Improving Energy Efficiency via Nonlinear Dynamics and Chaos

SAND2015-10599C

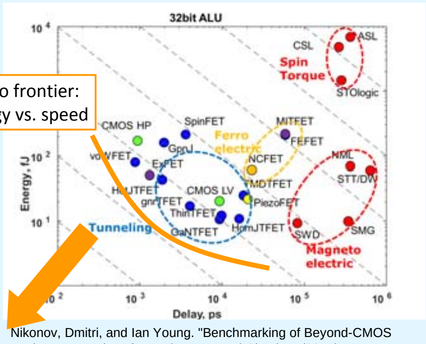
Erik P. DeBenedictis[1], Neal G. Anderson[2], Michael P. Frank[1], Natesh Ganesh[2], R. Stanley Williams[3],
[1]Center for Computing Research, Sandia, [2]ECE Department, University of Massachusetts Amherst, [3]Hewlett Packard Labs

1. The Problem

Comprehensive analytical comparisons strongly suggest transistor replacements and other advanced logic devices have limited potential for improved energy efficiency.

Diagram illustrates delay and energy consumption of over a dozen devices when composed into standard logic circuits. A 32-bit ALU is used as the benchmark circuit.

Pareto frontier: energy vs. speed



Community seeks a strategy to move significantly beyond the limit

Nikonov, Dmitri, and Ian Young. "Benchmarking of Beyond-CMOS Exploratory Devices for Logic Integrated Circuits." (2015).

2. Strategy: Avoid the Boolean Logic Abstraction

Simplifying assumptions are currently reducing energy efficiency:

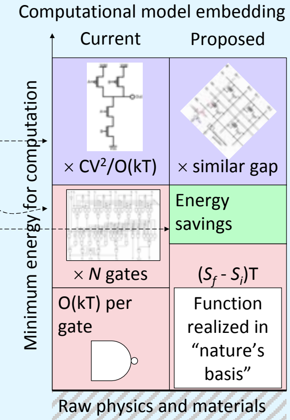
- The current approach has two steps: use physics to create Boolean logic gates, then use those gates to create the desired function
- The proposal is to use nonlinear dynamics and chaos in the behavior of new or existing devices to create the desired function in one step

Theory and practice

- There are well-defined theoretical minimums on energy consumption
- However, energy of practical systems tends to multiply minimum energy by a manufacturing factor

Effect of extra layers

- Theoretically $E_{min}(f(g(x))) \le E_{min}(f(g)) + E_{min}(g(x))$
- Basically, computing f in two parts will have higher minimum energy unless the parts exactly fit



New degrees of freedom

- Optimize devices for needed function rather than Boolean logic gates
- Realize function more efficiently than Boolean logic
- Aggregation lowers minimum energy $E_{min}(f(g(x))) \le E_{min}(f(x)) + E_{min}(g(x))$
- Exploit probabilities – optimize energy efficiency for likely data sets
- Use logic-in-memory

3. Learning Example

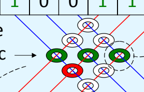
This “learning machine” example exceeds energy efficiency limits of Boolean logic. The learning machine monitors the environment for knowledge, yet usually just verifies that it has learned what it needs to know. Say “causes” (lion, apple, and night) and “effects” (danger, food, and sleep) have value 1.

Example input:
{lion, danger } {apple, food } {night, sleep } {lion, danger } {apple, food } {night, sleep } {lion, danger } {apple, food } {night, sleep } {lion, danger } {apple, food } {night, sleep } { lion, danger, food } {lion, danger }

continues indefinitely

lion	apple	night	danger	food	sleep
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	0	0	1	1	0

Old-style magnetic cores

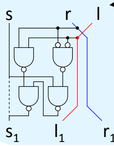


1	1	0	1	0	0
0	0	0	0	0	0

Signals create currents; core flips a ±1.5

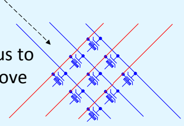
Functional example:
Machine continuously monitors environment for {1, 1} or {-1, -1} pairs and remembers them in state of a magnetic core. Theoretically, there is no need for energy consumption unless state changes

CMOS implementation:

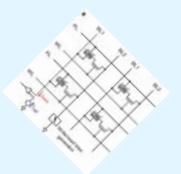


Notes: like Landauer’s “machine,” but r and l are trits & s, s_1 are state

Array analogous to cores above



Possible MeRAM implementation:
Magnetoelectric RAM is based on a device where voltage exceeding a threshold causes a nanomagnet to flip. Losses are negligible in absence of state change.



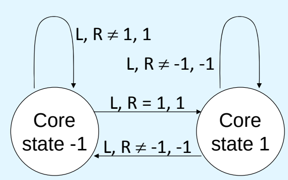
Hu, Jia-mian, et al. "High-density magnetoresistive random access memory operating at ultralow voltage at room temperature." *Nature communications* 2 (2011): 553

4. Theoretical Analysis

Diagram is the same calculation as in Landauer’s paper. In lieu of Boolean logic with O(kT) energy/gate, diagram is for a learning machine directly, with 1% probability of seeing input data to be learned and 0.01% probability of seeing contradictory data.

Probability of data to be learned:								**0.01**		
Probability of conflicting data:								**0.0001**		
Probability	left wire	right wire	field dir.		left wire	right wire	field dir.	Si (k's)	State	Sf (k's)
0.000001	-1	-1	-1	→	-1	-1	-1	0.000014	A	0.000921
0.001400	-1	0	-1	→	-1	0	-1	0.009201	B1	0.009201
Seven copies of row above for sequential input combinations (states C1-H1)										
0.000099	1	1	-1	→	1	1	1	0.000913	I	
0.000099	-1	-1	1	→	-1	-1	-1	0.000913	A	
0.140014	-1	0	1	→	-1	0	1	0.275269	B2	0.275269
Seven copies of row above for sequential input combinations (states C2-H2)										
0.009901	1	1	1	→	1	1	1	0.045694	I	0.046052
							Si (k's)	2.038824	Sf (k's)	2.038263
									Si-Sf (k's)	**0.000561**

Synapse as finite-state automata:



Learning cost lower bound

.00056 kT

per core per input, which is << O(kT)

See N. Ganesh and N. G. Anderson, “Irreversibility and Dissipation in Finite-State Automata” Phys Lett A(2013)

5. Generalization

The general design flow for using the nonlinear dynamics and chaos (if chaos is present) is as follows:

- Find the most theoretically energy-efficient implementation of the desired function in terms of manipulation of physical variables
- Try to exploit non-uniform probabilities in the problem and data
- Try to base devices on idealizations of known logic, memory, or state-containing logic devices
- Seek devices already invented with the required behavior, or discover new ones
- Optimize the devices to come as close as possible to physical limits

6. Conclusions

The Boolean logic abstraction offers intellectual elegance and reduces design effort, but may reduce energy efficiency. This poster gives one example where a new circuit based on a new MeRAM device theoretically improves energy efficiency by several orders of magnitude over accepted projections of Boolean logic gates. A route to improved energy efficiency was demonstrated for a “learning machine,” but generalization to other problems is beyond the scope of this poster.

Approved for unclassified unlimited release by Sandia. Tracking Number: 365436.