EE290C/EE194 SP24: Tapeouts in Intel 16

05/20/2024

Ryan Ma, Darwin Zhang, Jessica Fan, Jonathan Wang, Nico Castaneda,
Kevin He, Minh Nguyen, Ethan Gao, Naichen Zhao

Agenda

- Introduction and Overview
- BearlyML24
- DSP24
- Questions

EE290C/EE194 SP24 Overview

Timeline: From 0 to Tapeout in 15 Weeks

Week 1 1/16	TMA1 3/15	TMA2 4/17	Tapeout 5/10
Start	TMA1	TMA2	May 3
The SoC/Chisel/ VLSI Bootcamp	**First Mock Tapeout**	**Preliminary Review**	**Tapeout!**
Research papers Chipyard "Labs" A generic chip	Design freeze Implementation Flow debugging	Design reviews RTL freeze DRCs & sims	Integration feedback Final corrections Send design to Intel

Borivoje Nikolić

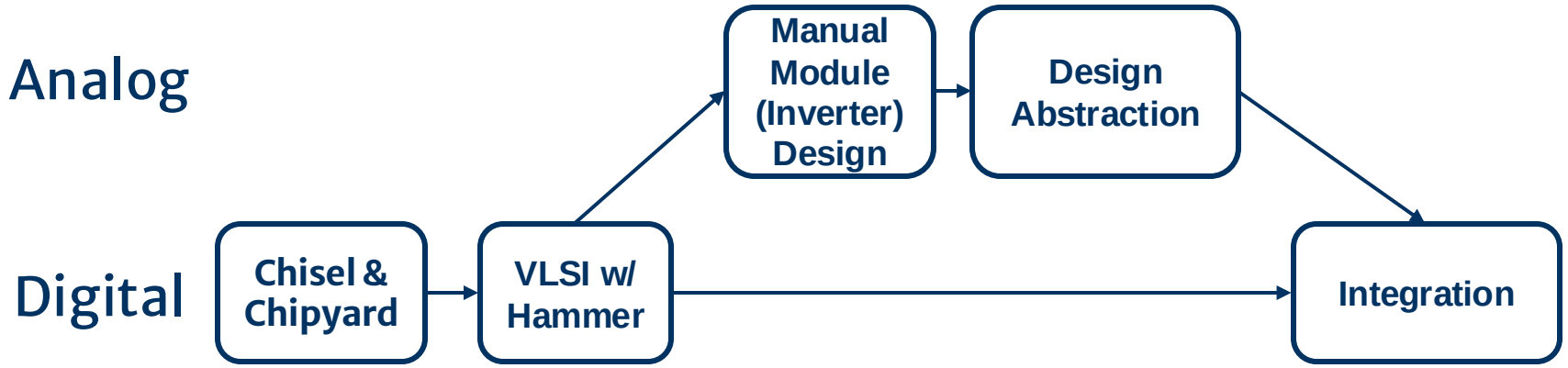
Kristofer Pister

~ 45 students*

*excluding our third chip, SCuM-V

Class Improvement: Revised Lab Content

- While students eventually specialize in parts of the chip-design process, an overarching goal for this iteration of the class is to ensure that all students are exposed to every step in the chip design process through the lab material
- As part of the revised lab content for the class, a completely new set of labs was developed



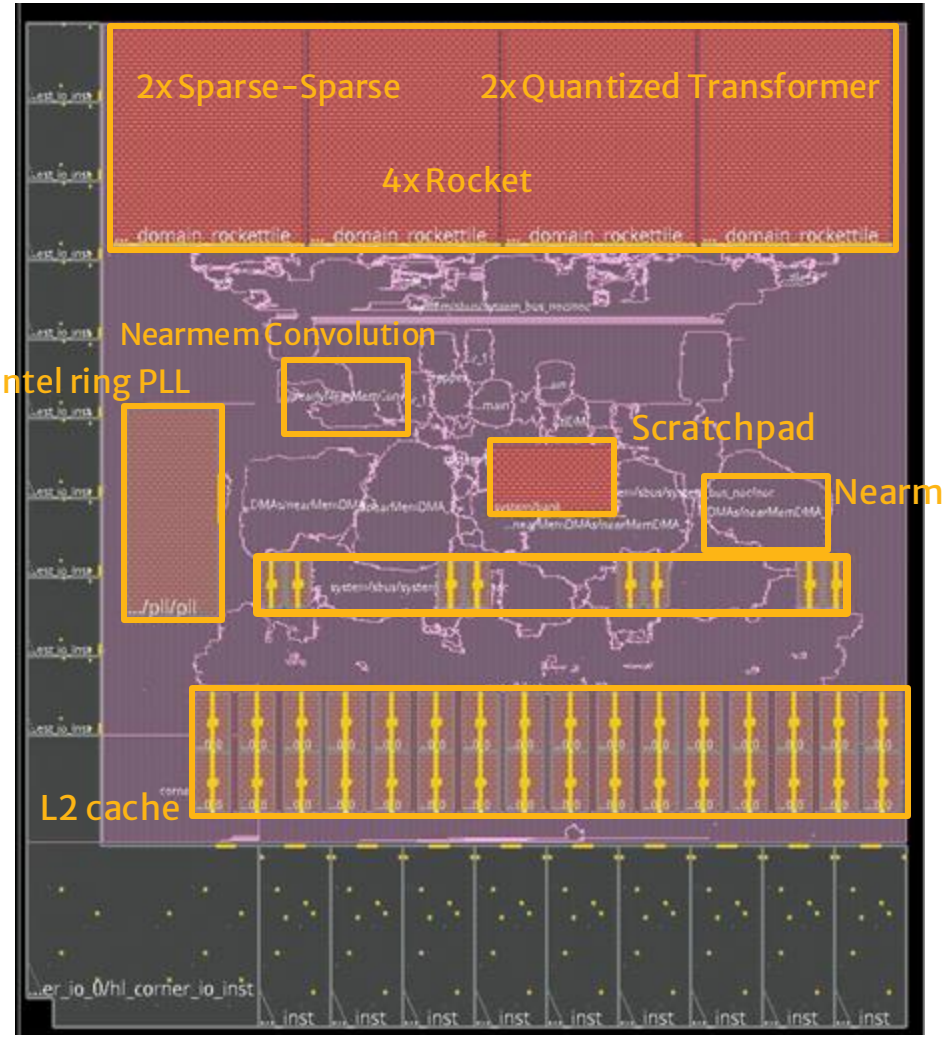
Two Digital Chips: BearlyML & DSP

Changes from Prior Iteration

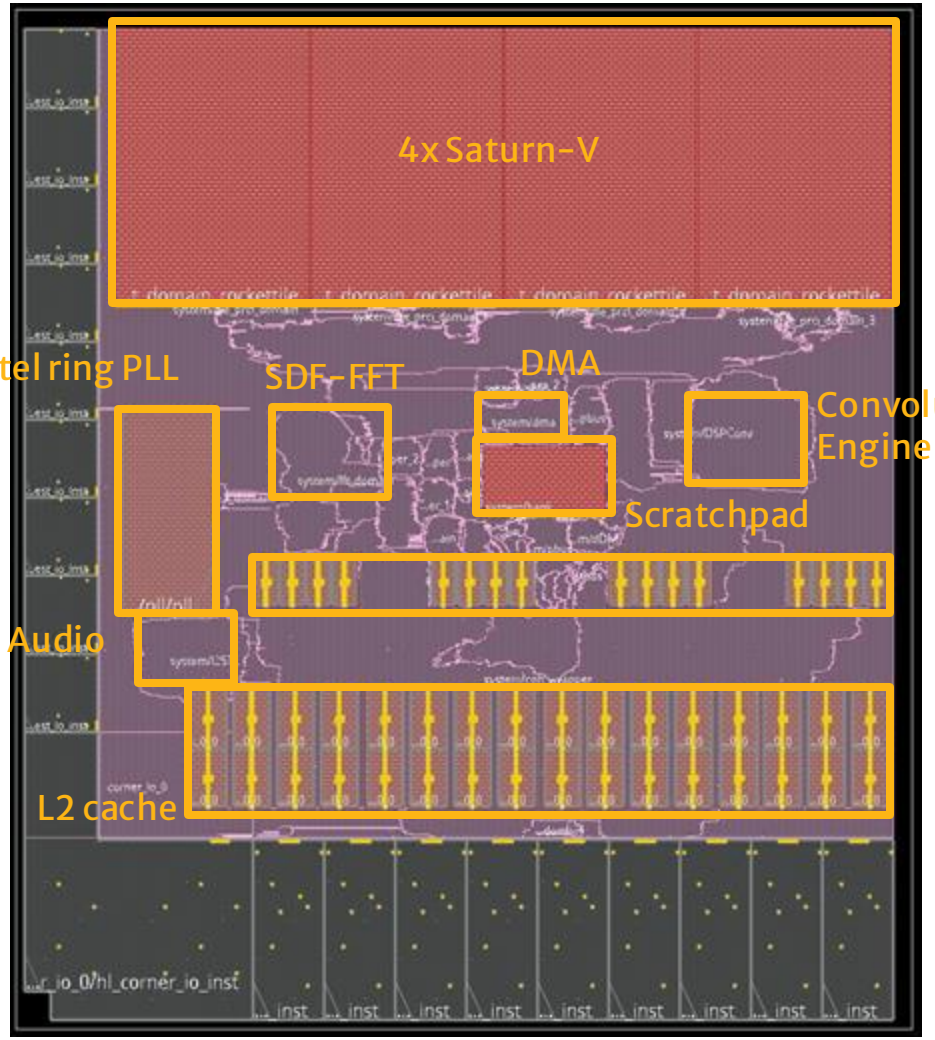
BearlyML'23	BearlyML'24	DSP'24
4 Rocket Cores	4 Rocket Cores	4 Saturn Vector Cores
Sparse Matrix Accel.	Sparse-Sparse Accel.	Convolution Accel.
Prefetcher	Convolution Engine	SDF-FFT Accel.
DMA Near Memory	Quantized Transformer	Audio I²S Interface
Scratchpad	NearMem MAC	General DMA
	Scratchpad	Scratchpad

Two Digital Chips: BearlyML & DSP

BearlyML'24



DSP'24

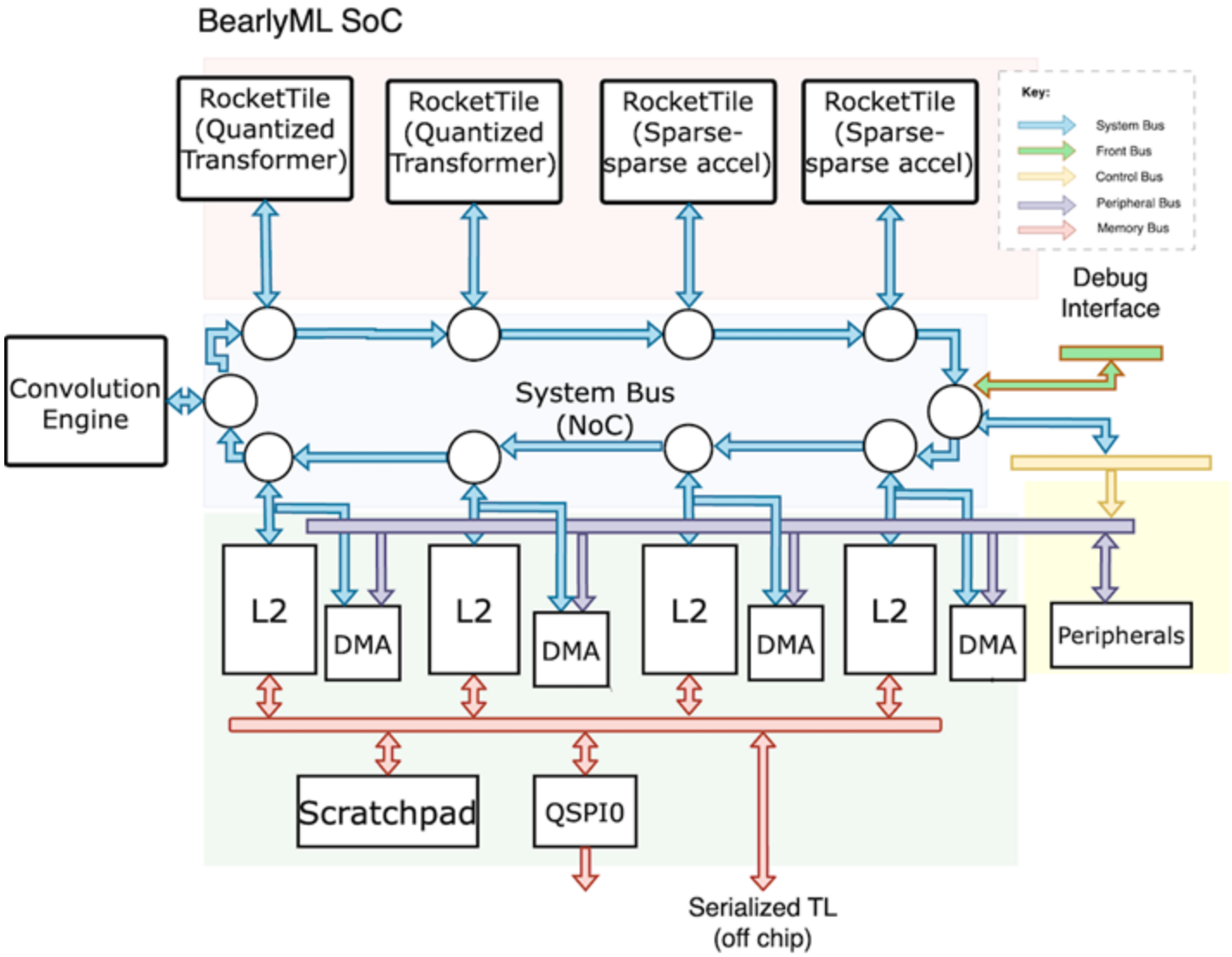


Agenda

- Introduction and Overview
- BearlyML
- DSP
- Questions

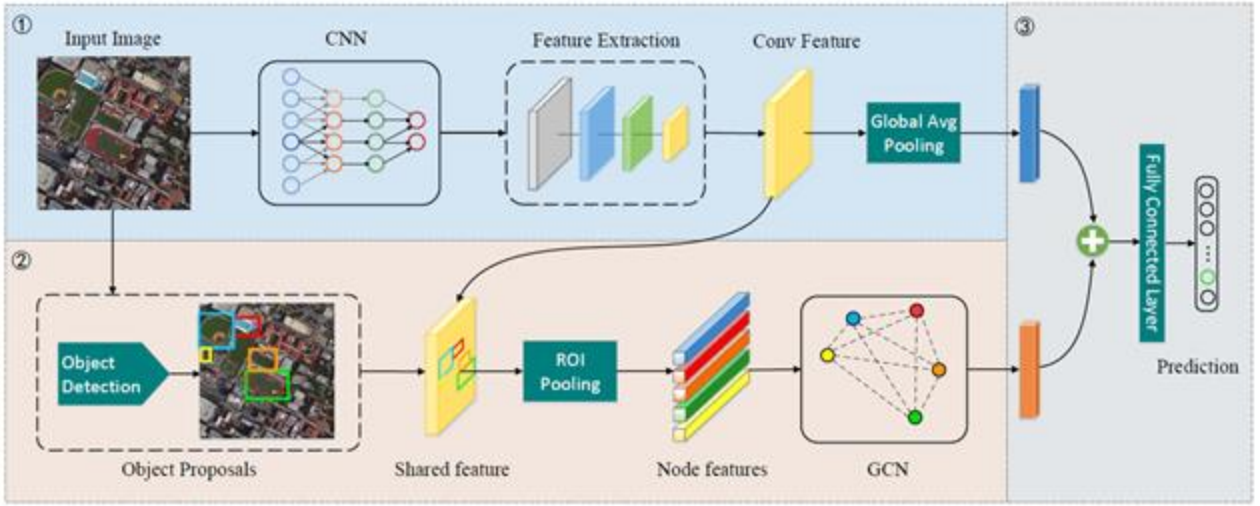
BearlyML Overview

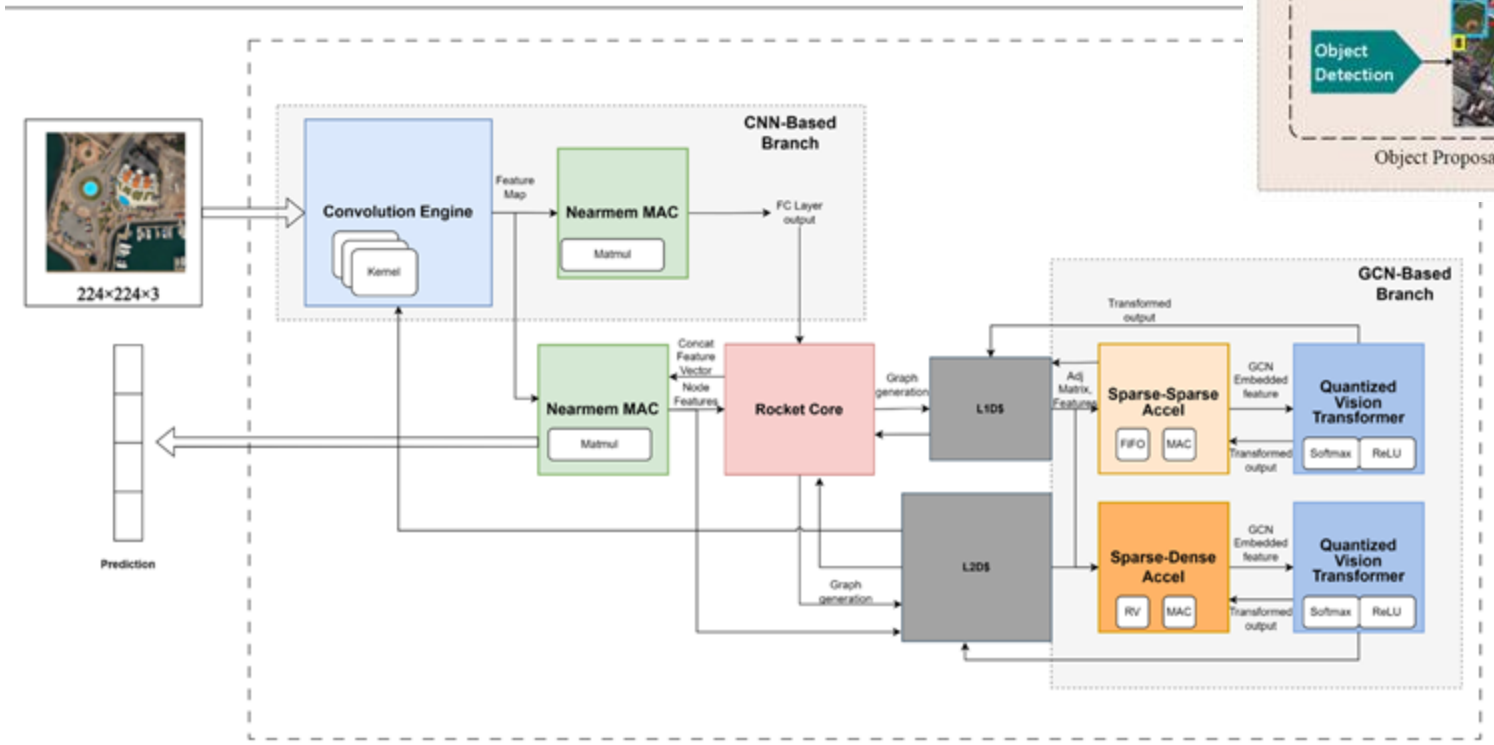
- Designed for ML applications
- Added features:
  - Nearmem MAC
  - Nearmem convolution
  - 256 KiB L2 cache
  - 32 KiB scratchpad
- 4x Rocket Tiles
  - 2 with Quantized Transformer
  - 2 with Sparse-Sparse matmul accelerator
- Buses widths expansion
  - 128-bit NoC by Constellation



BearlyML Applications

Interesting applications such as satellite imagery classification that fuse GCNs with CNNs (Ensemble models) can be run on BearlyML





GCN Layer

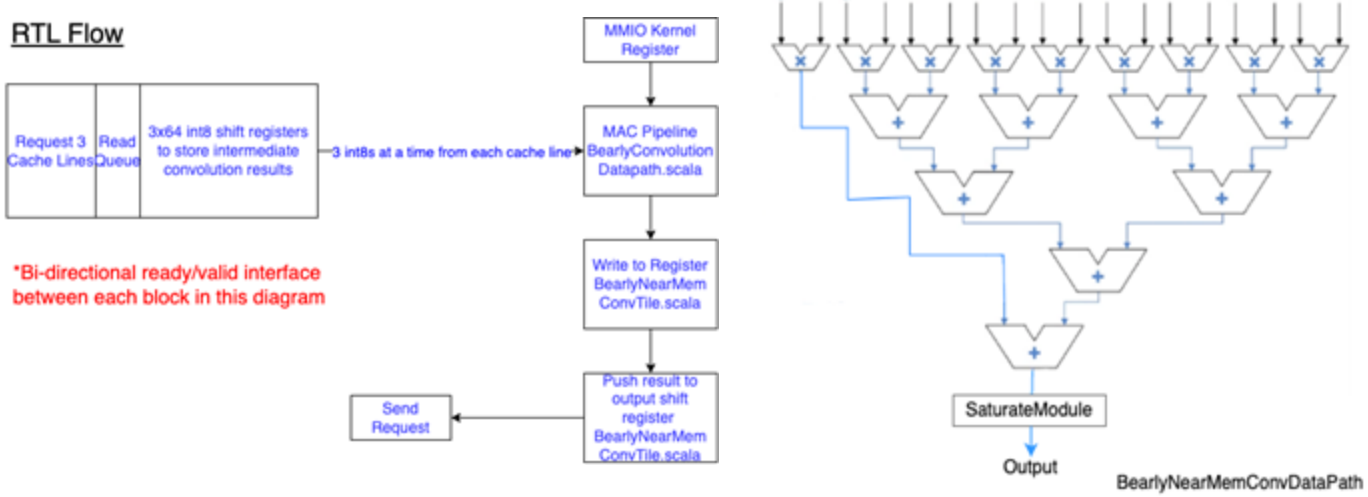
$$Z = f(X, A) = \mathrm{softmax}\left(\hat{A}\,\mathrm{ReLU}\left(\hat{A}XW^{(0)}\right)W^{(1)}\right)$$

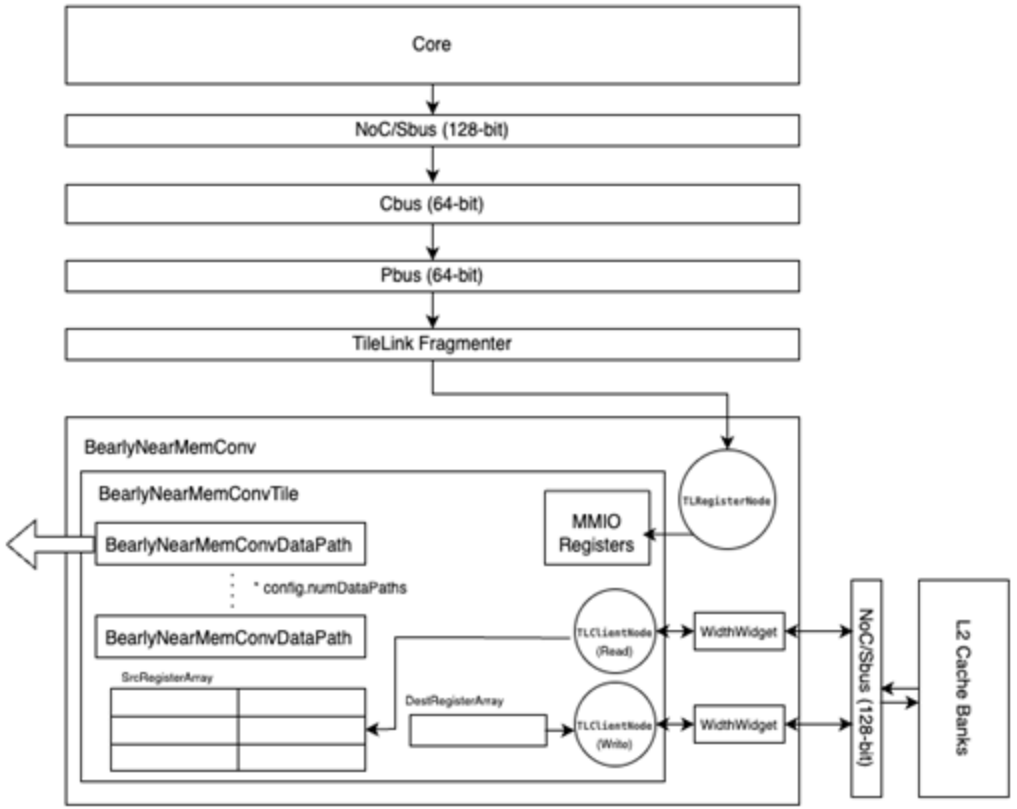
Feature vector matrix (*NxC*)
Trainable weights (*HxF*)
Trainable weights (*CxH*)
First layer
Scaled adjacency matrix (*NxN*)

Liang, Deng, Zeng, *A Deep Neural Network Combined CNN and GCN for Remote Sensing Scene Classification* (https://ieeexplore.ieee.org/document/9149910)

Convolution Engine

- Memory-mapped I/O (MMIO) convolution accelerator
- MMIO registers store image address, image height, image width, kernel, and status
- 3 x 3 kernel used for the convolution computations
- INT8 datatype for all matrices



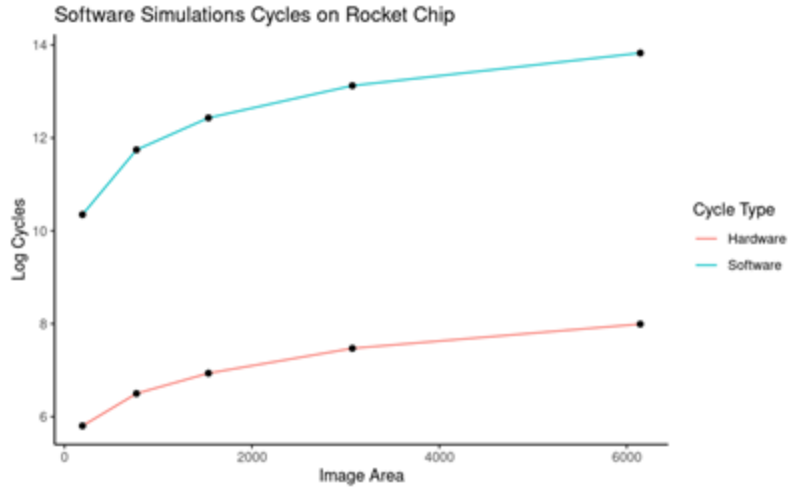


TileLink

- Enables L2 cache memory requests
- Ready/Valid protocol for accelerator status

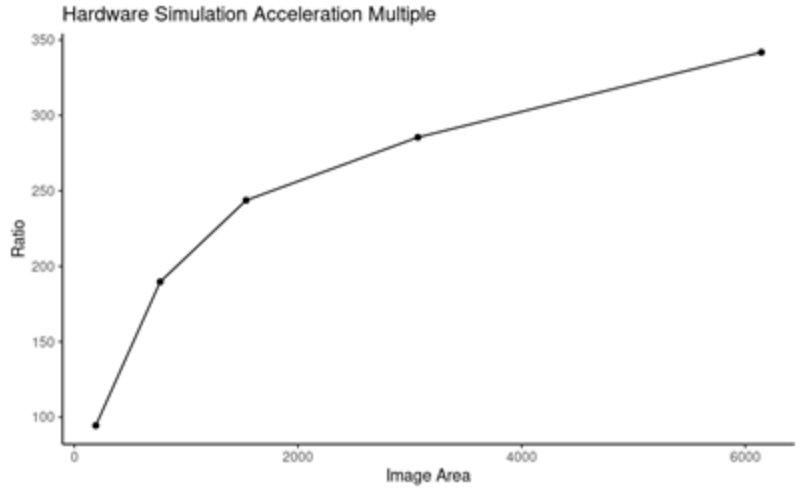
Convolution Engine

- On average, engine enables a minimum 95x convolution speedup
- As the image area increases, engine's MAC pipeline enables us to rapidly perform core accumulation faster, leading to speedup



Future Improvements

- Experiment with RTL and evaluation pipeline to figure out tradeoff between adding more processing capability to MAC pipeline vs. software speedup
- Currently require input matrix via MMIO to be cache width aligned; find a way to ease this restraint and pad non-aligned input images appropriately

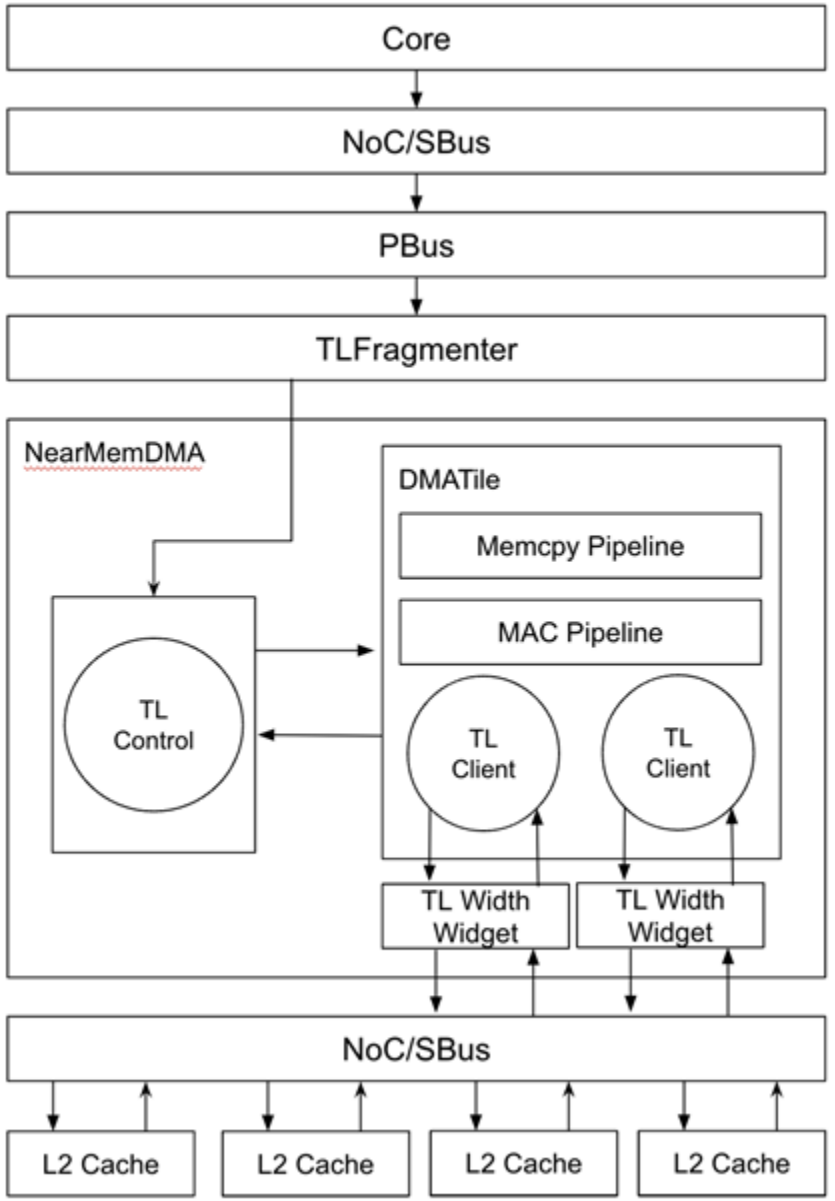


NearMem MAC

- MAC is on the DMA Engine, contains a direct pipeline to memory that increases the speed of our memory accesses
- Each DMA Tile has faster read/write to its corresponding L2 Cache, but not to the other 3

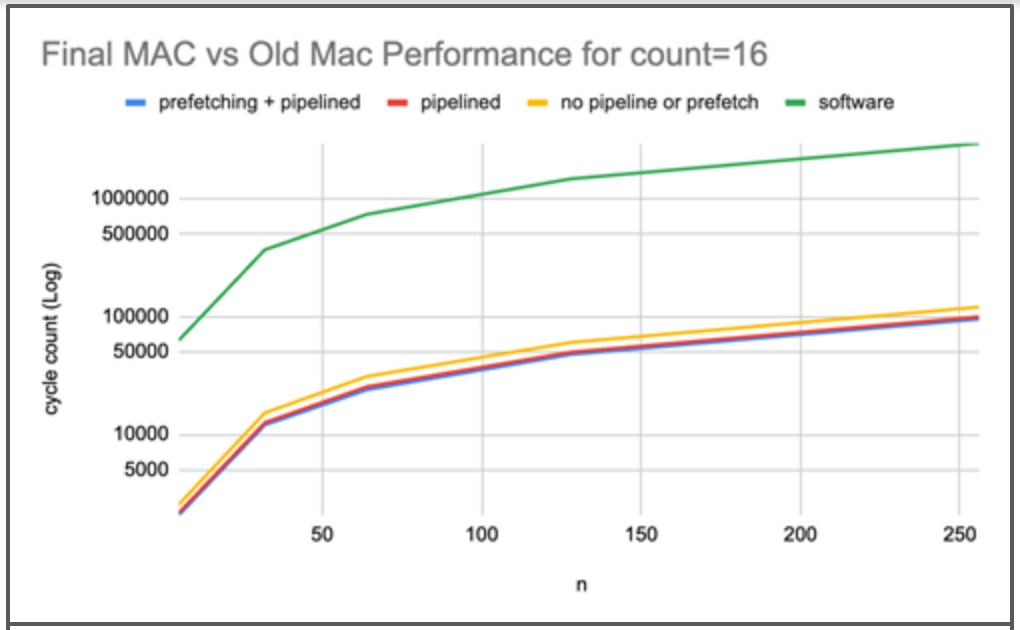
Interfaces:

- ReadPort
  - Read from memory using srcAddr, incremented by srcStride, count times
- ReadWritePort
  - For memcpy, after copying from srcAddr, write to destAddr
  - For MAC, after multiplying with operandRegister, write to destinationRegister

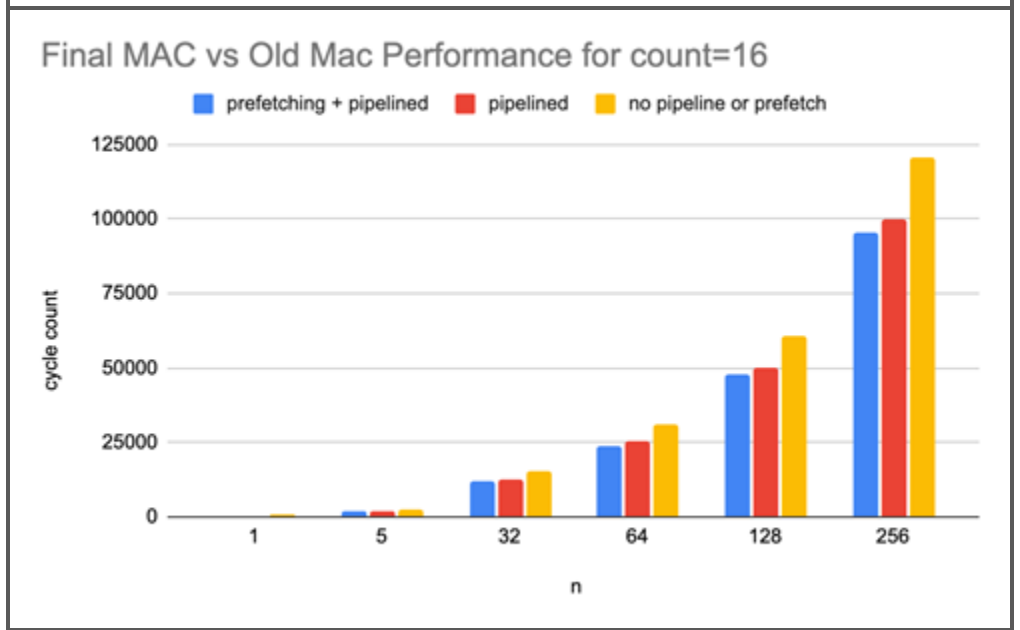


NearMem MAC

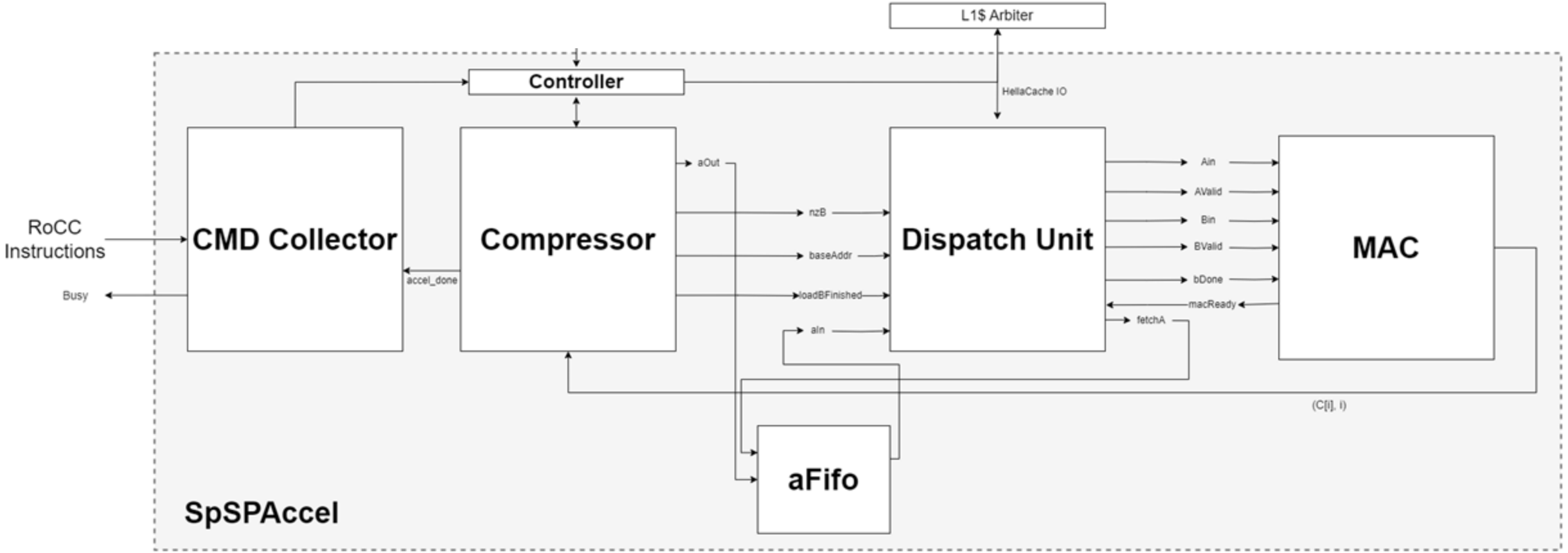
- ~10-32x speed-up for same L2 cache hits from single DMA tile
- ~15-85x speed-up for different L2 cache hits from single DMA tile



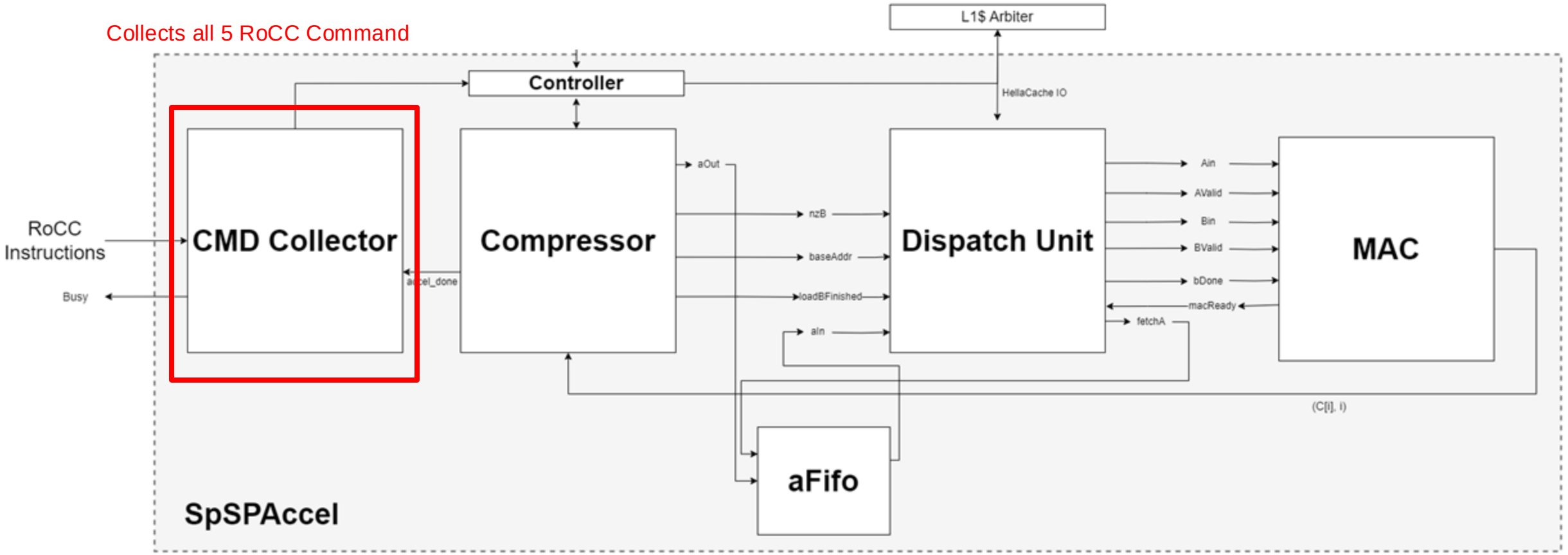
- Found minimal improvements in cycle count for prefetching compared to no prefetching
- If more area is allowed, could show more improvement in prefetching
- Conclusion: Performance improvement not worth the area increase for prefetching



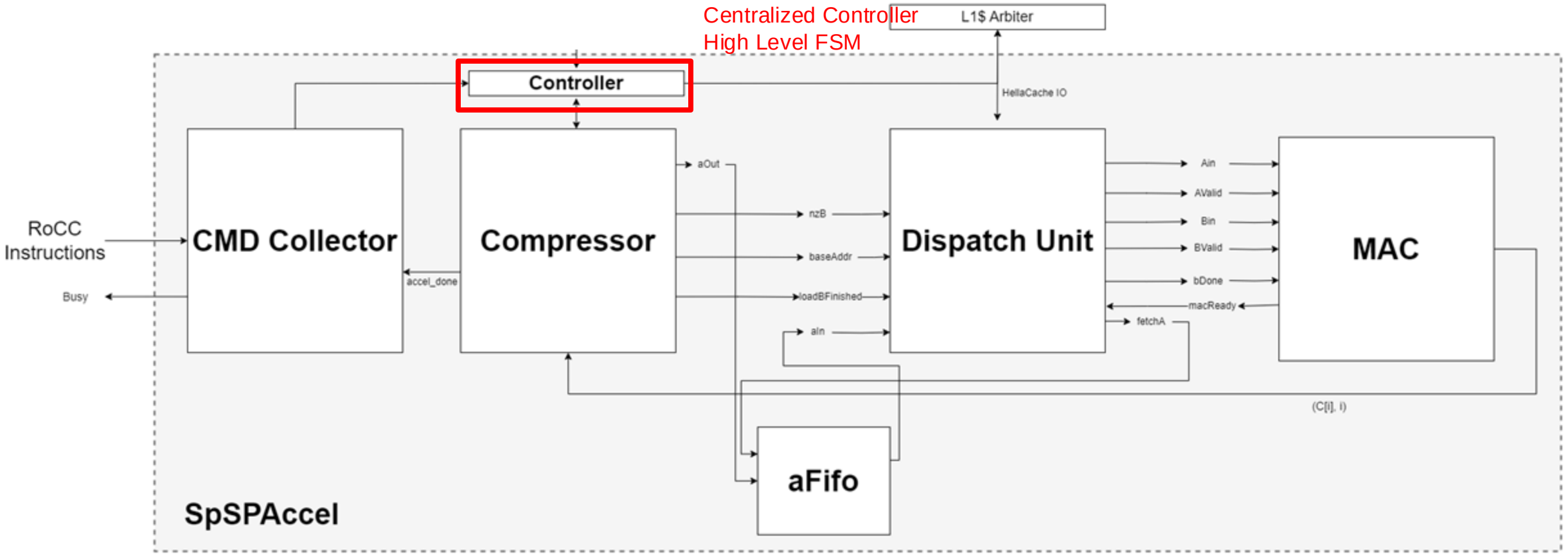
Sparse-Sparse Accelerator: Architecture



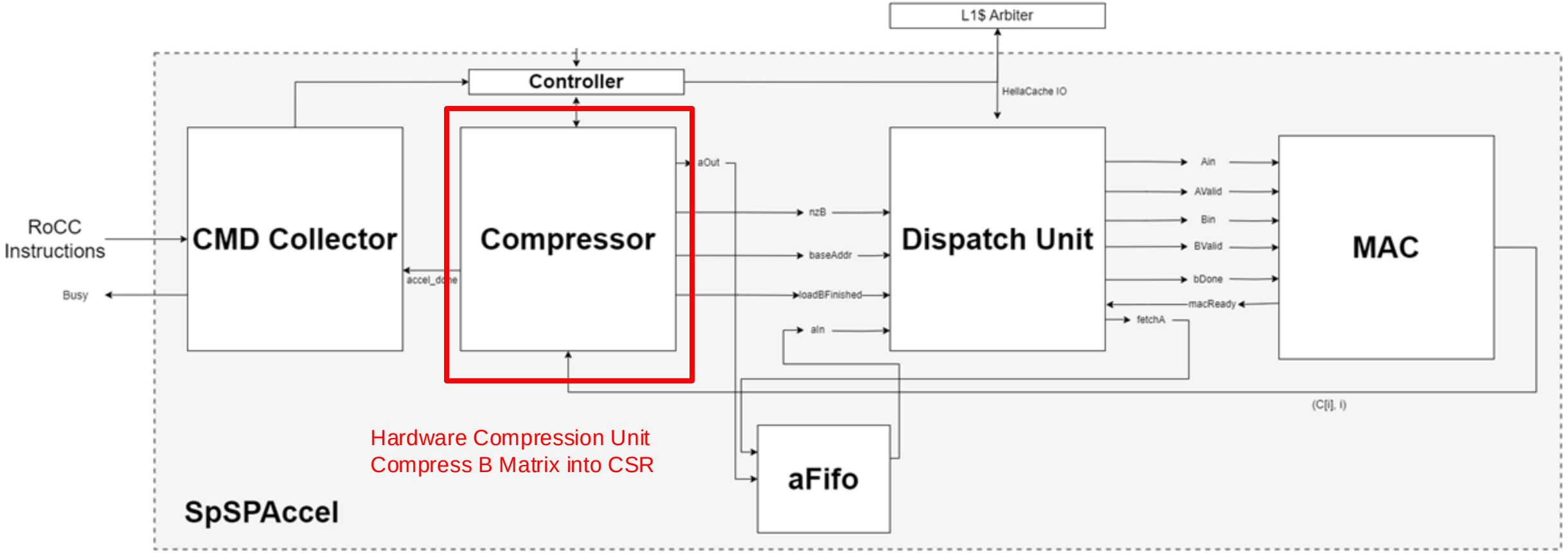
Sparse-Sparse Accelerator: Architecture



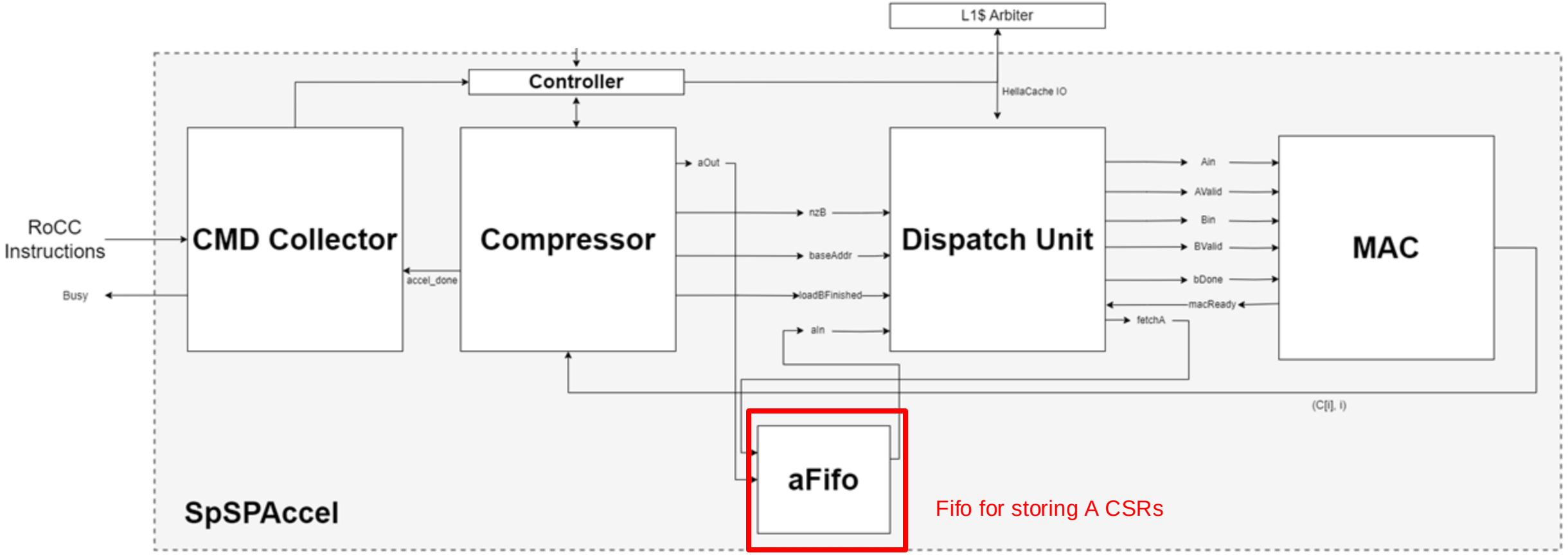
Sparse-Sparse Accelerator: Architecture



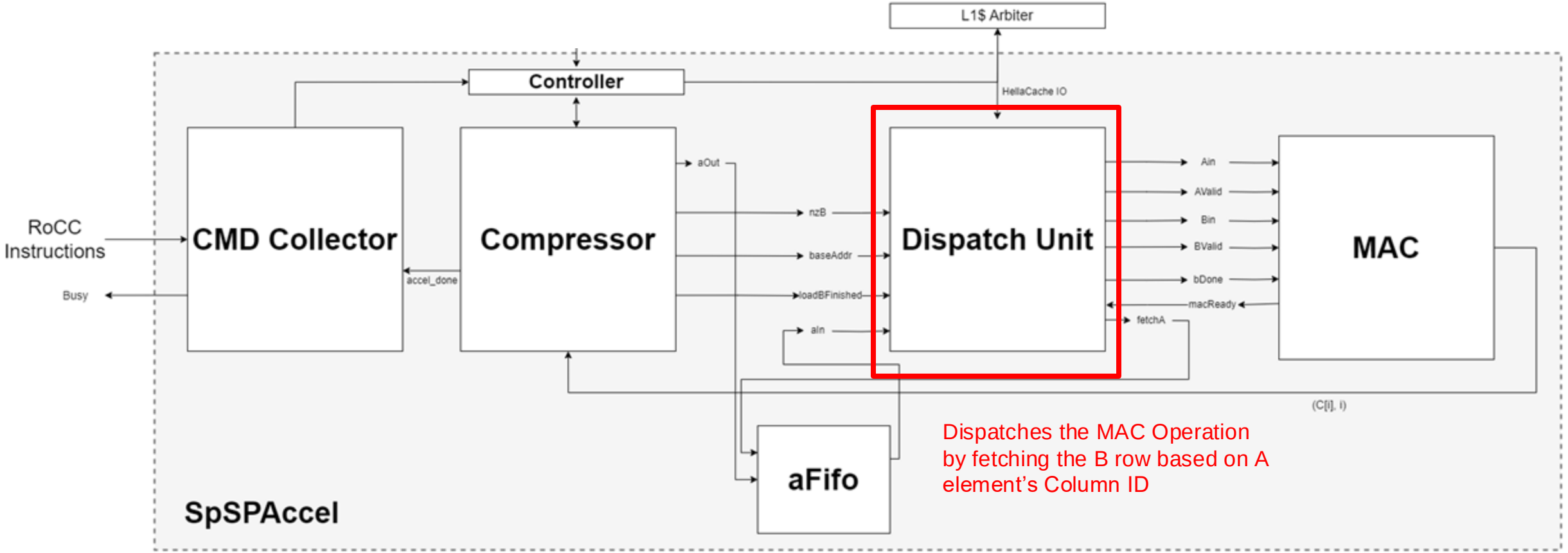
Sparse-Sparse Accelerator: Architecture



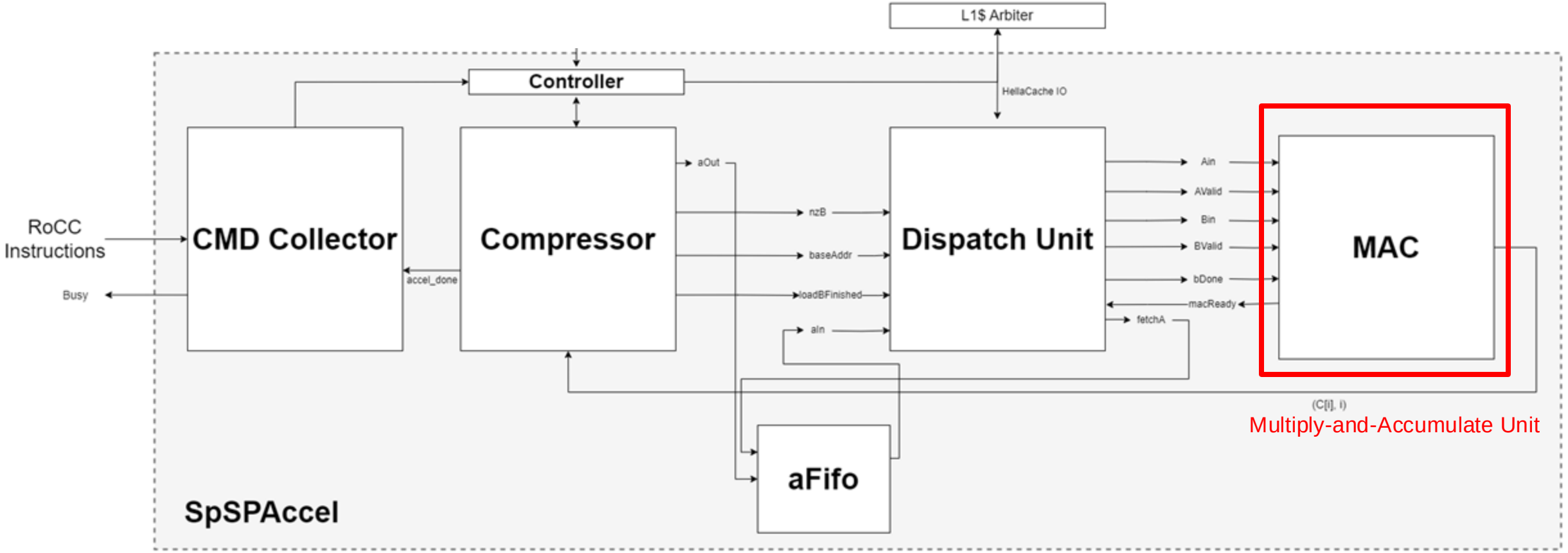
Sparse-Sparse Accelerator: Architecture



Sparse-Sparse Accelerator: Architecture



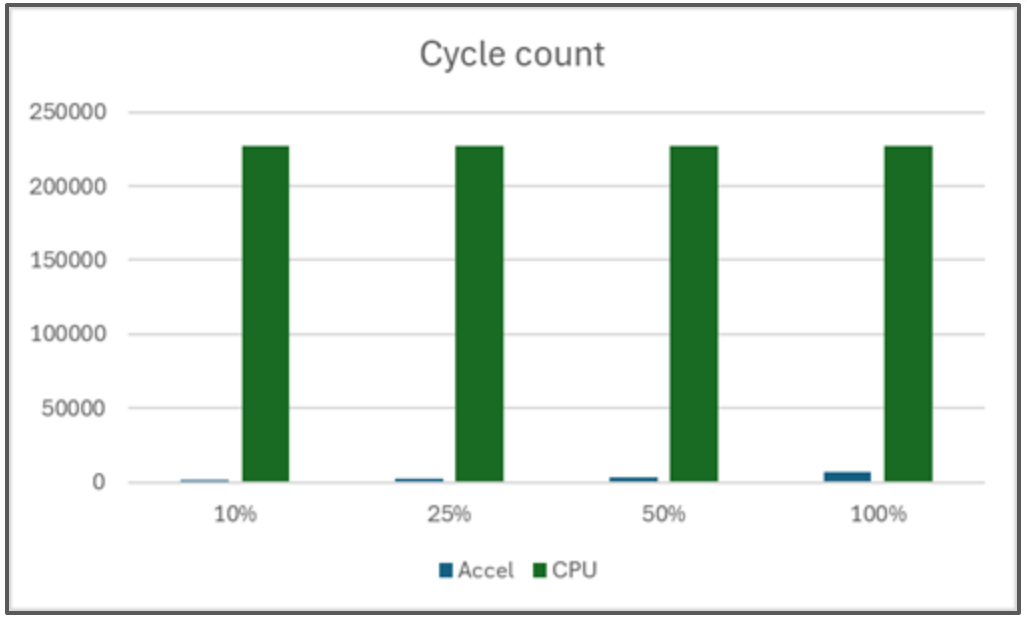
Sparse-Sparse Accelerator: Architecture



Sparse-Sparse Accelerator Performance

- See significant decreases in cycle counts for matrices of all levels of sparsity, with largest improvements on more sparse matrices.

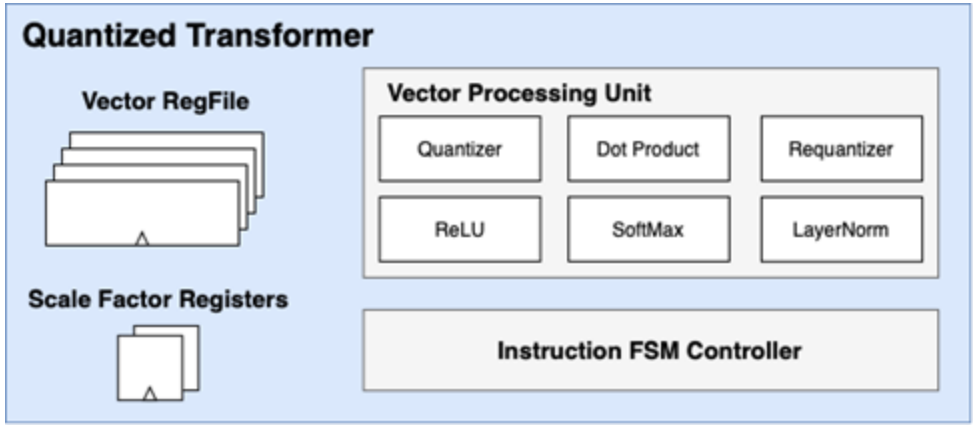
- Potential Improvements:
  - Add more MAC units to increase parallelism
  - Add support for hardware quantization
    - FP32 → INT8
    - Reduces inference latency for ML models

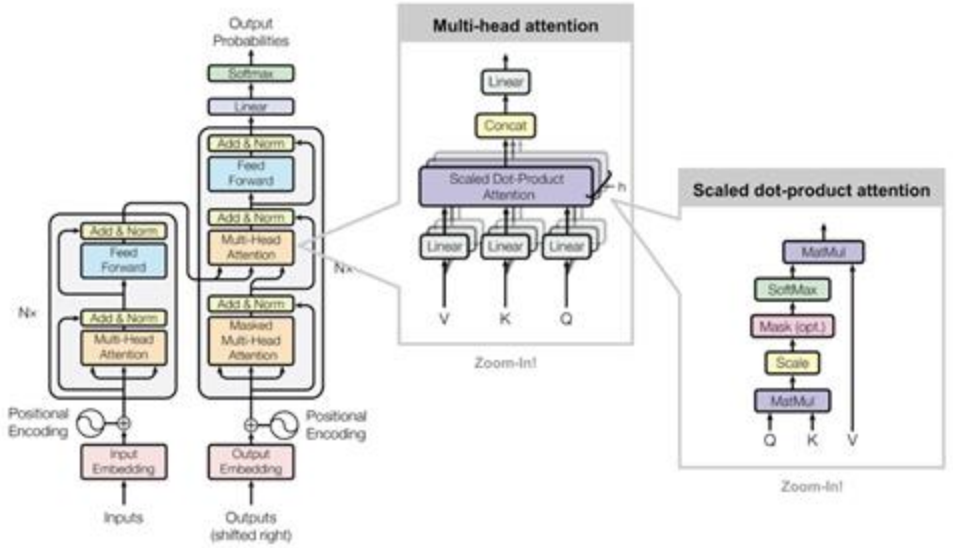


Density	Unit	Average	Improvement
10%-10%	Accel	1846	**x123**
	CPU	227101	
25%-25%	Accel	2271	**x100**
	CPU	227102	
50%-50%	Accel	3339	**x68**
	CPU	227105	
100%-100%	Accel	6679	**x34**
	CPU	227101	

Quantized Transformer

- Possesses independent modules to perform all operations needed for a quantized transformer
  - Vector-wise ReLU, Softmax, and LayerNorm
  - Dot Product and FP32 -> INT8 Quantization
- Instruction controller loads and stores vectors, passes input vectors to the correct module, and passes the output vector/scalar back to main memory
- Supports 8-wide INT8 vectors



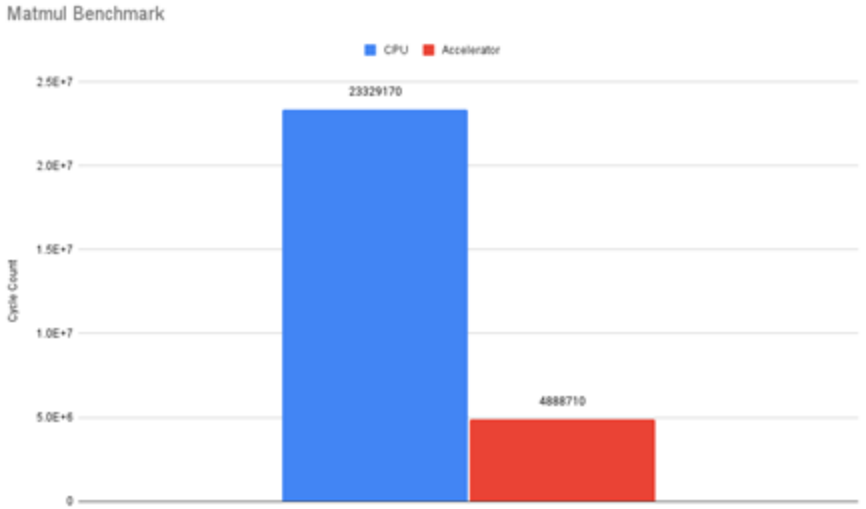


Interfaces:

- RoCC
  - Custom instructions pass data to the accelerator
  - Return 64-byte (scalar) outputs to CPU
- HellaCacheIO
  - Loading/Storing vectors through cache requests

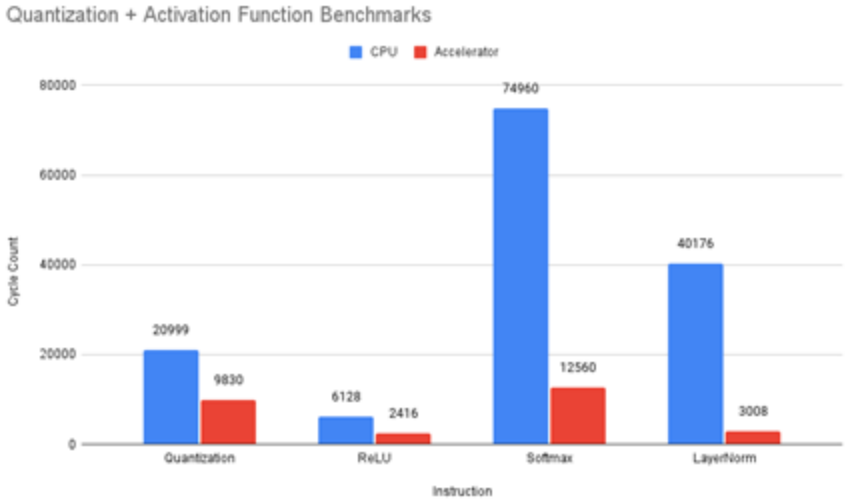
Quantized Transformer

- Benchmarks used 128-wide randomized INT8 vectors
- 128x128 Matmul - 4.77x speedup
- 128-vector Quantization - 2.14x speedup
- 128-vector ReLU - 2.54x speedup
- 128-vector Softmax - 5.97x speedup
- 128-vector LayerNorm - 13.36x speedup



Conclusions:

- Performance boosts even with load/store overhead
- Increasing vector size (area) can reduce overhead
- More hardware-software co-design can further optimize for specific transformer workloads

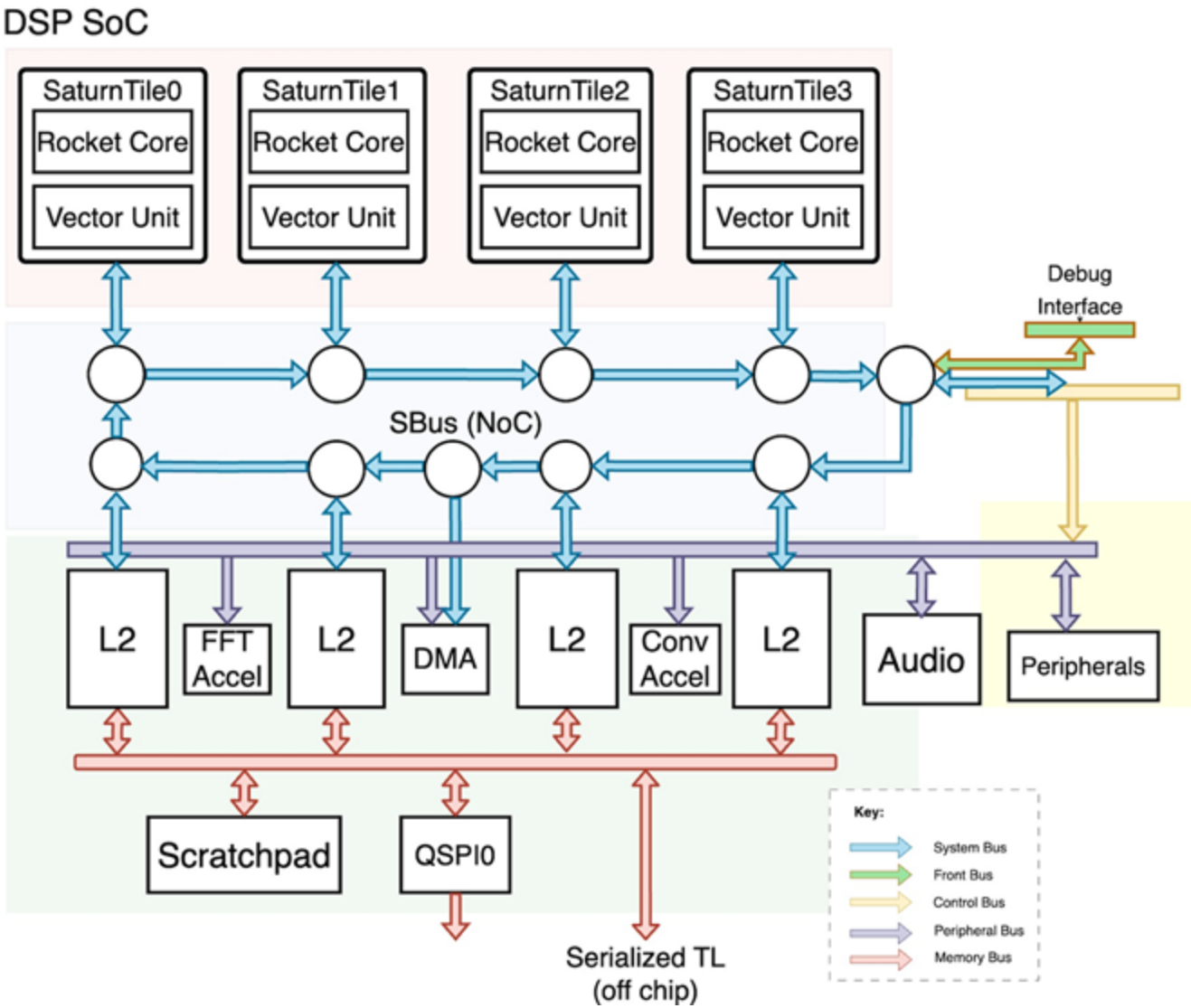


Agenda

- Introduction and Overview
- BearlyML
- DSP
- Questions

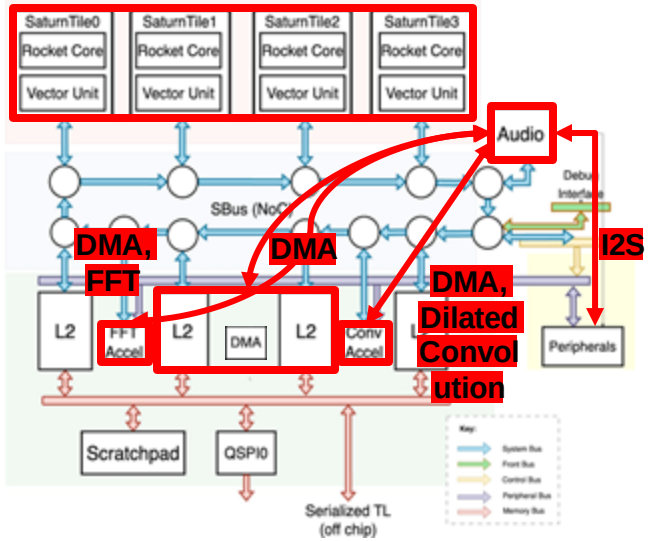
DSP Overview

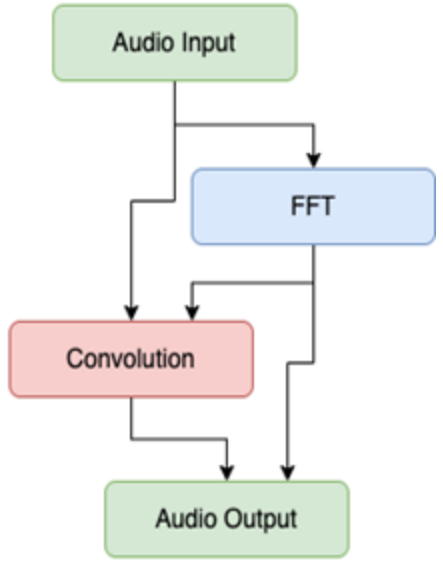
- Designed for Audio processing applications
- Added features:
  - Audio interface
    - I2S + Sigma-Delta DAC
  - Convolution accelerator
  - SDF-FFT accelerator
  - DMA engine
  - 256 KiB L2 cache
  - 32 KiB scratchpad
- 4x Saturn-V Tiles
  - Rocket core + Vector unit
- Buses widths expansion
  - 128-bit NoC by Constellation



DSP Applications: Audio Effects Processing

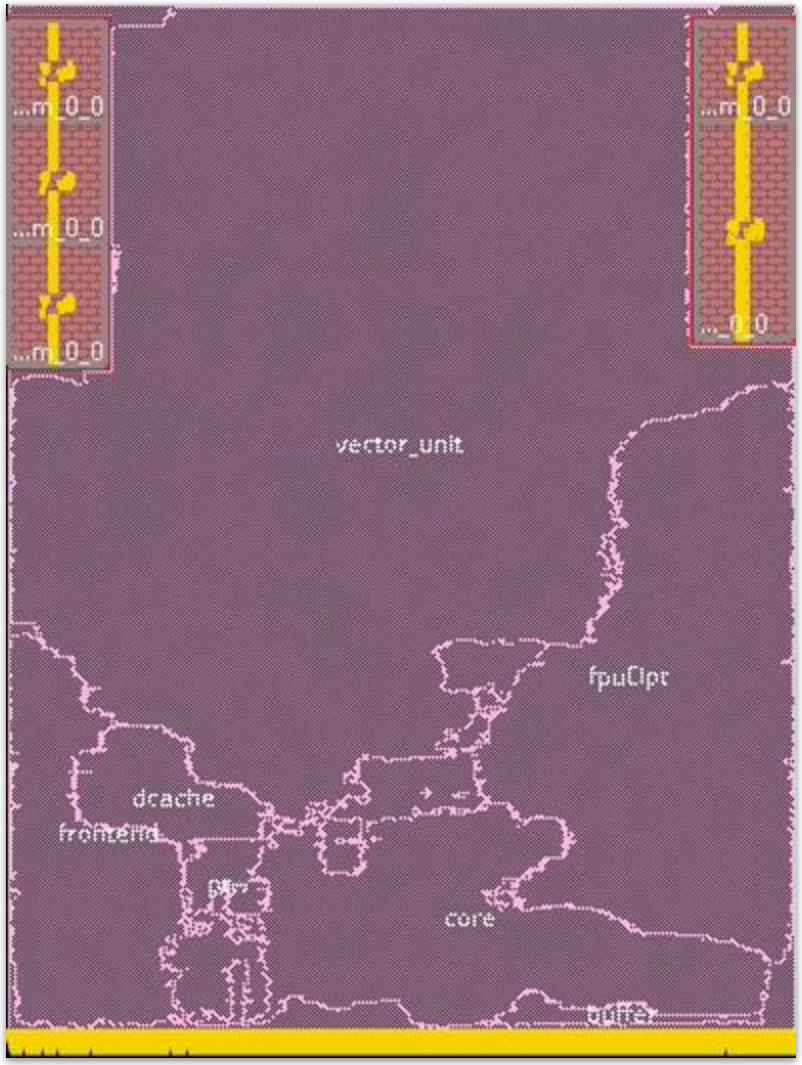
- 1. Audio Input/Output: Process audio input, ex: from an instrument.
- 2. FFT: Analyze the frequency components of the incoming signal:
  - Frequency-based Effects: Equalization or harmonic generation Pitch Detection: Detect the pitch of the incoming notes: tuning, pitch shifting effects, note detection
- 3. Convolution: Effects Simulation- By convolving the guitar signal with impulse responses of pedals like overdrive, distortion, chorus, or delay, the device can emulate the sound and response of these effects pedals digitally.
- 4. Saturn-V Vector Core: The addition of vector cores allows for more specialized algorithms (DCT, Wavelet Transforms, Filters) to be accelerated with minimal loss in performance.





Saturn RISC-V Vector Core

- 4x Rocket in-order cores with Saturn Vector units
- Direct-mapped 4 KiB ICache & 4 KiB DCache
- RISC-V Vector Extension Version 1.0 compliant
- Minimal area configuration targeting integer operations: VLEN=256 DLEN=64
- 423.36 x 550.98 um

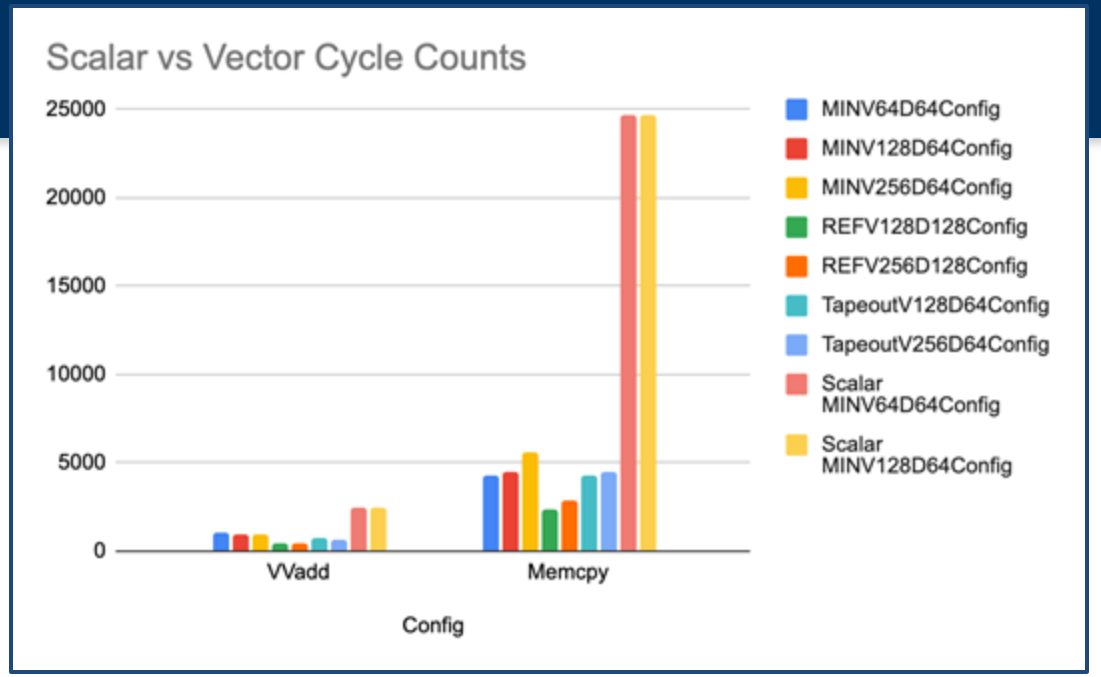


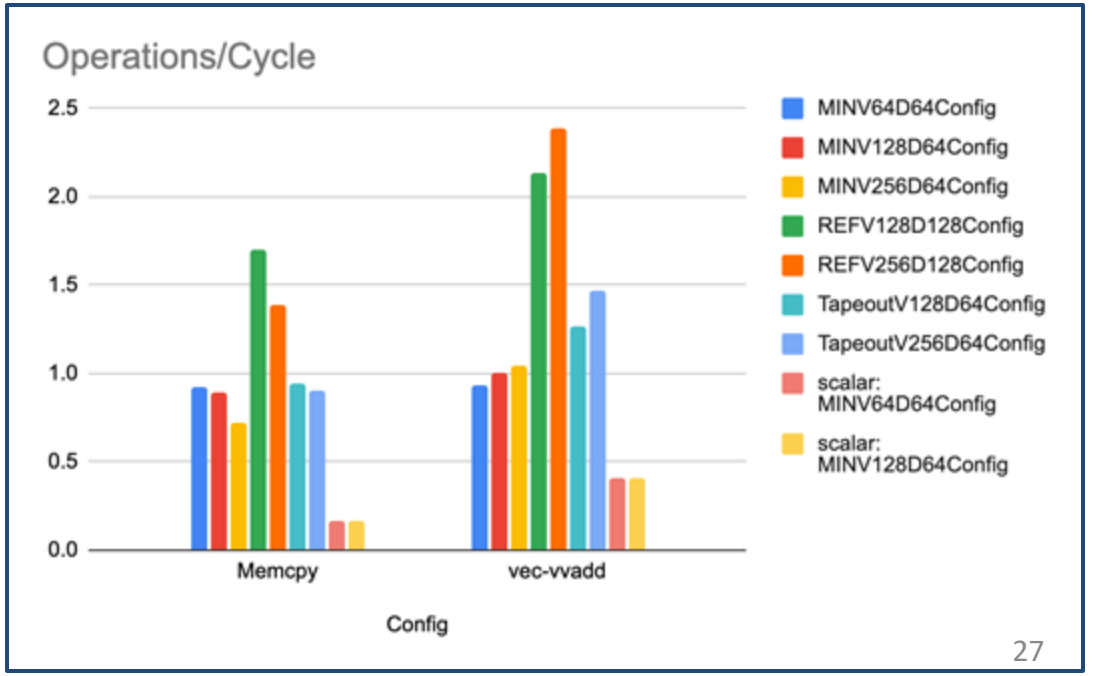
Saturn Core Floor Plan

Thank you, Saturn Team: Jerry Zhao, Daniel Grubb, Miles Rusch, Tianrui Wei, Ella Schwarz

Scalar vs Vector Performance

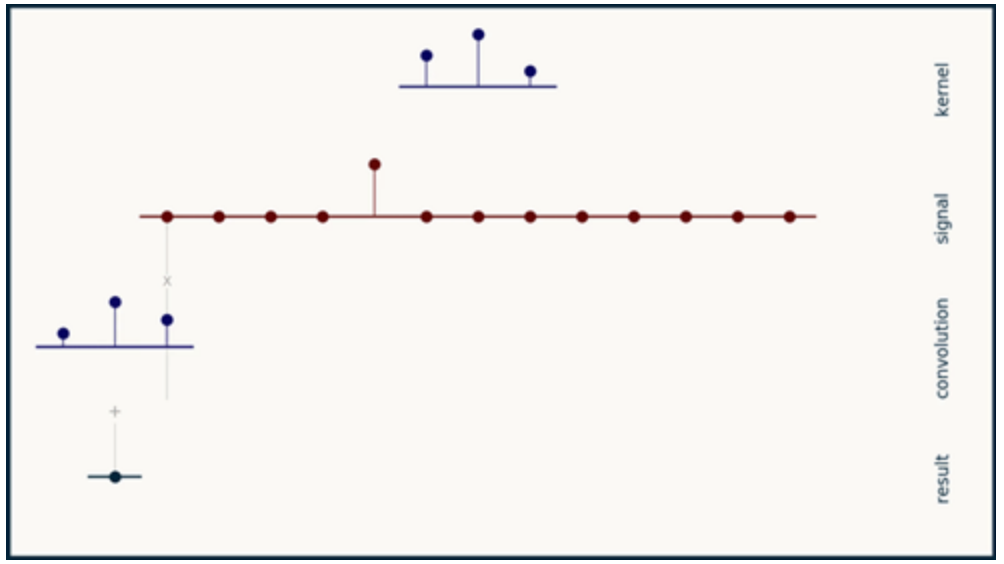
- Vector instructions significantly decrease instruction overhead with more ops/instruction
- Increased performance in data-parallel applications can match or even outperform small dedicated accelerators





DSP Convolution Accelerator

- Performs **1D dilated convolution** for audio processing applications on FP16 audio data
- Can alter, classify, or detect audio
- Dilated convolution used to increase effectiveness of CNNs



- Sends and receives data with the DMA
- Computes convolution with **multiplication units + an adder tree**
- FSM handles floating point and datapath errors

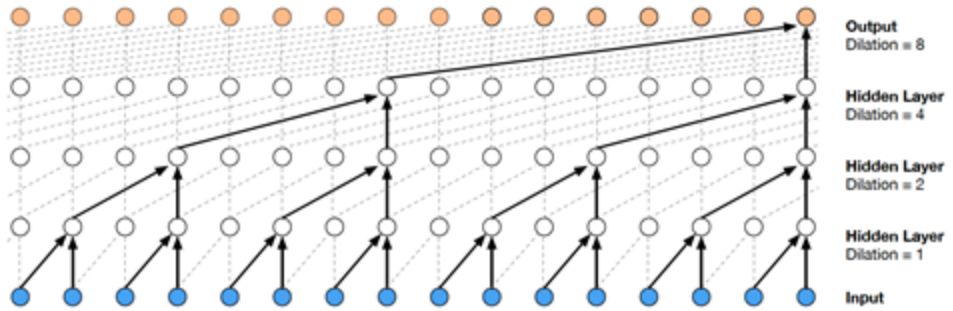
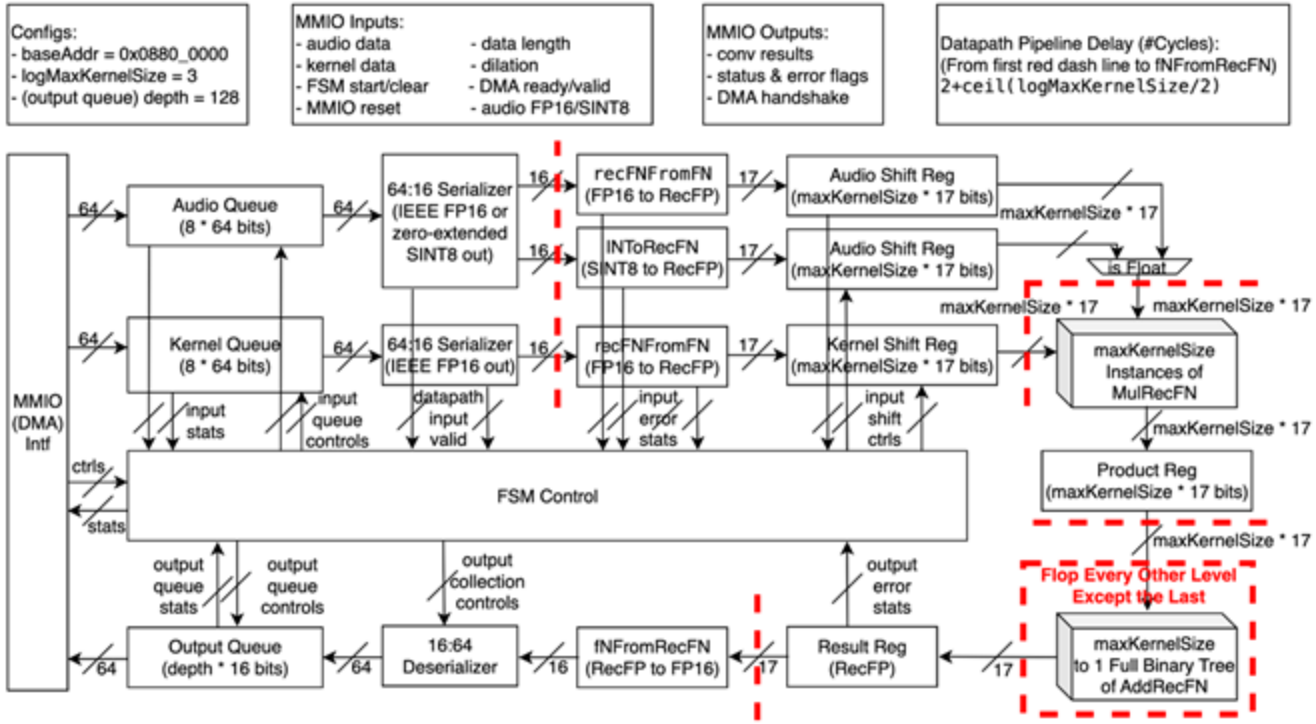


Figure 3: Visualization of a stack of *dilated* causal convolutional layers.

DSP Convolution Accelerator: Implementation



Input Length	CPU (Rocket) Cycles	Accelerator Cycles	Speedup
16	74303	910	82x
32	152894	1310	117x
64	306917	2105	146x
256	1230245	6516	189x

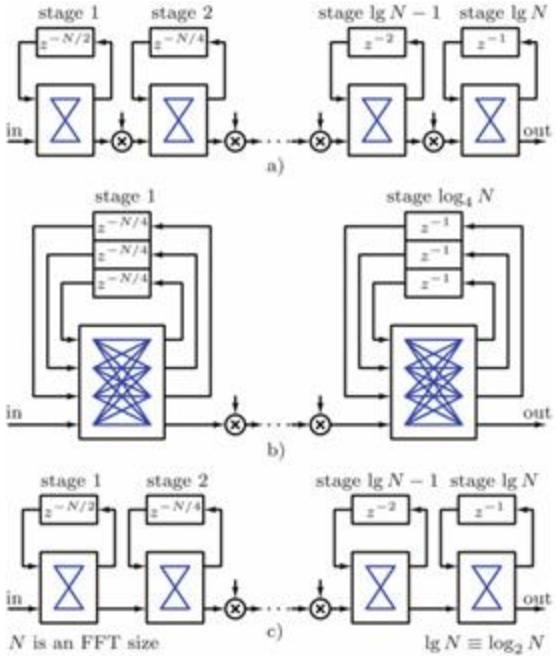
SDF-FFT Accelerator

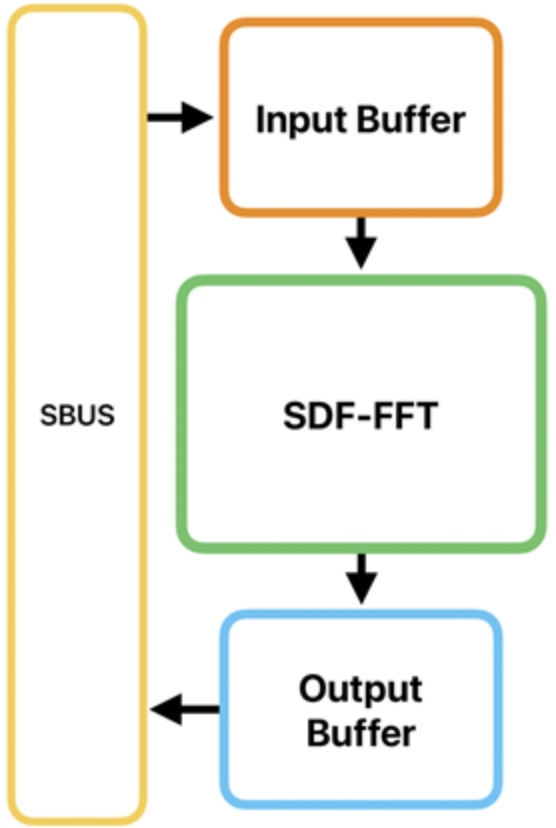
Chipyard integration of the SDF-FFT generato written by Vladimir Milovanović and Nikola Petrovic

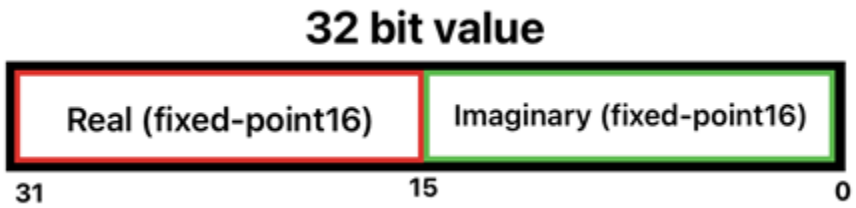
- Drastically smaller and scales better than the built-in chipyard FFT generator
- Custom TileLink Front-end with input and output buffers

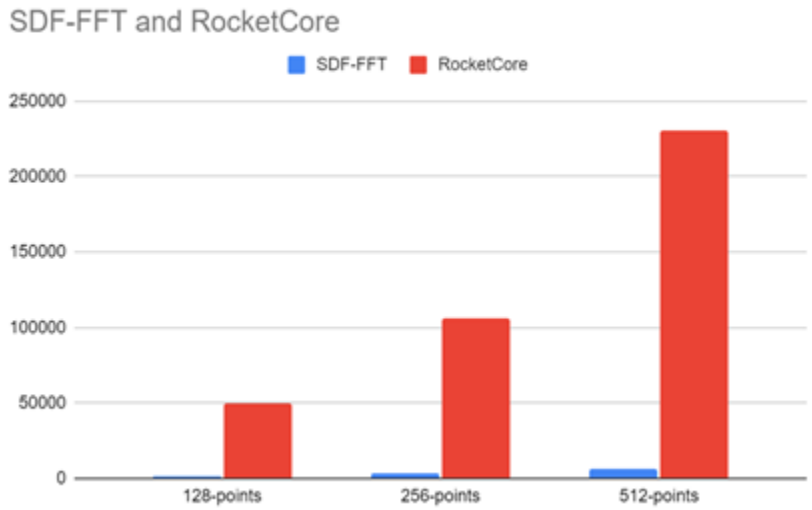
Our config has a **128-point FFT** accelerator

- takes in two stacked **16-bit fixed-poi** values





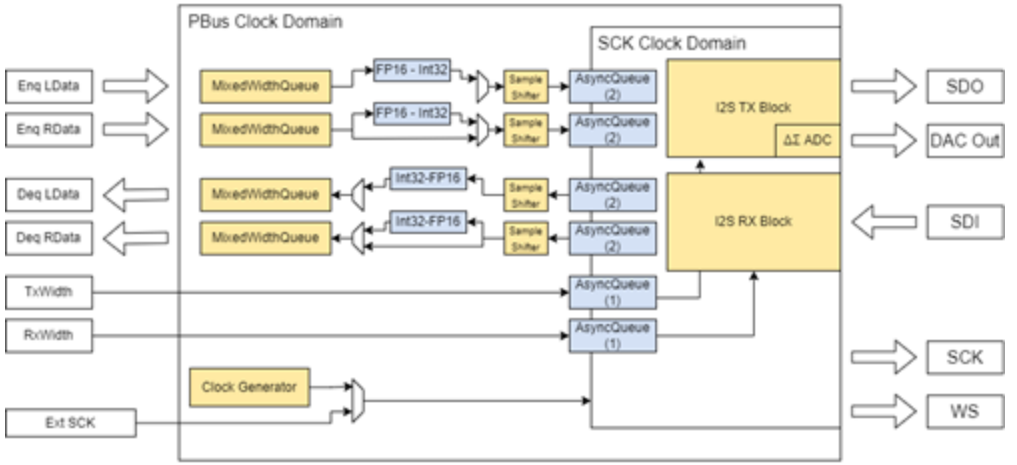


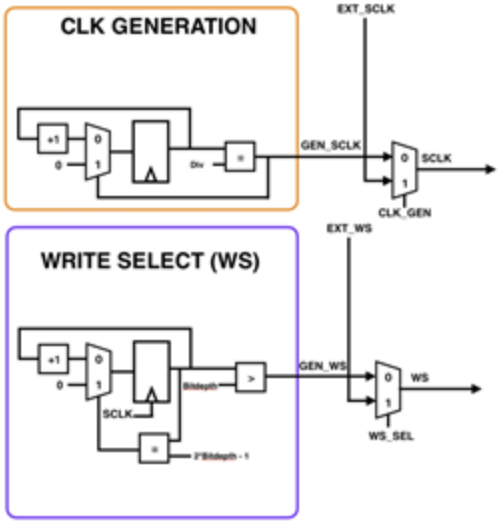


Audio I2S

Designed and Integrated 8 Channel Audio subsystem

- 4 Stereo Transmit and Receive Channels
- External Codec Support w/ I2S
- Built in 16 bit Delta-Sigma DAC
- Programmable Clock Generator
- 8-32 Bits/Sample or FP16
- 120Hz-4MHz Sample Rate
- Supports External Clock Generation





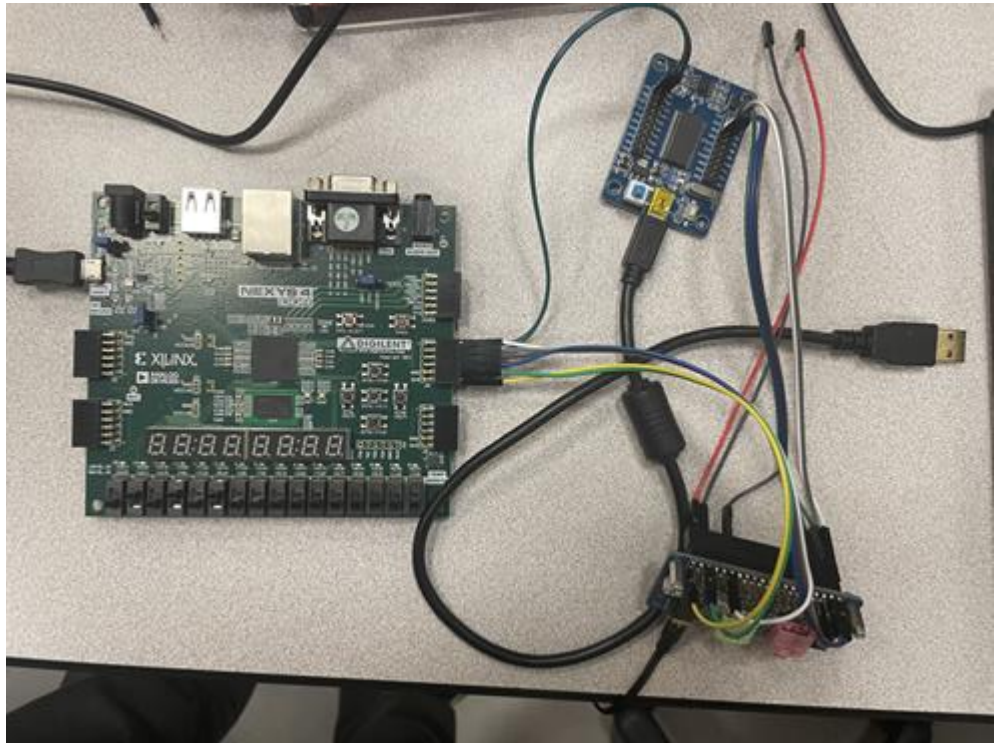
Audio I2S

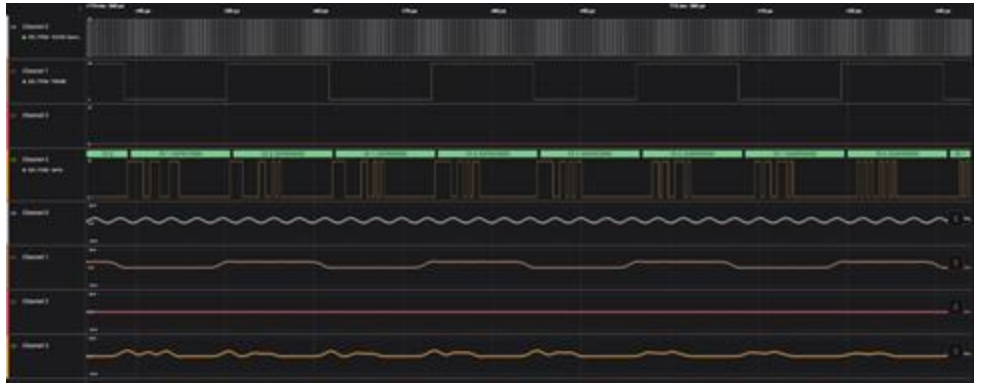
Programming Interface:

- 64 Bit MMIO FIFO Registers Packed
- 64 Byte Queues for each TX/RX channel
- Configuration Bits
- Watermark Registers for DMA Access

Testing:

- DAC + I2S FPGA Tested with FE-PI external CODEC + Nexys4DDR
- RTL Tested with custom VIP





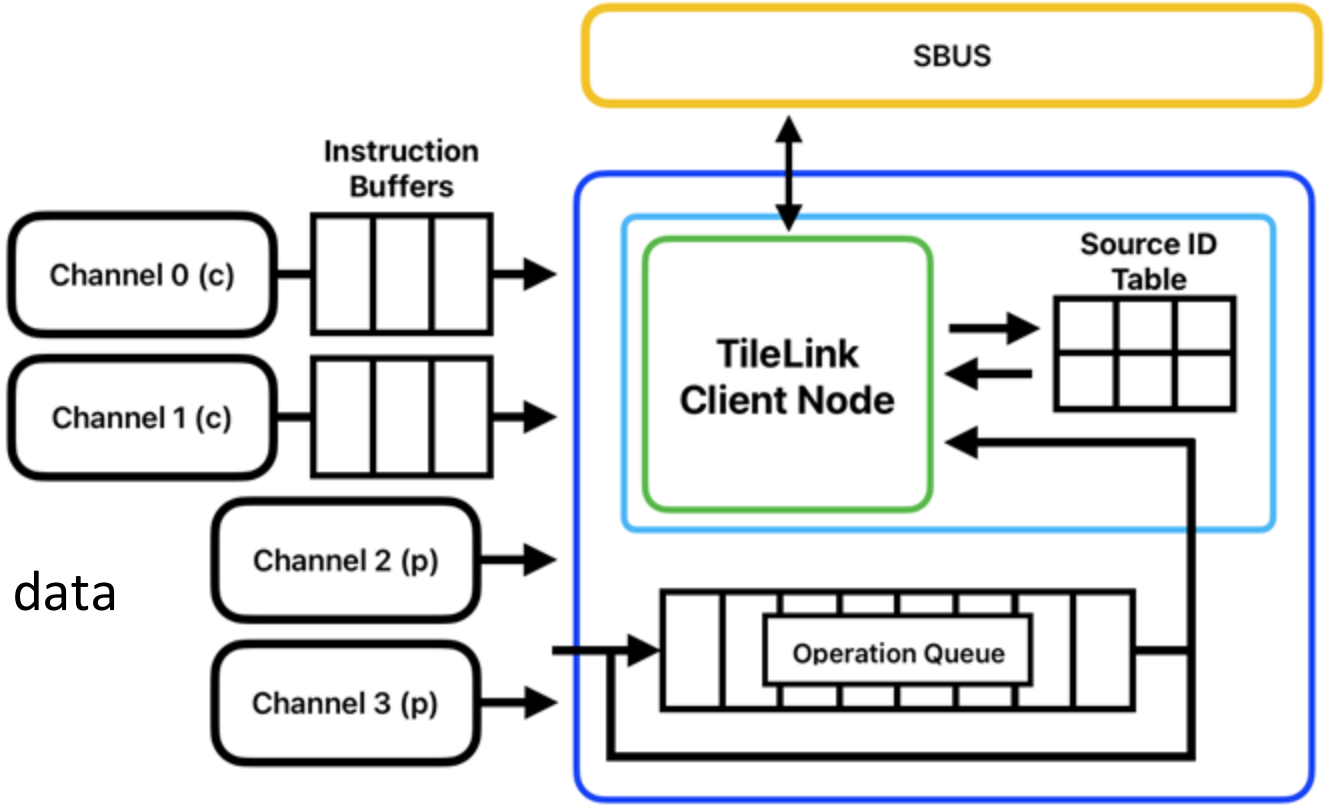
General DMA

DMA performs various TileLink memory operations

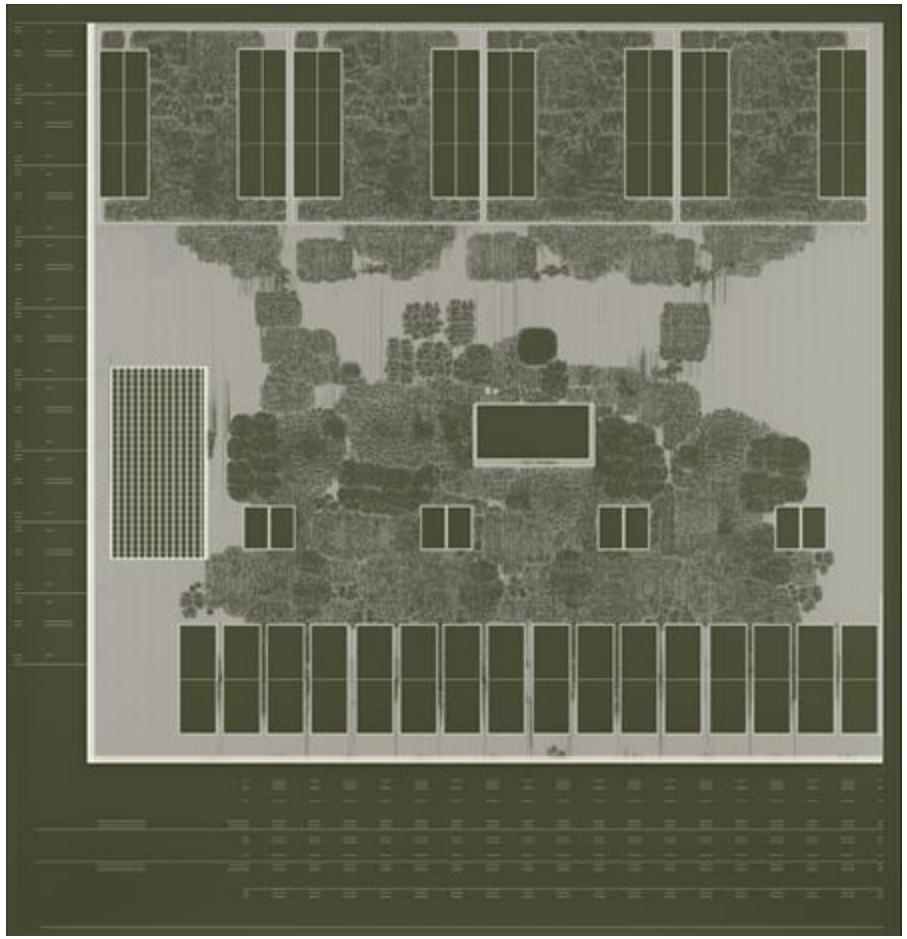
- Memory → Memory
- Peripheral → Memory
- Memory → Peripheral

We support:

- Multiple channels with load balancing
- Double-buffered operations
- Peripheral polling to see if they are ready for data transfer
- Variable transfer width/length
- Variable stride on read and write
- Multiple inflight operations (Currently set to 8)



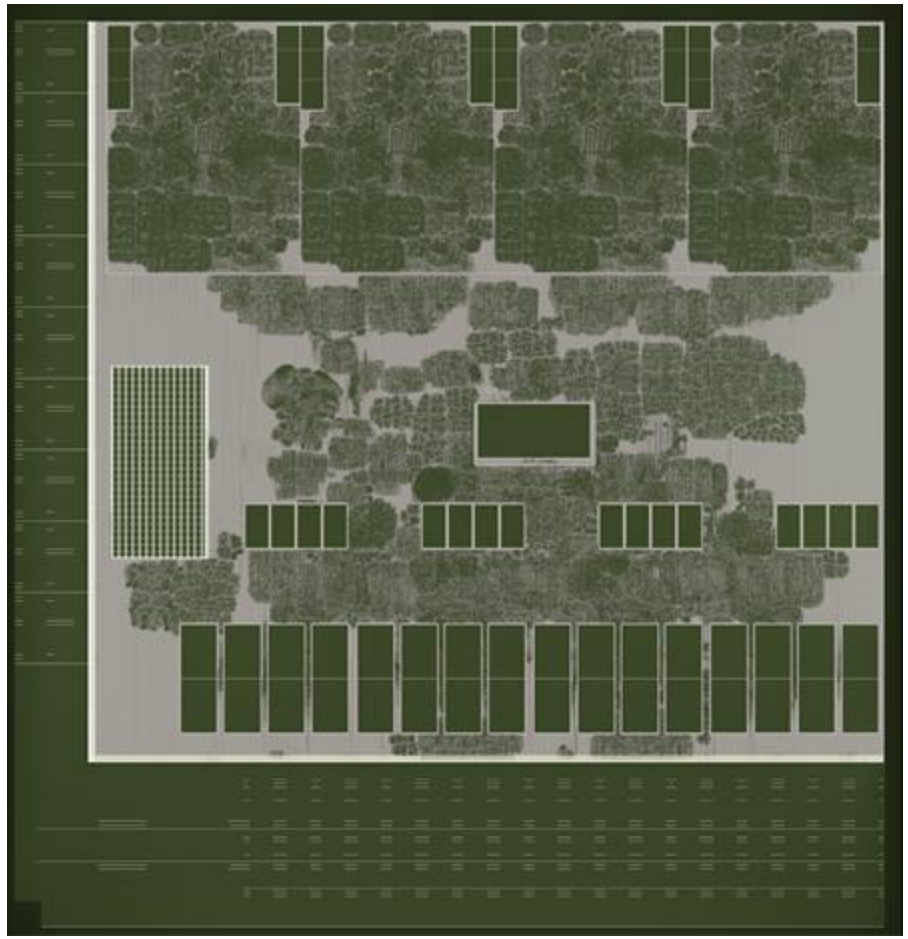
Questions?



BearlyML ‘24

DSP Final Design was sent to Intel on May 12, Bearly on May 13

All accepted on May 15



DSP

Special thanks to the Apple New Silicon Initiative & Intel University Shuttle Program!