RISC-V SUMMIT

NORTH AMERICA

RISC-V SUMMIT
NORTH AMERICA

The Benefits of Building New AI Accelerators with RISC-V

Martin Maas, Cliff Young
Google DeepMind

About Us

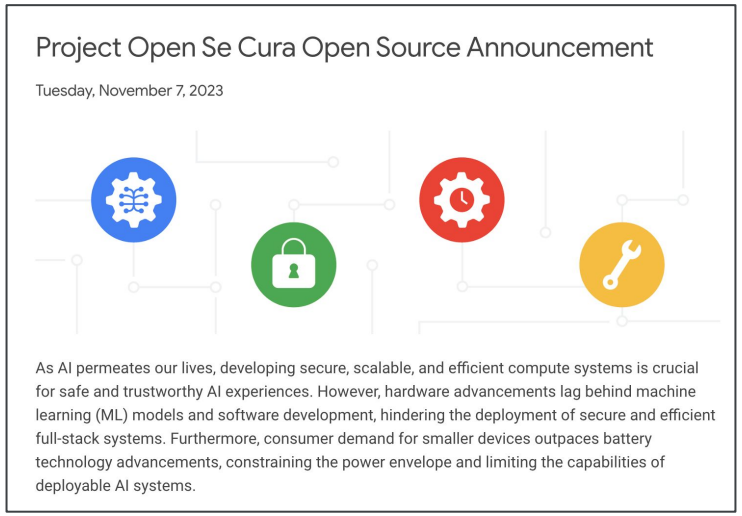
Research Scientist
Google DeepMind
ML for Systems, Systems for ML
J Extension TG Chair

Software Engineer
Google DeepMind
Anton, TPUv1, MLPerf
AI/ML Apps. SIG Act. Chair

Disclaimer

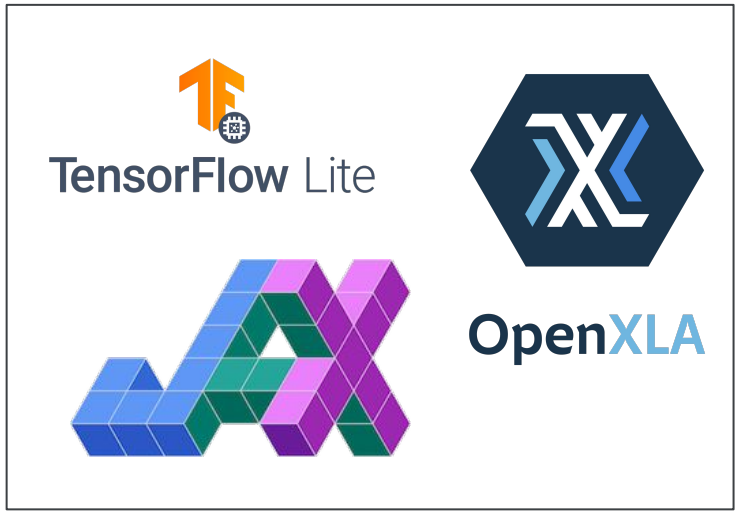
This talk is **not** about Google's products or systems. We are researchers and want to share relevant experience that could be helpful for RISC-V.

Increasing Interest in AI within RISC-V



RISC-V-based AI Accelerators & IP

AI-Related RISC-V ISA Standards



Growing ML Software Ecosystem

Current State of RISC-V & AI

So far, AI use cases **have not significantly affected the ISA**.

- Different AI accelerators use RISC-V in different ways, with custom non-standard extensions (e.g., for matrix multiplication).

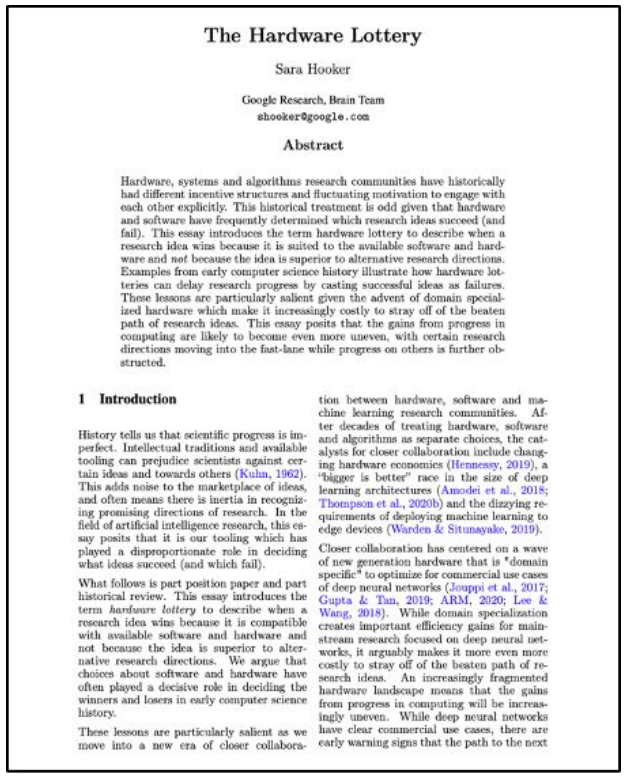
As RISC-V is considering the addition of **AI-specific standards**, **the ISA will have to decide on trade-offs**.

Agenda

1. RISC-V's Value Proposition for AI
2. Flexibility vs. Fragmentation
3. Reuse vs. Customizability
4. Many Core vs. Single Core
5. Numerics Considerations
6. Current vs. Future Models
7. A Call to Action

PART I
RISC-V's Value Proposition for AI

The Hardware Lottery



The Hardware Lottery

Sara Hooker

Google Research, Brain Team
shooker@google.com

Abstract

Hardware, systems and algorithms research communities have historically had different incentive structures and fluctuating motivation to engage with each other explicitly. This historical treatment is odd given that hardware and software have frequently determined which research ideas succeed (and fail). This essay introduces the term hardware lottery to describe when a research idea wins because it is suited to the available software and hardware and *not* because the idea is superior to alternative research directions. Examples from early computer science history illustrate how hardware lotteries can delay research progress by casting successful ideas as failures. These lessons are particularly salient given the advent of domain specialized hardware which make it increasingly costly to stray off of the beaten path of research ideas. This essay posits that the gains from progress in computing are likely to become even more uneven, with certain research directions moving into the fast-lane while progress on others is further obstructed.

1 Introduction

History tells us that scientific progress is imperfect. Intellectual traditions and available tooling can prejudice scientists against certain ideas and towards others (Kuhn, 1962). This adds noise to the marketplace of ideas, and often means there is inertia in recognizing promising directions of research. In the field of artificial intelligence research, this essay posits that it is our tooling which has played a disproportionate role in deciding what ideas succeed (and which fail).

What follows is part position paper and part historical review. This essay introduces the term *hardware lottery* to describe when a research idea wins because it is compatible with available software and hardware and not because the idea is superior to alternative research directions. We argue that choices about software and hardware have often played a decisive role in deciding the winners and losers in early computer science history.

These lessons are particularly salient as we move into a new era of closer collaboration between hardware, software and machine learning research communities. After decades of treating hardware, software and algorithms as separate choices, the catalysts for closer collaboration include changing hardware economics (Hennessy, 2019), a "bigger is better" race in the size of deep learning architectures (Amodei et al., 2018; Thompson et al., 2020b) and the dizzying requirements of deploying machine learning to edge devices (Warden & Situnayake, 2019).

Closer collaboration has centered on a wave of new generation hardware that is "domain specific" to optimize for commercial use cases of deep neural networks (Jouppi et al., 2017; Gupta & Tan, 2019; ARM, 2020; Lee & Wang, 2018). While domain specialization creates important efficiency gains for mainstream research focused on deep neural networks, it arguably makes it more even more costly to stray off of the beaten path of research ideas. An increasingly fragmented hardware landscape means that the gains from progress in computing will be increasingly uneven. While deep neural networks have clear commercial use cases, there are early warning signs that the path to the next

The systems we use determine the kinds of AI solutions that we can explore and build.

- Today, we enable huge, dense, matrix multiplication.
- Is that the only or most important area to make progress?

RISC-V can help, by making it easier to build new AI accelerator paradigms and support a wider range of workloads on existing ones.

The term was coined by Sara Hooker in her article “The Hardware Lottery”, in *Communications of the ACM* (Dec 2021).

Escaping the Hardware Lottery

Example #1: Real-time image processing that involves a mix of ML and conventional DSP-style processing

- If the latency between the two units is too large, we cannot implement it.

Example #2: Researchers develop a new type of activation function, but no hardware is available to train with it.

- Without general-purpose compute, we must wait for new hardware.

RISC-V's Value Proposition

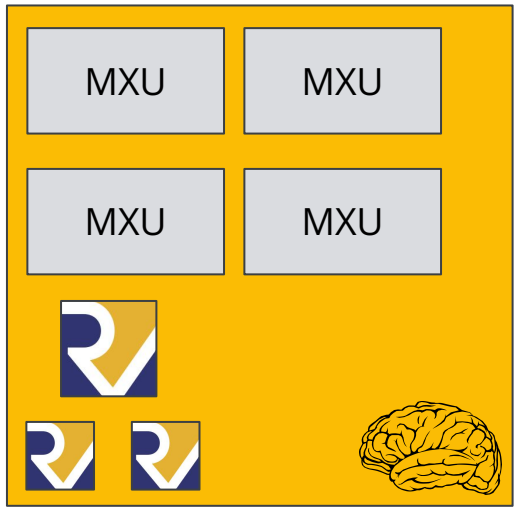
Some key benefits of RISC-V over existing, proprietary ISAs are:

- **Customizability**: Adding proprietary extensions to the ISA.
- **Scalability**: Can target a wide range of hardware designs.
- **Openness**: ISA can be used and adapted without restrictions.

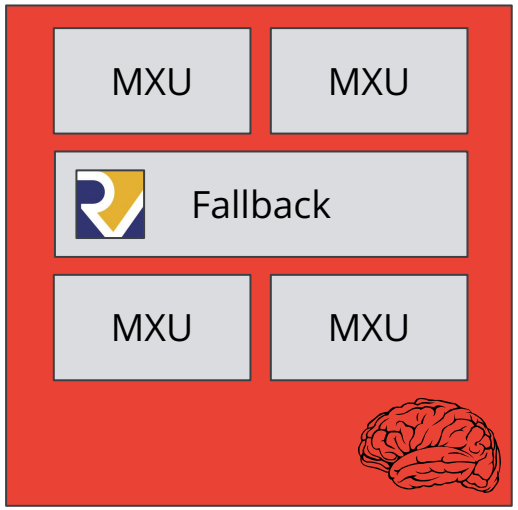
Not specific to AI, but it can be synergistic with AI use cases.

- Main downside for existing accelerators is compatibility/porting effort.
- In this talk, we only focus on "New AI Accelerators".

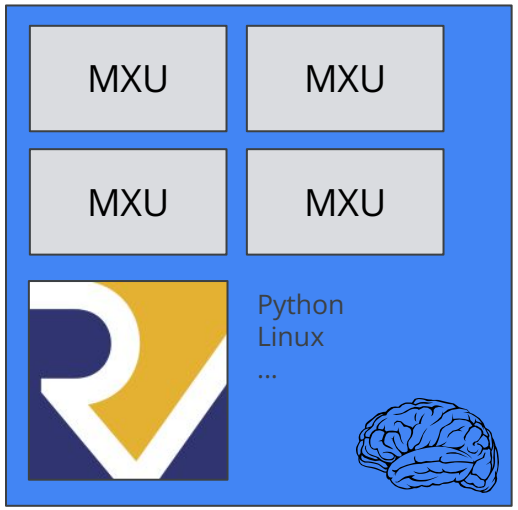
No Single Use Case in AI Accelerators



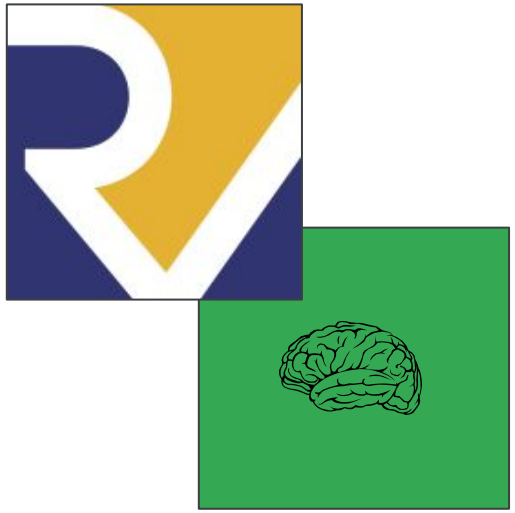
1) Replace custom-built sequencers, cores for management, boot, security, etc.



2) Fallback for less-common operations not handled by specialized hardware (e.g., elementwise).



3) AP-like CPU, perhaps running agentic code that complements the accelerator.



4) AI-specialized application processors or embedded/co-processor systems.

Implications for RISC-V

Because there is **no single use case**, focusing on or prescribing one single way of AI use may be detrimental to RISC-V.

The largest benefits are things RISC-V already does:

- Better **single-core performance** to keep up with the accelerator hardware.
- Better **compilation quality** for highly tuned compute kernels.
- Better **software ecosystem**: Being able to run Python in an AI accelerator benefits many use cases. Running an OS does as well.

Takeaway: There are many things hardware could do to help AI, but this does not mean that RISC-V needs to add them. Clarity is required on where the value-add of RISC-V is.

PART II
Flexibility vs. Fragmentation

RISC-V and Fragmentation

RISC-V provides the **minimum standards** so software “just works”.

- Binary compatibility and portability of optimizations is important.
- Mandating too little invites fragmentation. Mandating too much weakens the ecosystem, as it leads to poor quality implementations or harms adoption.

The benefit for AI accelerators is different: Flexibility means that it can be used in many use cases without mandating how it is used.

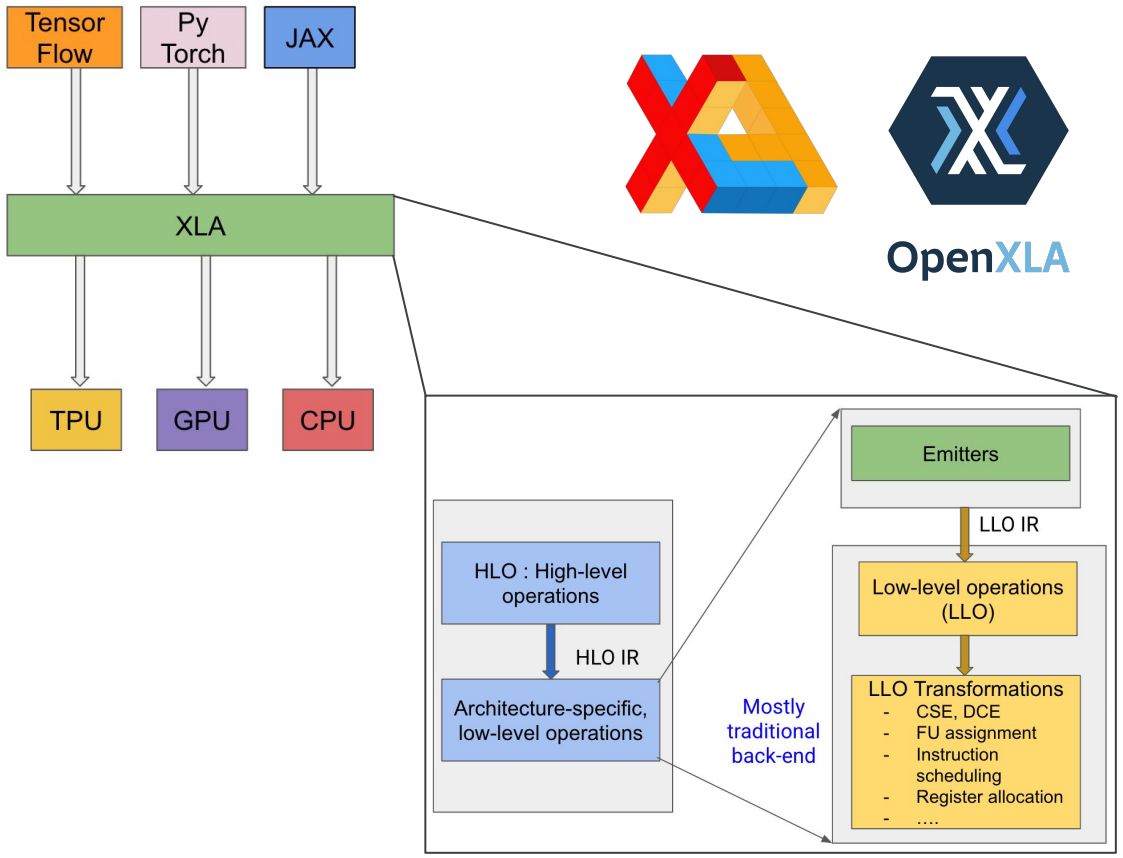
- ML compilers and kernel libraries abstract away the hardware details.
- Not every feature needs to be part of the RISC-V standard.

The Role of the ML Compiler

AI workloads heavily rely on **ML compilers** to target accelerators.

- Binary compatibility is less important.
- Even if the ISA is standardized, the compiler will not "just work" if two accelerators are very different.

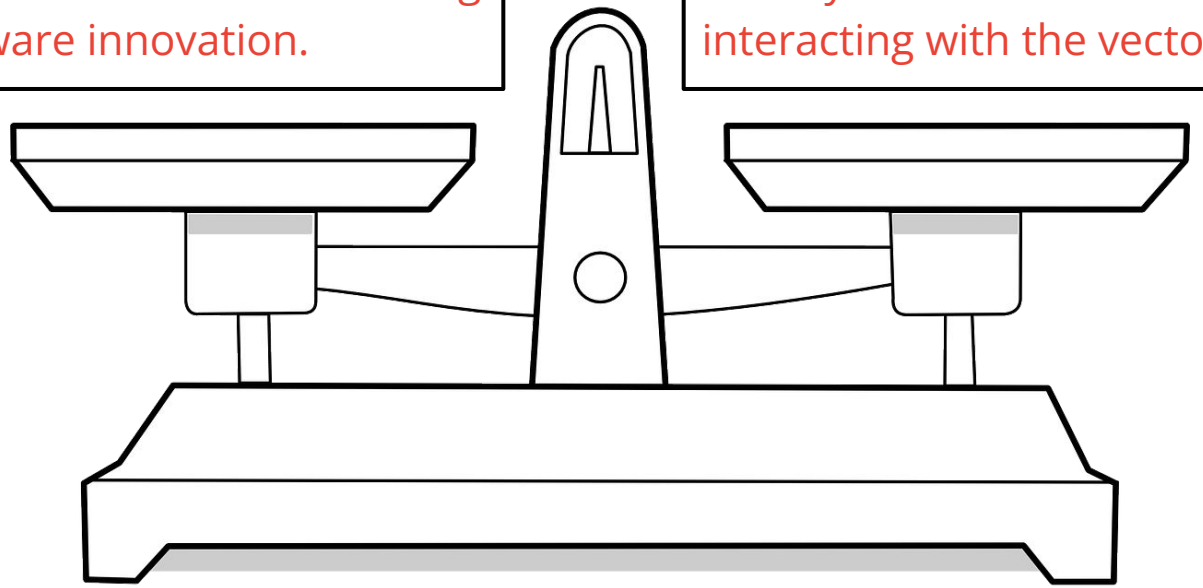
Fragmentation is okay as long as the interfaces are standardized.



RISC-V Needs to Strike a Balance

1) One "AI standard" (like RVV) that defines how AI workloads (e.g., matmul, embeddings) are written.
(+) The same binary runs on all RISC-V AI accelerators.
(–) Eliminates opportunities for **differentiation** + enabling new model capabilities with hardware innovation.

2) Restriction-free custom AI logic.
(+) Enables differentiation and innovation.
(–) Prevents **composing AI ideas and IP** (e.g., if every AI extension has a different way of interacting with the vector register file).



Takeaway: For AI, RISC-V's flexibility is the key advantage and the risk of baking assumptions into the ISA may sometimes be larger than the risk of fragmentation.

PART III
Reuse vs. Customizability

Reuse in AI Accelerators

The compiler can bridge the gap between software and hardware.

- However, standards are also needed to ensure **interoperability of IP** and allowing **mix+match-ing of functionality**.

Reuse happens at different levels of the design:

- Integrating **IP blocks** on the same chip (e.g., accelerator and general-purpose CPU).
- Integrating **custom AI functions** into the CPU (e.g., matrix multiply, activations).

Facilitating IP Integration

IP blocks should just plug into existing interconnect technology.

- Reuse network-on-chip, cache coherence, etc.
- May or may not need to be able to handle interrupts (the compiler should guarantee correctness).
- This is the integration model for **end-to-end AI accelerators**, **microcontrollers**, **general purpose compute on the accelerator SoC**.

RISC-V is already doing the right thing in this space; likely does not require anything special for AI, just remain compatible.

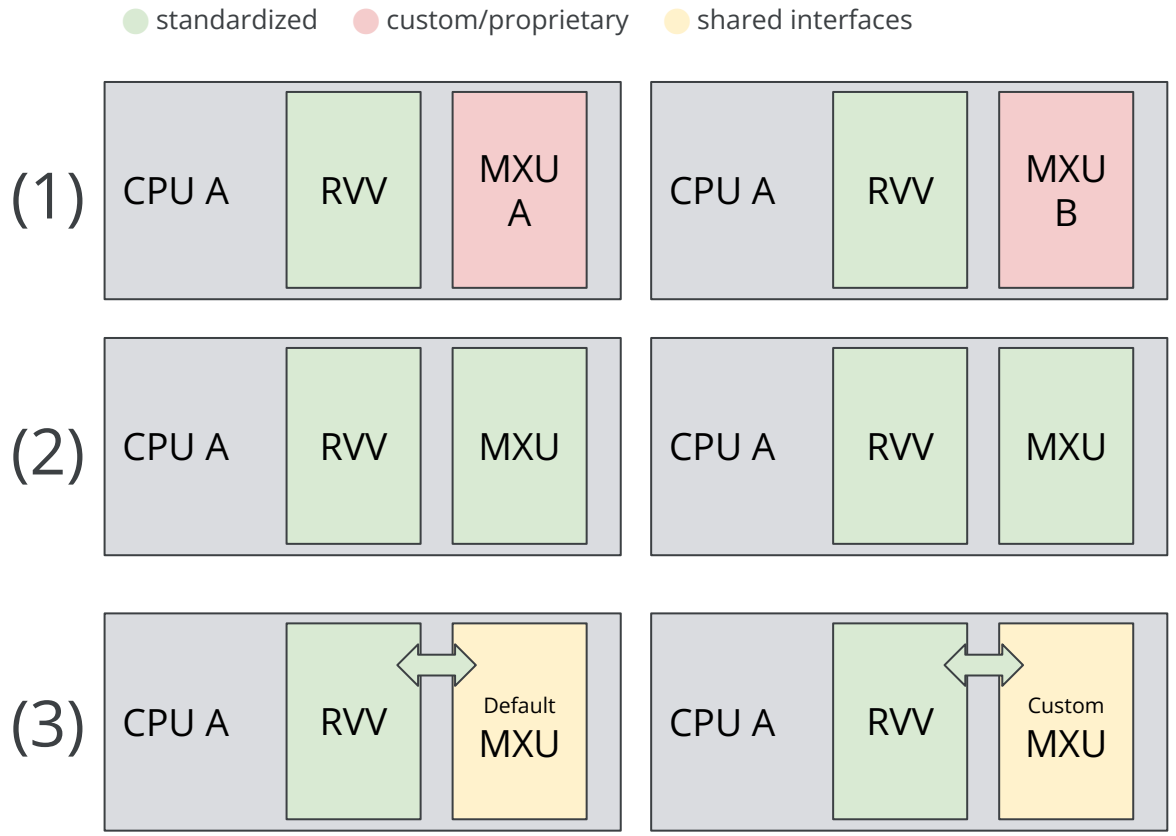
Custom AI Functions

There are three options:

1. Define **nothing** at all (today).
2. Add to RISC-V **standards** (like RVV).
3. Define a standard **interface** to enable customization (like VCIX).

Option 2 risks locking RISC-V into a particular paradigm.

- ISA extensions should be optional and not preclude customization.



e.g., matrix multiplication standards

Parallel Dispatch – A Missing Interface?

AI workloads often need to dispatch different work units to custom accelerator blocks/extensions in parallel.

- E.g., dispatching work to multiple MXUs.

In RISC-V, this currently does not have first class support.

- This may be a limitation in using RISC-V as a sequencer.
- Should RISC-V have an extension for this? (e.g., VLIW-like? Decoupled vector fetch? DMA-like interfaces?)

High Performance CPUs

A common view is that extensions are the biggest AI gap of RISC-V.

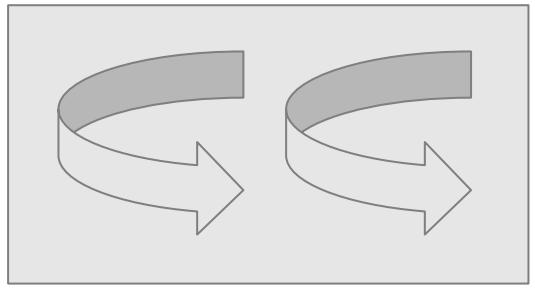
- We argue that the biggest gap is high-performance CPU IP with high-bandwidth interfaces to integrate custom extensions into the CPU (e.g., the vector register file).
- CPU needs to match throughput of the accelerator block.
- AI is becoming more dynamic and burstier, IPC matters a lot.
- AI workloads are increasingly mixing general purpose compute and AI computation.
- Multi-modal models shift this trade-off even further.

No obvious RISC-V contender vs. IP options in other ecosystems yet.

Takeaway: High-performance CPU IP that is customizable via standardized interfaces is more important than standardized AI extensions.

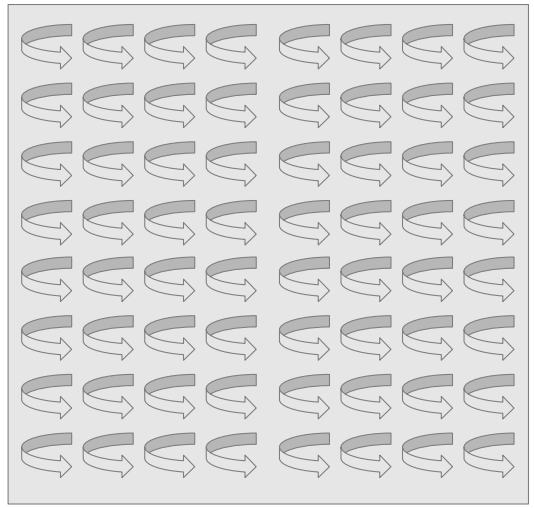
PART IV
Many Core vs. Single Core

Current Cores/Die Choices



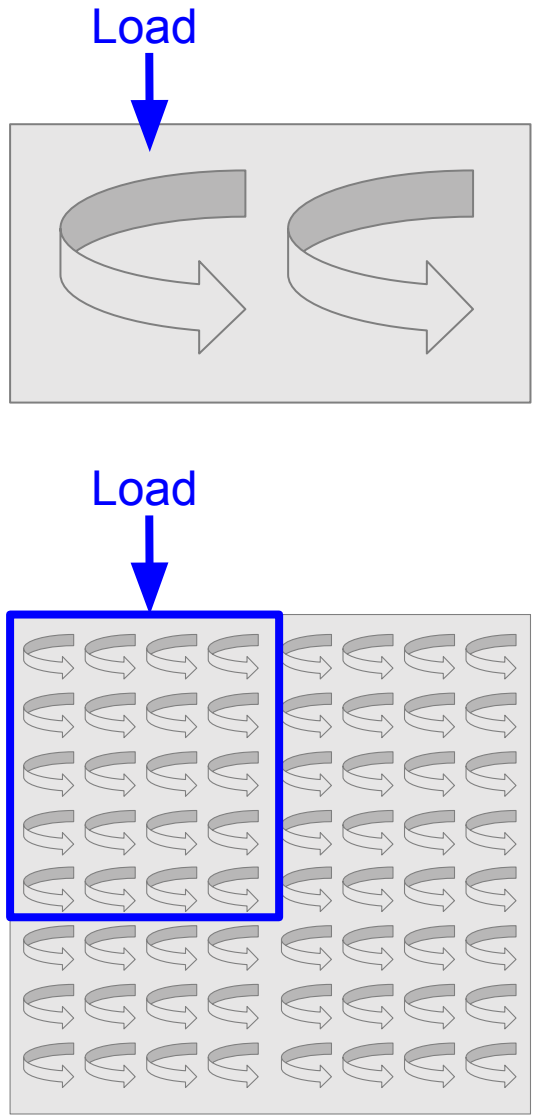
Different accelerators make different choices with regard to many-core vs. multi-core

- TPU: 1 or 2 cores per die
- MTIA: 64 PEs per die (2 RV cores per PE)
- GPU: >200 SMs per die



Cores/Die Implications

- Ease of programming: TPUv1 was single-thread
- What NOC width? CPU=64B; >1KB for TPUs
- Systolic reuse is your power/energy friend
  - Multicast and Reduce are done implicitly within MXU tile
  - But: short dims (e.g., batch=1) hurt utilization
- If manycore, can you reuse across cores?
  - Usual: get lucky in the cache
  - Some recent systems: “read multicast”



Takeaway: RISC-V ISA and IPs have natural strengths in the single-core and controller cases. With multicore, careful system design plays an increasingly important role.

PART V
Numerics Considerations

Lots of Numerical Formats!

fp32: Single-precision IEEE Floating Point Format [$1e^{-38}$,$3e^{38}$]

fp16: Half-precision IEEE Floating Point Format [$6e^{-8}$:65,504]

bfloat16: Brain Floating Point Format [$1e^{-38}$,$3e^{38}$]

E4M3 [$2e^{-3}$,448]

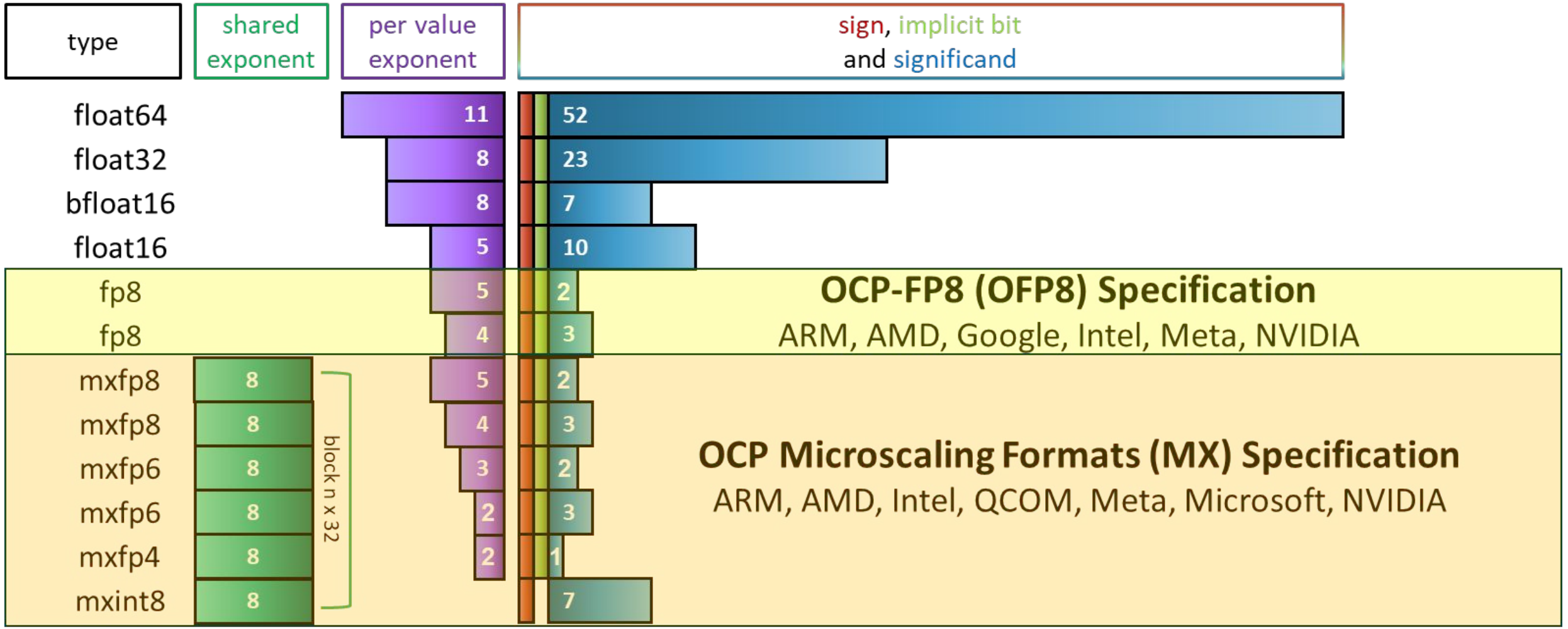
https://arxiv.org/pdf/2209.05433.pdf

E5M2 [$1.5e^{-5}$:57,344]

And more:

- See OCP/MX Alliance
- <8b per value
- microscaling

Previous and OCP Data Formats



RISC-V and Numerical Linear Algebra

- Select Element Width in the RISC-V Vector Extension doesn't go low enough and doesn't tell us enough.
- eXmY not enough—where are the scales, denorms, NaNs?
- Target is moving rapidly
  - Safe to standardize bf16, f16.
  - Must plan for weird futures in standardization.
- Scales, subchannel scales, and loss scales are all out-of-band
  - Where do they get stored in memory?
  - Do they mess up your nice tensor memory layout?

Takeaway: Numerics are a fast moving area, and RISC-V should anticipate a rapidly moving target.

PART VI
Current vs. Future Models

Components of Today's Models

Matmuls and their relatives:

- Fully connected layer
- Convolutional layer
- Attention (looks like a CAM to HW people: Query, Keys, and Values)

Everything else:

- Element-wise operations (LSTM gates)
- Embedding (sparse → dense, for recommenders)
- Pooling and Normalizations
- SoftMax
- Activation Functions (ReLU, sigmoid,...)

Are you sure that Attention is All You Need? Everything in Blue was in AlexNet (2012).

Algorithm and Model Innovation Continues

Promising Today:

- Increased context length
- Retrieval and retrieval-augmented generation (RAG)
- Multi-modal (text, audio, video)
- Reasoning
- Sparsity

Implications:

- Attention unlikely to be all we ever need.
- If you specialize to Transformer, what's your backup plan?
- What components might still be missing?

Takeaway: Algorithmic advancements continue. Some mechanisms (matrix multiplication, attention) will stick around, general-purpose compute preserves our options for the future.

PART VII
A Call to Action

A Call to Action

RISC-V should embrace enablement of shifting model paradigms with new requirements, as it makes it easy to adapt to these new paradigms.

- Need interfaces to integrate AI functions, without baking specific AI paradigms into the ISA (with some default, non-standard versions).
- Higher performance cores + IP are needed to enable agentic workloads and software implementations of novel ML paradigms before native support.
- Need well-established pathways to enable quick adoption of RISC-V in new settings without significant transition cost.
- The software ecosystem needs to ensures that new RISC-V accelerators can quickly be integrated into ML and traditional compilers.

Thank you!