tinyML® EMEA

Enabling Ultra-low Power Machine Learning at the Edge

June 26 - 28, 2023

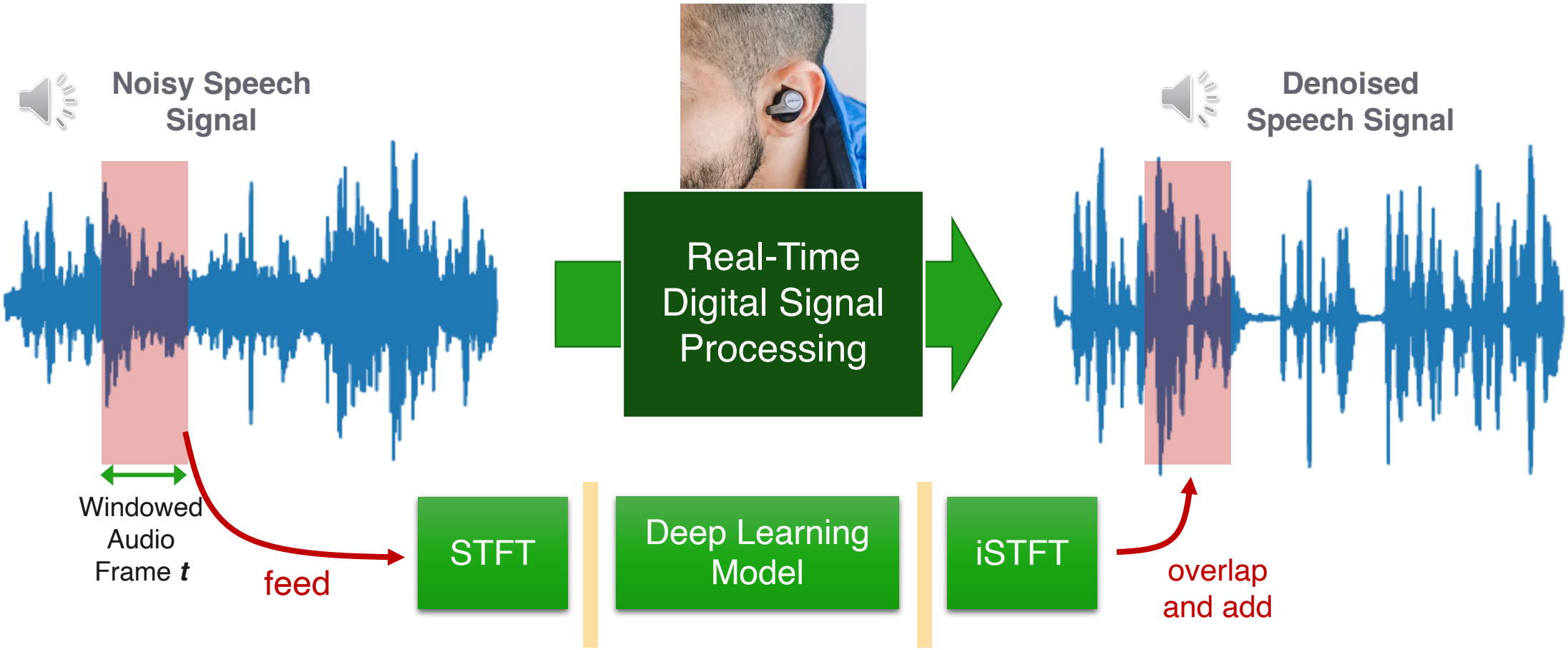
www.tinyML.org

TinyDenoiser: RNN-based Speech Enhancement on a Multi-Core RISCV MCU (GAP9) with Mixed FP16-INT8 Post-Training Quantization

Marco Fariselli - Manuele Rusci
Embedded ML Engineer
marco.fariselli@greenwaves-technologies.com

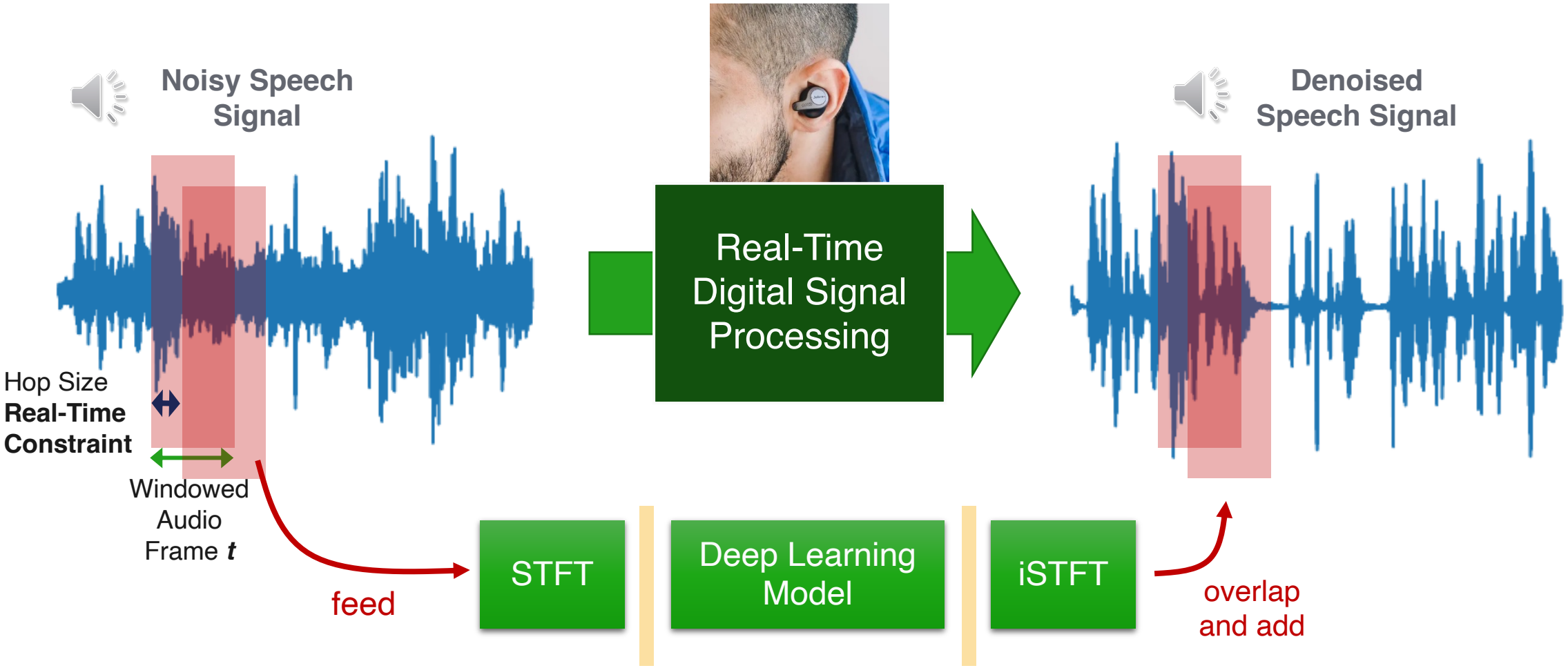
GREENWAVES TECHNOLOGIES

Speech Enhancement (or Denoising)



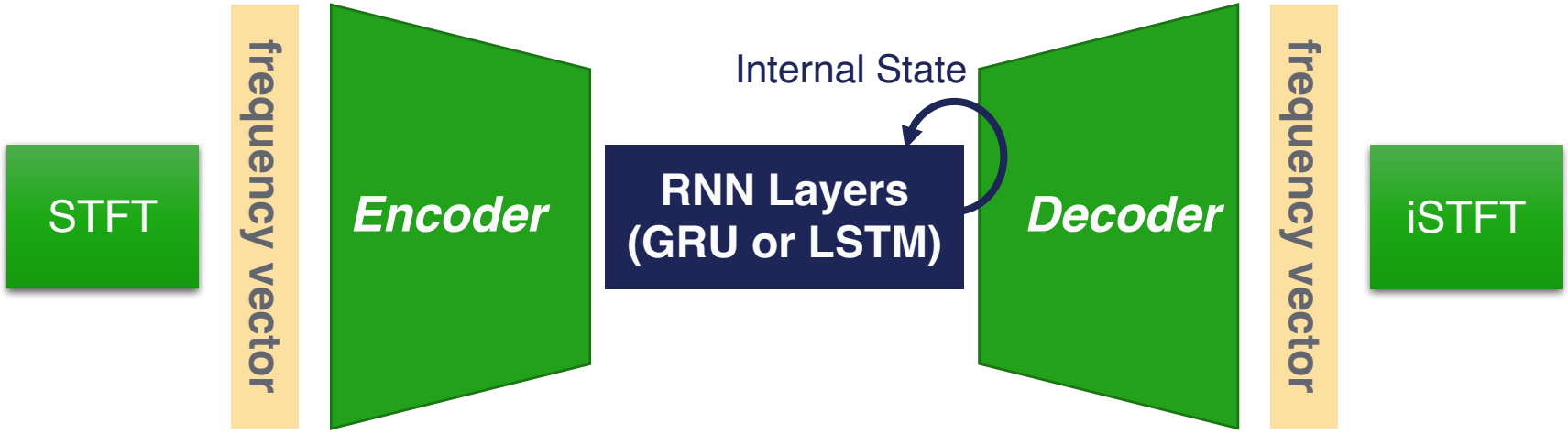
GREENWAVES TECHNOLOGIES

Speech Enhancement (or Denoising)



GREENWAVES TECHNOLOGIES

RNN for Speech Enhancement



- Prediction based on current and previous inputs (memory stored in internal state)
- No wide input receptive field
- Memory bounded (low Op/Param ratio)
- Audio pipelines are generally sensitive to quantization

Contribution

We present an optimized **HW/SW** design for **LSTM and GRU-based Speen Enhancement (SE)** models for **multi-core MCU** systems with limited memory space.

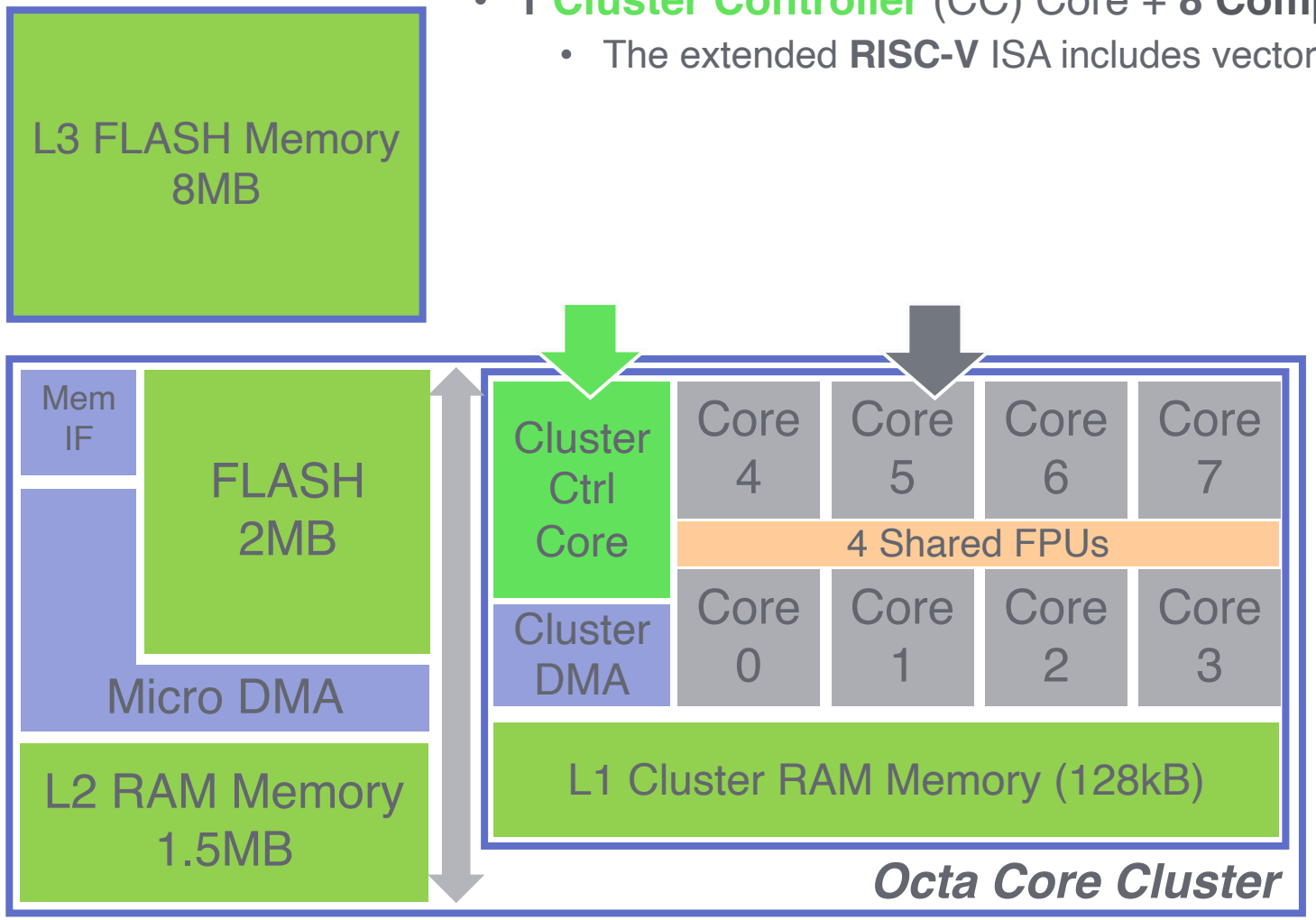
We propose an **almost lossless Mixed-Precision FP16-INT8 Post-Training Quantization** scheme to accelerate RNN-based SE on MCUs.

We provide a detailed analysis of **latency** and **HW/SW efficiency** on a 22-nm 1+8 RISC-V cores MCU.

6

RISC-V MultiCore MCU Platform (GAP9)

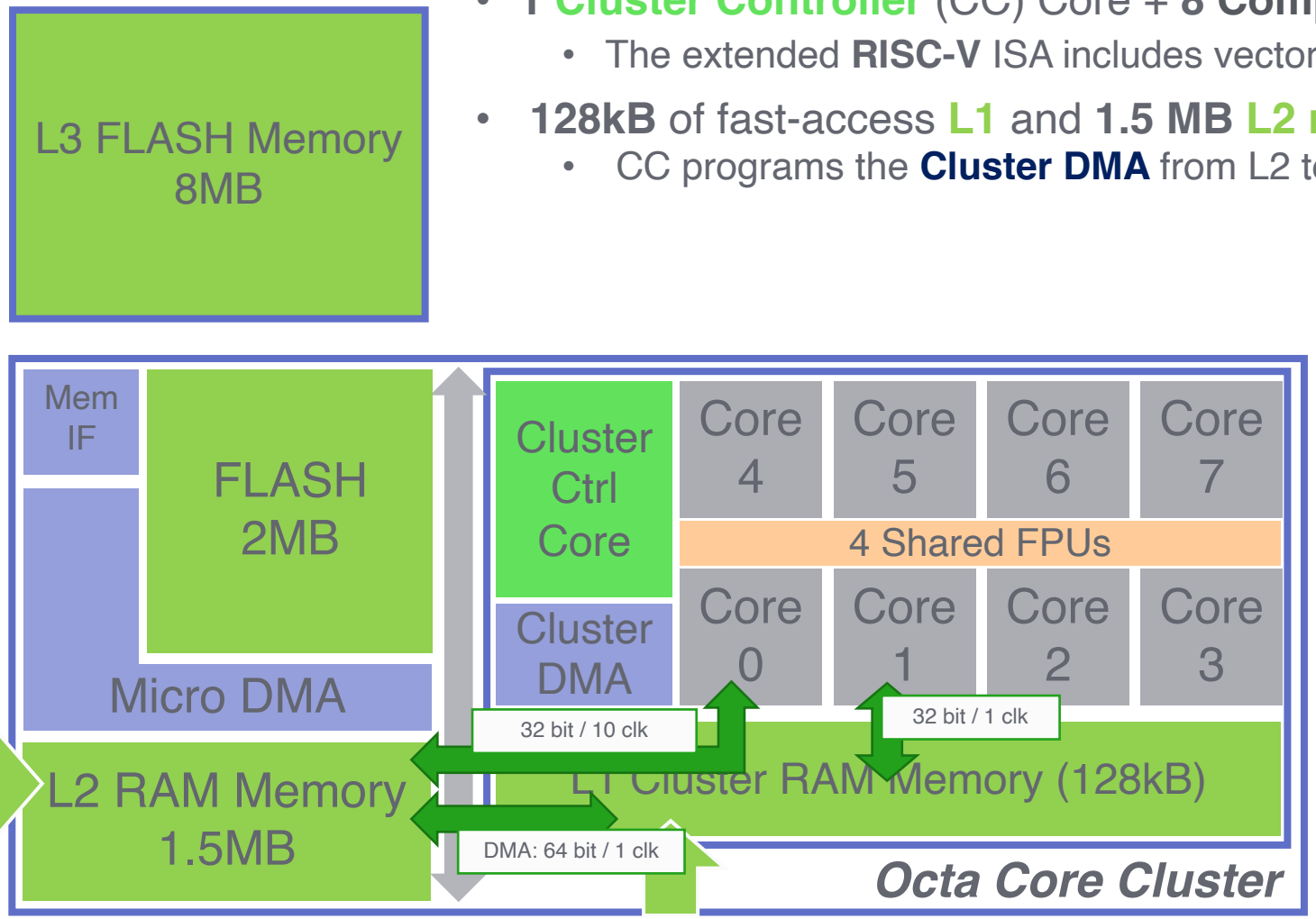
- **1 Cluster Controller** (CC) Core + **8 Computes Cores**
  - The extended **RISC-V** ISA includes vectorized INT8 MAC and **float16 (FP16) MAC**



SW Perspective

```
void foo() {
  // Some parallel computation
  ...
}

int main_cc_core() {
  ...
  task_offload_compute_cores(foo)
  ...
}
```

RISC-V MultiCore MCU Platform (GAP9)

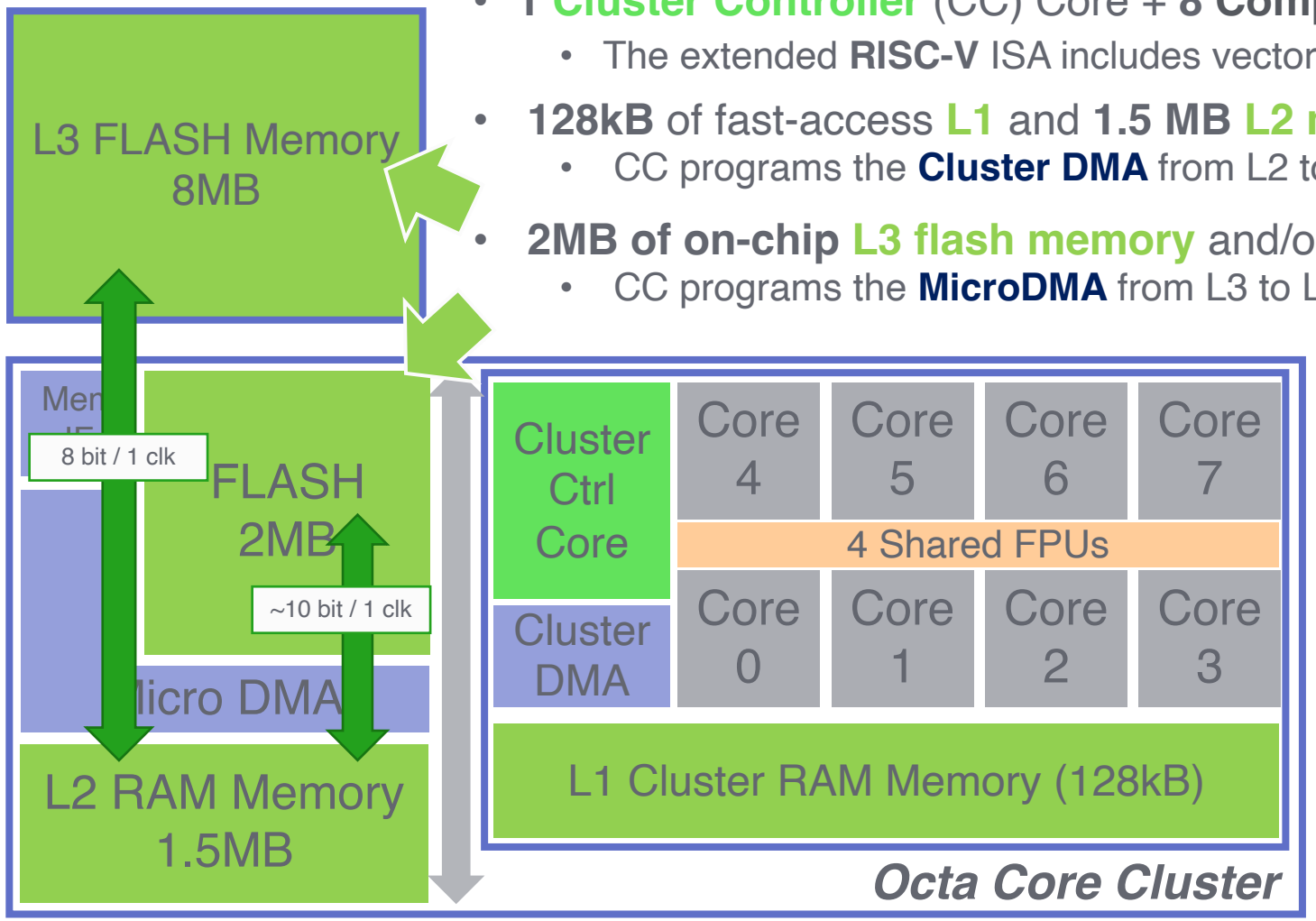


- **1 Cluster Controller** (CC) Core + **8 Computes Cores**
  - The extended **RISC-V** ISA includes vectorized INT8 MAC and **float16 (FP16) MAC**
- **128kB** of fast-access **L1** and **1.5 MB L2 memories**
  - CC programs the **Cluster DMA** from L2 to L1 to copy data between L2 and L1

SW Perspective

```
void foo() {
  // Some parallel computation
  ...
}

int main_cc_core() {
  ...
  task_offload_compute_cores(foo)
  ...
}
```

RISC-V MultiCore MCU Platform (GAP9)

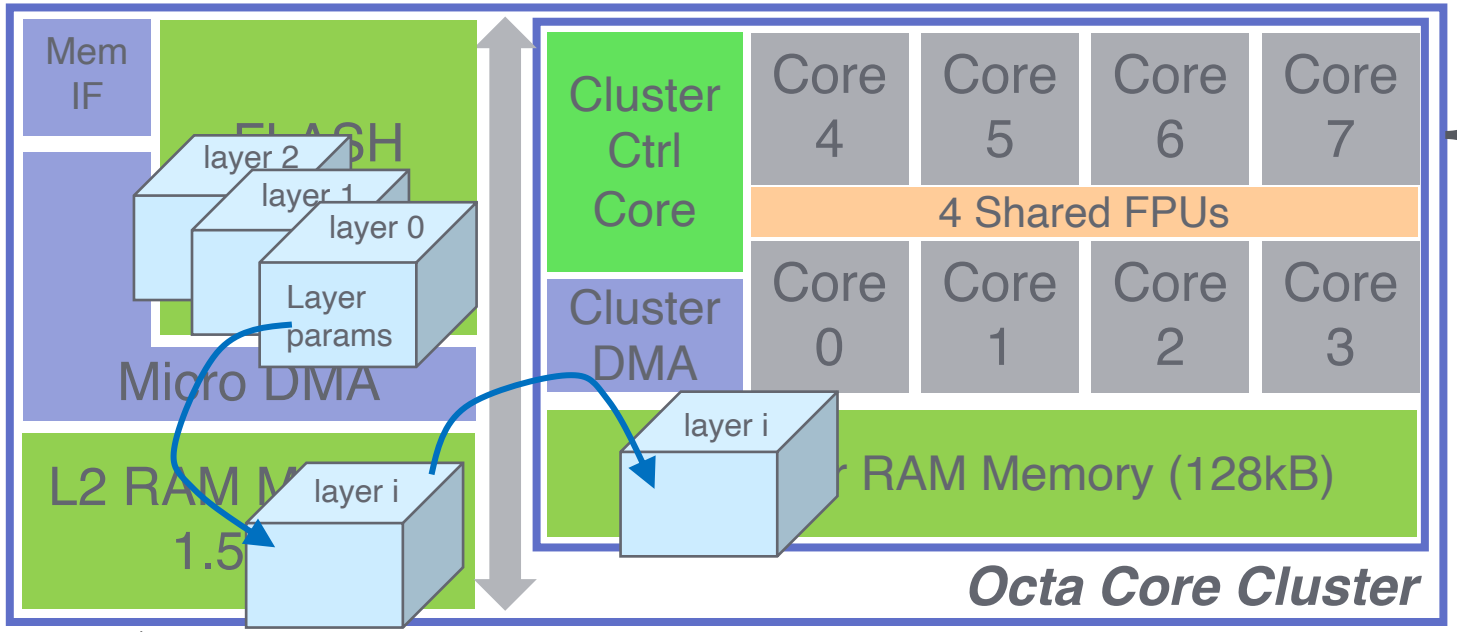


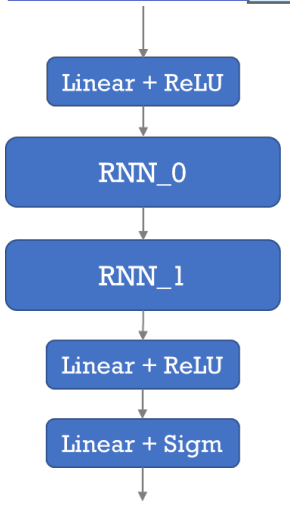
- **1 Cluster Controller** (CC) Core + **8 Computes Cores**
  - The extended **RISC-V** ISA includes vectorized INT8 MAC and **float16 (FP16) MAC**
- **128kB** of fast-access **L1** and **1.5 MB L2 memories**
  - CC programs the **Cluster DMA** from L2 to L1 to copy data between L2 and L1
- **2MB of on-chip L3 flash memory** and/or 8MB off-chip **L3 RAM memory**
  - CC programs the **MicroDMA** from L3 to L2 to copy data between L3 and L2

SW Perspective

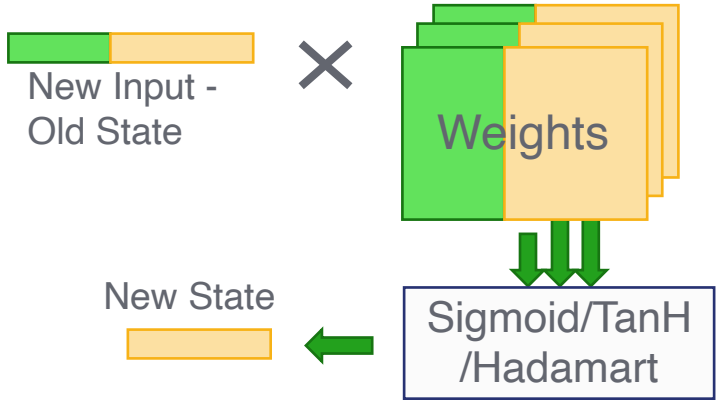
```
void foo() {
  // Some parallel computation
  ...
}

int main_cc_core() {
  ...
  task_offload_compute_cores(foo)
  ...
}
```

RNN Mapping on HW





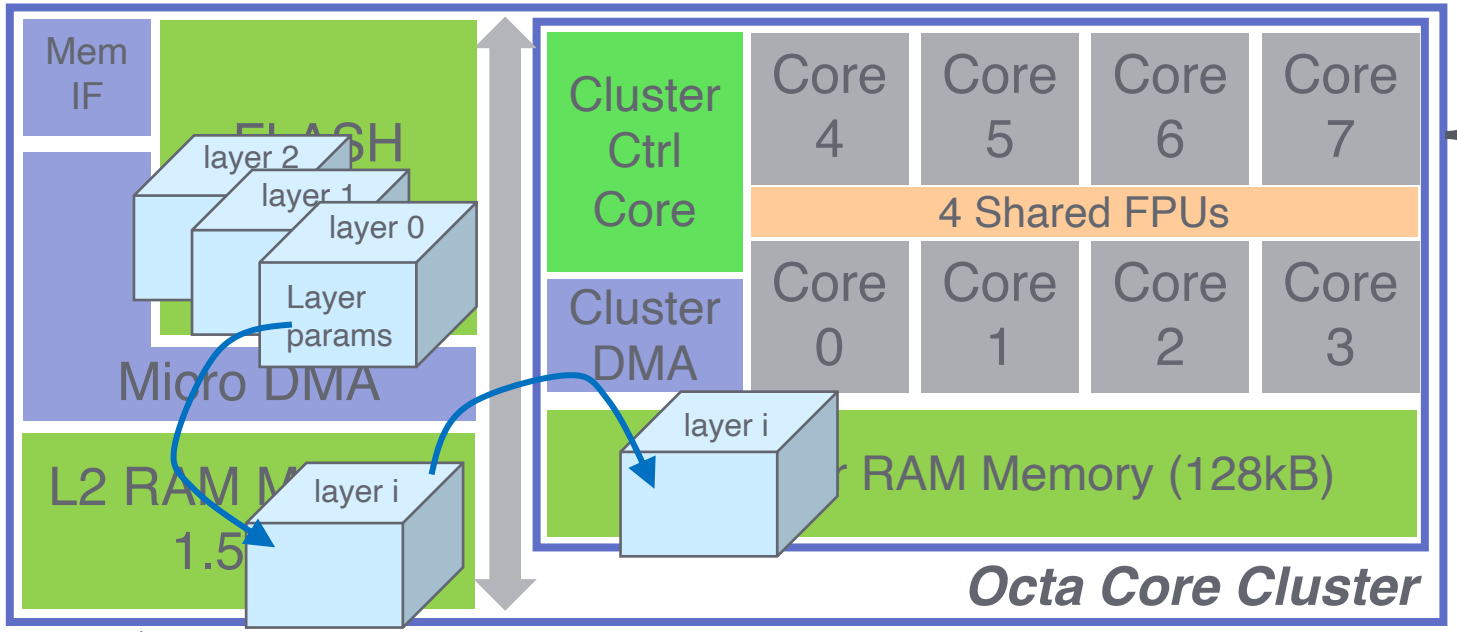
RNN Computational kernels



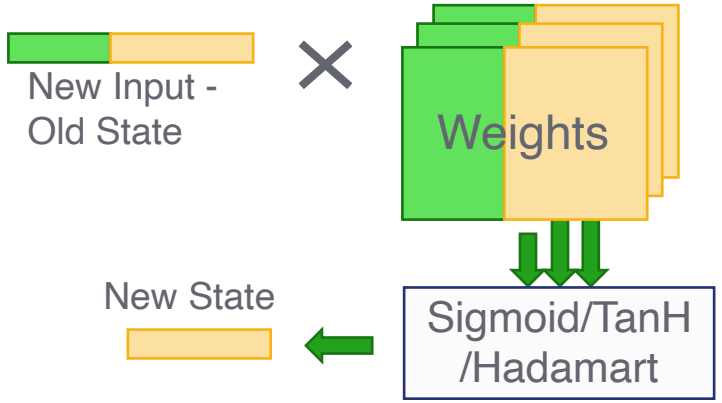
Most of the computation is in the ***MatrixVector*** Products

- Model Coefficient stored in non-volatile FLASH memory (on-chip, if fitting)
- Execution layer-by-layer
  - Data copied from L3 to L1
  - Parallel Execution on 8cores

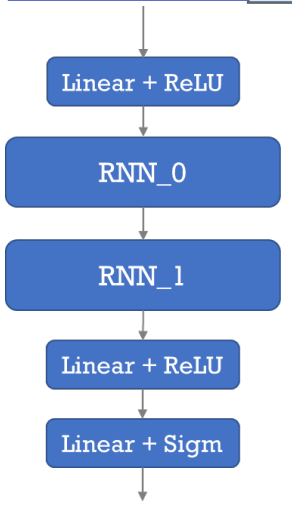
RNN Mapping on HW



RNN Computational kernels



Most of the computation is in the ***MatrixVector*** Products

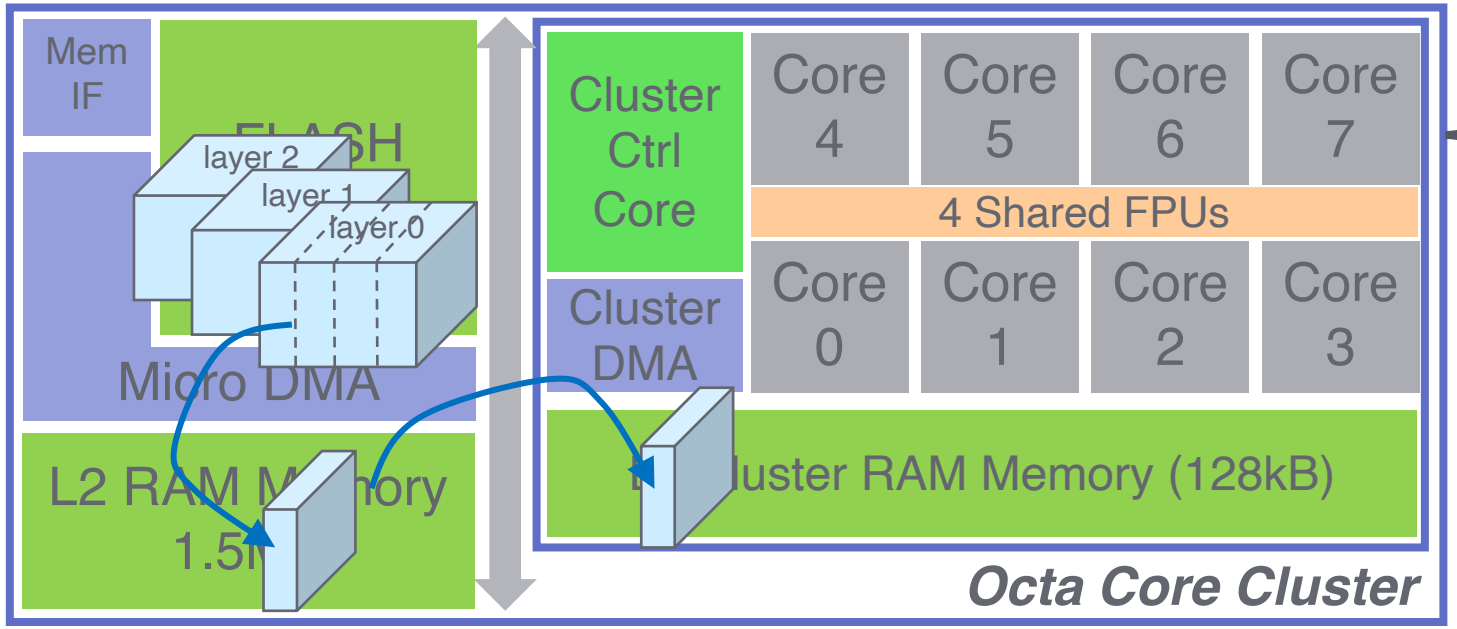


- Model Coefficient stored in non-volatile FLASH memory (on-chip, if fitting)
- Execution layer-by
  - Data copied fro
  - Parallel Execut

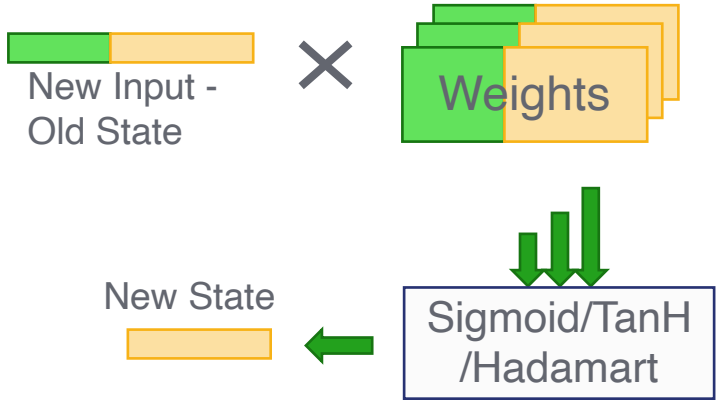
$$T = \sum_{i}^{N_{layer}} T_i = T_i^{L3-L2} + T_i^{L2-L1} + T_i^{cores}$$

Problem: coefficients may not fit L1 memory

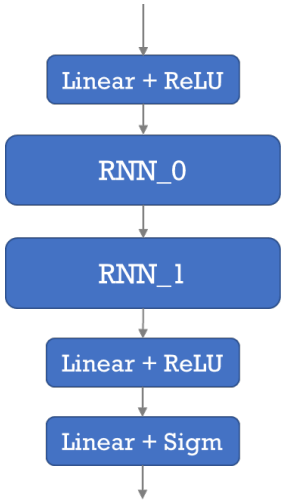
RNN Mapping on HW



RNN Computational kernels



Most of the computation is in the ***MatrixVector*** Products

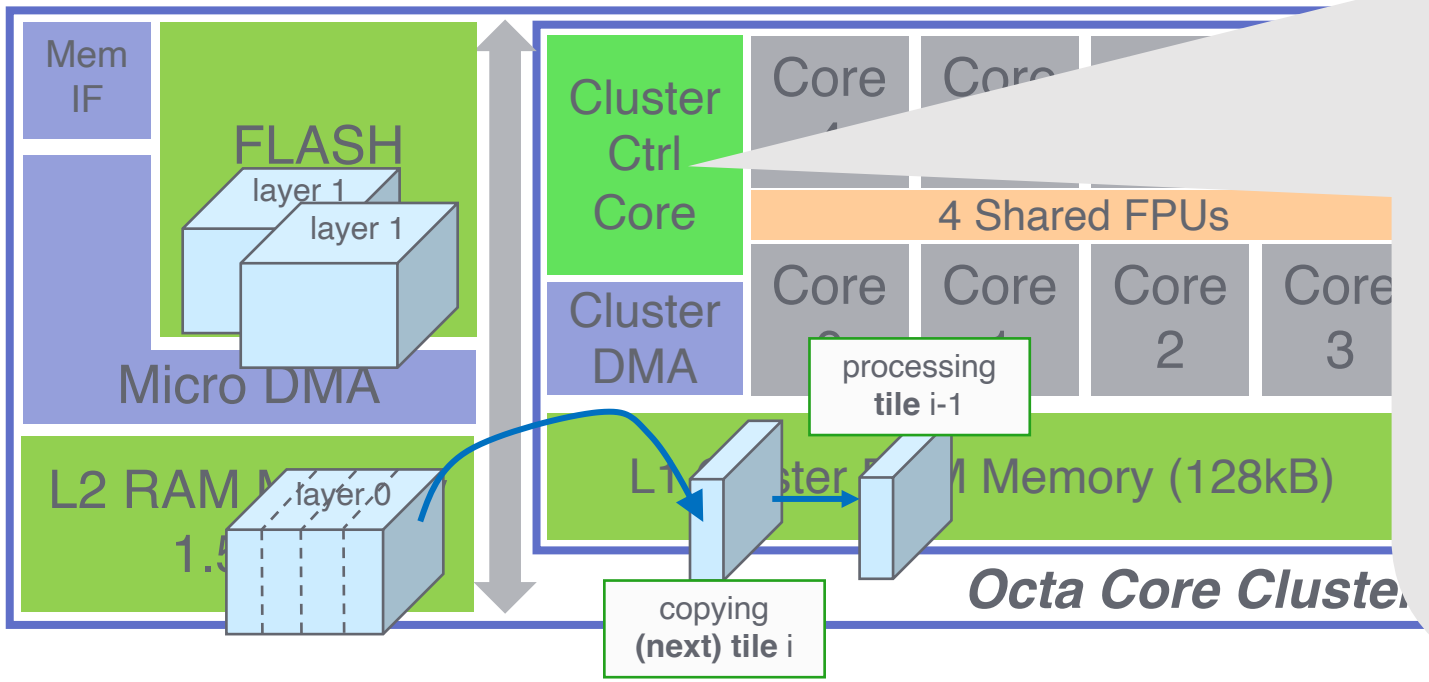


Problem: coefficients may not fit L1 memory

Need to split a weight tensor into N_{tile} sub-tensors: **Tiles**

$$T_i = N_{tile} \cdot (T_{i,tile}^{L3-L2} + T_{i,tile}^{L2-L1} + T_{i,tile}^{cores})$$

Optimizations: Double Buffering




```
void RNN_Layer_i(
   In,        // L2 Vector
   Weights,   // L3 or L2 Vector
   Out        // L2 Vector
)
// Tile sizes computed at compile time
// L1 buffers act as working mem
{
 (if not promoted)
  uDMA load first W tile L3->L2
  DMA load first In & W tile L2->L1
  for any tile t of In/Weights:
   (if not promoted)
   uDMA load next W tile L3->L2
   DMA load next In & W tile L2->L1
   Out[i] = ParRNN(In[t], Weights[t])
   DMA write Out tile L1->L2
}
```

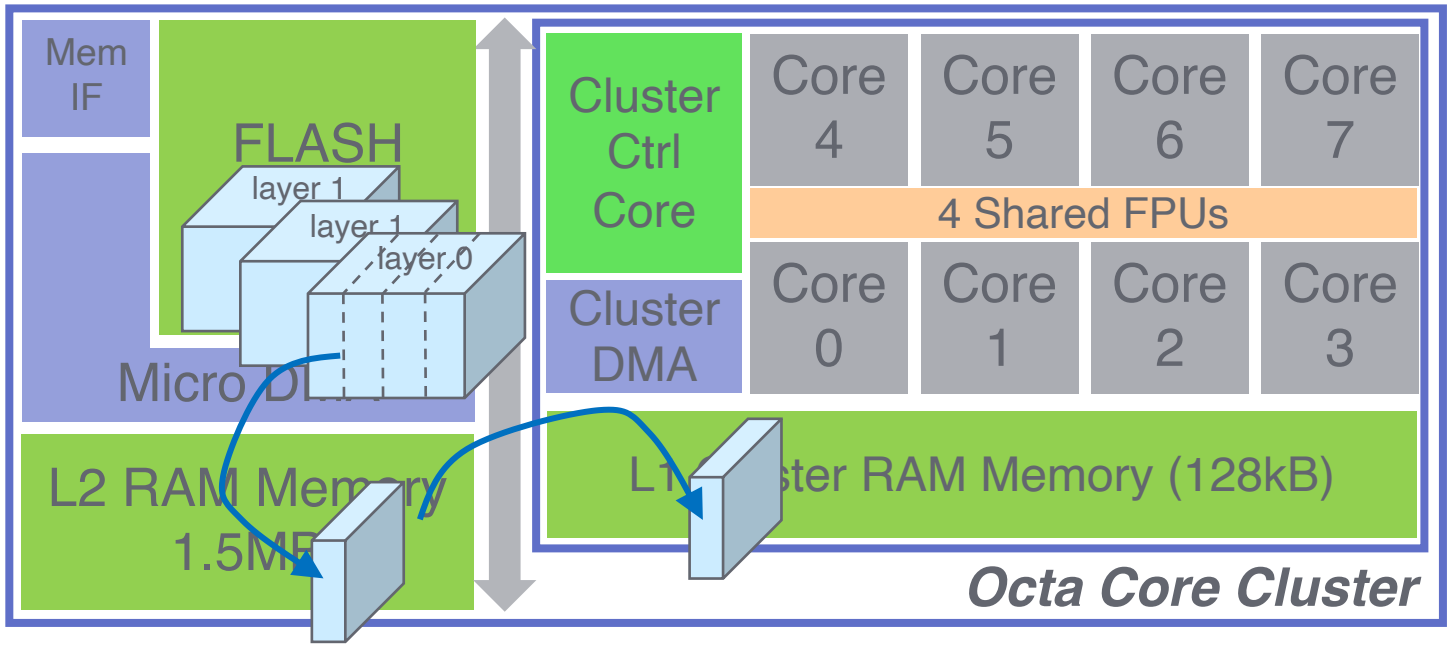
CC core interleaves memory transfers and compute tasks

Memory copies and computation happens **concurrently**

$$T_i = N_{tile} \cdot \boldsymbol{max}(T_{i,tile}^{L3-L2}, T_{i,tile}^{L2-L1}, T_{i,tile}^{cores})$$

GREENWAVES TECHNOLOGIES

Optimizations: Tensor Promotion



$$T_i = N_{tile} \cdot max(T_{i,tile}^{L3-L2}, T_{i,tile}^{L2-L1}, T_{i,tile}^{cores})$$

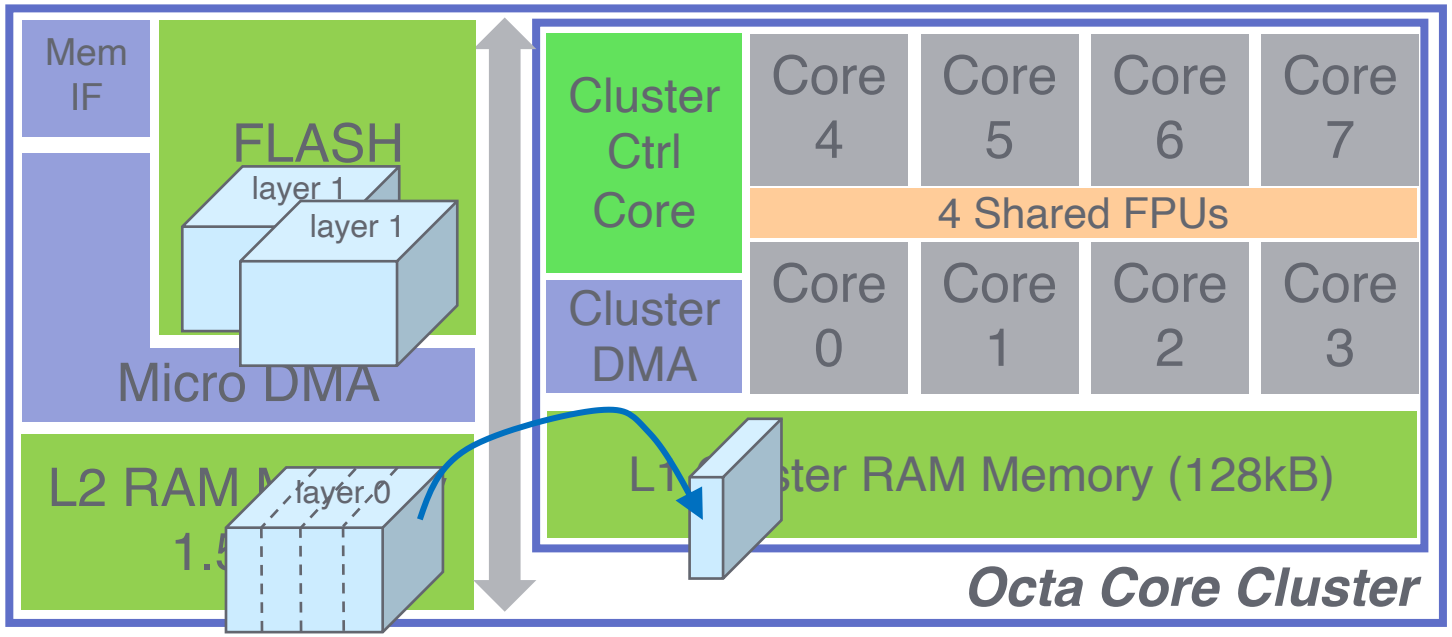
Bottlenecks (Int8 - Fp16):

- L3: 1B/Cyc = 1 - 0.5 MAC/Cyc
- L2: 8B/Cyc = 8 - 4 MAC/Cyc
- Parallel MV = 14 - 7 MAC/Cyc

For RNN Layers:
 # Params = # Ops

GREENWAVES TECHNOLOGIES

Optimizations: Tensor Promotion



Weight Tensor Promotion before inference

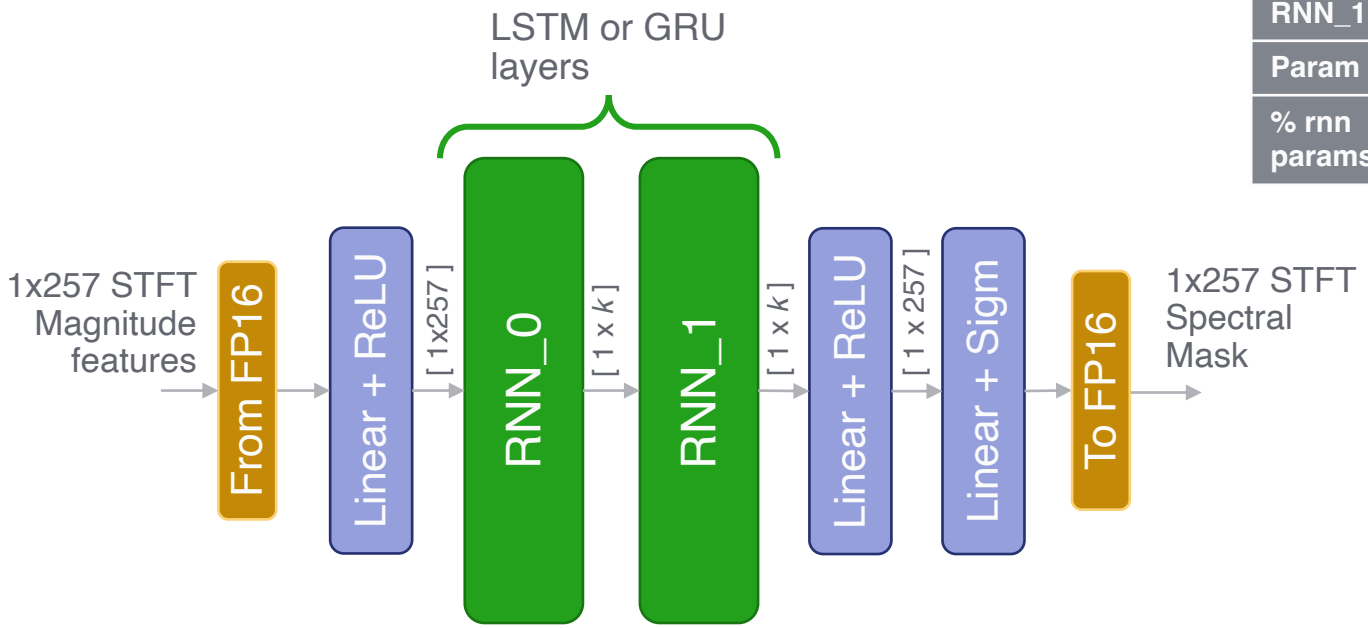
- Smaller models features a higher promotion rate and a faster execution

$$T_i = N_{tile} \cdot max(T_{i,tile}^{L3-L2}, T_{i,tile}^{L2-L1}, T_{i,tile}^{cores})$$

GREENWAVES TECHNOLOGIES

Post-Training Quantization

	LSTM256	GRU256	LSTM128	GRU128
k	256	256	128	128
RNN_0	LSTM(257,256)	GRU(257, 256)	LSTM(257,128)	GRU(257, 128)
RNN_1	LSTM(257,256)	GRU(257, 256)	LSTM(128 128)	GRU(128, 128)
Param	1.24 M	0.985 M	0.493 M	0.411 M
% rnn params	84%	80%	66.5%	59.8%



TinyDenoiser models trained on Valentini dataset: FP32 baseline

Quant Type	LSTM256			GRU256			LSTM128			GRU128		
	PESQ	STOI	Mem	PESQ	STOI	Mem	PESQ	STOI	Mem	PESQ	STOI	Mem
FP32	2.78	0.94	4.75	2.78	0.94	3.76	2.76	0.94	1.88	2.69	0.94	1.56
FP16	2.78	0.94	2.37	2.78	0.94	1.88	2.76	0.94	0.94	2.69	0.94	0.78

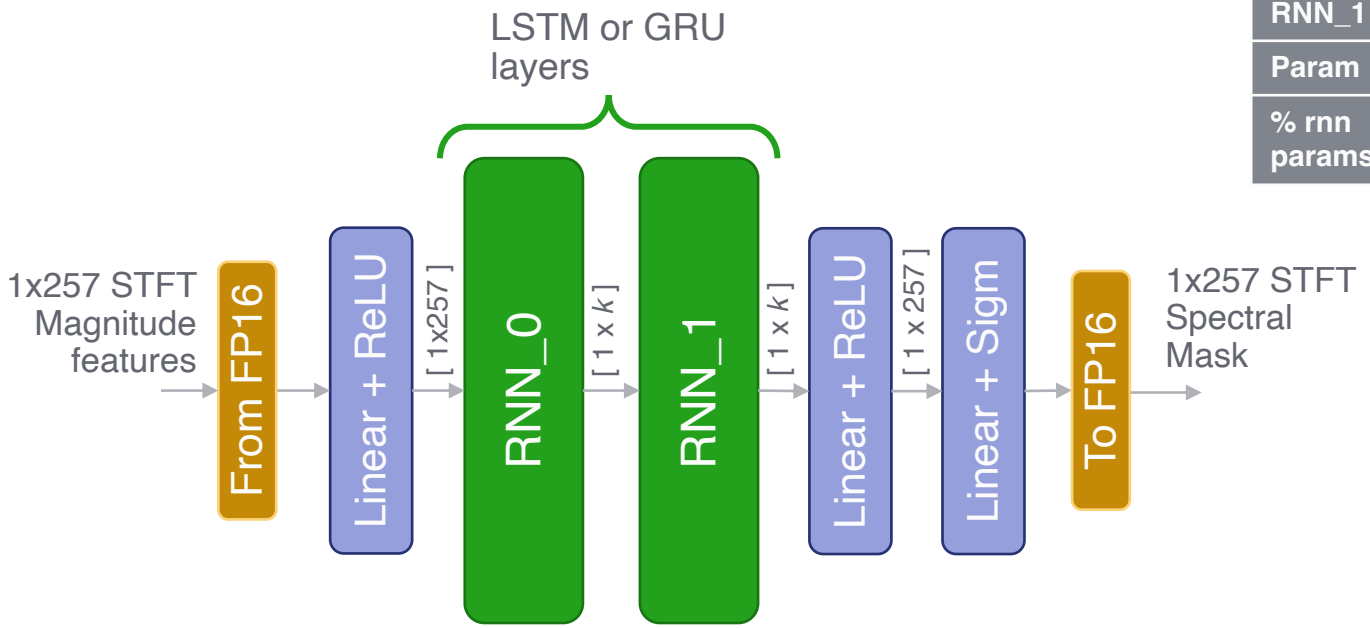
FP16, casting from FP32 to FP16, **lossless**

(and data-free)

GREENWAVES TECHNOLOGIES

Post-Training Quantization

	LSTM256	GRU256	LSTM128	GRU128
k	256	256	128	128
RNN_0	LSTM(257,256)	GRU(257, 256)	LSTM(257,128)	GRU(257, 128)
RNN_1	LSTM(257,256)	GRU(257, 256)	LSTM(128 128)	GRU(128, 128)
Param	1.24 M	0.985 M	0.493 M	0.411 M
% rnn params	84%	80%	66.5%	59.8%



The majority of the weights are due to RNN layers!

TinyDenoiser models trained on Valentini dataset: FP32 baseline

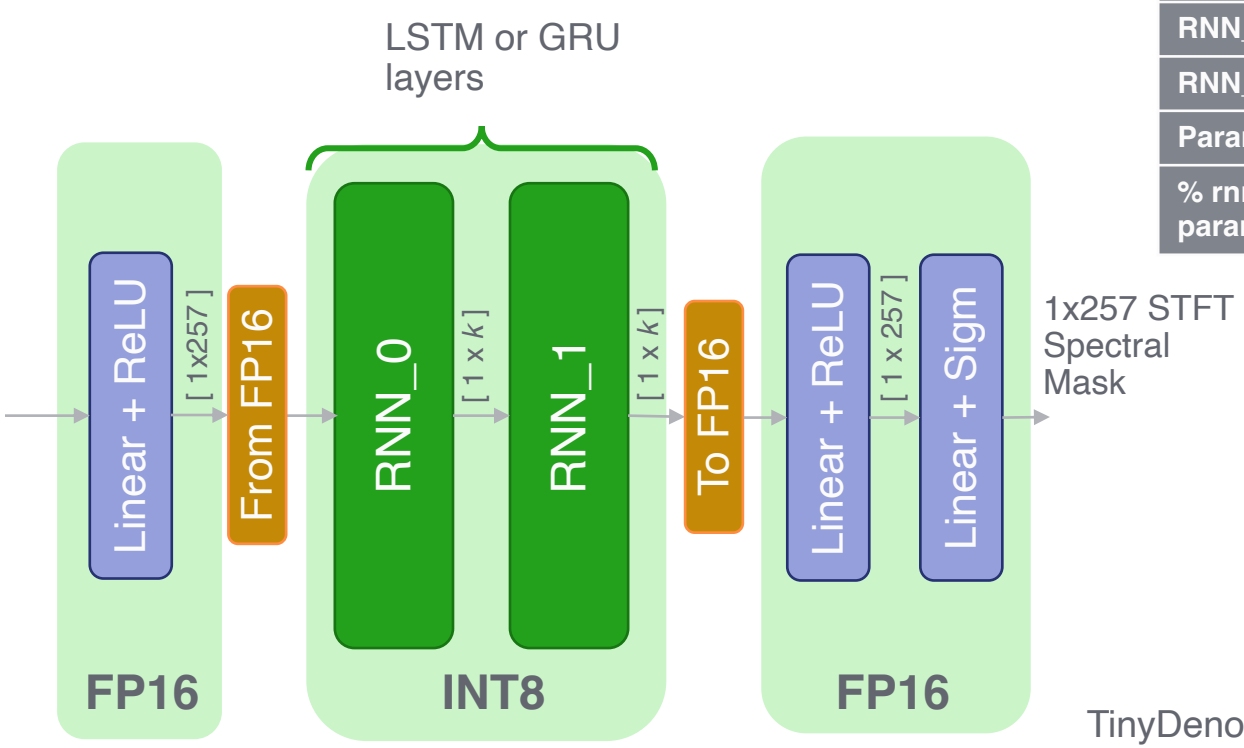
Quant Type	LSTM256			GRU256			LSTM128			GRU128		
	PESQ	STOI	Mem	PESQ	STOI	Mem	PESQ	STOI	Mem	PESQ	STOI	Mem
FP32	2.79	0.94	4.75	2.78	0.94	3.76	2.76	0.94	1.88	2.69	0.94	1.56
FP16	2.79	0.94	2.37	2.78	0.94	1.88	2.76	0.94	0.94	2.69	0.94	0.78
INT8	2.42	0.92	1.18	2.48	0.93	0.93	2.51	0.92	0.47	2.36	0.93	0.39

INT8 is lightweight but **LOSSY**

- avg PESQ loss: **-0.3**, STOI loss: **0.015**
- **2x** mem compression

GREENWAVES TECHNOLOGIES

Post-Training Quantization



	LSTM256	GRU256	LSTM128	GRU128
k	256	256	128	128
RNN_0	LSTM(257,256)	GRU(257, 256)	LSTM(257,128)	GRU(257, 128)
RNN_1	LSTM(257,256)	GRU(257, 256)	LSTM(128 128)	GRU(128, 128)
Param	1.24 M	0.985 M	0.493 M	0.411 M
% rnn params	84%	80%	66.5%	59.8%

The majority of the weights are due to RNN layers!

No need for expensive QAT !!!

TinyDenoiser models trained on Valentini dataset: FP32 baseline

Quant Type	LSTM256			GRU256			LSTM128			GRU128		
	PESQ	STOI	Mem	PESQ	STOI	Mem	PESQ	STOI	Mem	PESQ	STOI	Mem
FP32	2.78	0.94	4.75	2.78	0.94	3.76	2.76	0.94	1.88	2.69	0.94	1.56
FP16	2.78	0.94	2.37	2.78	0.94	1.88	2.76	0.94	0.94	2.69	0.94	0.78
INT8	2.42	0.92	1.18	2.48	0.93	0.93	2.51	0.92	0.47	2.36	0.93	0.39
MixFP16-INT8	2.73	0.93	1.37	2.72	0.94	1.13	2.69	0.93	0.67	2.63	0.94	0.55

MixedFP16-INT8 present low accuracy degradation (**PESQ: 0.06, STOI: <0.01**) while **1.4-1.7x** mem compression vs FP16

GREENWAVES TECHNOLOGIES

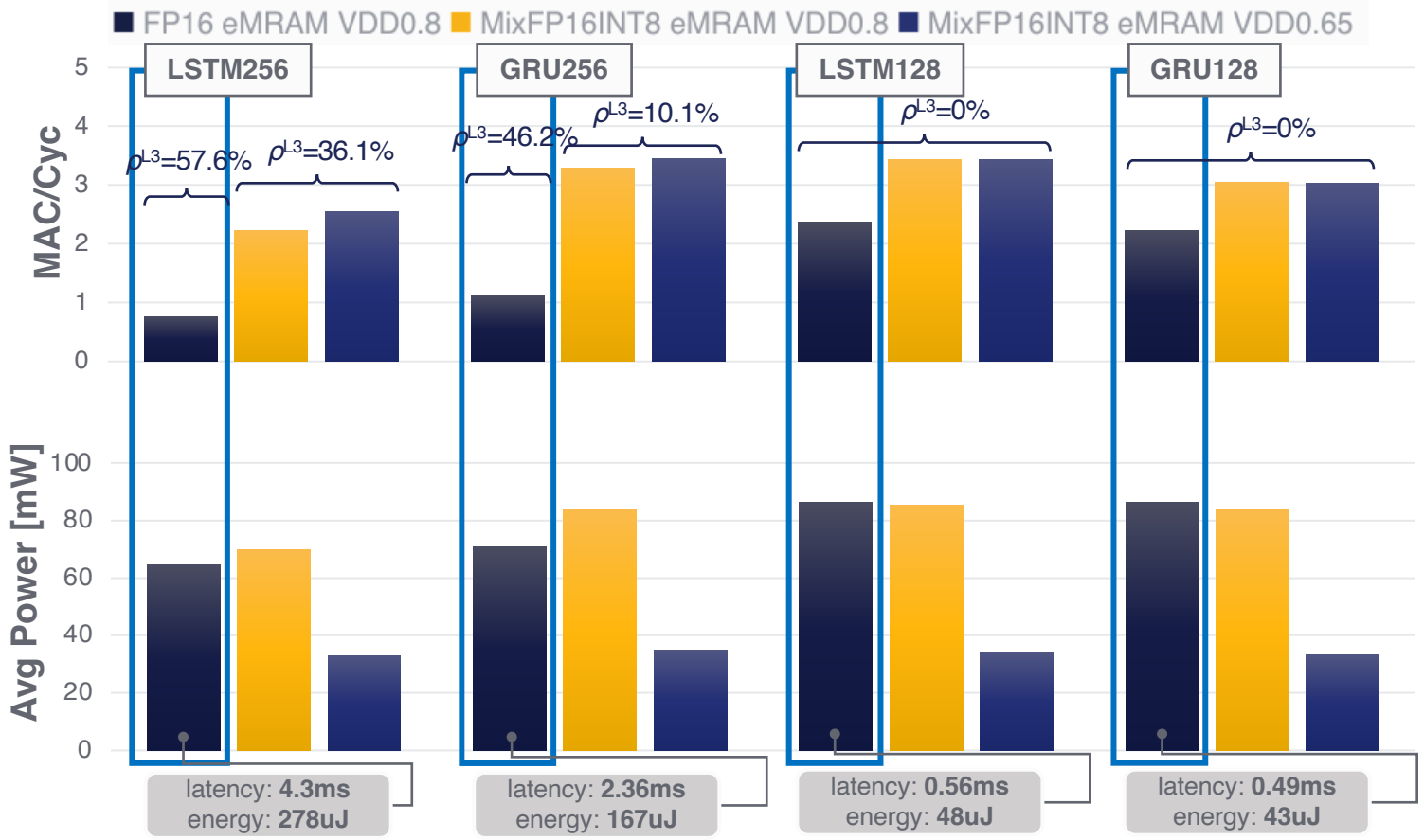
Latency & Power on target HW/SW

Due to their size, FP16 LSTM256 and GRU256 are **L3 memory bound** given high ρ^{L3}

- efficiency up to 1.13 MAC/cyc

FP16 LSTM128 and GRU128 can fully promote their tensors to L2 reducing ρ^{L3} to 0

- efficiency up to 2.2 MAC/cyc



Benchmark on 1+8 core 22nm chip (GAP9)

- V_{DD} = 0.8V, f_{max} = 370 MHz
- V_{DD} = 0.65V, f_{max} = 240 MHz

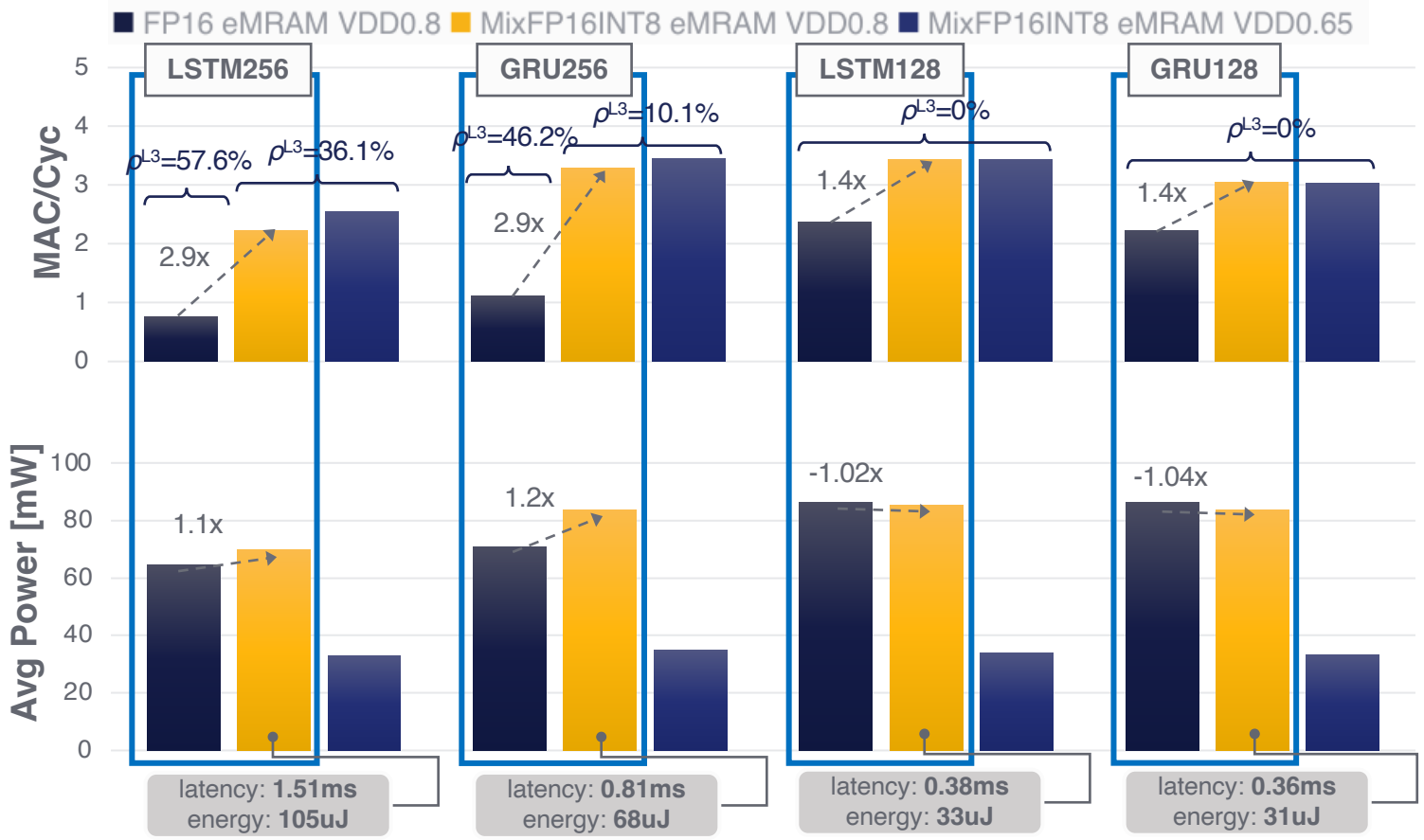
Metrics

- **MAC/cyc** to measure code efficiency
- **Power Consumption** (mW)
- ρ^{L3}: ratio of L3 params of the total

Latency & Power on target HW/SW

LSTM256 and GRU256 decrease ρ^{L3} thanks to Mixed-Precision
- avg. eff. up to 3.4 MAC/cyc
- higher avg. power cost because of higher operation density (but **lower energy** overall)

LSTM128 and GRU128 improves MAC/cyc thanks to Mixed Precision (SIMD int8 operations)



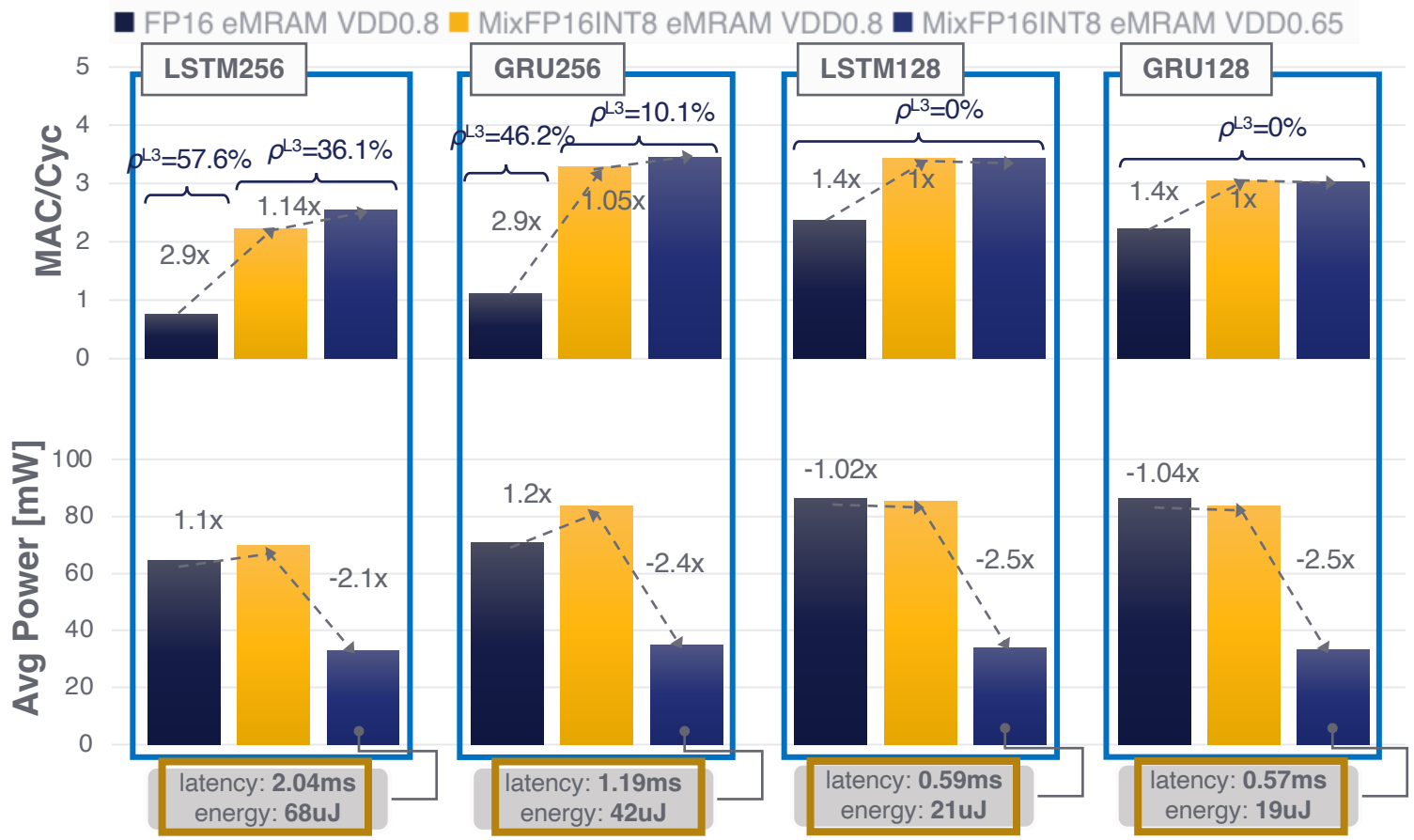
Benchmark on 1+8 core 22nm chip (GAP9)
- V_{DD} = 0.8V, f_{max} = 370 MHz
- V_{DD} = 0.65V, f_{max} = 240 MHz

Metrics
- **MAC/cyc** to measure code efficiency
- **Power Consumption** (mW)
- ρ^{L3}: ratio of L3 params of the total

Latency & Power on target HW/SW

Scaling down the voltage reduces the power by 2-2.5x

For LSTM256 and GRU256 efficiency slightly improves (5-15%) because on-chip FLASH increases BW at low freq since they still use L3



Benchmark on 1+8 core 22nm chip

- V_{DD} = 0.8V, f_{max} = 370 MHz
- V_{DD} = 0.65V, f_{max} = 240 MHz

- **Matched** the realtime Constraint of **6.25ms**
- Assuming **negligible power in sleep state:** avg power of **3mW**

- ρ^{L3}: ratio of L3 params of the total

Conclusions

Our optimized design for Speech Enhancement on MCUs consists of multiple elements:

- Multi-Core acceleration with vector FP16 and INT8 ISA support
- SW pipeline with interleaved memory transfers and computation
- Tensor promotion mechanism to reduce far-from-compute data transfers
- Mixed FP16-INT8 precision to gain INT8 speed and FP16 accuracy

Thank you

Email: marco.fariselli@greenwaves-technologies.com

TinyDenoiser Article: https://arxiv.org/pdf/2210.07692.pdf

GREENWAVES TECHNOLOGIES

Copyright Notice

www.tinyml.org