SiFive

SiFive Intelligence & VCIX

September, 2022

AI HARDWARE SUMMIT

Outline

Krste Asanovic SiFive Co-Founder and Chief Architect, RISC-V Chairman of Board, UC Berkeley Professor

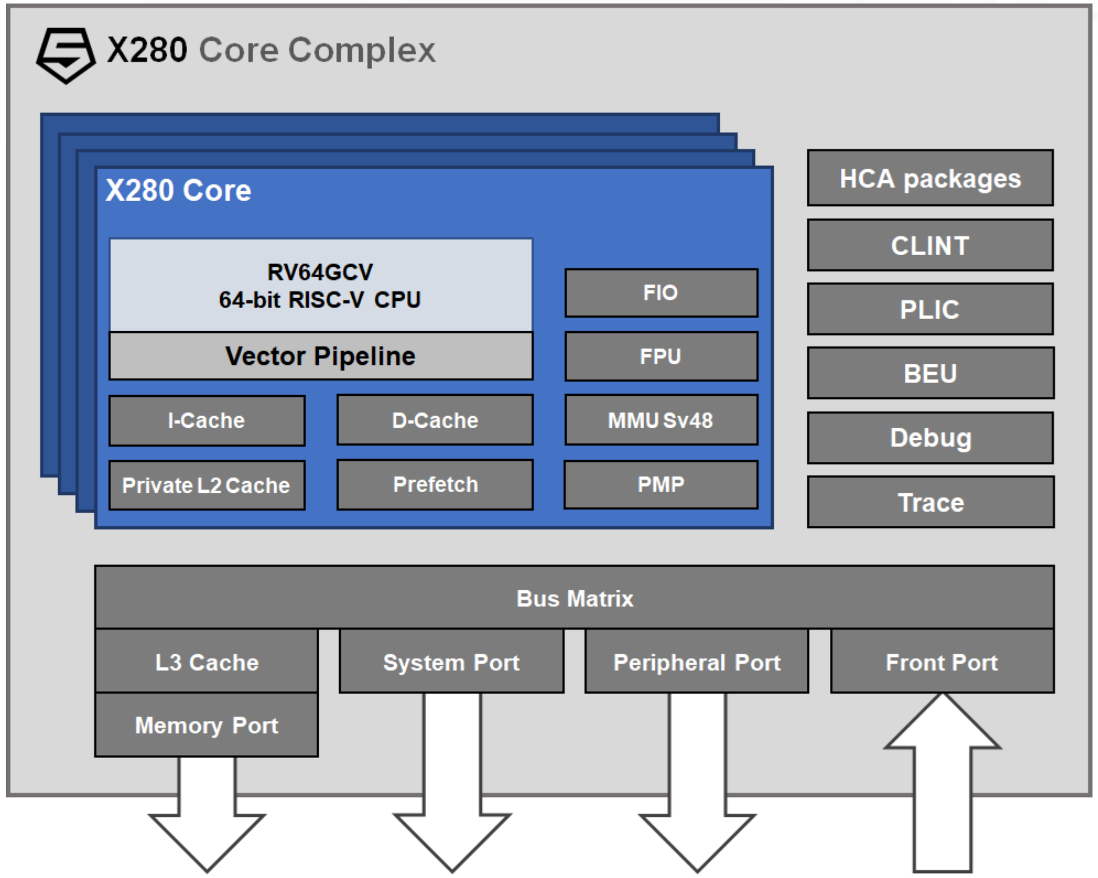
- SiFive Intelligence X280
- VCIX – Vector Coprocessor Interface
- RISC-V Toolchain Supports Scalar, Vector and Coprocessor Programming

Cliff Young TPU Architect, MLPerf Co-Founder, Google

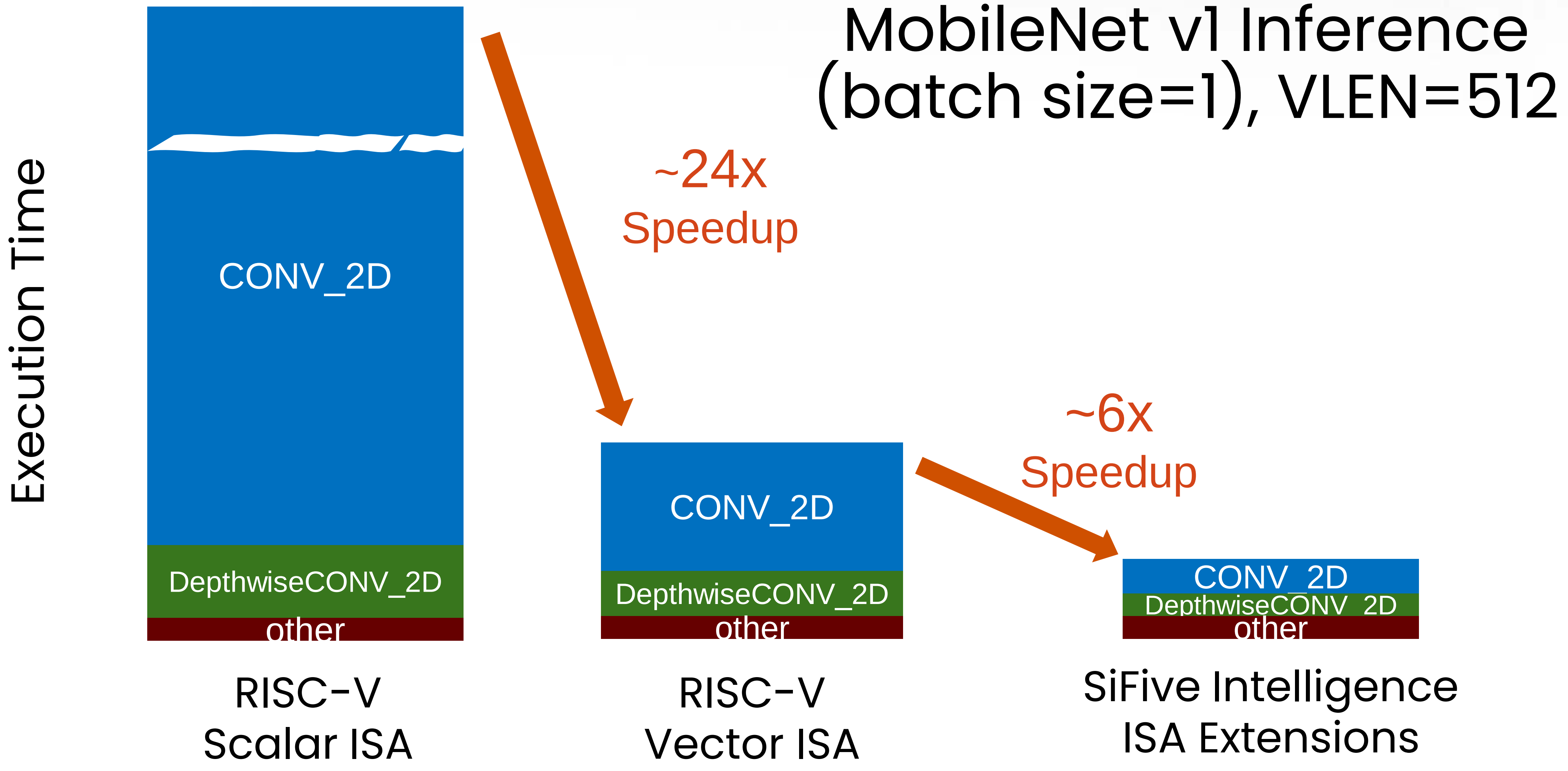
- How/Why X280 and VCIX built on Open RISC-V ISA can solve today's and future AI's Challenges

SiFive Intelligence X280

- RISC-V 64-bit scalar unit
  - 8-stage dual-issue in-order pipeline
- RISC-V vector unit with complete RVV v1.0 support
  - 32 x 512-bit vector registers
  - Up to 4096-bit vector operations (LMUL=8)
- SiFive Intelligence Extensions for AI/ML
  - Custom instructions accelerate critical AI/ML kernels
- Full Linux-capable applications processor (RVA22 profile)
  - Supports 48-bit virtual memory MMU (Sv48)
- Coherent multi-core configurations with up to 16 cores
- High-performance multi-level memory subsystem
  - Private L1 and L2 plus shared L3 for efficient data access
  - Stride prefetcher
- Performance
  - 5.7 CoreMarks/MHz   3.3 Dhrystone/MHz
  - 4.5 SpecINT2k6/GHz   3.4 SpecFP2k6/GHz (HiPerf config)



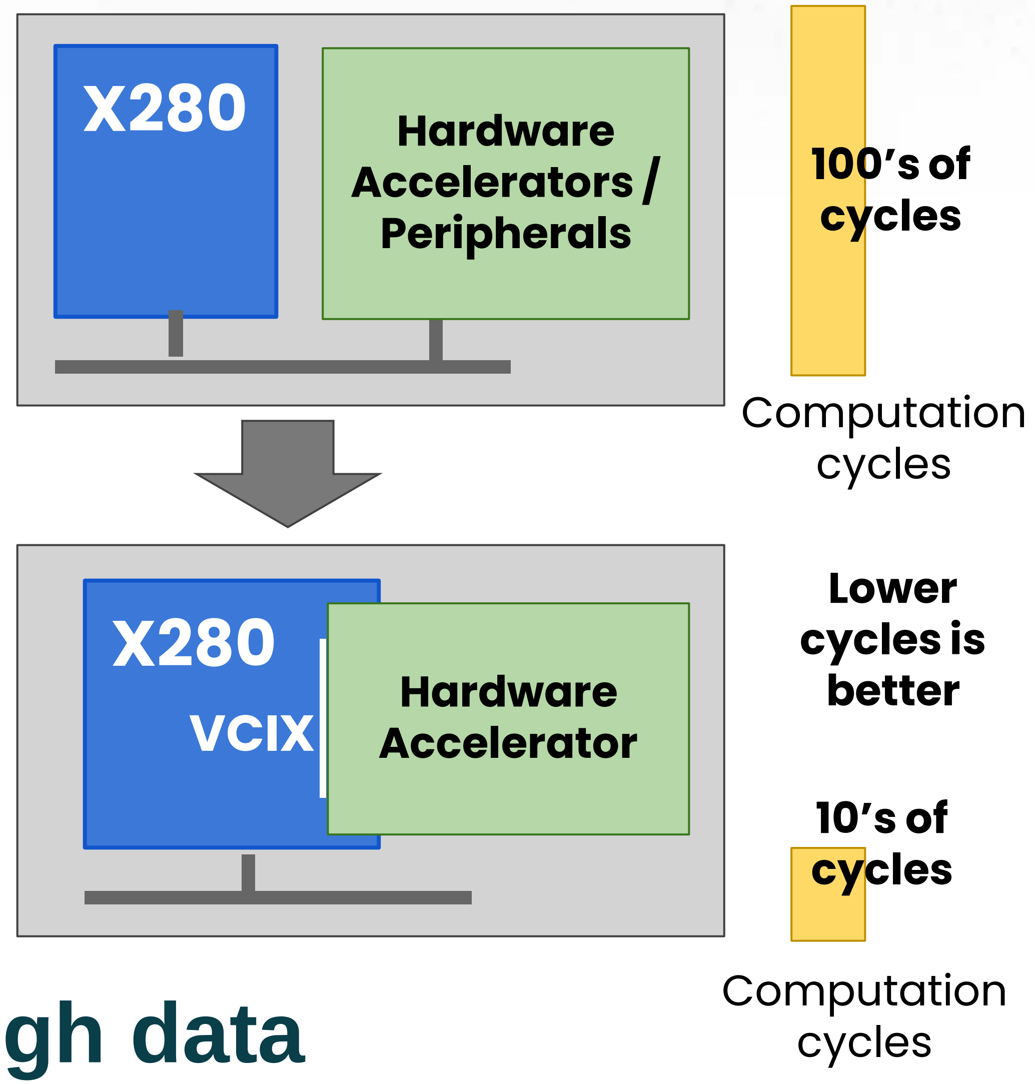
SiFive Intelligence: Accelerate end-to-end models



SiFive

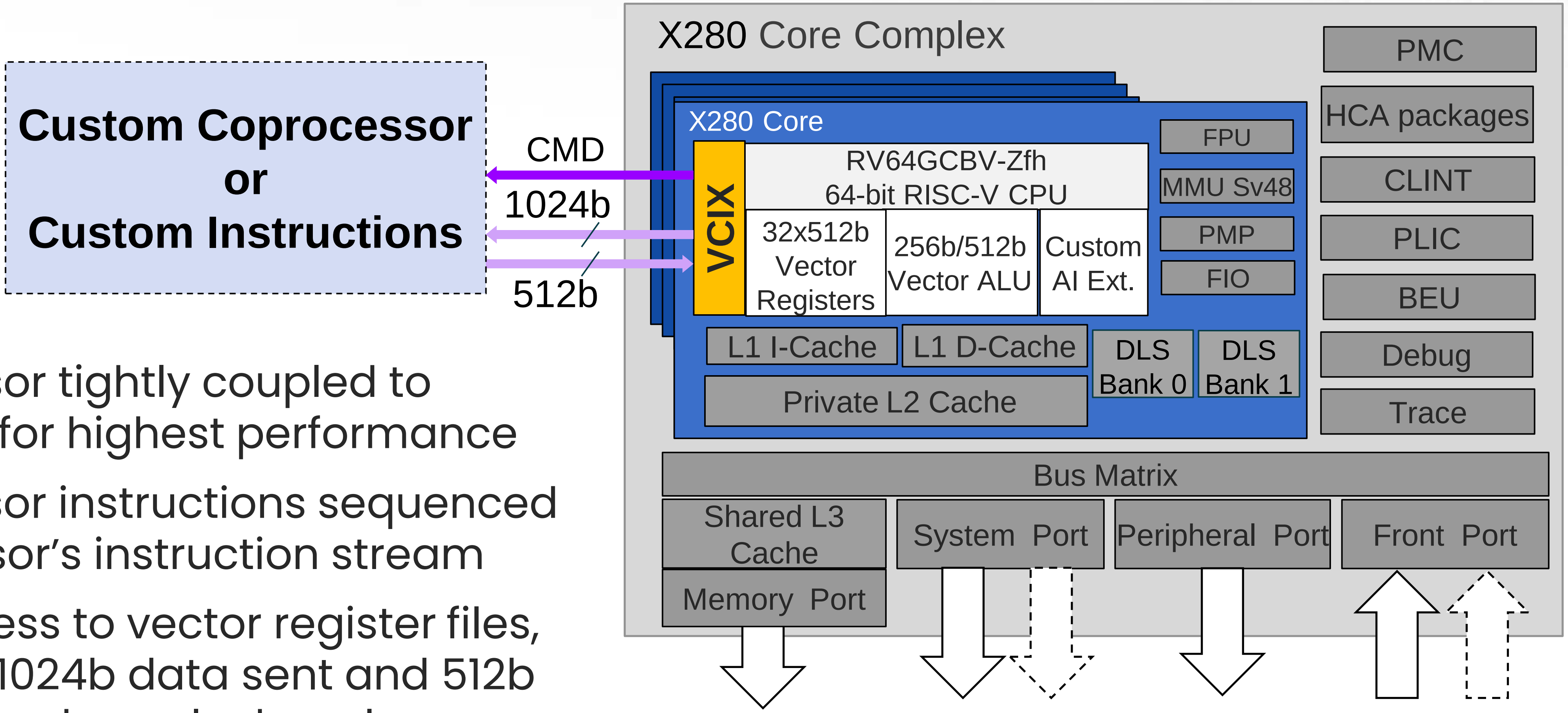
Vector Coprocessor Interface eXtension (VCIX)

- Strong demand for X280 coupled to hardware accelerators
- X280 "companion core" provides software and hardware "shell" for accelerator
- X280's benefit increased by bringing acceleration functionality into X280 core
- VCIX allows customers to easily add their **own vector instructions** and/or acceleration hardware to X280 vector processor
- Customers can greatly increase performance with custom instructions
  - FFT Butterfly, Matrix operations, Color Conversion, etc.



Increased performance, high data bandwidth, low latency, simpler software

Vector Custom Coprocesor Interface (VCIX)



- Coprocessor tightly coupled to processor for highest performance
- Coprocessor instructions sequenced by processor's instruction stream
- Direct access to vector register files, with up to 1024b data sent and 512b data returned per clock cycle

Bring Your Own Coprocessor (BYOC) into SiFive's Vector Machine while leveraging the entire RISC-V toolchain and software ecosystem!

SiFive X280 Vector Programming

Assembly

- Hand tune specific function using assembly
- Can mix intrinsics with inline assembly

Intrinsics

- Hand tune specific function using Intrinsics
- Easy to code using C-like program scheme
- Intrinsic names close to assembly mnemonics

Recode

- Migrate existing code to RISC-V vector, e.g., arm_neon.h
- Quick functional prototype
- 80/20 rule of converting majority of the existing code

Auto-Vectorizing Compiler

- LLVM compiler auto-vectorizes C code to vector instructions
- Rewrite C code based on what can be vectorized

Vector-Optimized Libraries

- Signal Processing
- Linear Algebra
- Nonlinear Functions
- Neural Networks
- Combinatorial Algorithms

Cliff Young

What's in a TPU (or accelerator)?

Compute, controlled by a VLIW sequencer

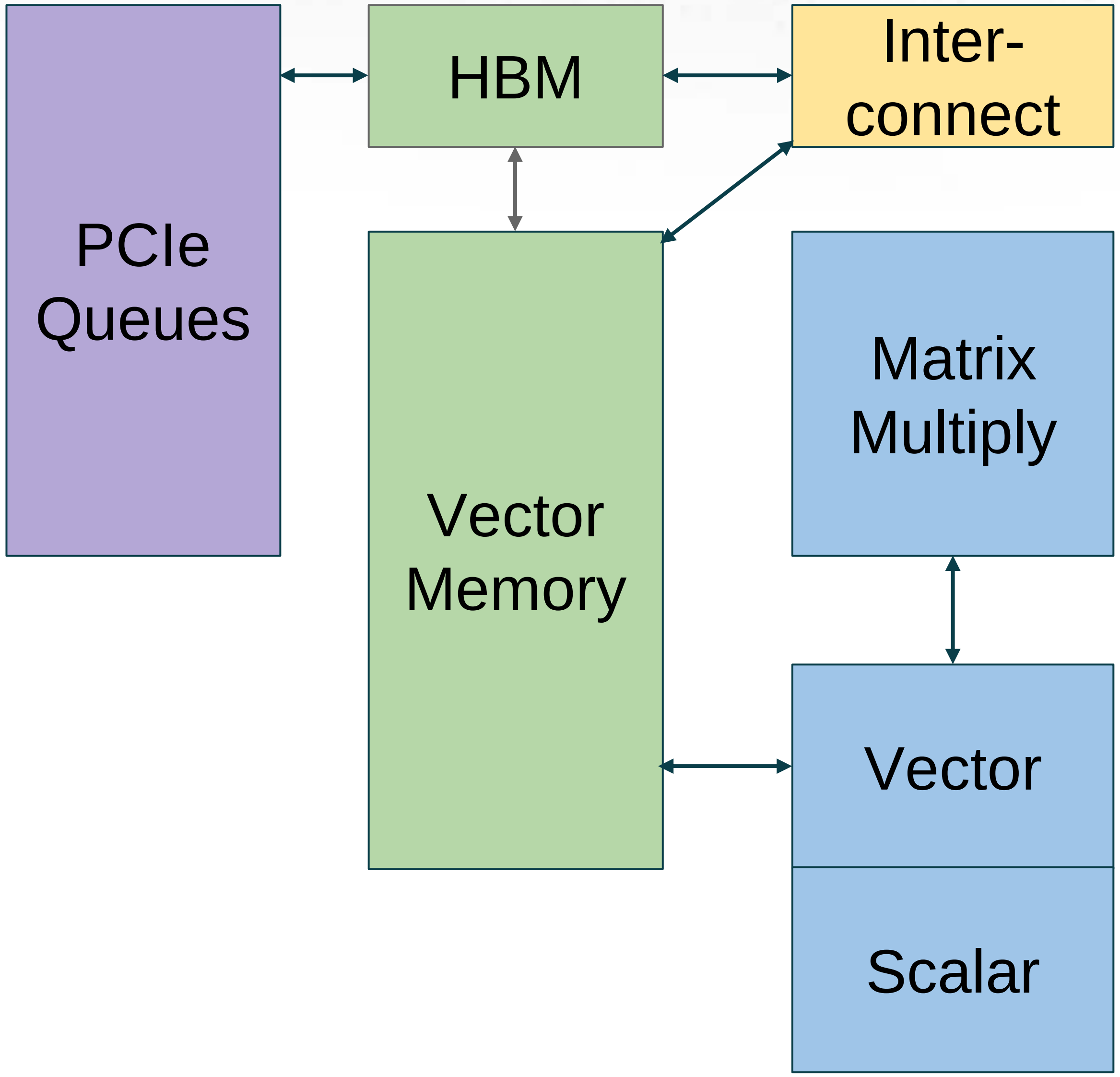
- Matrix Multiply Unit (Systolic Array)
- Vector Unit (1D, general, load/store)
- Scalar Unit (branches and addresses)

Memory

- On-chip SRAM
- Off-die but on-package HBM

Interconnect

- Inter-Chip Interconnect
- PCIe host interface



Adapted from "Google's Training Chips Revealed: TPUv2 and TPUv3", T. Norrie and N. Patil.

Cliff Young

What's in a TPU (or accelerator)?

Compute, controlled by a VLIW sequencer

- **Matrix Multiply Unit (Systolic Array)**
- Vector Unit (1D, general, load/store)
- Scalar Unit (branches and addresses)

Memory

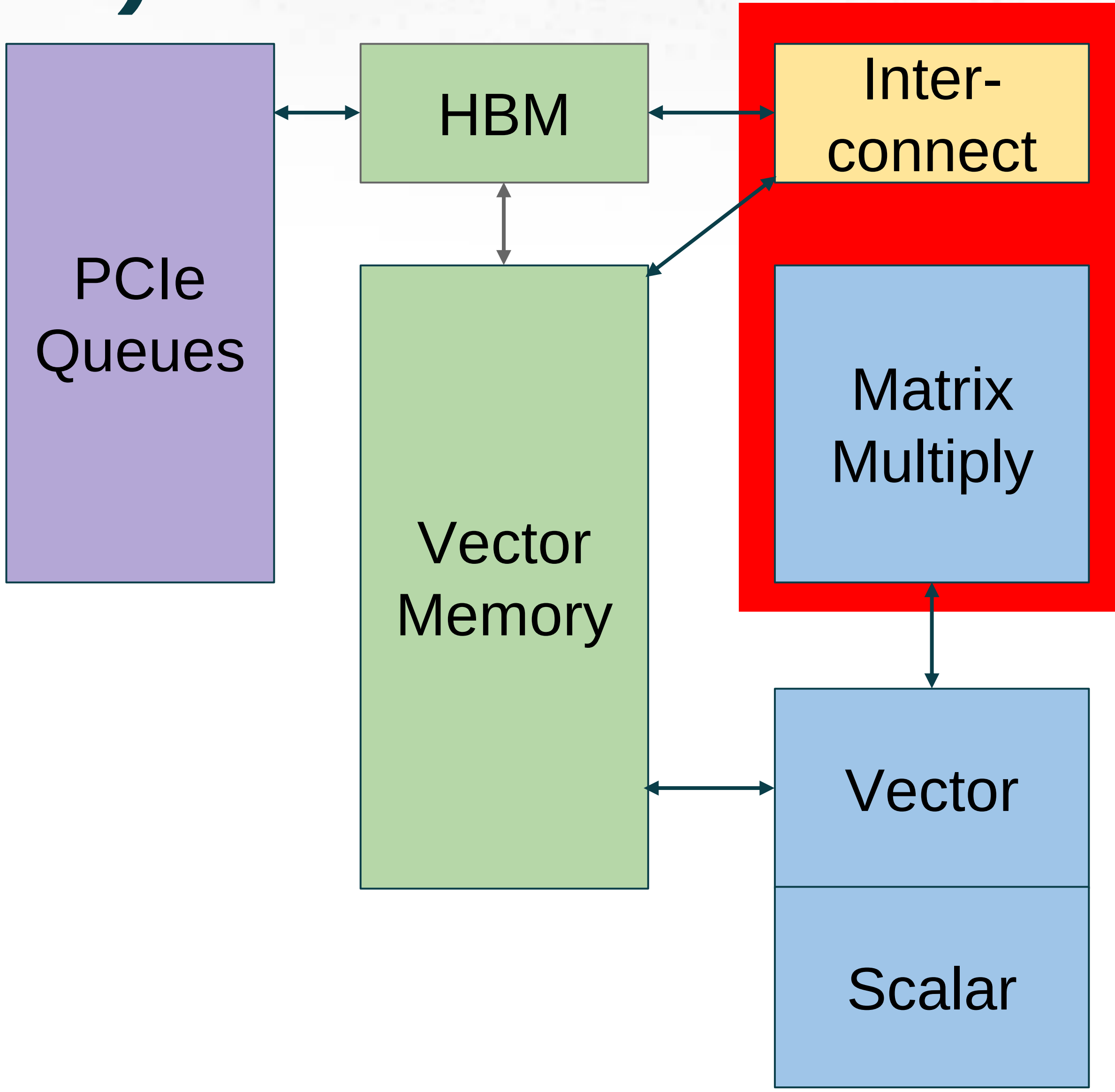
- On-chip SRAM
- Off-die but on-package HBM

Interconnect

- **Inter-Chip Interconnect**
- PCIe host interface

Only the **bold** items are unique to TPUs.

So why do we build them from scratch?



Adapted from "Google's Training Chips Revealed: TPUv2 and TPUv3", T. Norrie and N. Patil.

VCIX: An Elegant Division of Labor

Challenge: Can we combine a general-purpose core with a systolic matrix multiplier?

SiFive builds X280 with VCIX

- X280 is the "base" core
  - Integrated 64b scalar with sequencer
  - 512b vector unit, path to memory
  - SiFive doesn't have to mod the decoders
- RISC-V Standard RVV extensions
  - Future-proof vector programming code
- A single software toolchain
  - Write code in C/C++, assembly, etc.
- Well-defined VCIX interface
  - Push/pop vector instructions
  - Rich set allows overlay of functions
  - Single-cycle interface (no long-latency)
  - No exceptions or errors

Google focuses on the MXU

- VCIX supports familiar push/pop interface
  - Long latency through MXU handled by Google Software and Compilers
  - TPU SW stack already used to this model
- Tight coupling between CPU and MXU
  - Hard to beat vector register-level access
  - Handful of cycles, instead of PCIe latency
  - Base core runs in parallel with MXU
  - Generality and power in one system
- Programming model is simple:
  - One program with scalar, vector, and coprocessor instructions interleaved

SiFive

Thank you

SIFIVE.COM