1	0	1	0	1	0
	1	1	0	1	0	1	1	0	1	0
Context switch Next instruction address [MSB] = 1 → 0	0	1	0	0	0	0	1	0	1	1
	0	1	0	0	0	0	1	0	1	1
	0	0	1	0	0	1	0	1	0	0
	0	0	1	0	0	1	0	1	0	0
Program 1: Bubble-sort (x_1, x_2)	1	1	0	1	1	1	1	1	0	1
	1	1	0	1	1	1	1	1	0	1
	1	1	0	1	1	1	1	1	1	0
	1	1	1	1	1	0	1	1	1	0
Context switch Next instruction address [MSB] = 1 → 0	0	1	0	0	0	0	1	1	1	1
	0	1	0	0	0	0	1	1	1	1
	0	0	1	0	0	1	0	0	0	0
	0	0	1	0	0	1	0	0	0	0

Figure 1 | SUBNEG and program implementation. a, Flowchart showing the implementation of the SUBNEG instruction. **b,** Sample program on CNT computer. Each row of the chart is a full SUBNEG instruction. It is composed of two data addresses and a partial next instruction address. The (omitted) least significant bit (LSB) of the next instruction address is calculated by the arithmetic unit of the CNT computer, and the most significant bit (MSB) of the next instruction address indicates the running program, either a counter or bubble-sort algorithm in this instance.

CNT Schematic

The CNFET computer is composed of 178 CNFETs, with each CNFET comprising, 10–200 CNTs, depending on relative sizing of the widths of the CNFETs.



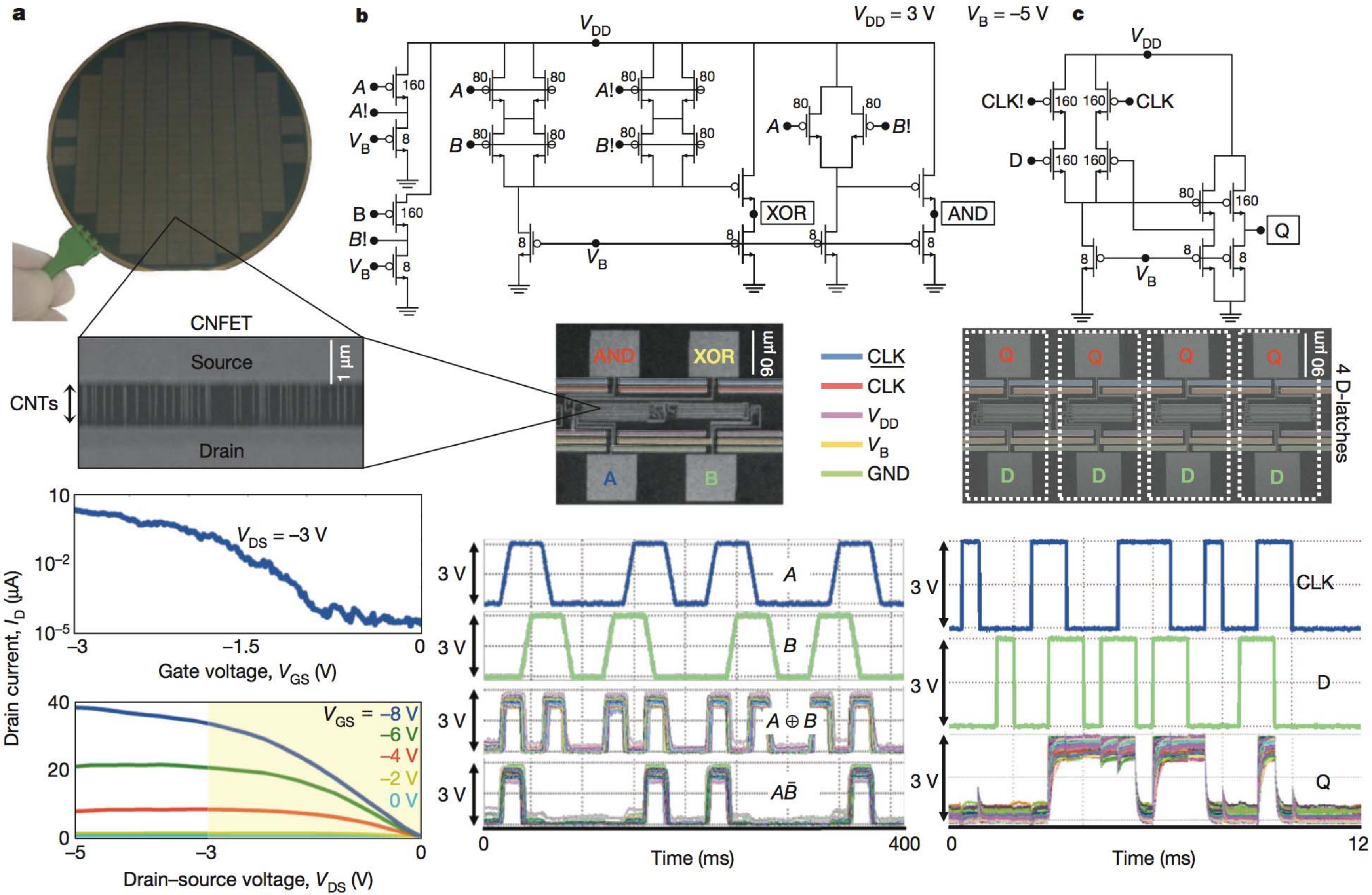


Figure 3 | **Characterization of CNFET subcomponents.** **a**, Top: Final 4-inch wafer after all fabrication. Middle: scanning electron microscope (SEM) image of a CNFET, showing source, drain and CNTs extending into the channel region. Bottom, Measured characterization (current–voltage) curves of a typical CNFET. The yellow highlighted region of the I_D–V_{DS} curve shows the biasing region that the CNFET operates in for the CNT computer. **b**, Top: transistor-level schematic of arithmetic unit. Numbers are width of transistors (in micrometres). Middle: SEM of an arithmetic unit. Bottom: measured outputs from 40 different arithmetic units, all overlaid. **c**, Top: transistor-level schematic of D-latches. Numbers are width of transistors (in micrometres). Middle: SEM of a bank of 4 D-latches. Bottom: measured outputs from 200 different D-latches, all overlaid.

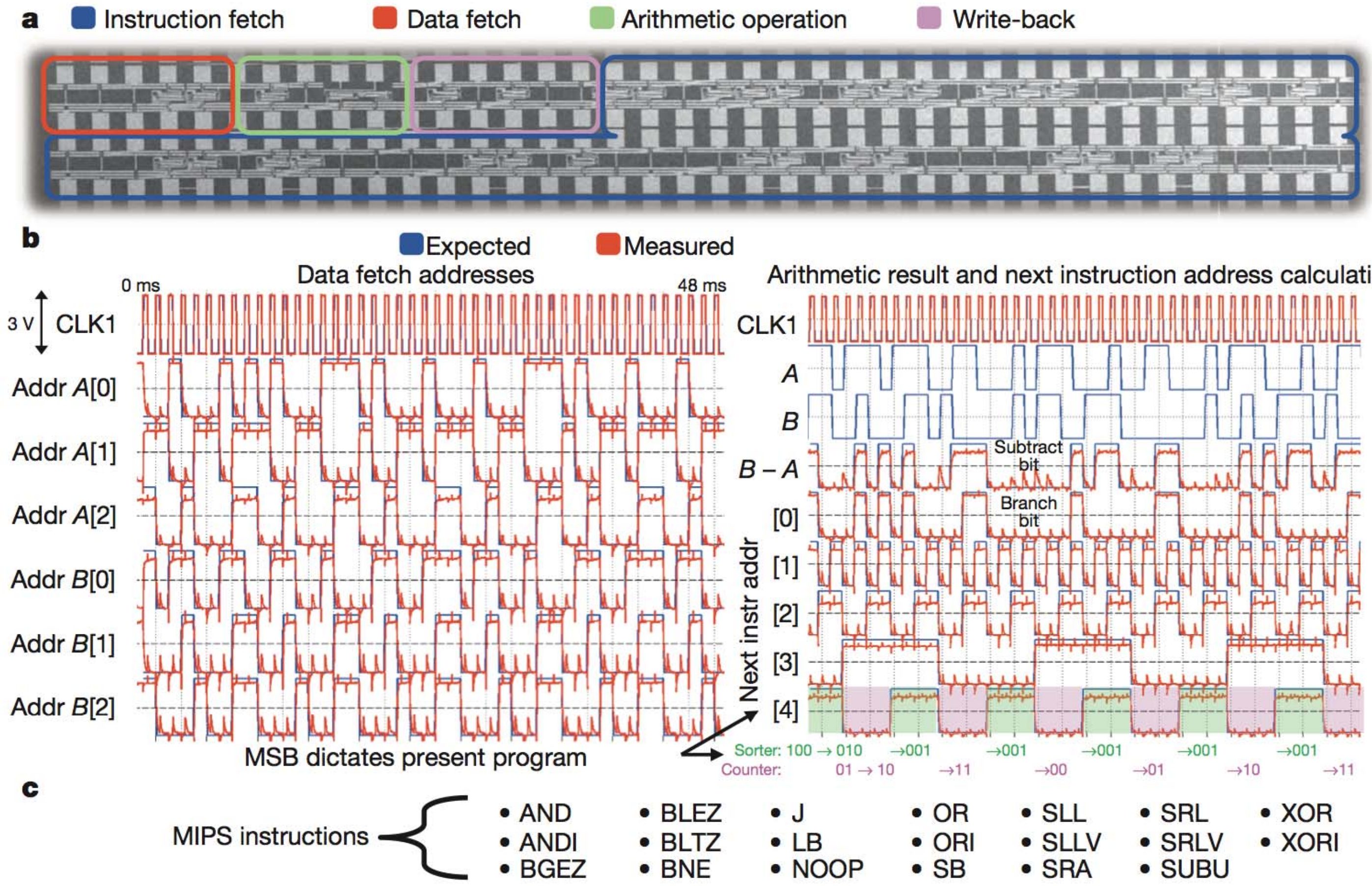


Figure 4 | CNT computer results. **a**, SEM of an entire CNT computer. **b**, Measured and expected output waveforms for a CNT computer, running the program shown in Fig. 1b. The exact match in logic value of the measured and expected output shows correct operation. As shown by the MSB (denoted [4]) of the next instruction address, the computer is switching between performing counting and sorting (bubble-sort algorithm). The running results of the counting and sorting are shown in the rows beneath the MSB of the next instruction address. **c**, A list of the 20 MIPS instructions tested on the CNT computer.

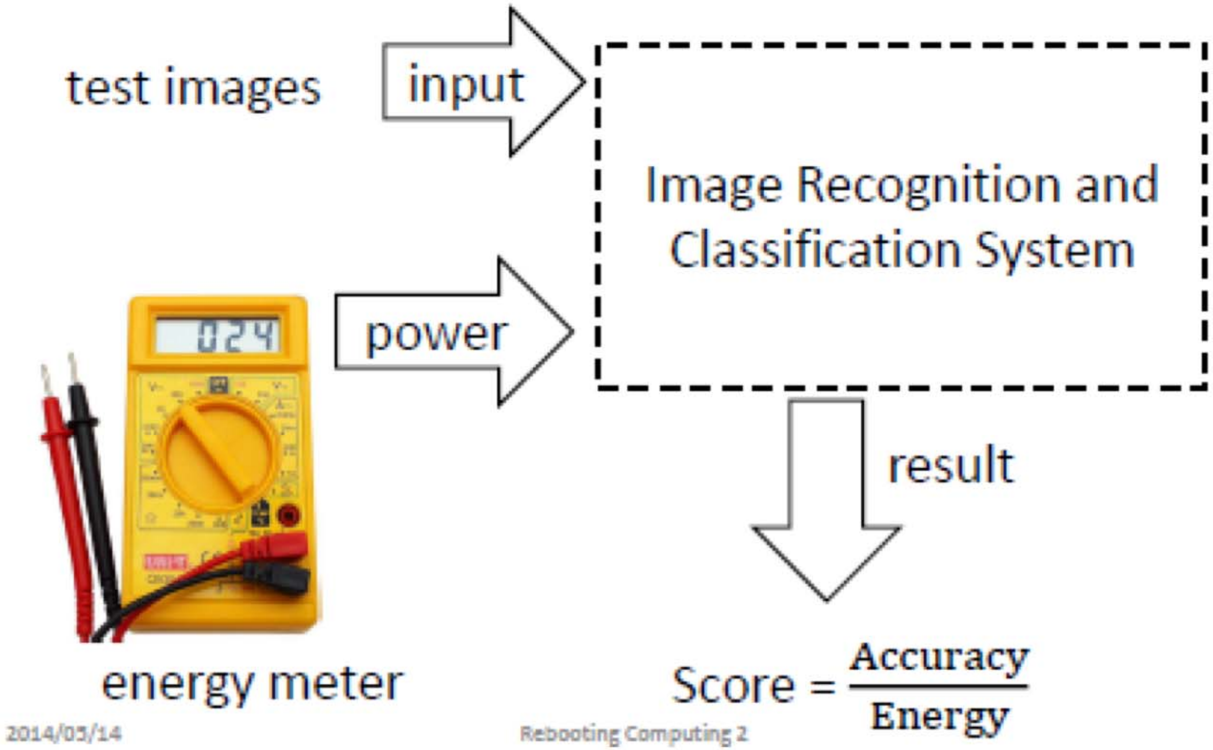
Some Other Things to Ponder

- Abstractions that can be used to evaluate future computing substrates (OISC or better)
  - Von neumann
  - Neuromorphics
- Programming models that encompass
  - Abstract models of data structures (e.g. IPM)
  - Data storage concepts into the language (e.g. NVRAM)
  - Computable Knowledge concepts into the language

Prizes for Rebooting Computing?

IEEE Competition for Low-Power Image Recognition, Yung-Hsiang Lu, Purdue

Prof. Lu proposed an IEEE prize competition, focusing on Low-Power Image Recognition using a mobile device, possibly for 2015. This would involve presentation of a set of test images to the device, and a limited time to accurately identify the images.



test images

input

Image Recognition and Classification System

power

energy meter

result

$$\text{Score} = \frac{\textbf{Accuracy}}{\textbf{Energy}}$$

2014/05/14

Rebooting Computing 2