**Post Exascale Landscape**

*MIND THE GAP!*

**Moore’s Law**

*Lithography Scaling*

2x increased density

2x lower power

Every 2 years!

**Post Moore Scaling**

New materials and devices introduced to enable continued scaling of digital electronics performance and efficiency.

---

**Now – 2025**

Moore’s Law continues through ~5nm -- beyond which diminishing returns are expected.

---

**End of Moore’s Law 2025-2030?**

---

**2025+**

---

**2016**

---

**2016-2025**

---
50 years of exponential technology scaling is an amazing thing *that is often taken for granted*.
Is it the end?

“I predict Moore’s Law will never end. That way I will only be wrong once!”

Alan Kay: Communications of the ACM 1989
Moore's Law is coming to an end. He also said they would have to "move away" from silicon!

"…we think we can continue Moore's Law for at least another 10 years…" 

"…eventually Moore's Law will slow down or come to an end…"

Bohr predicted that Moore's Law will not come to an abrupt halt, but will morph and evolve … scaling density by the 3D stacking of components rather than continuing to reduce transistor size.

http://www.v3.co.uk/v3-uk/news/2403113/intel-predicts-moores-law-to-last-another-10-years

Atomic scale limit case for 2D Lithography Scaling

End of Moore's Law
Technology Scaling Trends

Exascale in 2021… and then what?
Innovation is the Answer!

Moore’s Law is an economic theory. There are ways to continue scaling of digital technology after the end of classical lithographic scaling

(e.g. end of Dennard Scaling in ~2004
No more exponential clock frequency scaling
Move to exponentially increasing parallelism)
Beyond Moore Computing Taxonomy

Symbolic Computation, Arithmetic, Logic

Digital

Neuro-Inspired

Cognitive Computing, Pattern Recognition

Quantum

Combinatorial/NP, Annealing/Optimization, Simulated Atoms
Future of Computing

New Materials and Devices
20+ years (10 year lead time)

More Efficient Architectures and Packaging
The next 10 years after exascale
What about a new transistor?

How is that going?
Architectural schemes to extend digital computing aim to better manage energy, decrease power consumption, lower overall chip cost, and improve error detection and response.

Energy management

Current energy-management technologies are ubiquitous and typically coarse grained. Dynamic voltage and frequency scaling (DVFS) and thermal throttling lower both clock frequencies and voltages when computing demands do not require peak performance. Coarse-grained DVFS can save significant power in current consumer electronics devices, which are mostly idle. However, it only marginally benefits devices that operate near 100 percent utilization. Finer-grained power management might provide additional potential to recover energy, enabling faster transitions between power states by having the software direct state changes.

Circuit design

Studies have demonstrated approaches that enable wires to operate at a lower voltage for long-haul connections and then reamplify efficiently at the end points, although with some loss from reamplification. A recent NVIDIA paper estimated an opportunity for two to three times improvement using such advanced circuit design techniques with current technologies.

A more aggressive path to performance enhancement is clockless (or domino logic) design. Clock distribution consumes a large fraction of system power, and constrains a circuit to the operation speed of its slowest component. Practical and effective clockless designs have proven elusive, but recent examples show that this approach could be a viable way to lower dynamic power consumption for both neuromorphic and digital applications.

System-on-chip (SoC)

The core precept of SoC technology is that chip cost is dominated by component design and verification costs. Therefore, tailoring chips to include only the circuit components of value to the application is more economically efficient than designing one chip that serves a broad application range—the current commodity design practice. This tailoring strategy is common practice for cell-phone chips, such as:

<table>
<thead>
<tr>
<th>TABLE 1. Summary of technology options for extending digital electronics.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Improvement Class</strong></td>
</tr>
<tr>
<td>Architecture and software advances</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>3D integration and packaging</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Resistance reduction</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Millivolt switches (a better transistor)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Beyond transistors (new logic paradigms)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Accelerated Development
For Post CMOS Digital Computing
We might already be too late

Historically it is 10 years from lab to Fab…

But let’s talk about it anyways.
Borkar-Shalf Criteria for New Device Technology

1. Gain
2. Signal to Noise
3. Scalability
4. Manufacturability
New Breakthroughs in Transistor Technology Require Fundamentally New Principles of Operation

TFET (Tunneling FET): A More sensitive switch
Modulated by quantum tunneling instead of thermionic emission

Log Scale

Drain Current ($I_D$)

Voltage Range
Off vs On

Gate Voltage ($V_G$)

60 mV/dec

MOSFET
Alternatives to Conventional MOS Switches
(all require lower clock rate, and much more parallelism)

Need another 100x parallelism just to maintain status quo!

ENERGY [J]
1.0E-17
1.0E-16
1.0E-15

Performance [GHz]
0.01
0.1
1
10

MOSFET

TFET
Transition probability=0.01
Cap. per inverter=0.57fF

Today’s CMOS Technology
Tunneling FET advantage only at low clock rates
Comparing CMOS Technology Alternatives

• Favorable Spin devices based on logic
• With introduced SHE for lower energy
• With AFM for high speed

Nikonov/Young (Intel) Exploratory Integrated Circuits / Components Research

Better

Faster clock rate Slower
Comparing CMOS Technology Alternatives

(32 bit adder “benchmark” revolutionized field by focusing effort on device feature improvements that directly benefit the adder)

Nikonov/Young (Intel)

Exploratory Integrated Circuits

Components Research

10x-100x Slower (more parallelism)
Multiscale CoDesign Framework

Programming paradigms

System architecture modeling

Component hetero-integration

Device/Circuit integration

Device physics

Fundamental material science
The impact of advances in materials and structures must be understood at the systems level.

A major theme at this workshop is the need to develop a co-design approach.
Accelerated, bidirectional feedback paths are needed from scale to scale.

As the focus of innovation in architecture shifts from the general-purpose CPU to domain-specific and heterogeneous processors, we will need to achieve major breakthroughs in design time and cost. “(Hennessy-Patterson)
High-throughput screening of Novel Materials
Using first-principles calculations and Big Data Analytics on HPC

Kristin Persson

Today’s CMOS
Voltage limit

Interesting materials!!!

Accelerate discovery of new materials by factor of 1000x
Ab-Initio Full Electronic Device Simulations

LS3DF \(\rightarrow\) \(O(N)\) DFT Methods

Lin Wang Wang, Andrew Canning

- Combined several techniques for a holistic, ab-initio, atomistic (beyond TCAT) device simulative approach.
- LS3DF Device-size selfconsistent ab initio calculations to get atomistic potential profile, band alignment, based boundary conditioned Poisson solver.
- Based on the potential profile, and scattering state calculations to simulate the device transport, and leakage current etc.
- Using electron-phonon coupling to calculate the heat generation and dissipation at atomic scale.

A shallow defect state in Si.

The electron (left) and hole (right) localizations in a bulk CH\(_3\)NH\(_3\)PbI\(_3\) material. The small dots are atoms.
Gaps: Connections and Scaling

**Metrics:** Fan-out, area, switching speed, power

Compact Device Model to Verilog-A for analog circuit sim

**Metrics:** I-V curves, current drive, switching energy, transients

---

- **Systems**
- **Architectural Simulation**
- **Circuits**
- **Analog Simulation**
- **Device Physics**
- **Compact Models**

---

- **PARADISE**
- **OpenSoC System Architect**

---

**Metrics:** Fan-out, area, switching speed, power

Compact Device Model to Verilog-A for analog circuit sim

**Metrics:** I-V curves, current drive, switching energy, transients
Gaps: Connections and Scaling

Metrics: Application performance, system power

Models of digital logic components scaled to systems that run applications

Metrics: clock rate, area, power

Systems

Architectural Simulation

Circuits

Analog Simulation

Device Physics

Compact Models

Gaps: Connections and Scaling

PARADISE

OpenSoC System Architect
Rapidly Prototype/Synthesize SoCs (Synthesis & Simulation)

**Chisel**
- DSL for rapid prototyping of circuits, systems, and arch simulator components

**RISC-V**
- Open Source
- Extensible ISA/Cores
- Re-implement processor with different devices or extend w/accelerators

**OpenSOC**
- Open Source fabric
- To integrate accelerators and logic into SOC

Platform for experimentation with specialization to extend Moore’s Law

Back-end to synthesize HW with different devices or new logic families
In all likelihood, Weddington concedes, the resulting technology “will never be as good as what is commercially available.” But perhaps it could be made good enough “to bring the power and ability to design your own IC, or microprocessor, to smaller and smaller groups of people and drive down the enormous capital requirements of an entrenched, dinosaur industry.”

Similarly, Michael Cooney of Network World describes the state of open-source hardware today as roughly where open-source software was during the mid-1990s – waiting for commercial suppliers to provide higher levels of support. “What made open-source software acceptable for many businesses was the arrival of support for it, such as Red Hat,” he says, adding, “Something similar may take place with the hardware.”

- Rapid growth in the adoption and number of open source software projects
- More than 95% of web servers run Linux variants, approximately 85% of smartphones run Android variants
- Will open source hardware ignite the semiconductor industry? Is RISC-V the hardware industry’s Linux?
Industrializing the Discovery Process with CoDesign
(Accelerate Discovery of new Materials/Devices/Architectures by 1000x)

Automate Search for Better Materials

Rapid Prototype Hardware

OpenHPC System Architect

Ab-initio Full Device Simulation

More Efficient Computing Architecture

Better Devices

Better Materials

Interesting materials!!!
Skyrmions
Single-track Skyrmion-Based Spiking Neural Network

Z. He et al., 1705.02995v1 (2017)
Impact of End of Moore’s Law on Parallel Software
Impact on Programming of Parallel Systems

- Exponentially Increasing Parallelism (central challenge for Exascale)
  - Trend: End of exponential clock frequency scaling (end of Dennard scaling)
  - Consequence: Exponentially increasing parallelism

- End of Lithography as Primary Driver for Technology Improvements
  - Trend: Tapering of lithography Scaling
  - Consequence: Many forms of heterogeneous acceleration (not just GPGPUs)

- Data Movement Heterogeneity and Increasingly Hierarchical Machine
  - Trend: Moving data operands costs more than computation performed on them
  - Consequence: More heterogeneity in data movement performance and energy

- Performance Heterogeneity
  - Trend: Heterogeneous execution rates from contention and power management
  - Consequence: Extreme variability and heterogeneity in execution rates

- Diversity of Emerging Memory and Storage Technologies
  - Trend: Emerging memory technologies and stall in performance improvements
  - Consequence: Disruptive changes to our storage environment (**POSIX just won’t cut it anymore!!!**)
Towards Diverse Integrated Accelerators
Extreme Hardware Specialization is Happening Now

This trend is already well underway in broader electronics industry. Cell phones and even megadacenters (Google TPU, Microsoft FPGAs...) (and it will happen to HPC too... will we be ready?)

29 different heterogeneous accelerators in Apple A8 (2016)
Google Tensor Processing Unit (TPU)

- Deployed in datacenters since 2015
- 10-30x Faster than NVIDIA G80 or Intel Haswell for ML workloads *(64k arithmetic ops per cycle)*
- Could be faster with better memory subsystem.
- 8bit integer arithmetic (all that is needed for ML)

![TPU Image]

**Figure 4.** Systolic data flow of the Matrix Multiply Unit. Software has the illusion that each 256B input is read at once, and they instantly update one location of each of 256 accumulator RAMs.

<table>
<thead>
<tr>
<th>Model</th>
<th>mm²</th>
<th>nm</th>
<th>MHz</th>
<th>TDP</th>
<th>Measured Idle</th>
<th>Measured Busy</th>
<th>TOPS/s 8b</th>
<th>TOPS/s FP</th>
<th>GB/s</th>
<th>On-Chip Memory</th>
<th>Dies</th>
<th>DRAM Size</th>
<th>TDP</th>
<th>Measured Idle</th>
<th>Measured Busy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haswell E5-2699 v3</td>
<td>662</td>
<td>22</td>
<td>2300</td>
<td>145W</td>
<td>41W</td>
<td>145W</td>
<td>2.6</td>
<td>1.3</td>
<td>51</td>
<td>51 MiB</td>
<td>2</td>
<td>256 GiB</td>
<td>504W</td>
<td>159W</td>
<td>455W</td>
</tr>
<tr>
<td>NVIDIA K80 (2 dies/card)</td>
<td>561</td>
<td>28</td>
<td>560</td>
<td>150W</td>
<td>25W</td>
<td>98W</td>
<td>--</td>
<td>2.8</td>
<td>160</td>
<td>8 MiB</td>
<td>8</td>
<td>256 GiB (host) + 12 GiB x 8</td>
<td>1838W</td>
<td>357W</td>
<td>991W</td>
</tr>
<tr>
<td>TPU</td>
<td>NA*</td>
<td>28</td>
<td>700</td>
<td>75W</td>
<td>28W</td>
<td>40W</td>
<td>92</td>
<td>--</td>
<td>34</td>
<td>28 MiB</td>
<td>4</td>
<td>256 GiB (host) + 8 GiB x 4</td>
<td>861W</td>
<td>290W</td>
<td>384W</td>
</tr>
</tbody>
</table>
Why us and why now?

- The rest of industry is already moving forward with specialization
  - Gen2 Google TPU 2,300x faster than CPU for AI workloads
  - Microsoft FPGA accelerators for search acceleration
  - Numerous other specialized architectures in AI space

- Little incentive for industry to provide scientifically relevant accelerators
  - We need to learn how to do this ourselves!
  - When should we begin? **NOW!**

- Why will this work now (not very successful in past)
  - Already demonstrated successes in cell phones, datacenters and AI
  - Won’t be eclipsed by Moore’s Law performance growth **this time!**
Why DFT?

- Mature code that has large user base
  - Hard to specialize for moving target
  - Offers high scientific impact

- $O(N)$ (LS3DF) algorithmic alternative maximizes spatial locality

- Most users regard DFT codes as an “appliance”!
  (good candidate)
Why $O(N)$ divide & conquer method (e.g., LS3DF)?

Each Patch executes entirely within FPGA

*eliminate memory bottleneck*

Scale problem by adding FPGAs
The DFT kernel for each fragment
(to be performed within each FPGA)
Scale to larger problems by adding FPGAs

\[ h(i, j) = \langle \psi_i | H | \psi_j \rangle \]

\[ P_i = H\psi_i - \epsilon_i \psi_i \]

\[ P_i = A(P_i - \frac{\lambda_i}{\lambda_i^o} P_i^o) \]

\[ P_i = P_i - \sum_{j=1}^{i} \langle P_i | \psi_j \rangle \]

\[ \psi_i = \psi_i \cos \theta_i + P_i \sin \theta_i \]

\[ \psi_i = \psi_i - \sum_{j<i} \langle \psi_i | \psi_j \rangle \]

Sub_diag, *

Hpsi, *

Precond. CG step

Projection, *

Line minimiz.

Orth., *

Sub_diag, *

O(\(N^2 \log(N))\))
Comm bound if non-local

3D parallel FFT

TSQR & Cholesky

ZGEMM

O(\(N^3\))
Compute-bound

One patch per FPGA
400 atoms/patch

FY19 LDRD
Field Programmable Gate Arrays (FPGA)

**FPGA**

- **Cost for first FPGA (NRE):** $1,200-$3,000
- **Cost for 20,000th:** $1,200-$3,000
- **Clock Rate:** 0.1-0.3Ghz

**ASIC**

- **Cost for first ASIC (NRE):** $2M-$15M
- **Cost for 20,000th:** $100
- **Clock Rate:** 1-2 Ghz (10x)
- **Area Efficiency:** 10x FPGA
- **Energy Efficiency:** 10x-100x FPGA

*Use FPGA as a testbed for ASIC*

*If successful, can project ASIC performance*
On-detector processing
Putting our hardware design tool suite to work to augment existing HPC resources

- **The Problem:**
  - Future detectors threaten to overwhelm data transfer and computing capabilities with data rates exceeding 1 Tb/s
  - Data processing experiment driven

- **Proposed solution:**
  - Process the data *before it leaves the sensor*
  - Application-tailored, programmable processing allows data reduction to occur on-sensor
  - Programmability allows data reduction techniques to be tailored to the experiment – even *after* the sensor is built!
On-detector processing: CryoEM

768x performance density boost ½ when paired with RISC-V core
Net ~7x when on FPGA

Francetti: SPIRAL

• All FFT instructions are contained within:
  \( \text{opcode}[1:0] = 00b \)
  • All standard RV32 instructions are contained in:
  \( \text{opcode}[1:0] = 11b \)

<table>
<thead>
<tr>
<th>Instruction</th>
<th>opcode[3:2]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fft_config</td>
<td>10b</td>
<td>Configures FFT parameters</td>
</tr>
<tr>
<td>fft_status</td>
<td>01b</td>
<td>Reads FFTAccel status registers</td>
</tr>
<tr>
<td>fft_start</td>
<td>11b</td>
<td>Starts FFT processing</td>
</tr>
<tr>
<td>fft_stop</td>
<td>00b</td>
<td>Stops FFT processing</td>
</tr>
</tbody>
</table>

• This leaves \( \text{opcode}[6:4], \func3, \text{and func7} \) unused
  • Potentially used to address multiple FFTAccel
More opportunities for specialization

Custom computing could be applied and incorporated in many areas across DOE

- **JGI – Dedicated appliance for BLAST**
  - Majority of compute time at JGI spent on BLAST
  - Design, build and deploy custom BLAST computing appliance

- **BRAIN – Real-time signal processing**
  - To perform closed-loop experiments need to process and respond to incoming neural data in < 10ms

- **LSST – Transient detection**
  - Power constrained on-site, custom, low-power computing could reduce need for lossy compression or high-bandwidth, long-haul networks

- **All these applications augment rather than replace future HPC resources**
There are Multiple Ways to Specialize

1. **Extended ISA:** extend ISA to replace common motifs with Fixed function within each core.
   - Requires a more flexible/extensible compiler
   - Tensilica: ISA extensions exposed as intrinsics that could easily be executed as subroutine or compiler generates code

2. **Diverse function accelerators within each chip:** Smartphone, DE Shaw Anton and Amazon Cloud examples
   - Expose as software interfaces at first, and then invoke hardware
   - Currently labor-intensive to build underlying interfaces (need standardization of low-level interfaces (an accelerator ABI?)

3. **Diverse Discrete Accelerators In System:** Our current approach with GPU, but extended to diverse accelerators
   - We already see how this works out (need to better manage data movement)
Data Movement Could Undercut any Gains from Accelerators and Specialization

Under what circumstances?
How bad is it?

*Let's just do a simple thought experiment*
The problem with Wires

Energy to move data proportional to distance

♦ Energy Efficiency of copper wire:
  - Power = frequency * Length / cross-section-area
  - Wire efficiency *does not improve* as feature size shrinks

♦ Energy Efficiency of a Transistor:
  - Power = \( V^2 \) * frequency * Capacitance
  - Capacitance \(~=\) Area of Transistor
  - Transistor efficiency improves as you shrink it

♦ *Net result is that moving data on wires is starting to cost more energy than computing on said data*
Abstract

A 45nm 1.3GHz 16.7 Double-Precision GFLOPS/W RISC-V Processor with Vector Accelerators

Introduction

In this paper, we present a 64-bit dual-core RISC-V processor with vector accelerators in a 45 nm SOI process. This processor has a clock rate of 1.3 GHz at 1.2 V and peak energy efficiency of 16.7 double-precision GFLOPS/W at 0.65 V with an area of 3 mm$^2$.

Rocket Pipeline

Figure 1 shows the block diagram of the dual-core processor. Rocket is a 6-stage single-issue in-order pipeline that has an optional IEEE 754-2008-compliant FPU.

Hwacha Pipeline

The Hwacha vector accelerator is 1.8 times more energy-efficient than the IBM Cell processor, both fabricated in the same process. The dual-core RISC-V processor achieves maximum clock frequency of 1.3 GHz at 1.2 V and peak energy efficiency of 16.7 double-precision GFLOPS/W at 0.65 V with an area of 3 mm$^2$.

Verification

We utilize a verification suite, a Linux port, and additional documentation, developed at the University of California, Berkeley, which is open-source.

Computational Benchmark

Compared to ARM's Cortex-A5 score of 1.57 DMIPS/MHz, our 64-bit dual-core RISC-V processor outperforms ARM's comparable processor and 2.6% is 1.8 times more higher in DMIPS/MHz than the Cortex-A5, ARM's comparable processor.

RISC-V Software Toolchain

RISC-V software toolchain includes a GCC cross-compiler, OpenJDK, RISC-V software development kits for various fields, a Linux port, and additional documentation developed at the University of California, Berkeley, which is open-source.

Basic Stats

<table>
<thead>
<tr>
<th></th>
<th>Core Energy/Area est.</th>
<th>Wire Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Area: 12.25 mm$^2$</td>
<td><img src="#" alt="Wire Energy" /></td>
</tr>
<tr>
<td></td>
<td>Power: 2.5W</td>
<td>Assumptions for 22nm</td>
</tr>
<tr>
<td></td>
<td>Clock: 2.4 GHz</td>
<td>100 fJ/bit per mm</td>
</tr>
<tr>
<td></td>
<td>E/op: 651 pj</td>
<td>64-bit operand</td>
</tr>
<tr>
<td></td>
<td>Area: 0.6 mm$^2$</td>
<td>Energy:</td>
</tr>
<tr>
<td></td>
<td>Power: 0.3W (&lt;0.2W)</td>
<td>1 mm=~6 pj</td>
</tr>
<tr>
<td></td>
<td>Clock: 1.3 GHz</td>
<td>20 mm=~120 pj</td>
</tr>
<tr>
<td></td>
<td>E/op: 150 (75) pj</td>
<td>[1, †]</td>
</tr>
<tr>
<td></td>
<td>Area: 0.046 mm$^2$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Power: 0.025W</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Clock: 1.0 GHz</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E/op: 22 pj</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
When does data movement dominate?

<table>
<thead>
<tr>
<th>Core</th>
<th>Area: 12.25 mm²</th>
<th>Power: 2.5W</th>
<th>Clock: 2.4 GHz</th>
<th>E/op: 651 pj</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Movement Cost</td>
<td>Compute Op == data movement Energy @ 108mm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Energy Ratio for 20mm</td>
<td>0.2x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Core</th>
<th>Area: 0.6 mm²</th>
<th>Power: 0.3W (&lt;0.2W)</th>
<th>Clock: 1.3 GHz</th>
<th>E/op: 150 (75) pj</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Movement Cost</td>
<td>Compute Op == data movement Energy @ 12mm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Energy Ratio for 20mm</td>
<td>1.6x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Core</th>
<th>Area: 0.046 mm²</th>
<th>Power: 0.025W</th>
<th>Clock: 1.0 GHz</th>
<th>E/op: 22 pj</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Movement Cost</td>
<td>Compute Op == data movement Energy @ 3.6mm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Energy Ratio for 20mm</td>
<td>5.5x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SUBROUTINE SIMPLE(N, A, B)
INTEGER I, N
REAL B(N), A(N)
!$OMP PARALLEL DO !I is private by default
DO I=2,N
   B(I) = (A(I) + A(I-1)) / 2.0
ENDDO
!$OMP END PARALLEL DO
END SUBROUTINE SIMPLE
Data Movement Directives
(directives in red) John Levesque

Keep data on the accelerator with acc_data_region

!$acc data copyin(cix,ci1,ci2,ci3,ci4,ci5,ci6,ci7,ci8,ci9,ci10,ci11,&
!$acc& ci12,ci13,ci14,r,b,uxyz,cell,rho,grad,index_max,index,&
!$acc& ciy,ciz,wet,np,streaming_sbuf1, &
!$acc& streaming_sbuf1,streaming_sbuf2,streaming_sbuf4,streaming_sbuf5,&
!$acc& streaming_sbuf7s,streaming_sbuf8s,streaming_sbuf9n,streaming_sbuf10s,&
!$acc& streaming_sbuf11n,streaming_sbuf12n,streaming_sbuf13s,streaming_sbuf14n,&
!$acc& streaming_sbuf7e,streaming_sbuf8w,streaming_sbuf9e,streaming_sbuf10e,&
!$acc& streaming_sbuf11w,streaming_sbuf12e,streaming_sbuf13w,streaming_sbuf14w, &
!$acc& streaming_rbuf1,streaming_rbuf2,streaming_rbuf4,streaming_rbuf5,&
!$acc& streaming_rbuf7n,streaming_rbuf8n,streaming_rbuf9s,streaming_rbuf10n,&
!$acc& streaming_rbuf11s,streaming_rbuf12s,streaming_rbuf13n,streaming_rbuf14s,&
!$acc& streaming_rbuf7w,streaming_rbuf8e,streaming_rbuf9w,streaming_rbuf10w,&
!$acc& streaming_rbuf11e,streaming_rbuf12w,streaming_rbuf13e,streaming_rbuf14e, &
!$acc& send_e,send_w,send_n,send_s,recv_e,recv_w,recv_n,recv_s)

do ii=1,ntimes
  o o o
  call set_boundary_macro_press2
  call set_boundary_micro_press
  call collisiona
  call collisionb
  call recolor

Where is the Abstraction?
OMP4 taking on Data-movement/Locality Issues
And you have to do this for EVERY SINGLE LOOP!

```c
#pragma omp target data if(N>THRESHOLD) map(from: p[0:N])
{
    #pragma omp target if (N>THRESHOLD) map(to: v1[:N], v2[:N])
    #pragma omp parallel for
    for (i=0; i<N; i++)
        p[i] = v1[i] * v2[i];
    init_again(v1, v2, N);
    #pragma omp target if (N>THRESHOLD) map(to: v1[:N], v2[:N])
    #pragma omp parallel for
    for (i=0; i<N; i++)
        p[i] = p[i] + (v1[i] * v2[i]);
}
```

Where is the Abstraction?
What do you mean by “Data Centric Programming Model?”

♦ Old Model (Compute-Centric)
  - Describe how to parallelize loop iterations
  - Parallel “DO” divides loop iterations evenly among processors
  - . . . but where is the data located?

♦ New Model (Data-Centric) *also in big data*
  - Describe how data is laid out in memory
  - Loop statements operate on data where it is located
  - Similar to MapReduce, but need more sophisticated descriptions of data layout for scientific codes

```c
forall_local_data(i=0;i<NX;i++;A)
C[j]+=A[j]*B[i][j]);
```

*Its “owner computes” with runtime info for RTS to map*
Loops Should Bind to Data Layout and not the other way around!

Building up a hierarchical layout

- Layout block coreblk \{blockx,blocky\};
- Layout block nodeblk \{nnx,nny,nnz\};
- Layout hierarchy myheirarchy \{coreblk,nodeblk\};
- Shared myhierarchy double \[a[nx][ny][nz]\];

Then use data-localized parallel loop

\[
\text{doall}\_\text{at}(i=0;i<nx;i++;a)\
\text{doall}\_\text{at}(j=0;j<ny;j++;a)\
\text{doall}\_\text{at}(k=0;k<nz;k++;a)\
\]

\[a[i][j][k]=C*a[i+1]...>\]

And if layout changes, loop remains the same

a) Logical Tiles (CPU)  
b) Separated Tiles (GPU)  
c) Regional Tiles
Need Data-Centric Programming Models

Whether Future Looks Homogeneous Or Stacked Or Heterogeneous

A Programming Model that Efficiently Expresses Data Locality so that Runtime Can Optimize Is Essential (see http://www.padalworkshop.org)
Beyond Bulk Synchronous

How will we handle performance diversity and also hardware diversity?

*IPAM speakers have made similar observation*
Assumptions of Uniformity is Breaking (many new sources of heterogeneity)

Bulk Synchronous Parallel Execution Model

Asynchronous / DAG Execution Model
Beyond Bulk-Synchronous Execution

Bulk Synchronous (current practice)

Asynchronous / DAG Model / static schedule (production interface is still topic of research)

Need to re-examine Functional semantics for programming systems or (taking page from C++) Dysfunctional Languages!
Future of Digital Electronics?
Von Neumann Emulates Logic Array
PIM is NOT Non-Von Neumann
Its just better packaging
Organizing principles for Non-Von Digital Design

- Data Movement will remain a challenge even with exotic materials, but especially CMOS (*perhaps not with superconducting… ask Burton*)
- Copper is as good of a conductor as you can expect at room temperature
- With even lower power switches, challenges skews even more to data movement (*NEED Spatial Computing approach*)
- Push towards more parallelism (more tessellation of the memory structures)… Strong Scaling

Strong Scaling extrapolates to **limit case** with no separation of memory and compute (*e.g. one PDE cell per processing element*)
PDE on a Block Structured Grid
Extrapolated to Non-Von Neumann
PDE on a Block Structured Grid Extrapolated to Non-Von Neumann
PDE on a Block Structured Grid Extrapolated to Non-Von Neumann
PDE Solvers on Block Structured Grid
"Solid State Digital Fluid"

PDE Element

2D Slice

3D problem domain

NOT a CPU
**Concept: Solid State Virtual Fluid**

*Extreme (spatial) Specialization + New Devices + New programming models*

---

**PDEcell / PICcell**: Ultra-simple compute engine (50k gates) calculates finite-difference updates, and particle forces from neighbors. Microinstructions specify the PDE equation, stencil, and PIC operators. **Novel features**: variable length streaming integer arithmetic and novel PIC particle virtualization scheme.

**Computational Lattice**: PDECells are tiles in a lattice/array on each 2D planar chip layer. Target 120x120 tiles per mm² @28nm lithography. **Novel Features**: each tile represents single cell of computational domain (pushes to limit of strong-scaling).

**Monolithic 3D Integration**: Integrate layers of compute elements using emerging monolithic 3D chip stacking. **Novel Features**: 1000 layer stacking (20x more than current practice). Area efficient inter-layer connectivity and new energy efficient transistor logic (ncFET). 1 Petaflop equivalent performance in 300mm² for < 200Watts.

---

Virtualize the program that is applied to the data.

---

Scalar waves in 3D are solutions of the hyperbolic wave equation:

\[-f_{,tt} + f_{,xx} + f_{,yy} + f_{,zz} = 0\]

Initial value problem: given data for \(f\) and its first time derivative at initial time, the wave equation says how it evolves with time.

---

Compiles to MicroOps:

\[
\begin{align*}
R[n+1][0,0,0] &= 0 \\
R[n+1][0,0,0] &= 2*R[n][0,0,0] \\
R[n+1][0,0,0] &= 2*R[n][1,0,0] \\
R[n+1][0,0,0] &= 2*R[n][0,0,0] \\
R[n+1][0,0,0] &= 2*R[n][1,0,0] \\
R[n+1][0,0,0] &= 2*R[n][1,0,0] \\
\end{align*}
\]

Executes in Wavefronts:

1. **1st compute wave**
2. **2nd compute wave**
3. **3rd compute wave**
4. **4th compute wave**

---

1000 Layers Monolithic Stacking
Spatial Computing

locality aware domain-specialized hardware

Challenge
- Data Movement dominates energy losses
- Copper is as good of a conductor as you can expect at room temperature
- Challenge is exacerbated improving energy efficiency of the devices

Approach
- Identify common computational patterns
- *Design machine that conforms SPATIALLY to a computational pattern in 3D*
- Design for limit case of parallelism

Result: Spatial Computing

<table>
<thead>
<tr>
<th>Extreme Specialization</th>
<th>Extreme Data Locality</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Giants of Data (NRC)</td>
<td>7 Motifs of Simulation</td>
</tr>
<tr>
<td>Basic statistics</td>
<td>Monte Carlo methods</td>
</tr>
<tr>
<td>Generalized N-Body</td>
<td>Particle methods</td>
</tr>
<tr>
<td>Graph-theory</td>
<td>Unstructured meshes</td>
</tr>
<tr>
<td>Linear algebra</td>
<td>Dense Linear Algebra</td>
</tr>
<tr>
<td>Optimizations</td>
<td>Sparse Linear Algebra</td>
</tr>
<tr>
<td>Integrations</td>
<td>Spectral methods</td>
</tr>
<tr>
<td>Alignment</td>
<td>Structured Meshes</td>
</tr>
</tbody>
</table>

---

*On-die Interconnect*

*Graph*:
- Compute energy
- On-die IC energy
- Technology (nm)

*Graph Note*:
- On-die IC energy (per mm) reduces slower than compute data movement energy will start to dominate
- Copyright © 2014 Intel Corporation
Spatial Computing Concept for HEP

Following Images from Ted Liu of FNAL
(thliu@fnal.gov)
Why do we need more performance?  
*(Jim Seigrist HEP, May 2018)*

Working group began by conducting an initial survey of the computing needs from each of the three physics Frontiers, and assembled this into a preliminary model:

- Energy Frontier portion alone was a large factor beyond the current computing budget.
- Large data volumes with the HL-LHC require correspondingly large amounts of computing to analyze it:
  - Grid-only solution: \( \$850M \pm 200M \)
  - Using the experiments’ estimates of future HPC use reduces this to \( \$650M \pm 150M \)

---

**LHC/HL-LHC Computing Costs**

Fall 2017

![Graph showing LHC/HL-LHC Computing Costs](image)

---

**Track Reconstruction**

- Hits from one particle
- Hits from a fraction of particles in HL-LHC environment
- Seeding: triplet, quadruplet, n-plet finding
- Track finding: combinatorial Kalman filtering
- Track fitting & classification:
  - Kalman filtering, smoothing
  - \( \chi^2 \) minimization
  - Track classification, etc.
Pattern recognition for tracking is naturally a task in 3D.
Conclusion

- The end of lithography scaling as we know it is coming within a decade (*close to when Exascale is done*)

- Consequence is that parallel processing will be even more central to our future, but has NEW challenges
  - Diverse Hardware Specialization (*many in single node*)
  - Data Centric Computing (*Spatial Computing*)
  - Non-Bulk Synchronous execution models

- Cannot ignore this until after exascale
  - *Top Science challenges should guide our investments*
  - *Our next generation of machines should not be judged by “extensive metrics” (scale is a poor metric for success)*
  - *Judge based on the quality of science enabled.*
Extra
Accelerated feedback path to focus device and material discovery process

**Length Scales**

- **Processor/System:** 10k-100k Circuits
- **Circuit/Std. Cell:** 10-100 Devices
- **One Device:** ~1M Atoms
- **One Junction:** ~100k Atoms
- **Bulk Material:** ~100 Atoms
Device level and materials advances must be understood at the “systems” level! An end-to-end process is required!

Clock-Rates, Power, Area

Current Drive, switching energy, transients

Junction Physics, I-V curves

Material Physics
Carrier Mobility

Bulk Material:
~100 Atoms

One Junction:
~100k Atoms

One Device:
~1M Atoms

Circuit/Std. Cell:
10-100 Devices

Processor/System:
10k+ Circuits

Application Performance
System-Power

Switch Speed, Power,
Area, Fan-out, Stability

Interface-level
Losses/Performance

Materials Metrics

Length Scales

Architectural
Simulation

Analog
Simulation

Compact
Models

Junctions

Bulk
Materials

PARADISE

LS3DF
Essence of a Neural Network (for context)

A. Dendrites, axon, cell body, terminal axon.

B. Function $f(x)$ with inputs $x_1, x_2, \ldots, x_n$ and output $y_1$.

C. Synapse in neural network.

D. Neural network with multiple layers and synapses.
Skyrmions
Single-track Skyrmion-Based Spiking Neural Network

Z. He et al., 1705.02995v1 (2017)

Crosspoint

Pre-synaptic

Post-synaptic

Integrate-and-Fire Neuron

Skyrmion driven current Pulse

Fire spike

Launched Skyrmion

Neuron input terminal

As input

Insulating Coupling Layer

Gate Electrode

Spin Polarizer

Integrate-and-Fire Neuron

Heavy Metal Substrate

Ultrathin Magnetic Film
Scalar waves in 3D are solutions of the hyperbolic wave equation:

\[-\phi_{,tt} + \phi_{,xx} + \phi_{,yy} + \phi_{,zz} = 0\]

**Initial value problem:** given data for \(\phi\) and its first time derivative at initial time, the wave equation says how it evolves with time.

![Diagram showing wave evolution over time and space](image)
Numerical solve by discretising on a grid, using explicit *finite differencing* (centered, second order)

\[
\phi_{i,j,k}^{n+1} = 2\phi_{i,j,k}^n - \phi_{i,j,k}^{n-1} \\
+ \frac{\Delta t^2}{\Delta x^2}(\phi_{i+1,j,k}^n - 2\phi_{i,j,k}^n + \phi_{i-1,j,k}^n) \\
+ \frac{\Delta t^2}{\Delta y^2}(\phi_{i,j+1,k}^n - 2\phi_{i,j,k}^n + \phi_{i,j-1,k}^n) \\
+ \frac{\Delta t^2}{\Delta z^2}(\phi_{i,j,k+1}^n - 2\phi_{i,j,k}^n + \phi_{i,j,k-1}^n)
\]
Decomposing Into PDE Weights

\[
\phi^{n+1}_{i,j,k} = 2\phi^n_{i,j,k} - \phi^{n-1}_{i,j,k} \\
\Delta t^2/\Delta x^2(\phi^n_{i+1,j,k} - 2\phi^n_{i,j,k} + \phi^n_{i-1,j,k}) \\
+ \Delta t^2/\Delta y^2(\phi^n_{i,j+1,k} - 2\phi^n_{i,j,k} + \phi^n_{i,j-1,k}) \\
+ \Delta t^2/\Delta z^2(\phi^n_{i,j,k+1} - 2\phi^n_{i,j,k} + \phi^n_{i,j,k-1})
\]

\[
R_{[t=n+1]}(0,0,0) = 0 \\
R_{[t=n+1]}(0,0,0) += 2*R_{[t=n]}(0,0,0) \\
R_{[t=n+1]}(0,0,0) -= R_{[t=n-1]}(0,0,0) \\
R_{[t=n+1]}(0,0,0) += C * R_{[t=n+1]}(+1,0,0) \\
R_{[t=n+1]}(0,0,0) -= C * 2 * R_{[t=n]}(0,0,0) \\
R_{[t=n+1]}(0,0,0) += C * R_{[t=n]}(-1,0,0) \\
R_{[t=n+1]}(0,0,0) += C * R_{[t=n+1]}(0,+1,0) \\
R_{[t=n+1]}(0,0,0) -= C * 2 * R_{[t=n]}(0,0,0) \\
R_{[t=n+1]}(0,0,0) += C * R_{[t=n]}(0,-1,0) \\
R_{[t=n+1]}(0,0,0) += C * R_{[t=n+1]}(0,0,+1) \\
R_{[t=n+1]}(0,0,0) -= C * 2 * R_{[t=n]}(0,0,0) \\
R_{[t=n+1]}(0,0,0) += C * R_{[t=n]}(0,0,-1) \\
\text{Rotate Registers}
\]
$\phi_{i,j,k}^{n+1} = 2\phi_{i,j,k}^{n} - \phi_{i,j,k}^{n-1} + \frac{\Delta t^2}{\Delta x^2}(\phi_{i+1,j,k}^{n} - 2\phi_{i,j,k}^{n} + \phi_{i-1,j,k}^{n})$
+$\frac{\Delta t^2}{\Delta y^2}(\phi_{i,j+1,k}^{n} - 2\phi_{i,j,k}^{n} + \phi_{i,j-1,k}^{n})$
+$\frac{\Delta t^2}{\Delta z^2}(\phi_{i,j,k+1}^{n} - 2\phi_{i,j,k}^{n} + \phi_{i,j,k-1}^{n})$

PDE Weights

$R[n+1](0,0,0) = 0$
$R[n+1](0,0,0) += 2*R[n](0,0,0)$
$R[n+1](0,0,0) -= R[n-1](0,0,0)$
$R[n+1](0,0,0) += C * R[n+1](+1,0,0)$
$R[n+1](0,0,0) -= C * 2 * R[n](0,0,0)$
$R[n+1](0,0,0) += C * R[n](-1,0,0)$
$R[n+1](0,0,0) += C * R[n+1](0,+1,0)$

Fed Sequentially to MicroOp Entry Point
Challenges of Limit-Case Non-Von Neumann Digital

Von Neumann ~= Instruction Processor

```c
int main()
{
    int n = 0;
    while(n < 100)
    {
        n = n + 5;
        printf("n = %d\n", n);
        pause(200);
        if(n == 50) break;
    }
    printf("All done!\n");
}
```

What the heck is this?

PDE in Domain Specific Representation

\[
\phi_{i+1,j,k}^{n+1} = 2\phi_{i,j,k}^{n+1} - \phi_{i,j,k}^{n} + \frac{\Delta t}{\Delta x^2} (\phi_{i+2,j,k}^{n} - 2\phi_{i+1,j,k}^{n} + \phi_{i,j,k}^{n}) + \frac{\Delta t}{\Delta y^2} (\phi_{i,j+2,k}^{n} - 2\phi_{i,j+1,k}^{n} + \phi_{i,j,k}^{n}) + \frac{\Delta t}{\Delta z^2} (\phi_{i,j,k+2}^{n} - 2\phi_{i,j,k+1}^{n} + \phi_{i,j,k}^{n})
\]

Compiled to MicroOps

- **1st compute wave**
- **2nd compute wave**
- **3rd compute wave**
- **4th compute wave**

MicroOp Entry Point

Fed Sequentially to

Modern languages (including many classes of DSLs) Were designed with instruction processors in mind
PDEcell / PICcell: Ultra-simple compute engine (50k gates) calculates finite-difference updates, and particle forces from neighbors. Microinstructions specify the PDE equation, stencil, and PIC operators. **Novel features:** variable length streaming integer arithmetic and novel PIC particle virtualization scheme.

Computational Lattice: PDECells are tiles in a lattice/array on each 2D planar chip layer. Target 120x120 tiles per mm² @28nm lithography. **Novel Features:** each tile represents single cell of computational domain (pushes to limit of strong-scaling).

Monolithic 3D Integration: Integrate layers of compute elements using emerging monolithic 3D chip stacking. **Novel Features:** 1000 layer stacking (20x more than current practice). Area efficient inter-layer connectivity and new energy efficient transistor logic (ncFET). 1 Petaflop equivalent performance in 300mm² for < 200Watts.

**Scalar waves in 3D are solutions of the hyperbolic wave equation:**

\[ \phi_n^{i,j,k} + \phi_n^{i,j,k} + \phi_n^{i,j,k} = 0 \]

Initial value problem: given data for \( \phi \) and its first time derivative at initial time, the wave equation says how it evolves with time.