Beyond Moore's Law and Implications for Computing in Space

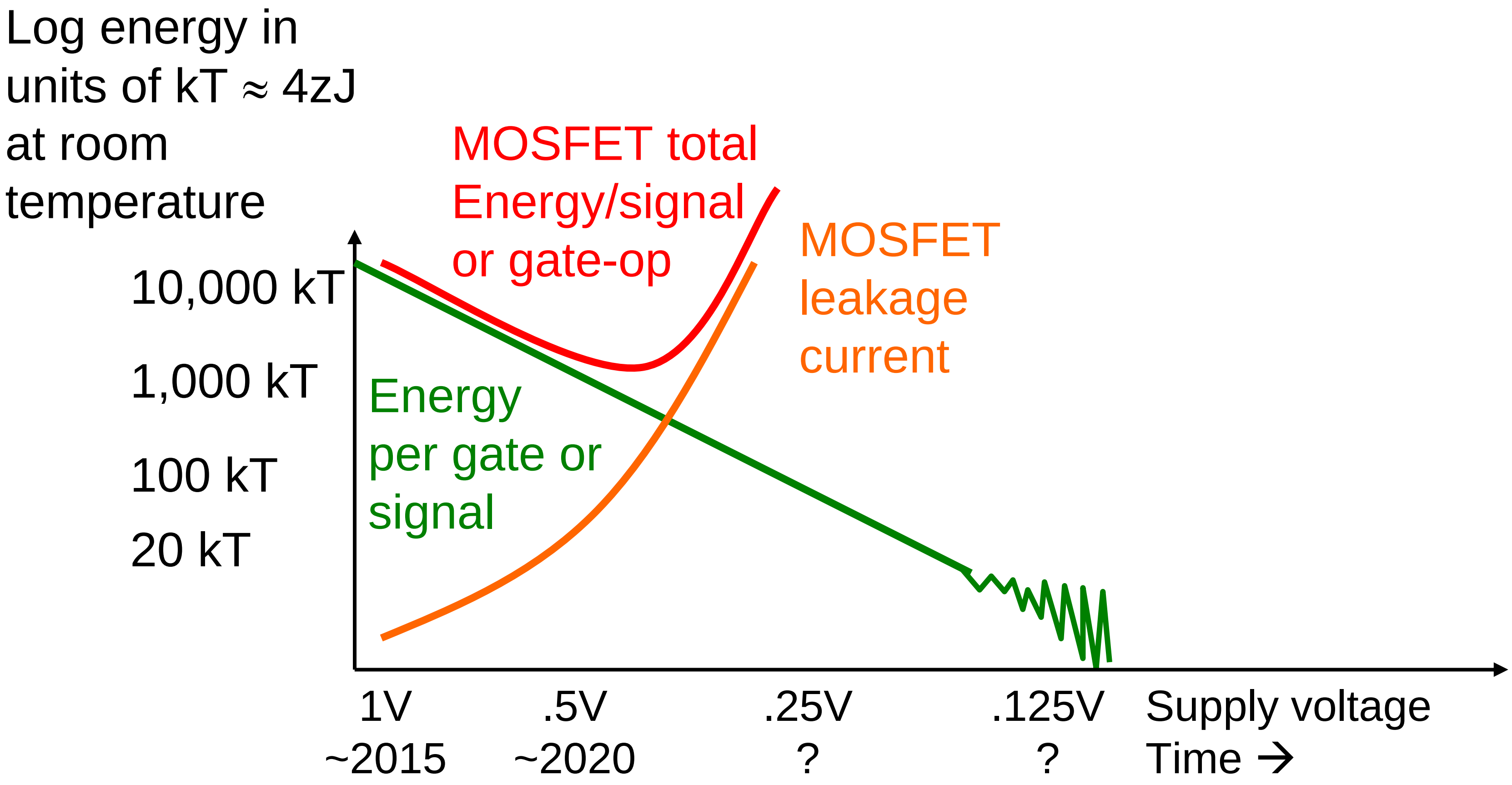
Erik P. DeBenedictis, Jeanine Cook, Tzevetan Metodi, Mark Hoemmen,
Matt Marinella, Rich Schiek, Center for Computing Research, Sandia
Hans Zima, Jet Propulsion Laboratory, Caltech

AFRL Presentation, July 2, 2015

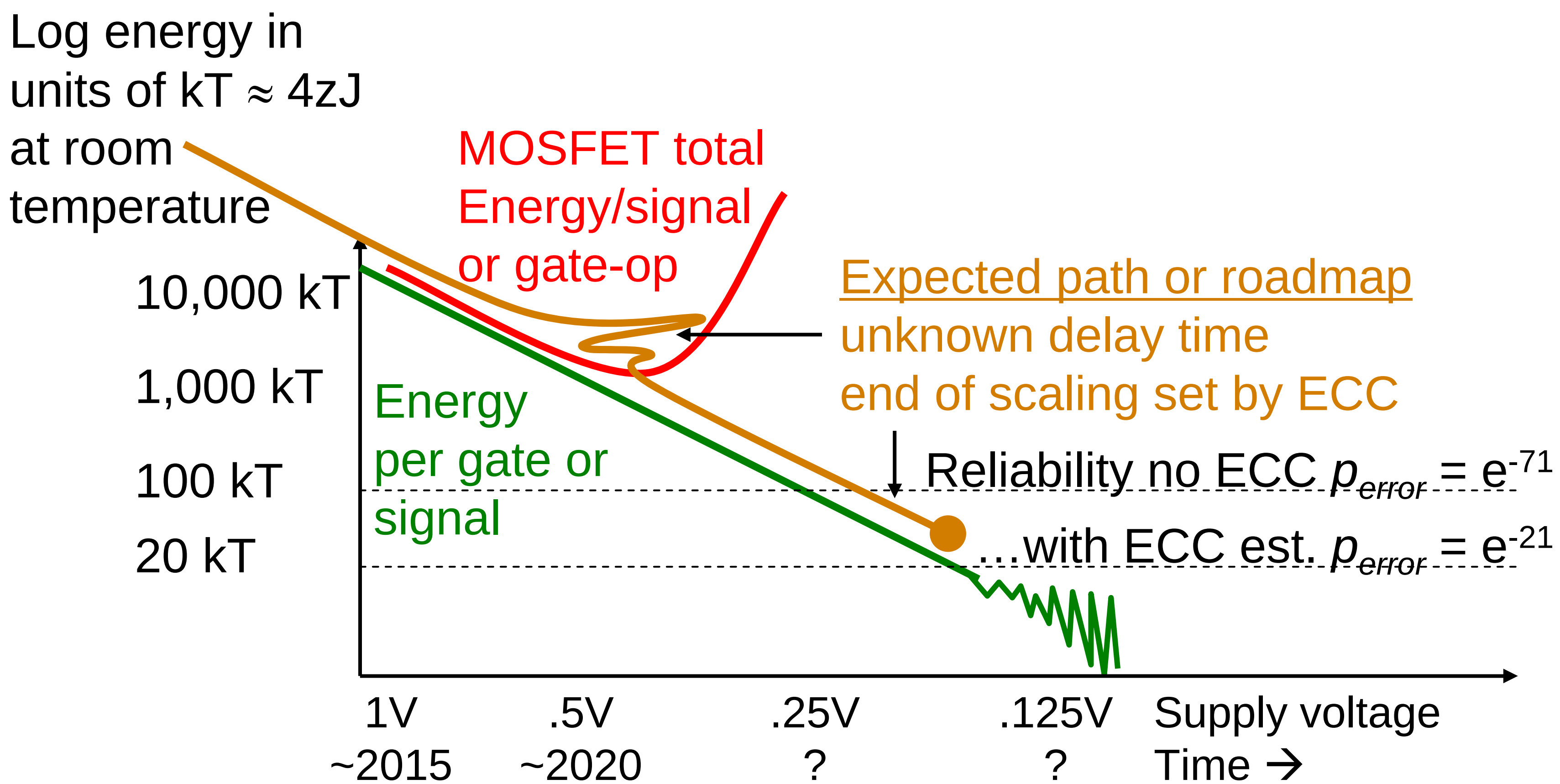
Approved for Unclassified Unlimited Release
SAND2015-5312 PE

Consider supply voltage's impact on scaling

- Scaling stuck in local minimum due to leakage current
- "Millivolt switch" could restore scaling to reliability limit



Roadmap for von Neumann architecture



What to do?

- Evolve architecture only
  - Baseline plan
- Adiabatic circuits
  - Recycle signal energy
- Scale but correct errors
  - Need a new architecture
- Scale but tolerate errors
  - Approximate computing
- Neural networks
  - Very different
- Quantum computing

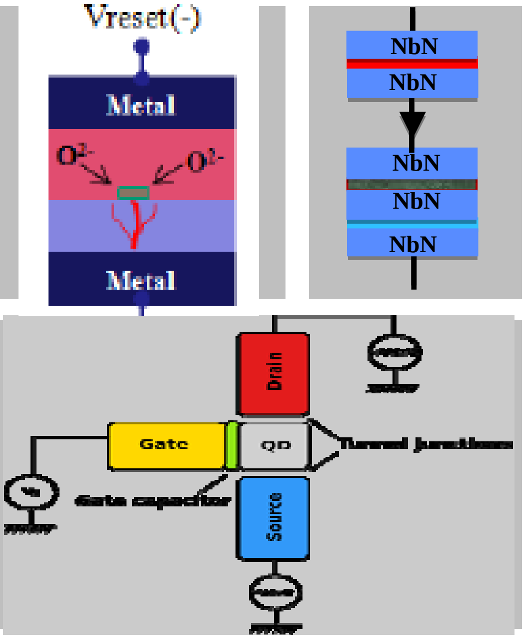
Sandia activities/talk agenda

- Space-specific issues
  - Space computing approach
  - Sandia Beyond Moore Computing Research Challenge
- Sandia project: Processor-In-Memory-and-Storage (PIMS)
- Sandia project: “Creepy” architecture (a code name)
- Sandia’s Rebooting Computing option: PIMS + Creepy
- Conclusions

Space-specific issues

- It is anticipated that space computers will become more processor and memory intensive
  - Our PIMS architecture addresses this need
  - (See Beyond Moore Computing Research Challenge, next slides)
- Space computers must be rad hard
  - The ultimate energy-efficient mobile phone should have logic errors
    - (otherwise the manufacturer should reduce energy some more)
  - If industry fixes logic errors for mobile phones, the solution should reduce radiation-induced errors for space as well
  - Our "Creepy" architecture addresses logic errors – for mobile phones or otherwise

LT mtg 4/28/2015



Beyond Moore Computing RC
Leadership Team Meeting

[Vacant] – RC Director

John Aidun (1425) – RC Deputy

jbaidun@sandia.gov

Sandia National Laboratories

Exceptional service in the national interest

U.S. DEPARTMENT OF ENERGY

NNSA

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.
SAND NO. 2014-XXXXP

5-10yr Focus (exemplar problem):

Design & prototype a special-purpose processor for smart data collection from an advanced sensor

Problem - Multiple Mission Areas are faced with
a deluge of sensor data

Solution –
A high performing computer system for an autonomous vehicle or embedded system that is capable of handling massively increased sensor data flows.

Outline

- Evolve architecture only
  - Baseline plan
- Adiabatic circuits
  - Recycle signal energy
- Scale but correct errors
  - Need a new architecture
- Scale but tolerate errors
  - Approximate computing
- Neural networks
  - Very different
- Quantum computing

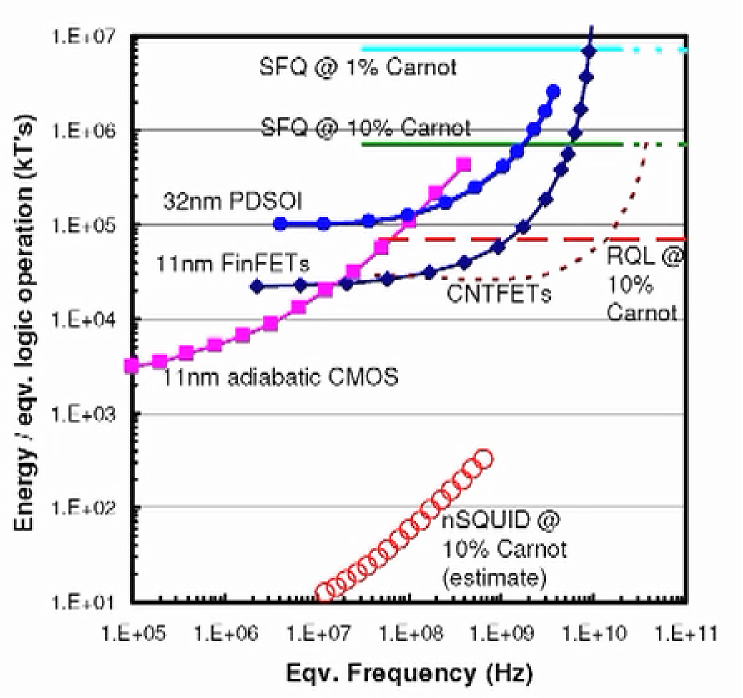
Sandia activities/talk agenda

- Space-specific issues
  - Space computing approach
  - Sandia Beyond Moore Computing Research Challenge
- Sandia project: Processor-In-Memory-and-Storage (PIMS)
- Sandia project: "Creepy" architecture (a code name)
- Sandia's Rebooting Computing option: PIMS + Creepy
- Conclusions

Energy efficiency can depend on clock rate

- David Frank (IBM) studied energy efficiency variance by clock rate
- Can make a scaling rule out of *f* vs energy efficiency dependence?

- Adiabatic circuits have behavior close to
  - Energy/op $\propto f$ (clock rate)
  - Power $\propto f^2$



From David Frank's presentation at RCS 2; viewgraph 23. "Yes, I'm ok with the viewgraphs being public, so it's ok for you to use the figure. Dave" (10/31/14)

A plot will reveal what we will call "optimal adiabatic scaling"

- Impact of manufacturing cost
  - Computer costs should include both purchase cost and energy cost.
  - However, let's adapt this idea to a situation where manufacturing cost drops with time, as in Moore's Law
- Let's plot economic quality of a gate or chip:

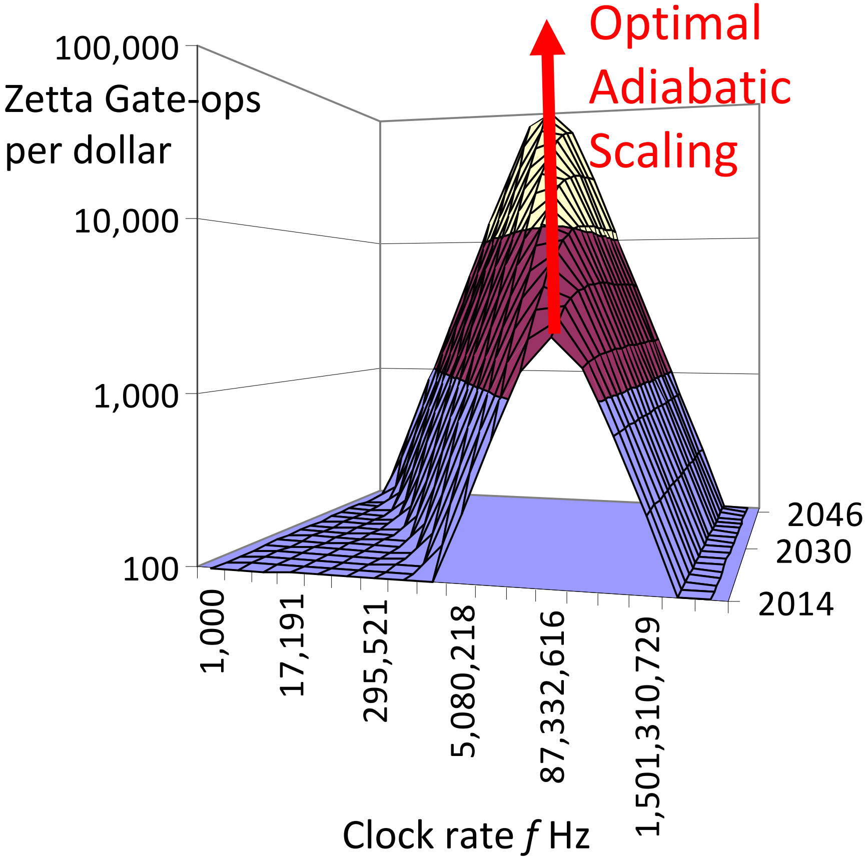
$$Q_{chip} = \frac{Ops_{lifetime}(f)}{\$_{purchase} + \$_{energy}(f^{2})}$$

Where $\$_{purchase} = A\ 2^{-t_{year}/3}$

$Ops_{lifetime} = Bf$, *and*

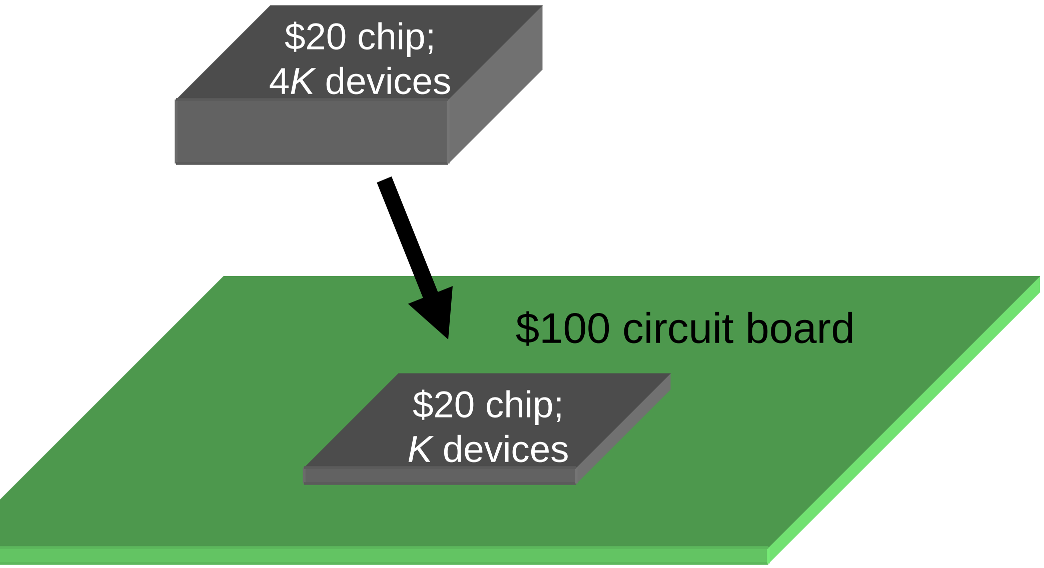
$\$_{energy} = Cf^{2}$ (A, B, and C constants)

- Assume manufacturing costs drops by ½ every three years
- Top of the ridge rises with time



How to derive a scaling rule

- Chip vendor says: "How would you like a chip with 4× as many devices for the same price?"

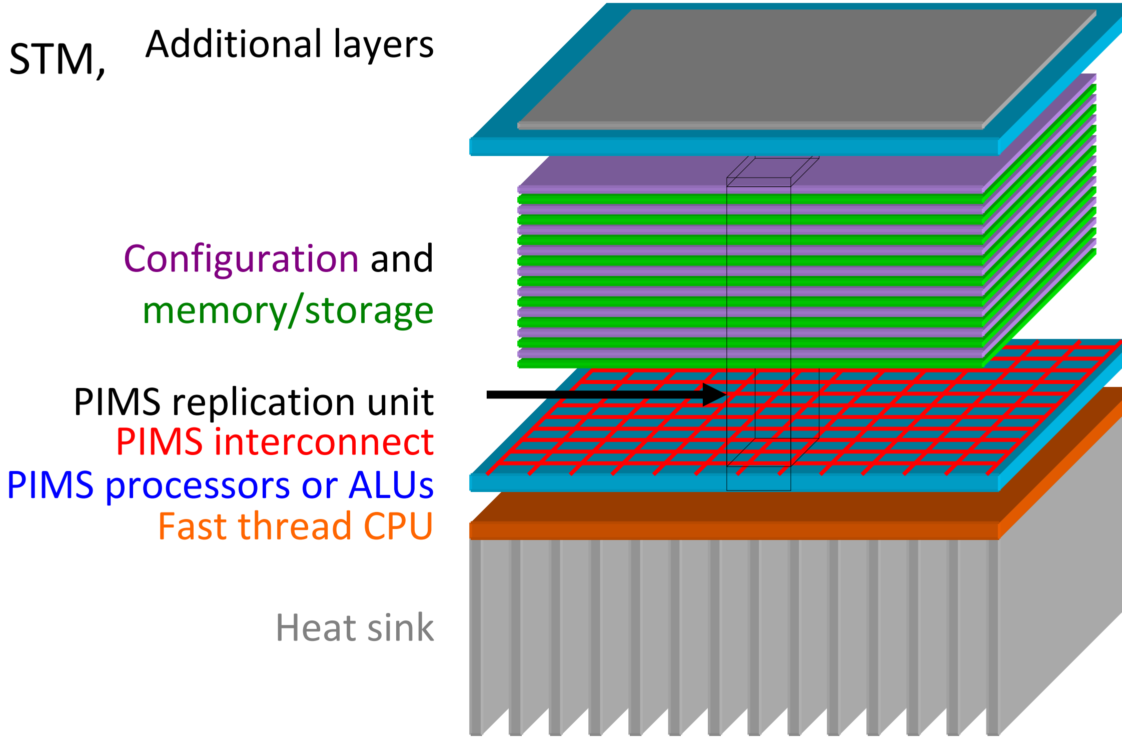


- Optimal adiabatic scaling says:
  - Cut clock rate to 1/√4× (halve)
  - Power per device drops to 1/4×
  - Power per chip stays same
  - Throughput doubles: 4× as many devices runn at 1/√4× the speed, for a net throughput increase of √4×

Processor-In-Memory-and-Storage (PIMS) Physical implementation vision

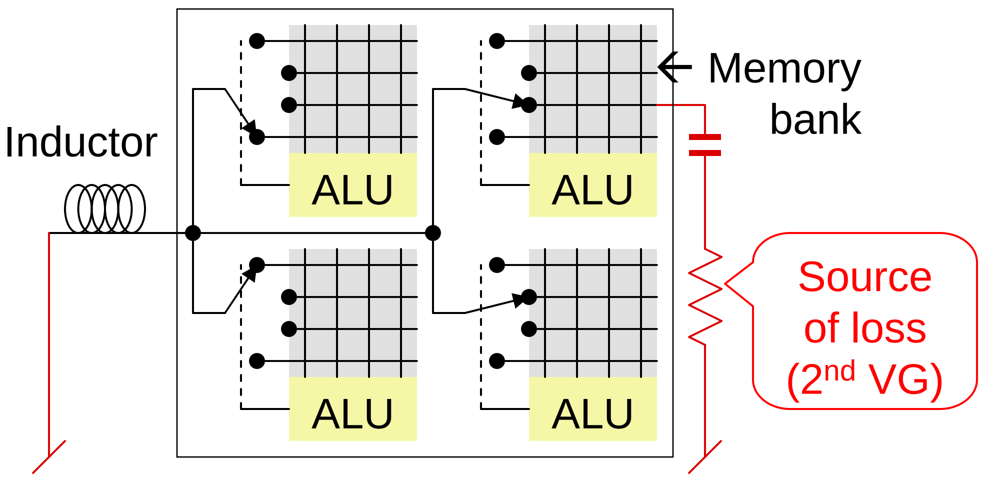
From a different project

- Storage/Memory
  - Flash, ReRAM (memristor), STM, DRAM
- Base layer
  - PIMS logic
- Fast-thread CPU
  - Some algorithms will need a conventional processor



Design for energy management

- Make principal energy pathway into a resonant circuit
  - Recycle the energy that the competitor's system turns into heat
- Size expectations for 128 Gb
  - 1024×1024 bits/memory bank
  - 128×128 banks/chip
- Chip



Tile programming

x

1	2	3	4

A

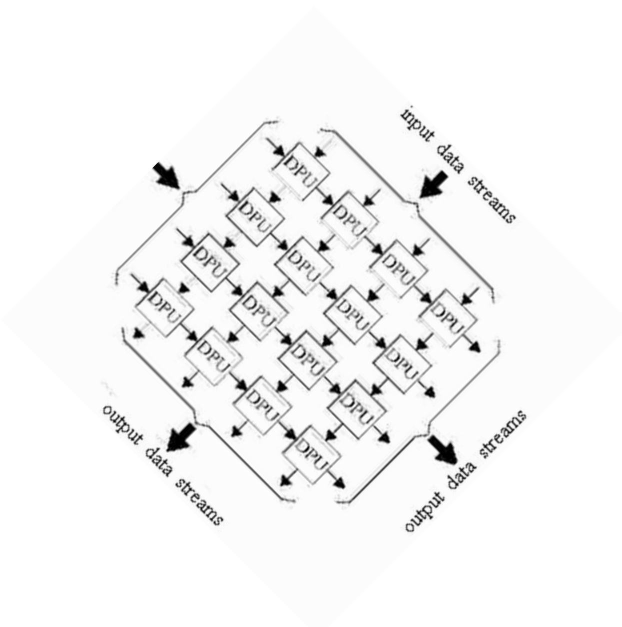
1	0	0	2
0	0	3	0
0	4	0	5
6	0	0	0

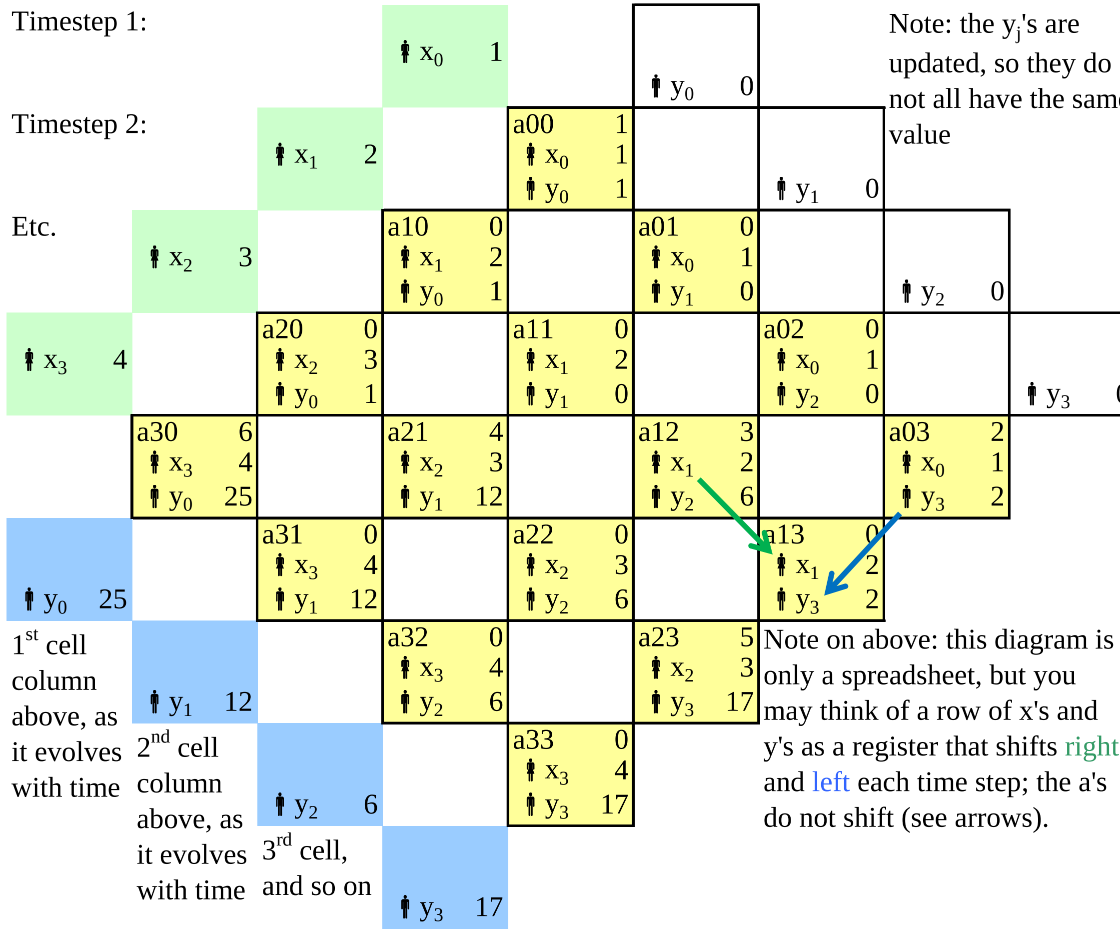
=

y

25	12	6	17

Vector-matrix multiply on left implemented by dataflow-like spreadsheet below.





Note: the y_j's are updated, so they do not all have the same value

Note on above: this diagram is only a spreadsheet, but you may think of a row of x's and y's as a register that shifts right and left each time step; the a's do not shift (see arrows).

Tile programming

x

1	2	3	4

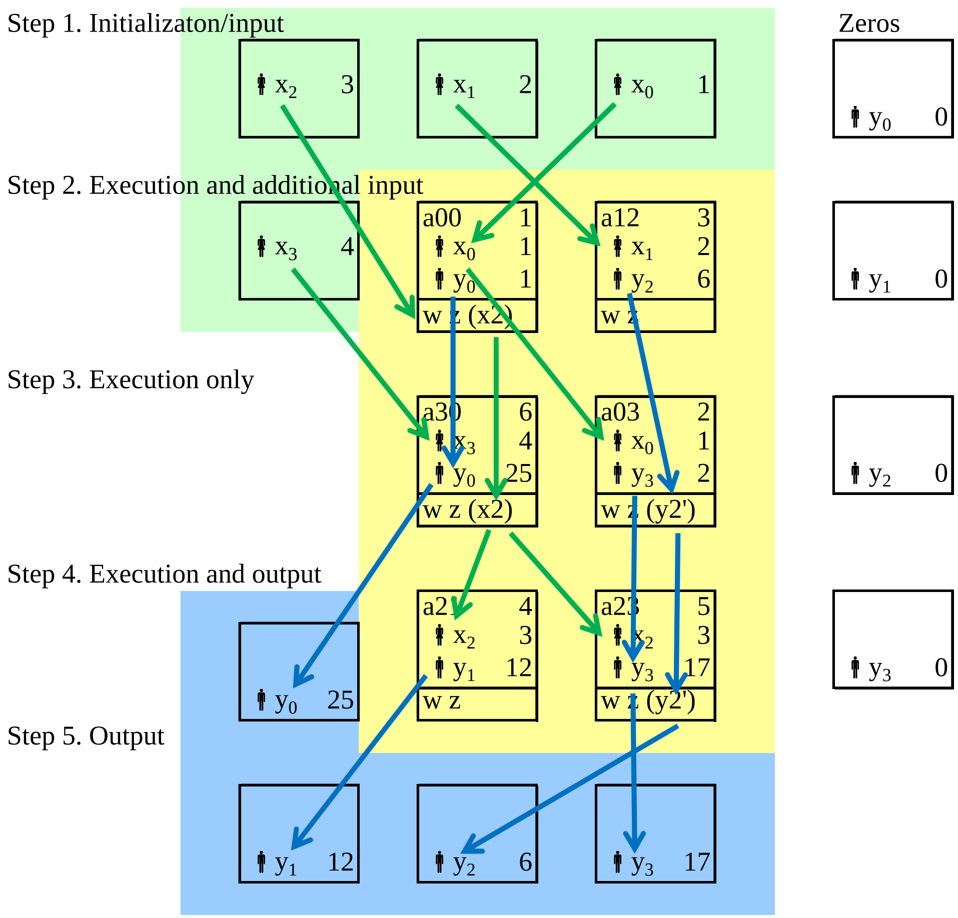
A

1			2
		3	
	4		5
6			

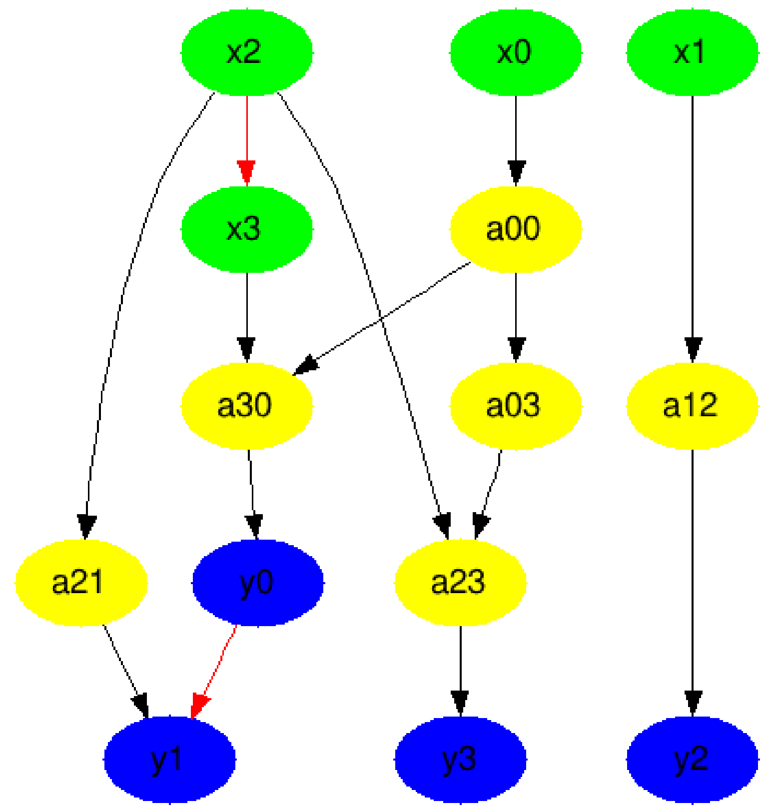
= y

25	12	6	17

Arrows indicate data flow; wth no data flow faster than nearest neighbor per step. Sometimes dance steps for ladies and gents.



GraphViz:



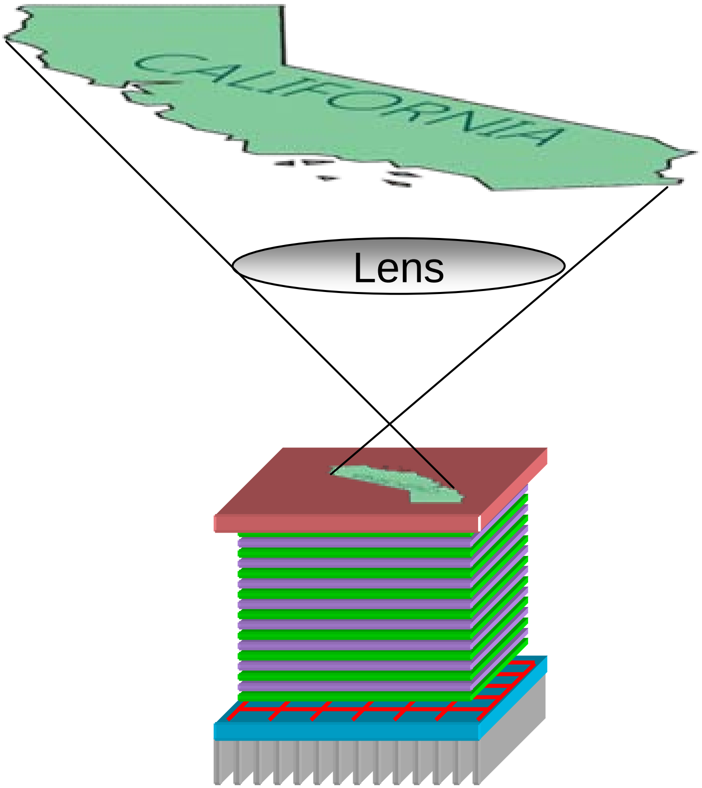
PIMS applications

Applications analyzed

- Sparse Matrix operations, used in
  - deep learning
  - supercomputer simulations
  - graph analytics
- Sorting
- Parsing
- Database storage and access
- LINPACK

Space computing vision

- Sensor, storage, and analysis unit
- Cubesat?



Outline

- Evolve architecture only
  - Baseline plan
- Adiabatic circuits
  - Recycle signal energy
- Scale but correct errors
  - Need a new architecture
- Scale but tolerate errors
  - Approximate computing
- Neural networks
  - Very different
- Quantum computing

Sandia activities/talk agenda

- Space-specific issues
  - Space computing approach
  - Sandia Beyond Moore Computing Research Challenge
- Sandia project: Processor-In-Memory-and-Storage (PIMS)
- Sandia project: "Creepy" architecture (a code name)
- Sandia's Rebooting Computing option: PIMS + Creepy
- Conclusions

Need for error handling in semiconductor scaling

- Logic scaling has been connected quantitatively to redundancy and error correction
  - See →
  - See also Mike Frank
- We have queried the authors, but have not found
  - Examples of the needed error correction technique
  - A Turing-complete architecture

- Theis and Solomon*

1) *Conventional Logic:* Reduce the stored energy $(1/2)CV^2$. For conventional FETs, as V approaches a small multiple of kT/e, we must accept reduction in switching speed. New device concepts, discussed below, may allow more significant reduction in V and facilitate the reduction of stored energy towards kT. As thermal voltage fluctuations become significant, we must incorporate redundancy and error correction in the logic to keep the error rate in bounds. Refrigeration can reduce T, but in a power-constrained environ-

ment with Meindl and Davis. Since Johnson–Nyquist voltage noise is Gaussian with a standard deviation of V_n, a stored logic voltage of m standard deviations, or a stored energy of m^2kT, would be needed to achieve a reliability of $(1/2)\mathrm{Erfc}[m/\sqrt{2}]$. (Eight standard deviations give an error probability of $\sim 10^{-15}$.)

Note that,

$$p_{error} = ½\mathrm{Erfc}[m/\sqrt{2}] \approx \exp(-E_{signal} / kT)$$

*Theis, Thomas N., and Paul M. Solomon. "In Quest of the" Next Switch": Prospects for Greatly Reduced Power Dissipation in a Successor to the Silicon Field-Effect Transistor." *Proceedings of the IEEE* 98.12 (2010): 2005-2014.

Primer on Redundant Residue Number System (backup)

Residue Number System (RNS)

- Given a set of relatively prime moduli m_1, m_2, m_3, m_4, e. g.
  - 199, 233, 194, 239
- Any number < $m_1 \times m_2 \times m_3 \times m_4$ can be represented by the four remainders (residues) upon division by m_j
- Addition and multiplication become vector-wise modular add and multiply
- Comparison, shifting, conversion are *residue interacting functions*

- Redundant RNS (RRNS)
- Add extra moduli, m_5, m_6, e. g.
  - 251, 509
- Up to two bad residues can be detected
- Up to one bad residue can be corrected

- NOTE: Covers the math, not just the storage!

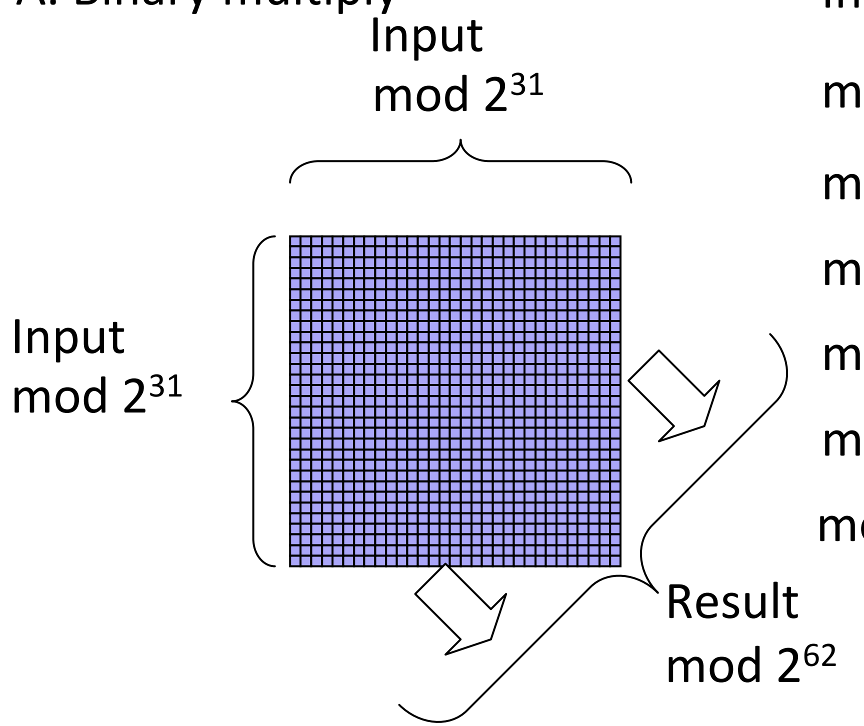
Trivia: This is the Ph. D. thesis of Dick Watson, LLNL, retired

This is the RNS used in Watson, Richard W., and Charles W. Hastings. "Self-checked computation using residue arithmetic." *Proceedings of the IEEE* 54.12 (1966): 1920-1931.

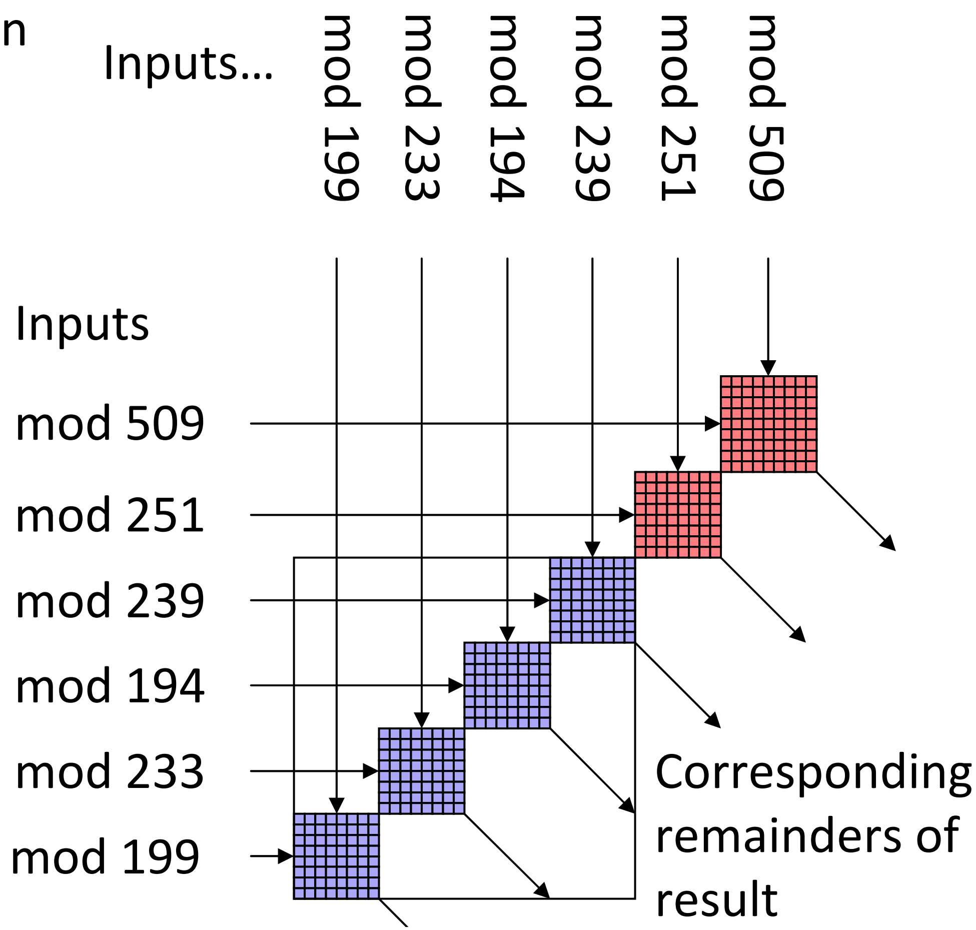
Example where we gain energy efficiency

- Added energy for redundancy in part B is about 50%, so energy efficiency improves given baseline on earlier VG.

A. Binary multiply

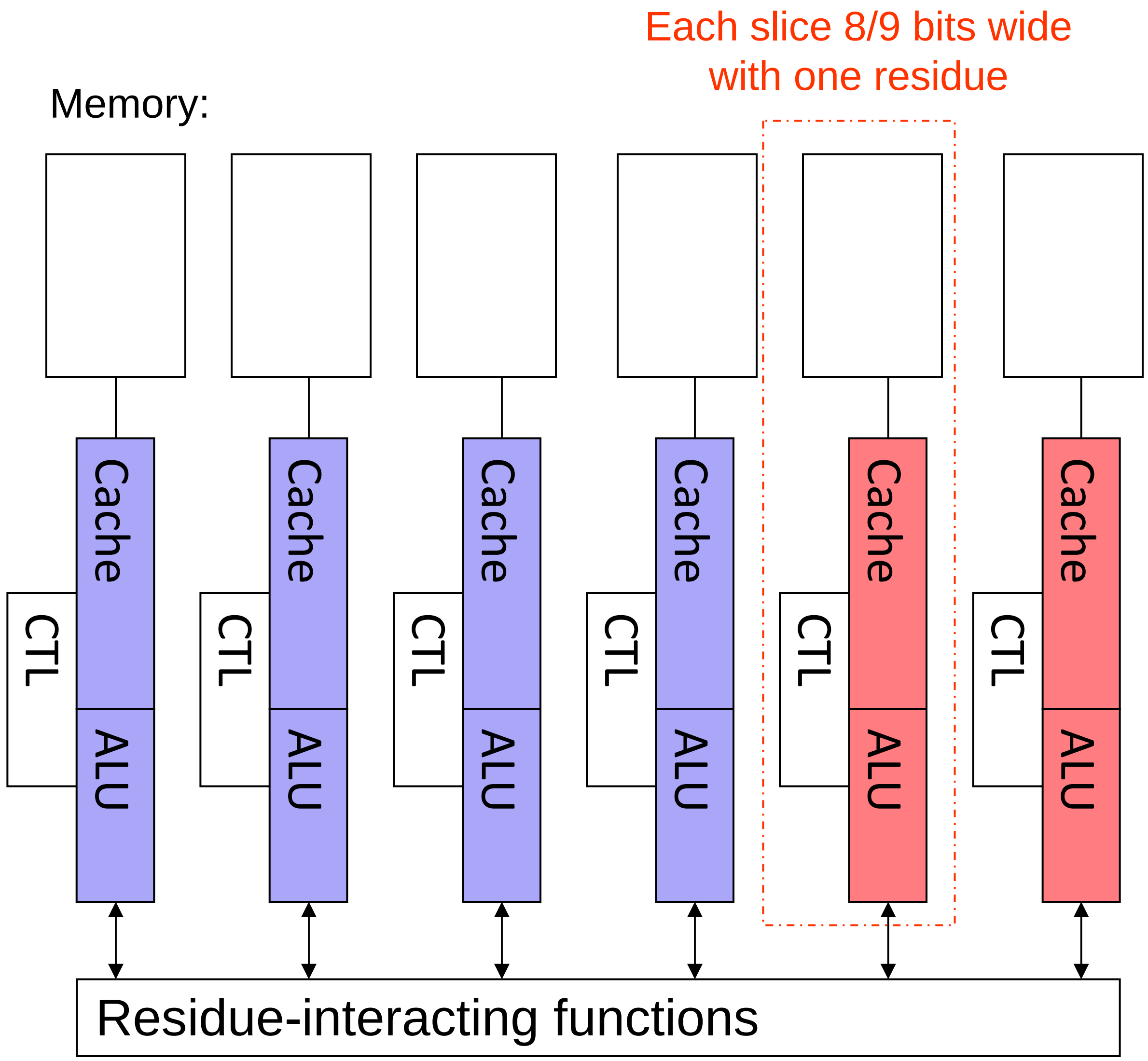


B. Redundant Residue Number System



This is the RNS used in Watson, Richard W., and Charles W. Hastings. "Self-checked computation using residue arithmetic." *Proceedings of the IEEE* 54.12 (1966): 1920-1931.

Creepy architecture (temporary name)



Purple slices are the the non-redundant residues; red slices are the checks

Overhead: 50% on ALU and cache; 6× on control

Programming with assertion language (Hans Zima)

RRNS structure definition with assertions (*ED*=error detect; *EC*=error correct):

```
struct RRN { int r199:8, r233:8, r194:8, r239:8, r251:8, r509:9; }
    assert(ED(...)) error(EC(x,...));
```

Multiply:

```
struct RRN mul (RRN a, RRN b) { v, p_u(...), p_d(...), E(...) } {
    return RRN (a.r199*b.r199%199,
        a.r233*b.r233%233,
        a.r194*b.r194%194,
        a.r239*b.r239%239,
        a.r251*b.r251%251,
        a.r509*b.r509%509);
}
```

p_u(...), p_d(...), E(...) are pragmas conveying information on error probabilities and energy consumption to the system

Hans P. Zima, Erik DeBenedictis, Jacqueline Chame, Pedro C. Diniz, Robert F. Lucas, *The FailSafe Assertion Language Version 8.0*, Technical Report, Information Sciences Institute, University of Southern California, May 2015

Backup: At stake? Maybe one generation

- Scaling will not stop abruptly, but it will be stopped by an exponential rise in error rate with declining energy
- But how much energy efficiency improvement is possible if we can tolerate errors? Spreadsheet →
  - No ECC 71 kT
  - ECC scenarios 24 kT – 28 kT
  - 2:1 after overhead, +/-
- A trillion dollar question

Exascale reliability requirement

100000	Gates-ops per floating point op where an error would cause a wrong answer
1.00E+18	ops/second (definition of Exascale)
60	seconds per minue
60	minutes per hour
24	hours per day
365	days per year
3	years for a computer's lifetime (before it becomes obsolete)
9.46E+30	number of gate operations per lifetime where an error would cause a wrong answer
71.33211	If we have Esignal equal this many kT's, error rate will be inverse of previous line

Say an operation is this many gate-ops	1000	20000	1.00E+05	1.00E+06	1.00E+18
Steps in lifetime (serial and parallel)	9.46E+27	4.73E+26	9.46E+25	9.46E+24	9.46E+12

RRNS using system in Watson and Hastings					
Gate ops per residue (four non-redundant residue	250	5000	25000	250000	2.5E+17
perror target for exaflops over lifetime	1	1	1	1	1
perror per step	1.06E-28	2.11E-27	1.06E-26	1.06E-25	1.06E-13
perror per residue; 3 errors in a step must go und	7.02E-09	1.91E-08	3.26E-08	7.02E-08	7.02E-04
Es = this many kTs will meet reliability in line abov	24.30	26.30	27.37	28.90	47.33
Energy savings	2.94	2.71	2.61	2.47	1.51
However, we need 6 total residues, not 4	1.96	1.81	1.74	1.65	1.00

Additional beneficial factors
Fixes Cosmic Ray hits
Fixes weak and aging components
Could support overclocking; i. e. catches an "excessive overclocking" error

Outline

- Evolve architecture only
  - Baseline plan
- Adiabatic circuits
  - Recycle signal energy
- Scale but correct errors
  - Need a new architecture
- Scale but tolerate errors
  - Approximate computing
- Neural networks
  - Very different
- Quantum computing

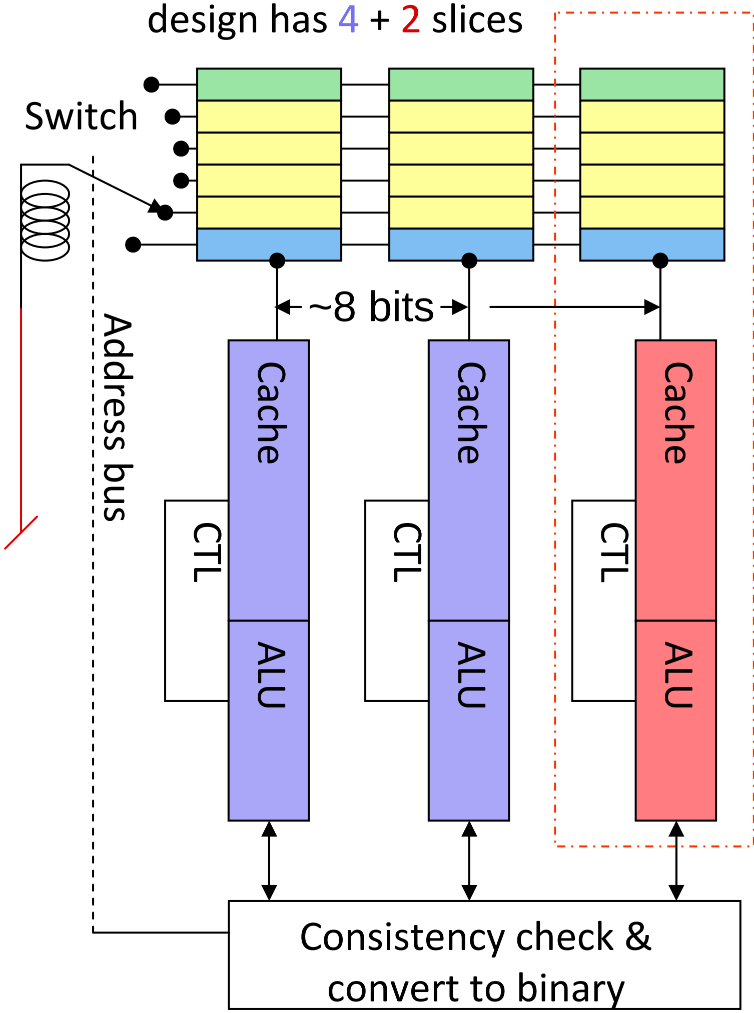
Sandia activities/talk agenda

- Space-specific issues
  - Space computing approach
  - Sandia Beyond Moore Computing Research Challenge
- Sandia project: Processor-In-Memory-and-Storage (PIMS)
- Sandia project: "Creepy" architecture (a code name)
- Sandia's Rebooting Computing option: PIMS + Creepy
- Conclusions

Power-efficient architecture overview

PIMS (memory) + Creepy (processor) architecture:

Each slice 8 or 9 bits wide; baseline design has 4 + 2 slices



Features:

Adiabatic memory = energy efficiency by recycling

Extreme energy efficiency in computation by RRNS* error correction (main/check)

Parallelism by pre-sorting

SpMV example:

x

1	2	3	4

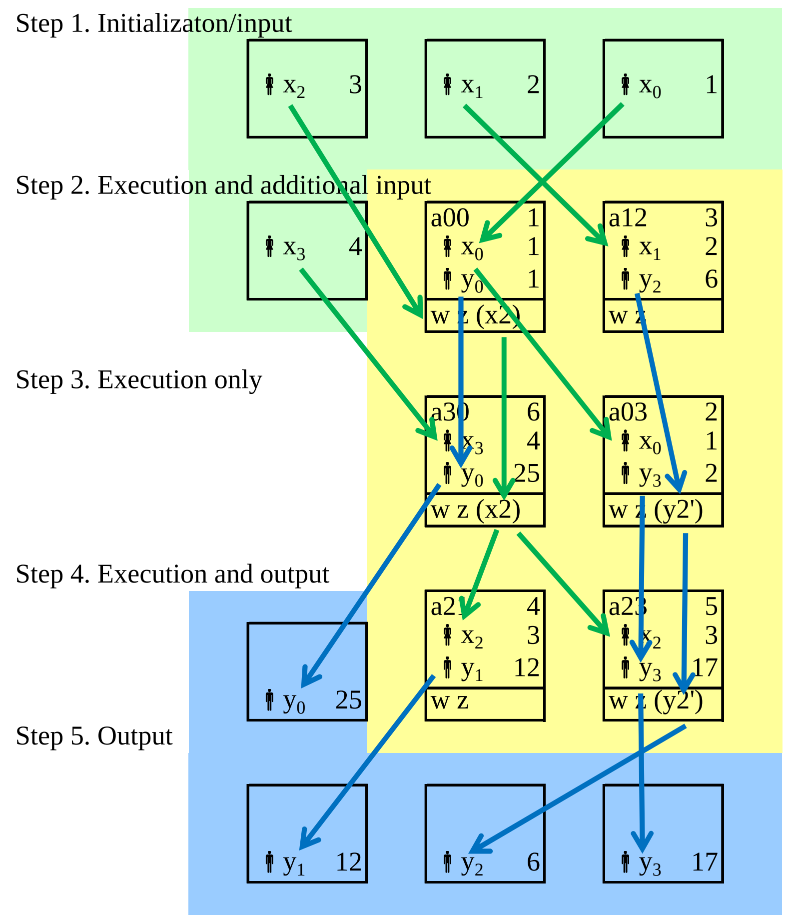
A

1			2
		3	
	4		5
6			

=

y

25	12	6	17



Outline

- Evolve architecture only
  - Baseline plan
- Adiabatic circuits
  - Recycle signal energy
- Scale but correct errors
  - Need a new architecture
- Scale but tolerate errors
  - Approximate computing
- Neural networks
  - Very different
- Quantum computing

Sandia activities/talk agenda

- Space-specific issues
  - Space computing approach
  - Sandia Beyond Moore Computing Research Challenge
- Sandia project: Processor-In-Memory-and-Storage (PIMS)
- Sandia project: "Creepy" architecture (a code name)
- Sandia's Rebooting Computing option: PIMS + Creepy
- Conclusions

Status and future work

Status

- OAS, PIMS, and Creepy
  - Tech report, two publications, patent in progress, half-dozen presentations
  - Software simulations
  - Circuit simulations
  - Contract with Georgia Tech
- Public initiatives
  - These topics are used as illustrations in the IEEE "Rebooting Computing" new initiative
  - Same with ITRS

Future work

- The overall project has immediately implementable technology and a grandiose vision; this VG deck is mostly the grandiose vision
- Immediately implementable technology
  - Software for a DRAM- and/or Flash-based conventional Processor-In-Memory (PIM)
  - PIM projects exist (DARPA-, DOE-, industry-funded)

Conclusions

- Computer performance growth slowing, so lots of people are looking for new approaches to computing, including us
- We discussed Sandia projects:
  - Optimal Adiabatic Scaling (OAS)
  - Processor-In-Memory-and-Storage (PIMS)
  - Low energy architecture (Creepy)
  - Beyond Moore Computing Research Challenge
- Applicable to space too
  - Right applications and SWaP
  - Might be rad hard as side effect of quest for low power

Abstract (AFRL)

Beyond Moore's Law and Implications for Computing in Space
Erik DeBenedictis and Hans Zima
July 2, 2015, 10 AM, AFRL Kirtland Building 914

The talk will first discuss transistor scaling limits and the implications to what is colloquially called Moore's Law.

Building on the scaling discussion, the talk will describe a research-level computing approach with two important properties: (1) it could extend scaling for terrestrial computers by an estimated one generation and (2) the resulting computers would be radiation hard, thus eliminating the need for additional radiation hardening if used in space.

The approach can be summarized as follows: The audience will understand that industry is not currently inclined to produce rad-hard computers, leading to high costs for the government. The novel approach is to tie error detection and correction to power efficiency, based on the fact that continued power efficiency scaling eventually leads to an exponential rise in logic errors. If the terrestrial computer industry is to achieve the highest power efficiency for consumer products, industry will have to employ error detection and correction against the power-related errors. However, the needed error handling works irrespective of the error's source. Thus, the technology for power efficiency on Earth will also correct Cosmic ray-induced errors in space.

The example processor architecture is called "Creepy" and uses a Redundant Residue Number System (RRNS) as a suitable error correction method. Creepy is tied to a memory architecture called Processor-In-Memory-and-Storage (PIMS), which is essential to creating a general-purpose but low-power architecture. The software architecture involves an assertion language created by Hans Zima. The assertion language comprises extensions to languages like C or FORTRAN that allow assertions for correctness (the basis of error detection) and responses to failed assertions (the basis of error correction).