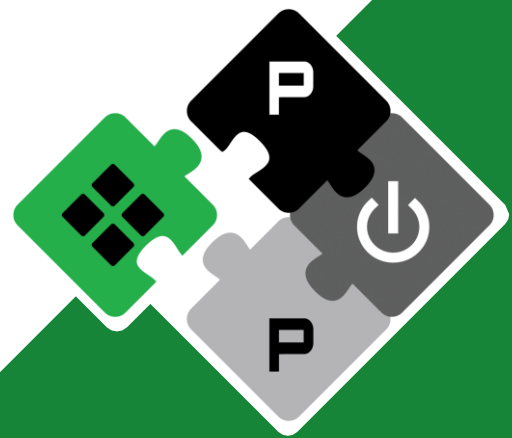
ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Towards Neuromorphic Acceleration through Register-Streaming Extensions on RISC-V Cores

University of Bologna

Simone Manoni s.manoni@unibo.it
Paul Scheffler
Alfio Di Mauro
Luca Zanatta
Andrea Acquaviva
Luca Benini
Andrea Bartolini



PULP Platform
Open Source Hardware, the way it should be!

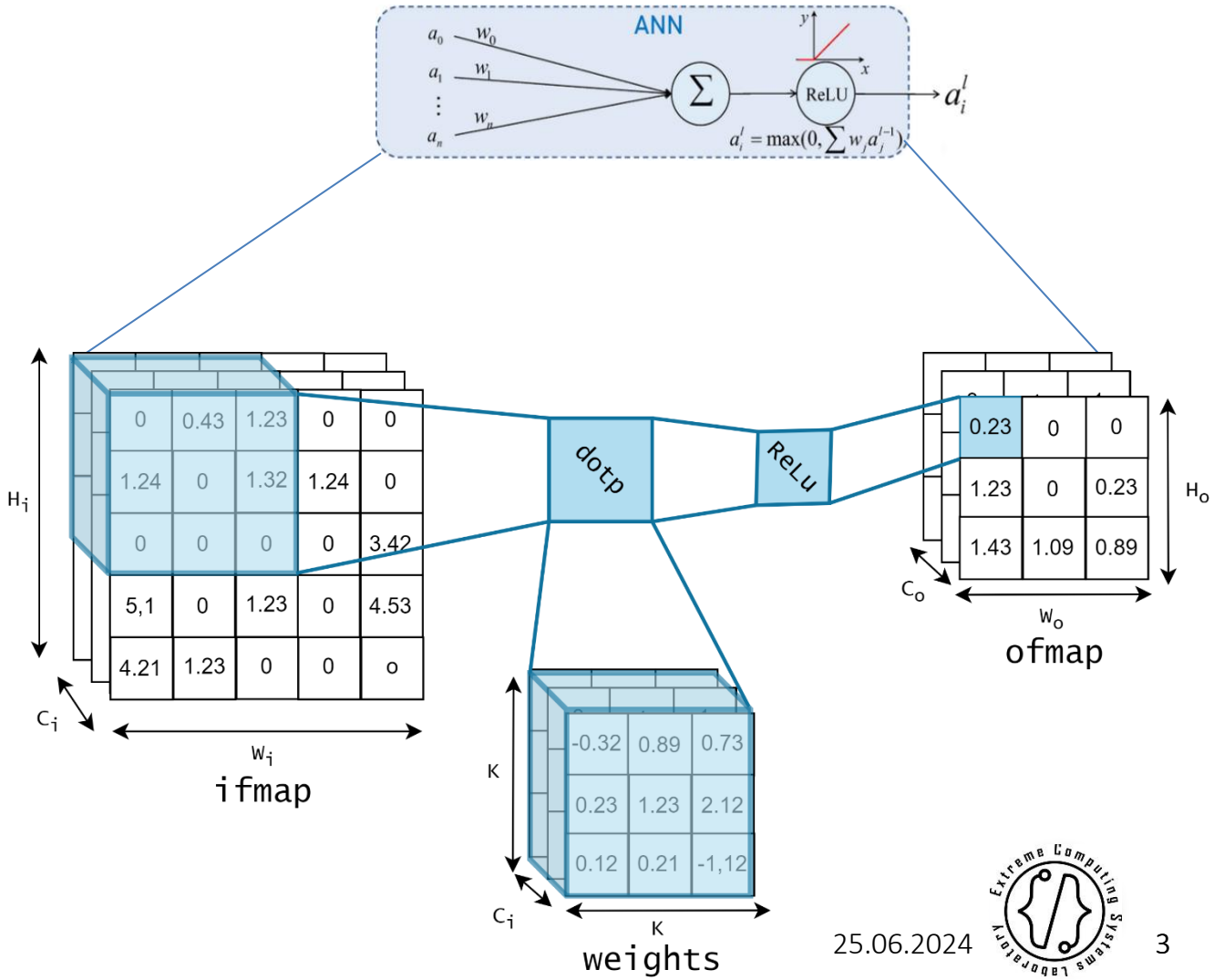
@pulp_platform
pulp-platform.org
youtube.com/pulp_platform

Outline

- **Neuromorphic Computing/SNNs**
- Neuromorphic/SNNs Accelerators
- Snitch Cluster architecture
- Contributions & Sparse conv layer example:
  - Compressed format
  - Dataflow in Snitch
  - Parallelization in Snitch cluster
- Results & Conclusions

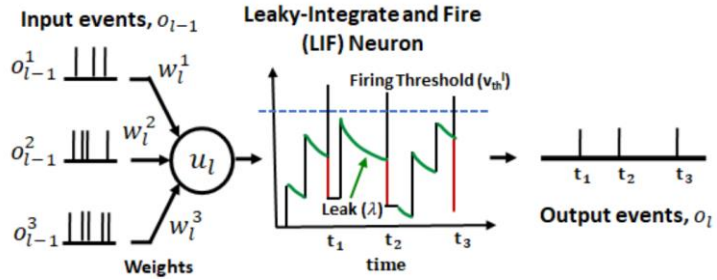
Neuromorphic Computing

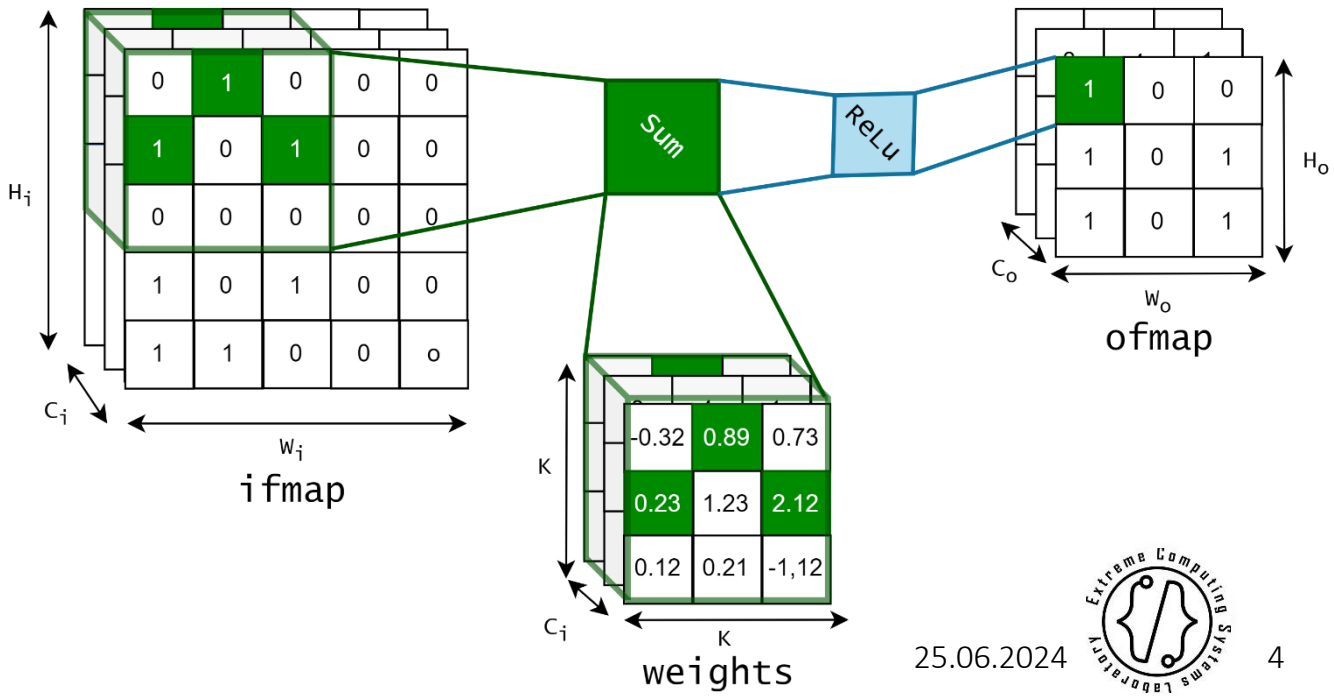
- **Computing technology based on SNNs**
- **SNNs vs. ANNs – key differences**



Neuromorphic Computing

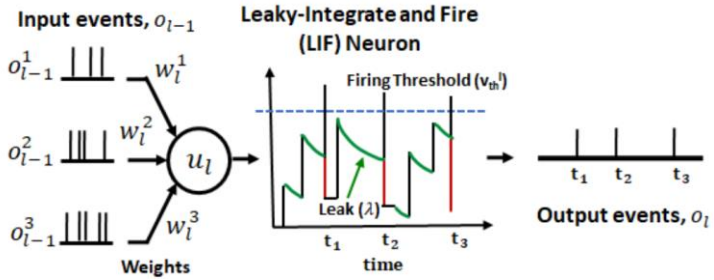
- **Computing technology based on SNNs**
- **SNNs vs. ANNs – key differences**
  - Neurons emit spikes to comunicate each other (sequences of 1s and 0s)

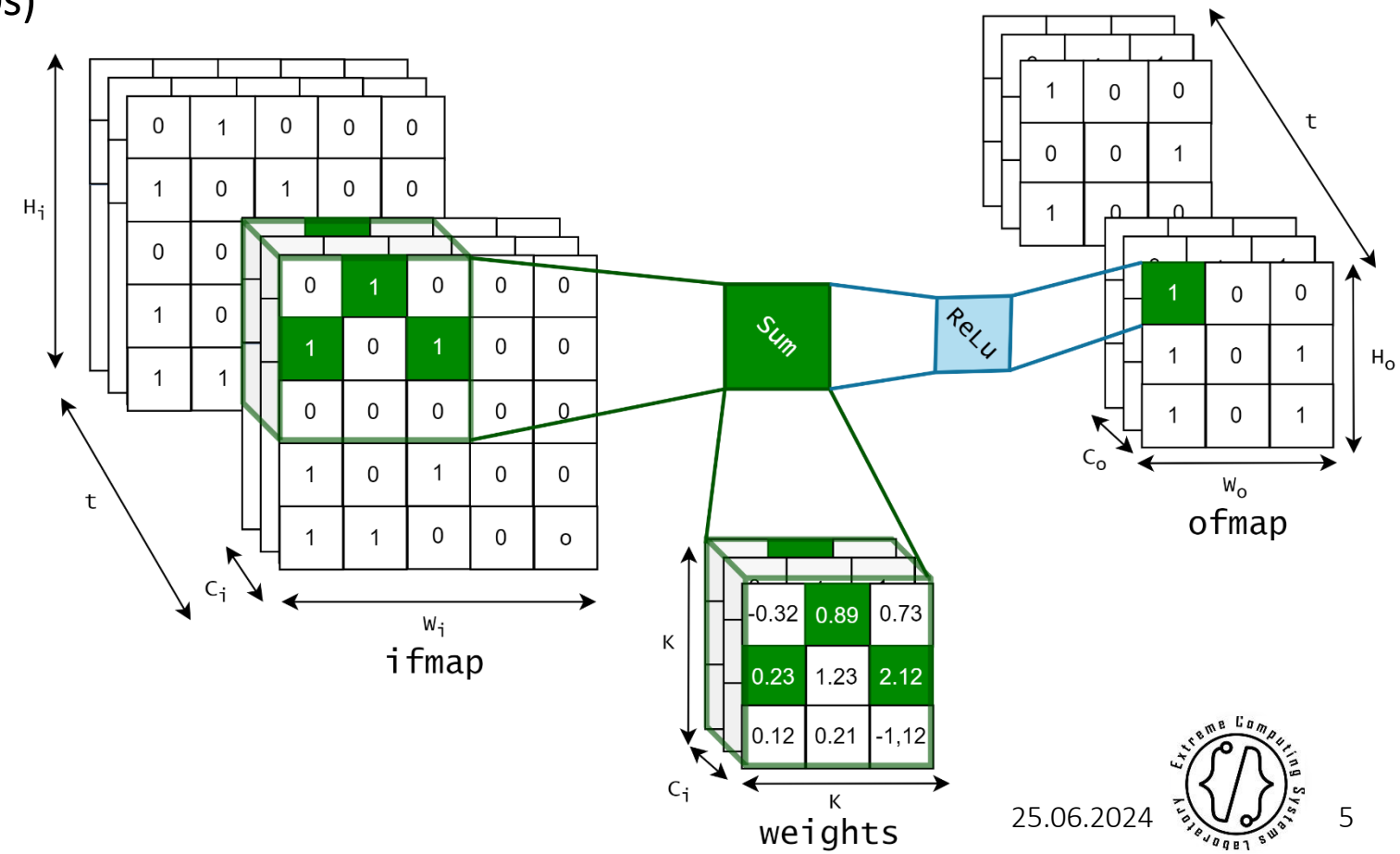




Neuromorphic Computing

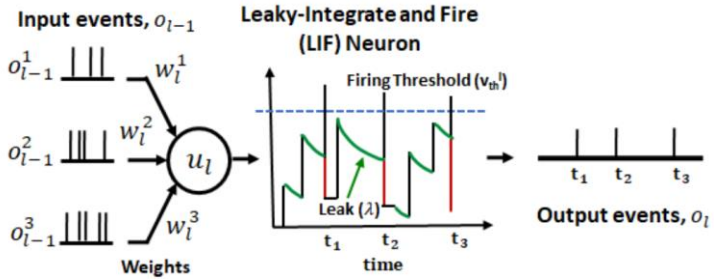
- **Computing technology based on SNNs**
- **SNNs vs. ANNs – key differences**
  - Neurons emit spikes to comunicate each other (sequences of 1s and 0s)
  - Incorporate temporal dimension

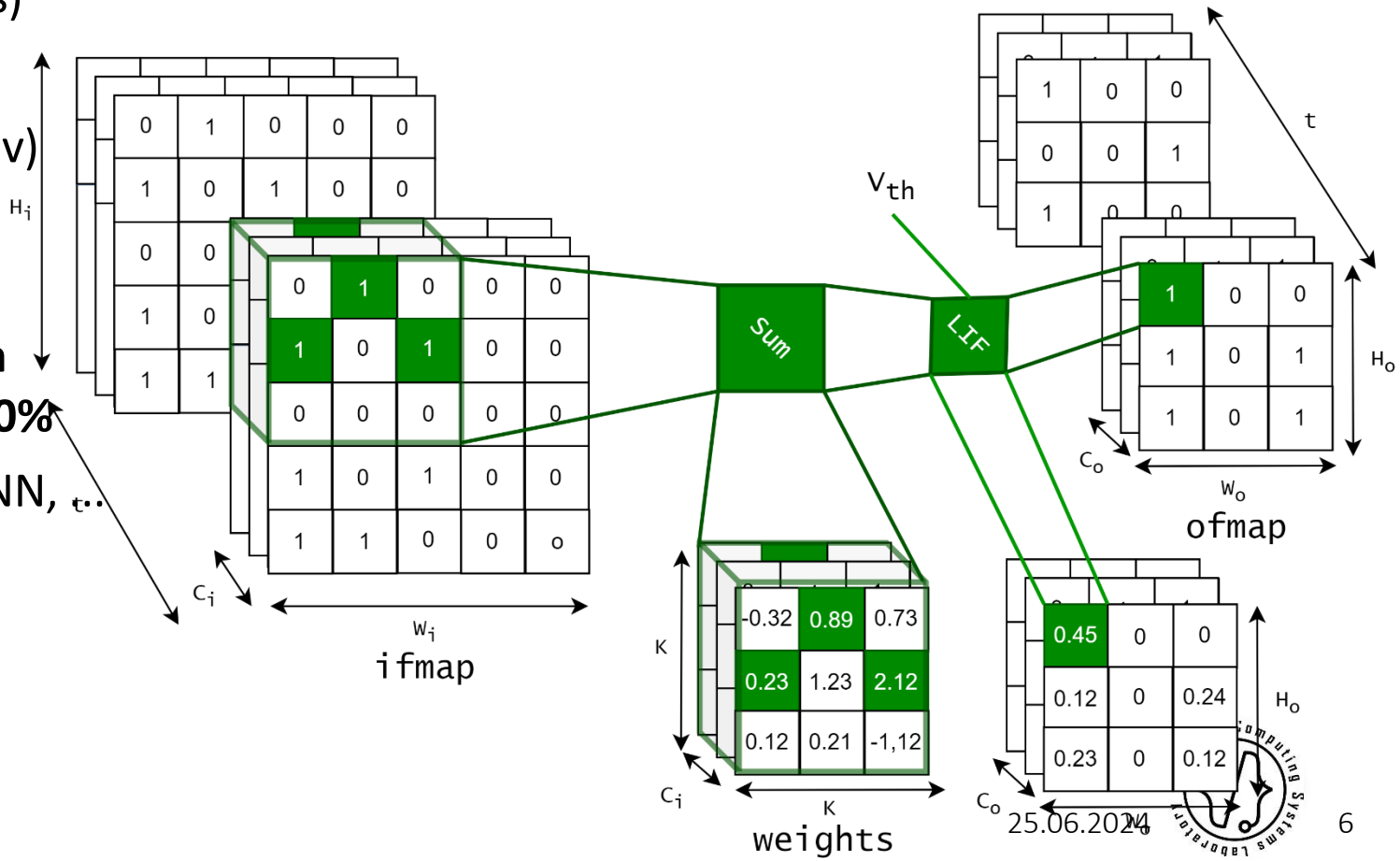




Neuromorphic Computing

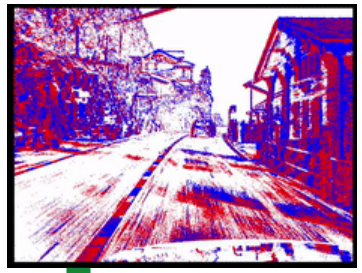
- **Computing technology based on SNNs**
- **SNNs vs. ANNs – key differences**
  - Neurons emit spikes to comunicate each other (sequences of 1s and 0s)
  - Incorporate temporal dimension
  - Each neuron has an internal state (v)
  - Dynamic activation function: Leaky-Integrate and Fire (LIF)
  - **Exhibit a high degree of sparsity in the input features (ifmaps) 60%-90%**
  - Same ANNs topology: FC, RNNs, CNN, ...



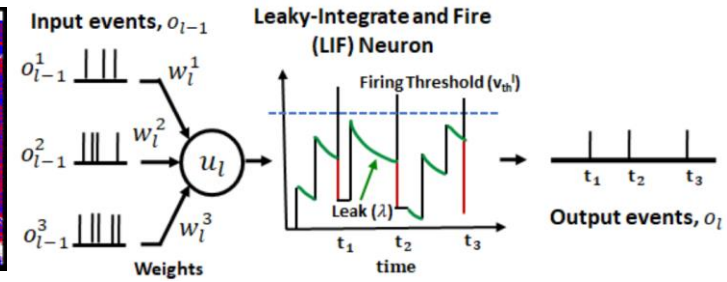


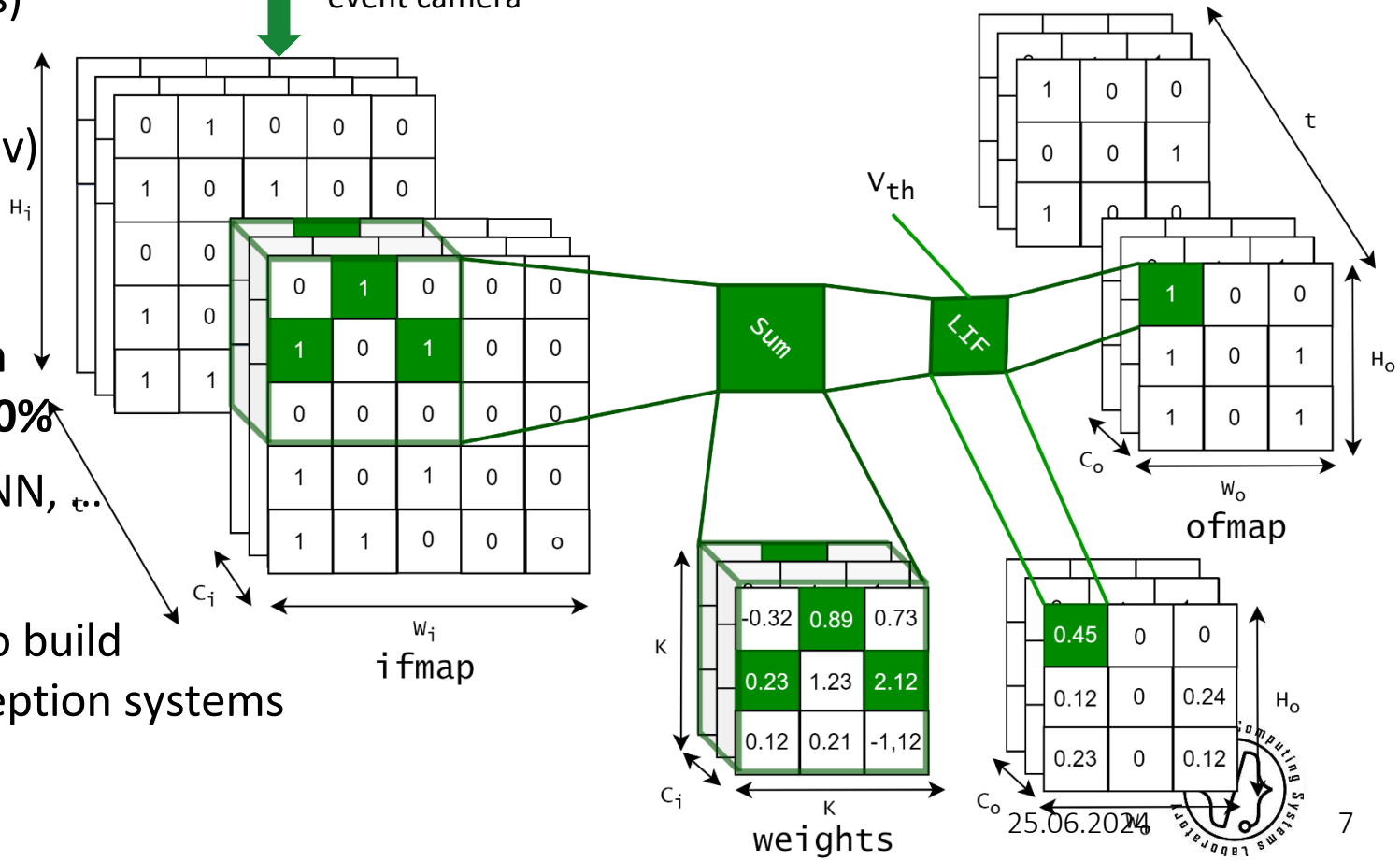
Neuromorphic Computing

- **Computing technology based on SNNs**
- **SNNs vs. ANNs – key differences**
  - Neurons emit spikes to comunicate each other (sequences of 1s and 0s)
  - Incorporate temporal dimension
  - Each neuron has an internal state (v)
  - Dynamic activation function: Leaky-Integrate and Fire (LIF)
  - **Exhibit a high degree of sparsity in the input features (ifmaps) 60%-90%**
  - Same ANNs topology: FC, RNNs, CNN, ...
  - **MAC replaced by ADD (Syn. Op.)**
  - **Spiking CNNs** promising solution to build energy-efficient event-based perception systems



event camera

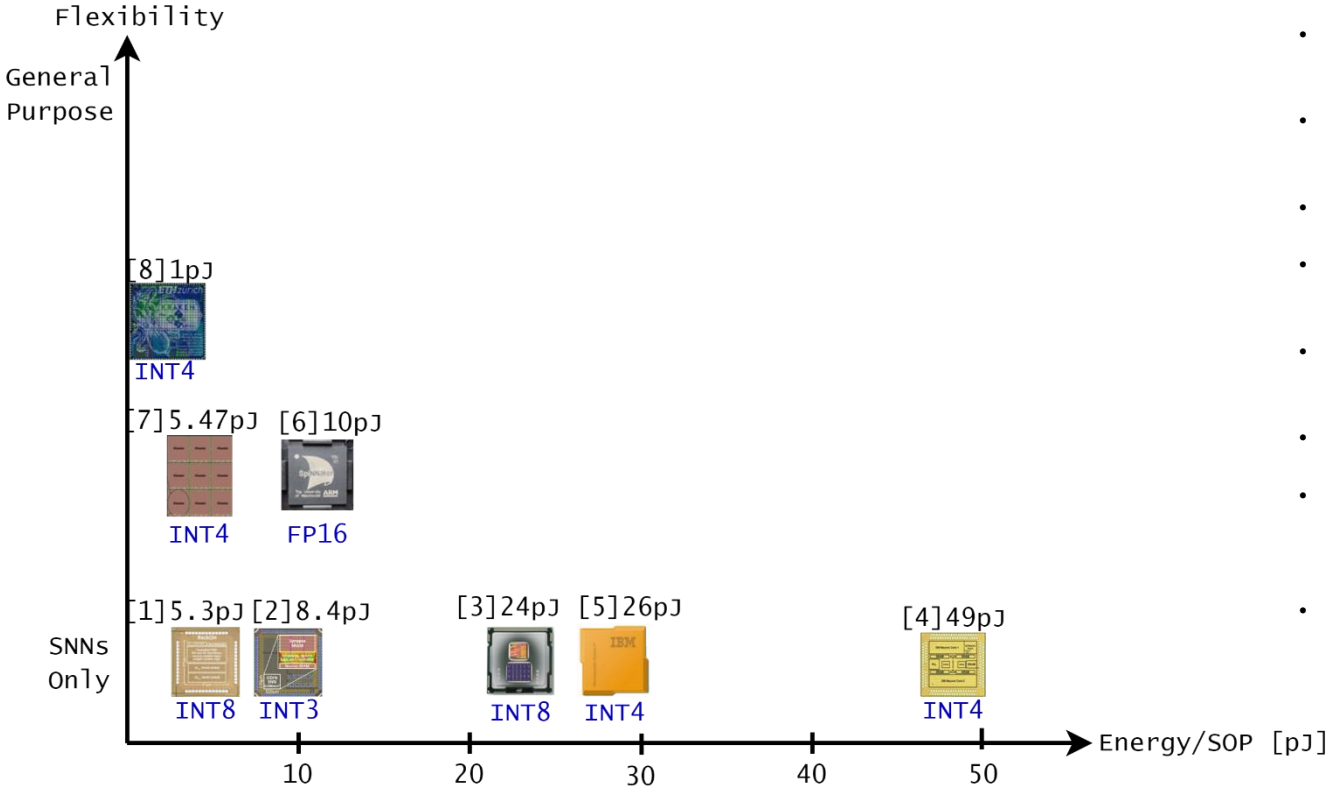




Outline

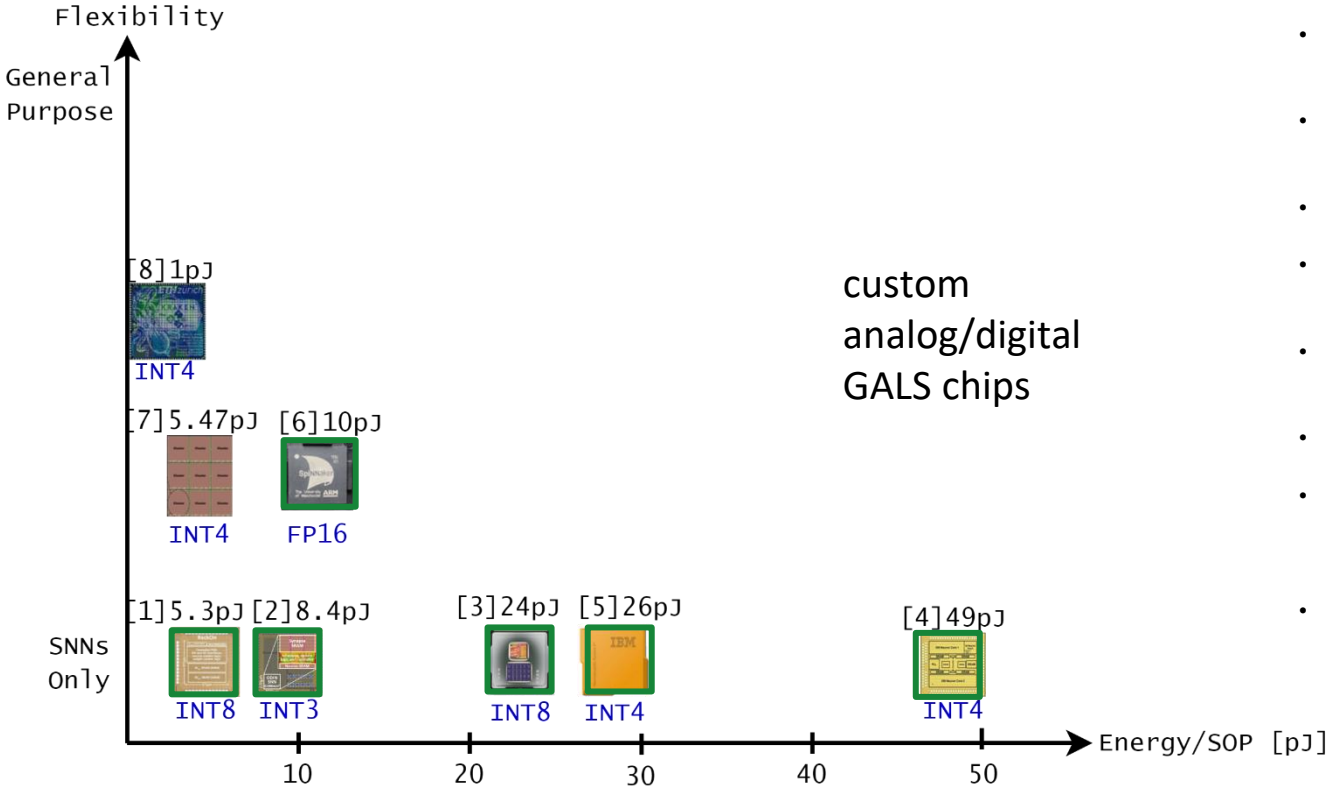
- Neuromorphic Computing/SNNs
- **Neuromorphic/SNNs Accelerators**
- Snitch Cluster architecture
- Contributions & Sparse conv layer example:
  - Compressed format
  - Dataflow in Snitch
  - Parallelization in Snitch cluster
- Results & Conclusions

Neuromorphic Accelerators



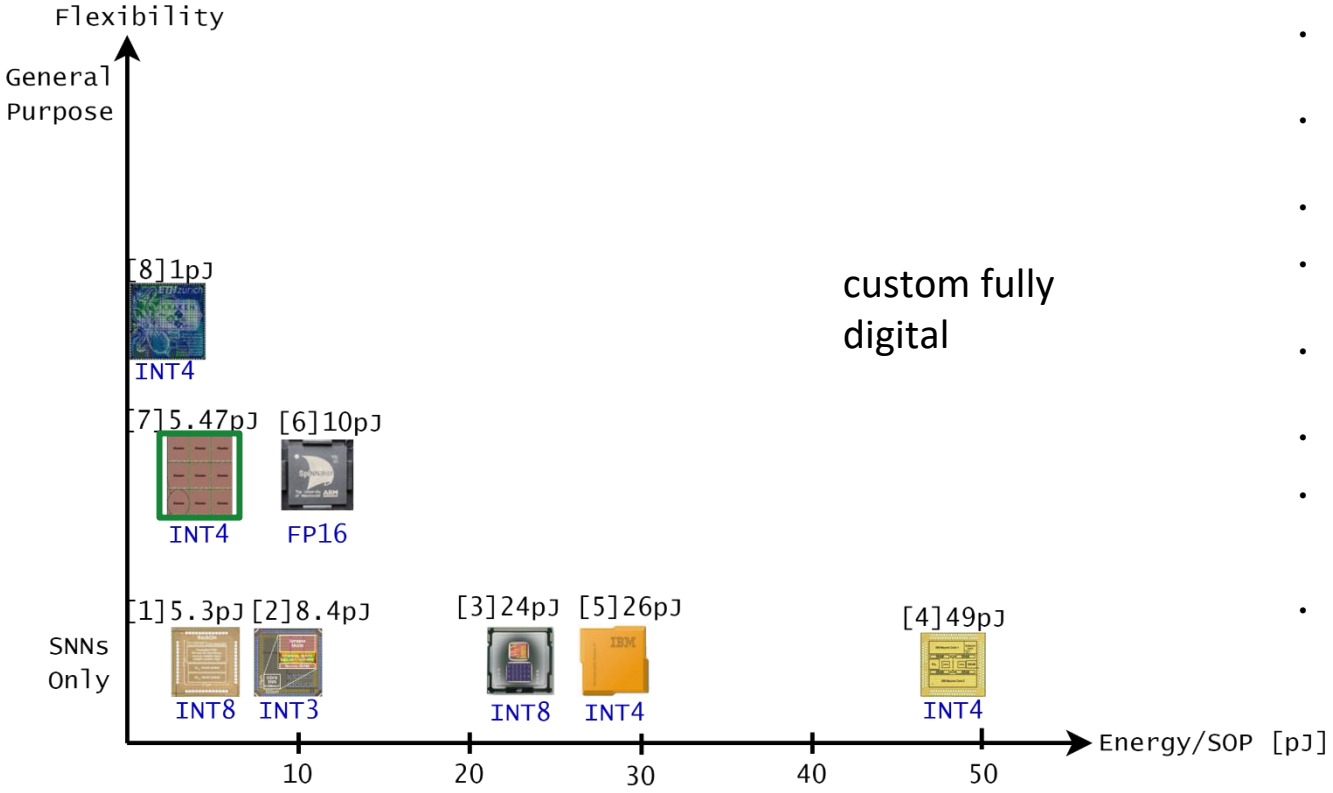
- *[1] C. Frenkel, M. Lefebvre, J. -D. Legat and D. Bol, "A 0.086-mm2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS," in IEEE Transactions on Biomedical Circuits and Systems*
- *[2] C. Frenkel and G. Indiveri, "ReckOn: A 28nm Sub-mm2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales," 2022 IEEE International Solid-State Circuits Conference*
- *[3] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro*
- *[4] F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang and M. P. Flynn, "A 3.43TOPS/W 48.9pJ/pixel 50.1nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS," 2017 Symposium on VLSI Circuits*
- *[5] F. Akopyan et al., "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*
- *[6] E. Painkras et al., "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," in IEEE Journal of Solid-State Circuits*
- *[7] Zhijie Yang, Lei Wang, Yao Wang, Linghui Peng, Xiaofan Chen, Xun Xiao, Yaohua Wang, and Weixia Xu. 2022. Unicorn: a multicore neuromorphic processor with flexible fan-in and unconstrained fan-out for neurons. In Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC '22)*
- *[8] Di Mauro, A., Scherer, M., Rossi, D., and Benini, L., "Kraken: A Direct Event/Frame-Based Multi-sensor Fusion SoC for Ultra-Efficient Visual Processing in Nano-UAVs", Hot Chips Symposium 2022 (HCS '22).*

Neuromorphic Accelerators



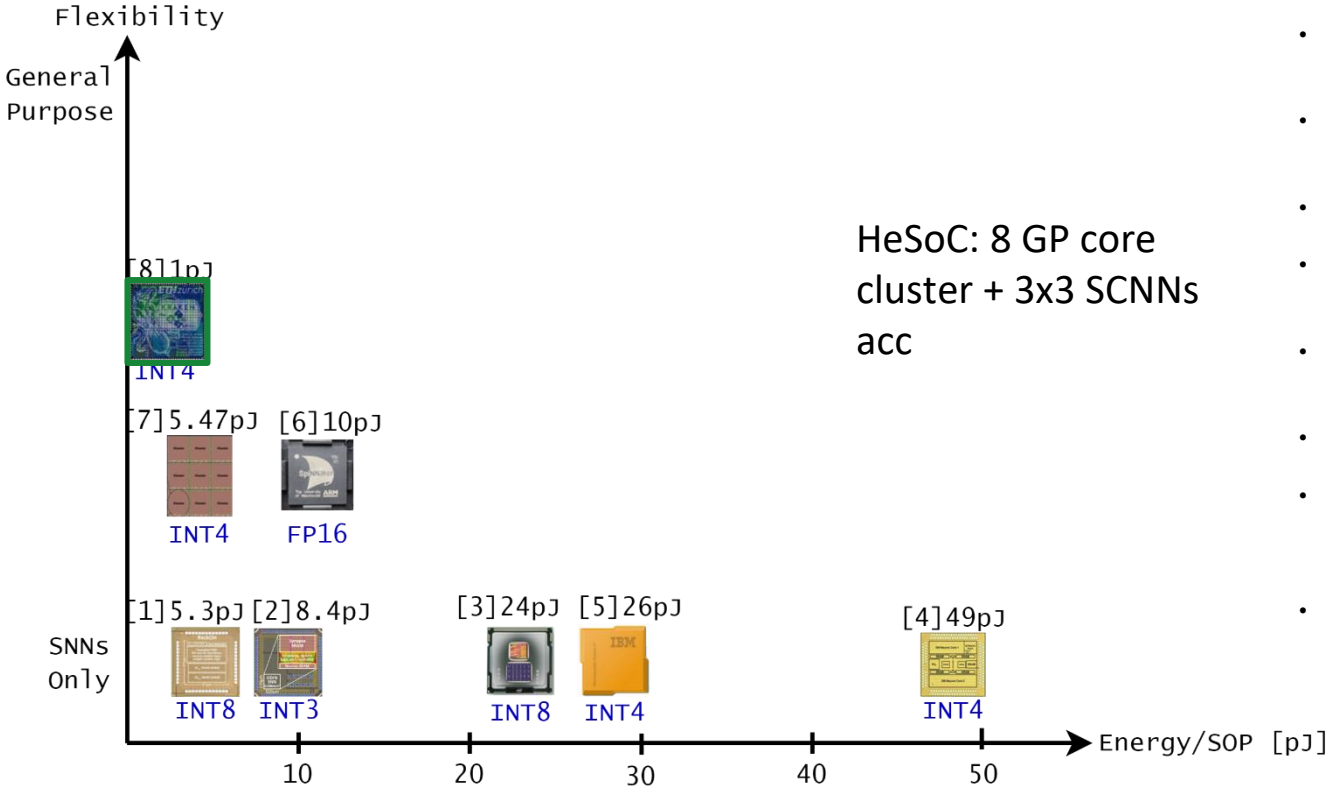
- *[1] C. Frenkel, M. Lefebvre, J. -D. Legat and D. Bol, "A 0.086-mm2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS," in IEEE Transactions on Biomedical Circuits and Systems*
- *[2] C. Frenkel and G. Indiveri, "ReckOn: A 28nm Sub-mm2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales," 2022 IEEE International Solid-State Circuits Conference*
- *[3] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro*
- *[4] F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang and M. P. Flynn, "A 3.43TOPS/W 48.9pJ/pixel 50.1nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS," 2017 Symposium on VLSI Circuits*
- *[5] F. Akopyan et al., "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*
- *[6] E. Painkras et al., "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," in IEEE Journal of Solid-State Circuits*
- *[7] Zhijie Yang, Lei Wang, Yao Wang, Linghui Peng, Xiaofan Chen, Xun Xiao, Yaohua Wang, and Weixia Xu. 2022. Unicorn: a multicore neuromorphic processor with flexible fan-in and unconstrained fan-out for neurons. In Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC '22)*
- *[8] Di Mauro, A., Scherer, M., Rossi, D., and Benini, L., "Kraken: A Direct Event/Frame-Based Multi-sensor Fusion SoC for Ultra-Efficient Visual Processing in Nano-UAVs", Hot Chips Symposium 2022 (HCS '22).*

Neuromorphic Accelerators



- *[1] C. Frenkel, M. Lefebvre, J. -D. Legat and D. Bol, "A 0.086-mm2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS," in IEEE Transactions on Biomedical Circuits and Systems*
- *[2] C. Frenkel and G. Indiveri, "ReckOn: A 28nm Sub-mm2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales," 2022 IEEE International Solid-State Circuits Conference*
- *[3] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro*
- *[4] F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang and M. P. Flynn, "A 3.43TOPS/W 48.9pJ/pixel 50.1nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS," 2017 Symposium on VLSI Circuits*
- *[5] F. Akopyan et al., "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*
- *[6] E. Painkras et al., "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," in IEEE Journal of Solid-State Circuits*
- *[7] Zhijie Yang, Lei Wang, Yao Wang, Linghui Peng, Xiaofan Chen, Xun Xiao, Yaohua Wang, and Weixia Xu. 2022. Unicorn: a multicore neuromorphic processor with flexible fan-in and unconstrained fan-out for neurons. In Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC '22)*
- *[8] Di Mauro, A., Scherer, M., Rossi, D., and Benini, L., "Kraken: A Direct Event/Frame-Based Multi-sensor Fusion SoC for Ultra-Efficient Visual Processing in Nano-UAVs", Hot Chips Symposium 2022 (HCS '22).*

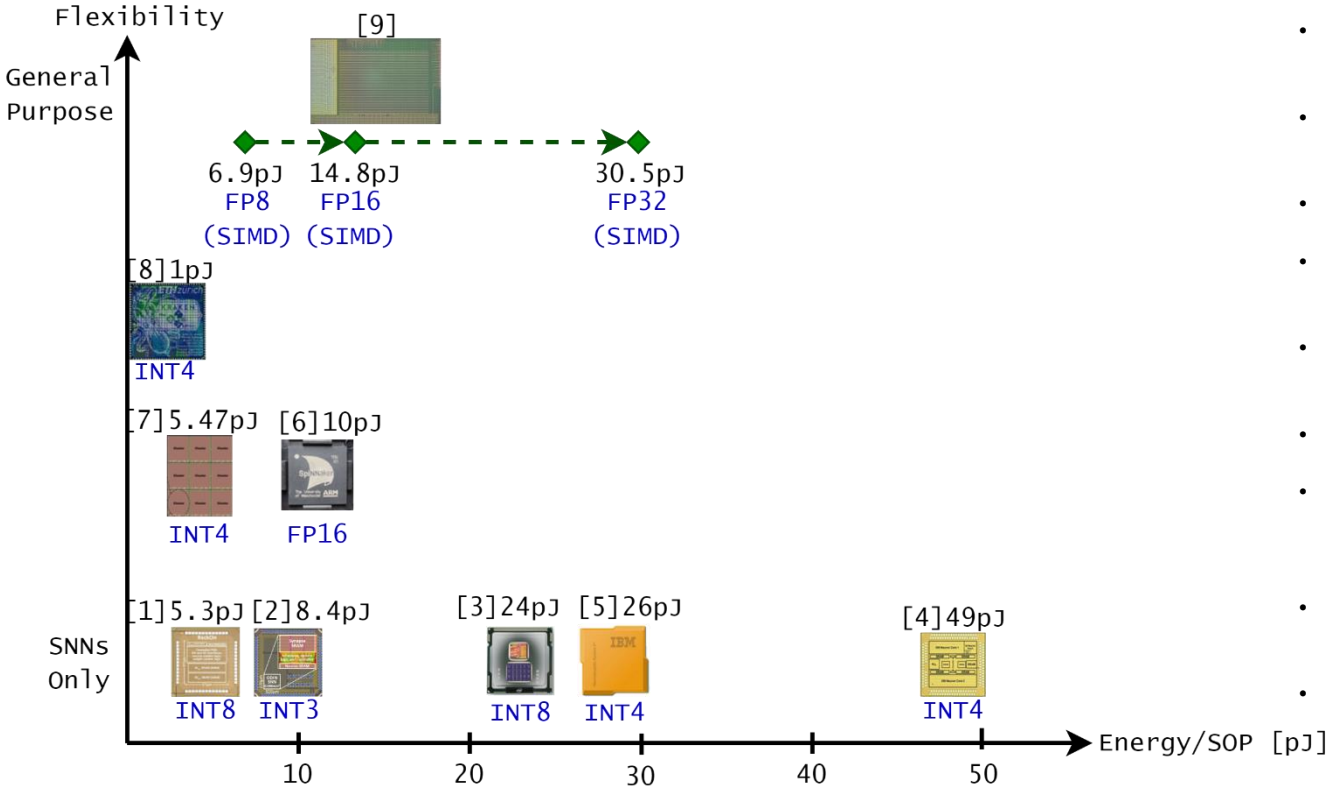
Neuromorphic Accelerators



HeSoC: 8 GP core cluster + 3x3 SCNNs acc

- *[1] C. Frenkel, M. Lefebvre, J. -D. Legat and D. Bol, "A 0.086-mm2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS," in IEEE Transactions on Biomedical Circuits and Systems*
- *[2] C. Frenkel and G. Indiveri, "ReckOn: A 28nm Sub-mm2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales," 2022 IEEE International Solid-State Circuits Conference*
- *[3] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro*
- *[4] F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang and M. P. Flynn, "A 3.43TOPS/W 48.9pJ/pixel 50.1nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS," 2017 Symposium on VLSI Circuits*
- *[5] F. Akopyan et al., "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*
- *[6] E. Painkras et al., "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," in IEEE Journal of Solid-State Circuits*
- *[7] Zhijie Yang, Lei Wang, Yao Wang, Linghui Peng, Xiaofan Chen, Xun Xiao, Yaohua Wang, and Weixia Xu. 2022. Unicorn: a multicore neuromorphic processor with flexible fan-in and unconstrained fan-out for neurons. In Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC '22)*
- *[8] Di Mauro, A., Scherer, M., Rossi, D., and Benini, L., "Kraken: A Direct Event/Frame-Based Multi-sensor Fusion SoC for Ultra-Efficient Visual Processing in Nano-UAVs", Hot Chips Symposium 2022 (HCS '22).*

Neuromorphic Accelerators + Snitch



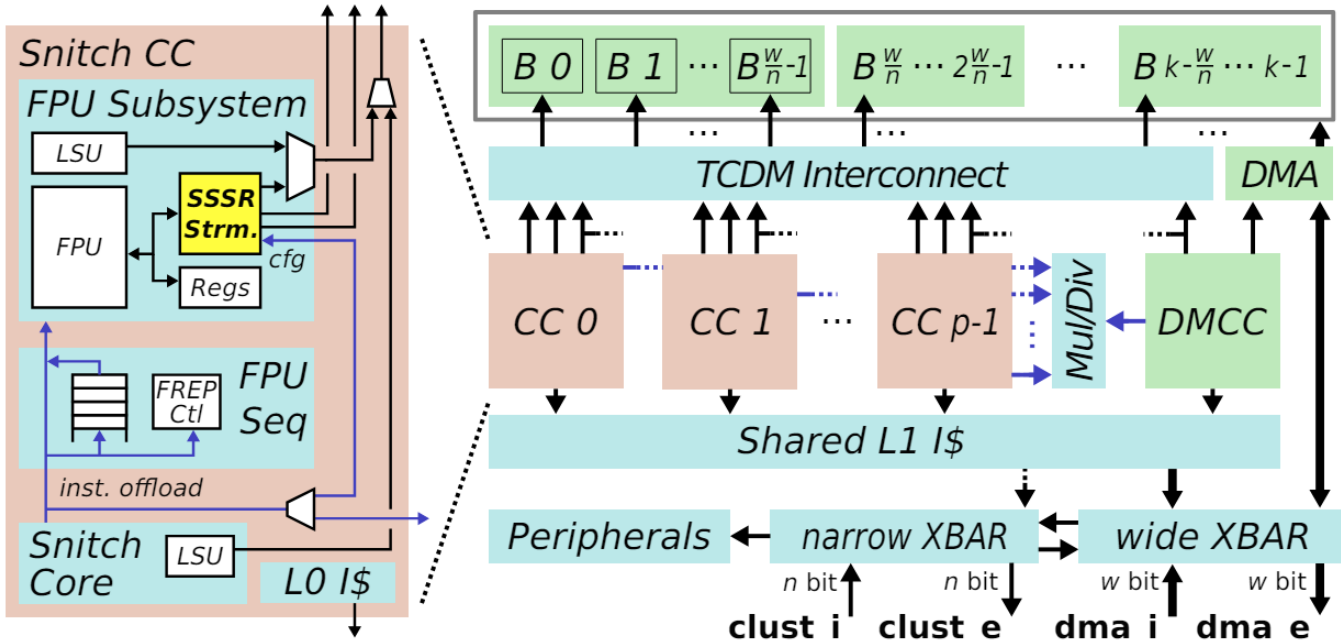
- Already many SNNs accelerators, but their flexibility is limited
- Lack of kernel libraries on GP platforms for SNNs inference
- Snitch core (inside *[9]*) offers support for Sparse Linear Algebra and a competitive Energy/SOP (add)

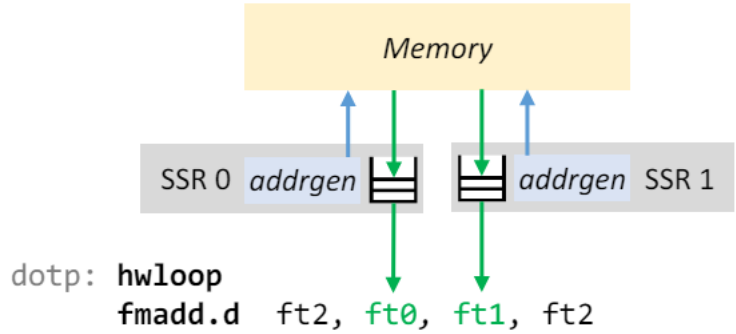
- *[1] C. Frenkel, M. Lefebvre, J. -D. Legat and D. Bol, "A 0.086-mm2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS," in IEEE Transactions on Biomedical Circuits and Systems*
- *[2] C. Frenkel and G. Indiveri, "ReckOn: A 28nm Sub-mm2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales," 2022 IEEE International Solid-State Circuits Conference*
- *[3] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro*
- *[4] F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang and M. P. Flynn, "A 3.43TOPS/W 48.9pJ/pixel 50.1nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS," 2017 Symposium on VLSI Circuits*
- *[5] F. Akopyan et al., "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*
- *[6] E. Painkras et al., "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," in IEEE Journal of Solid-State Circuits*
- *[7] Zhijie Yang, Lei Wang, Yao Wang, Linghui Peng, Xiaofan Chen, Xun Xiao, Yaohua Wang, and Weixia Xu. 2022. Unicorn: a multicore neuromorphic processor with flexible fan-in and unconstrained fan-out for neurons. In Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC '22)*
- *[8] Di Mauro, A., Scherer, M., Rossi, D., and Benini, L., "Kraken: A Direct Event/Frame-Based Multi-sensor Fusion SoC for Ultra-Efficient Visual Processing in Nano-UAVs", Hot Chips Symposium 2022 (HCS '22).*
- *[9]* G. Paulin et al., "Occamy: A 432-core 28.1 dp-gflop/s/w 83% FPU utilization dual-chiplet, dual-HBM2E RISC-V-based accelerator for stencil and sparse linear algebra computations with 8-to-64-bit floatingpoint support in 12nm FinFET," VLSI Symposium, 2024

Outline

- Neuromorphic Computing/SNNs
- Neuromorphic/SNNs Accelerators
- **Snitch Cluster architecture**
- Contributions & Sparse conv layer example:
  - Compressed format
  - Dataflow in Snitch
  - Parallelization in Snitch cluster
- Results & Conclusions

Target Platform: Snitch Cluster





Streaming Semantic Registers

- **Snich Cluster**: 8 open-source RV32G cores each one paired with a double-precision FPU + DMA + Tightly Coupled Data Memory (TCDM) and logarithmic interconnect
- **Enhanced with RISC-V ISA Extensions:**
  - **Streaming Semantic Registers** (SSRs): Map registers to memory streams using address generators and data movers: Supports affine, **indirect** and sparse data accesses
  - **Floating-point repetition** (FREP): Hardware loops over floating point instructions.
  - **SmallFloat**: FP SIMD support down-to to FP8

[10]: P. Scheffler et al., "Sparse Stream Semantic Registers: A Lightweight ISA Extension Accelerating General Sparse Linear Algebra," in *IEEE Transactions on Parallel and Distributed Systems*

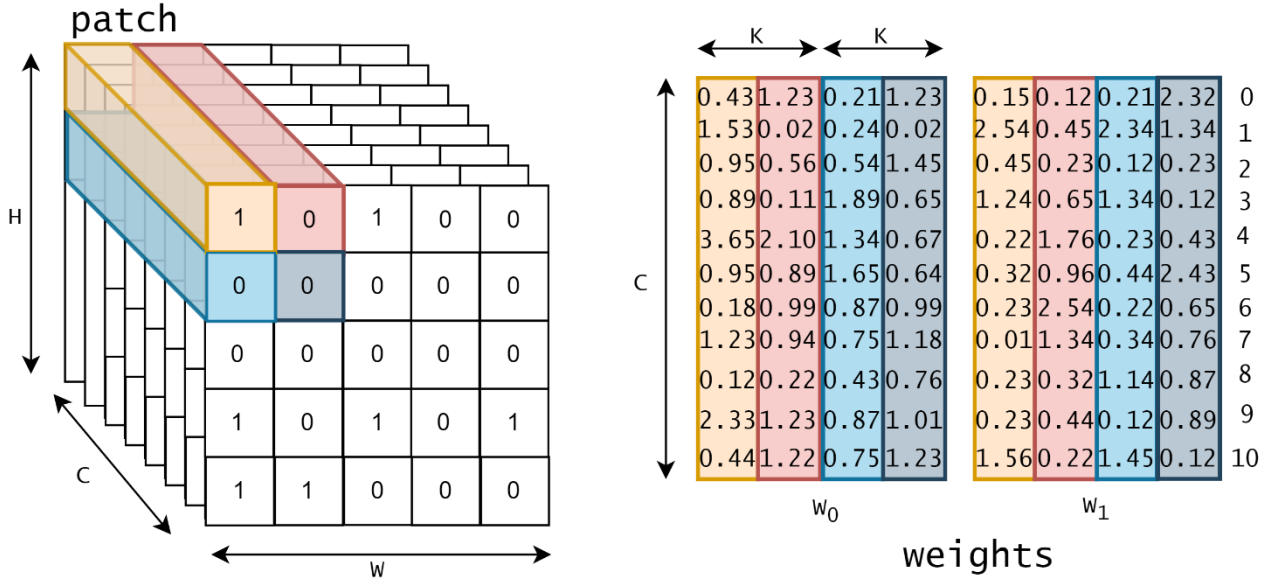
Outline

- Neuromorphic Computing/SNNs
- Neuromorphic/SNNs Accelerators
- Snitch Cluster architecture
- **Contributions & Sparse conv layer example:**
  - Compressed format
  - Dataflow in Snitch
  - Parallelization in Snitch cluster
- Results & Conclusions

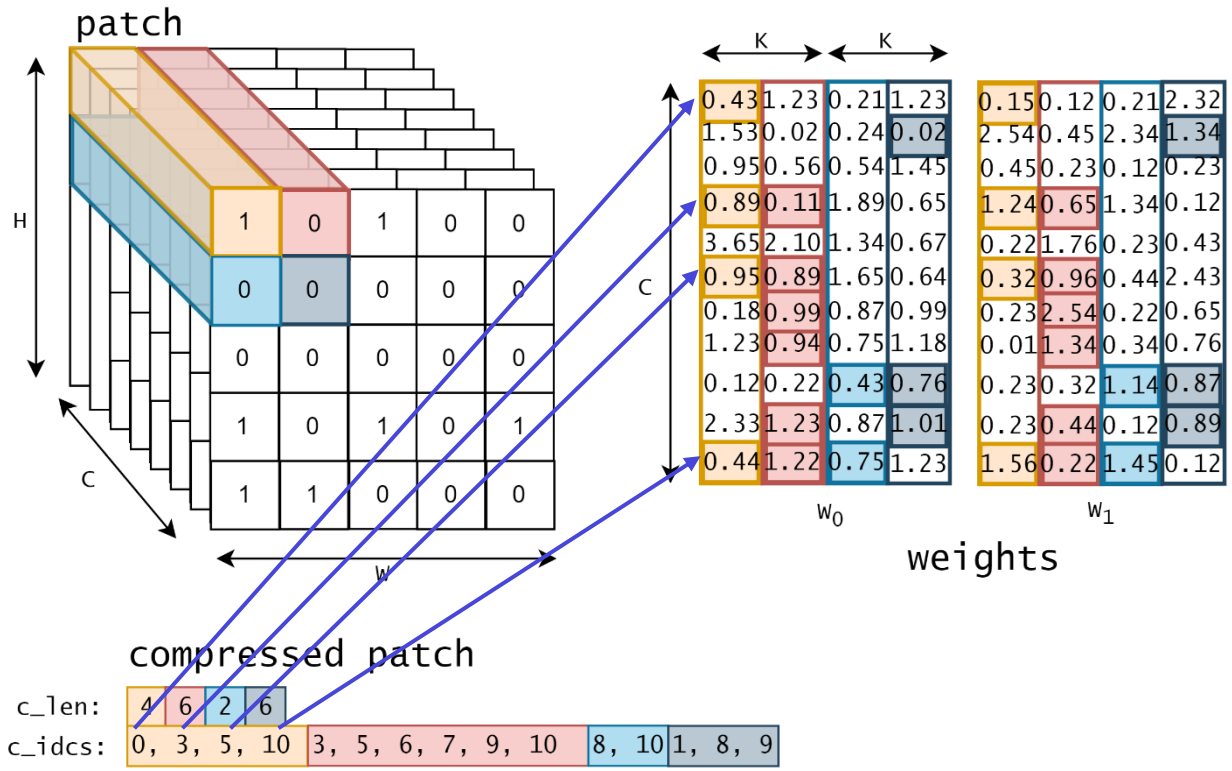
Contributions

- We propose a kernel library for SNNs Snitch based platforms
- We leverage Snitch's RISC-V ISA extensions to accelerate the baseline sparse computation
- We evaluate our optimized kernels w.r.t. the baseline in a cycle accurate simulation

Example Layer: Sparse Convolution – Baseline kernel



Example Layer: Sparse Convolution – Baseline kernel




```
for(int i = 0; i < K*K: i++)
        for(j=0; j < c_len[i]; j++)
                v += weights[c_idcs[j] + i * K];
```

```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        vfadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

- We adopt a compressed run-length format for each ifmap channel
- Compression brings in an indirection operation in the inner loop
- Only 1 (vfadd.s) out of 7 is a useful instruction in the baseline inner loop → Low FPU utilization: 14%

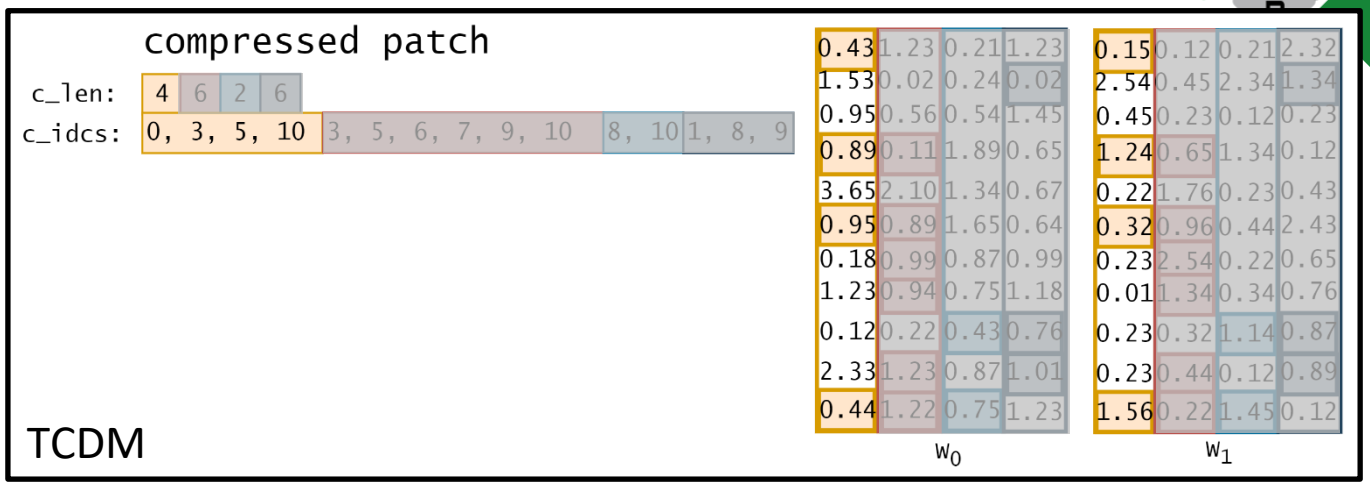
Example Layer: Sparse Convolution – Dataflow in Snitch

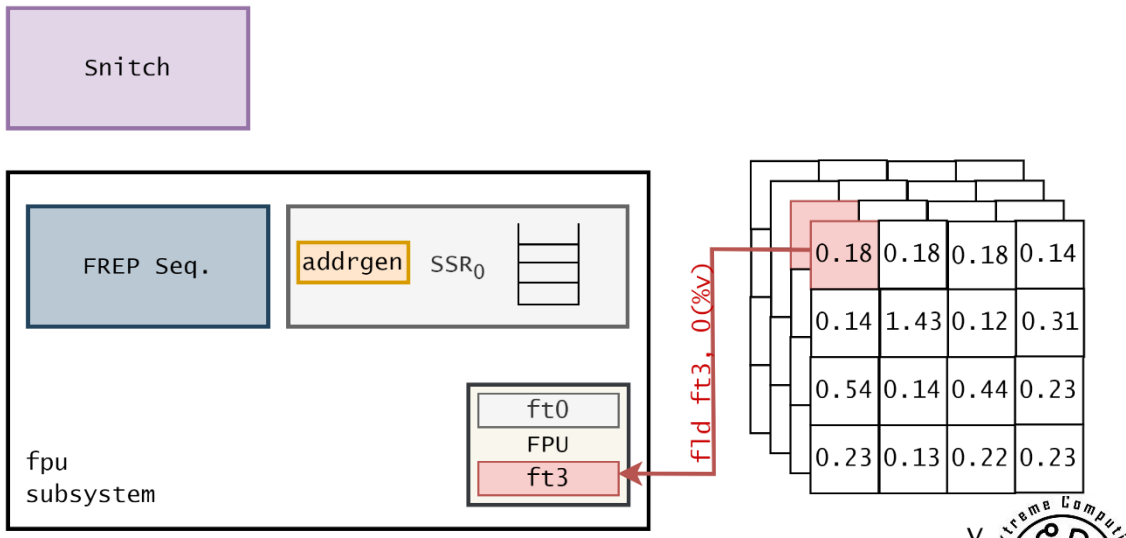
```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        fadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

Sparse baseline kernel

```
load v { fld ft3, 0(%v)
```

Snitch optimized kernel





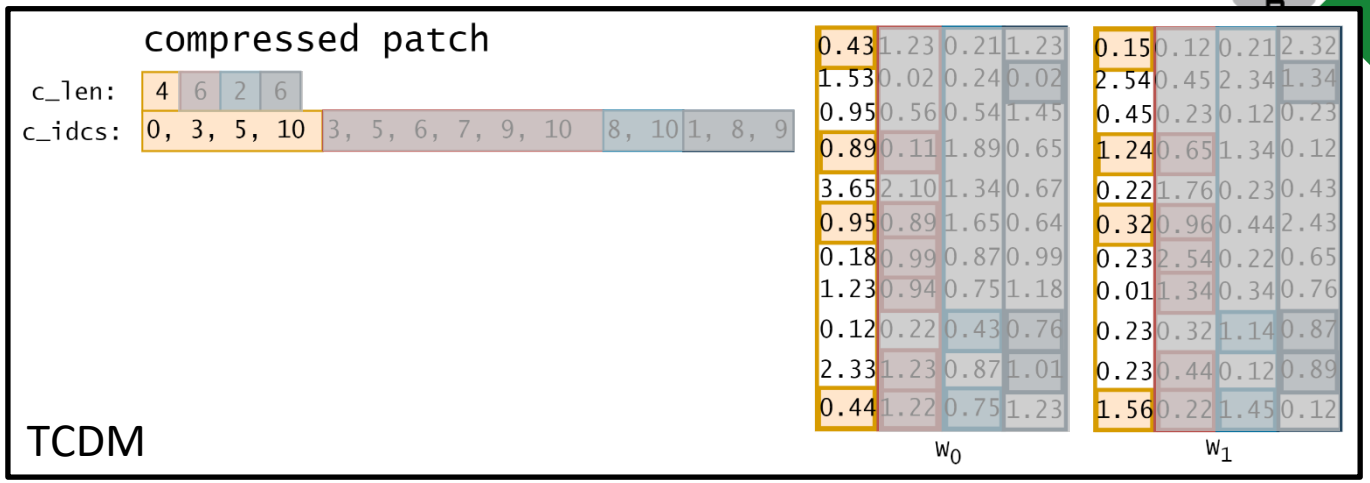
Example Layer: Sparse Convolution – Dataflow in Snitch

```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        fadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

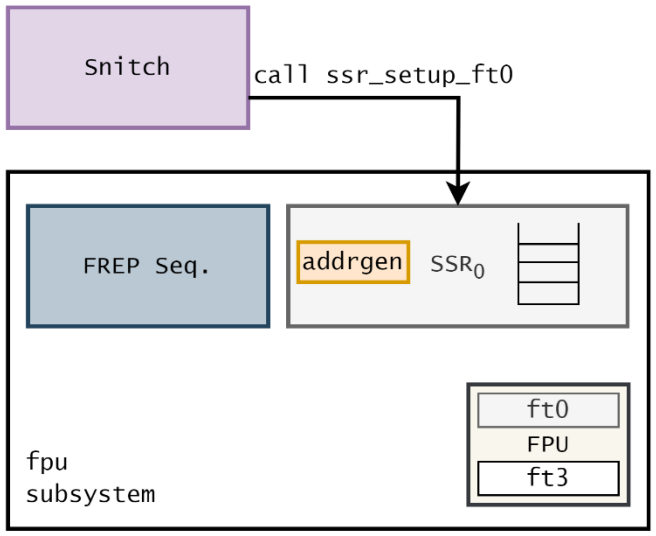
Sparse baseline kernel

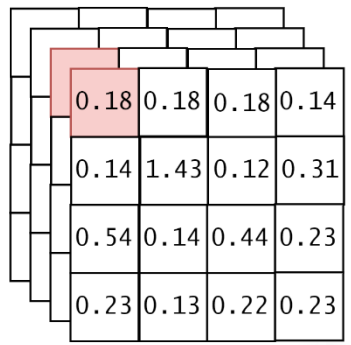
```
load v {fld ft3, 0(%v)
L0:
setup streamer {call ssr_setup_ft0
```

Snitch optimized kernel



- Load the membrane potential
- Setup the streamer: Provide W base addr to SSRs
- Repeat `vfadd` instruction `c_len` times
- SIMD over different weights





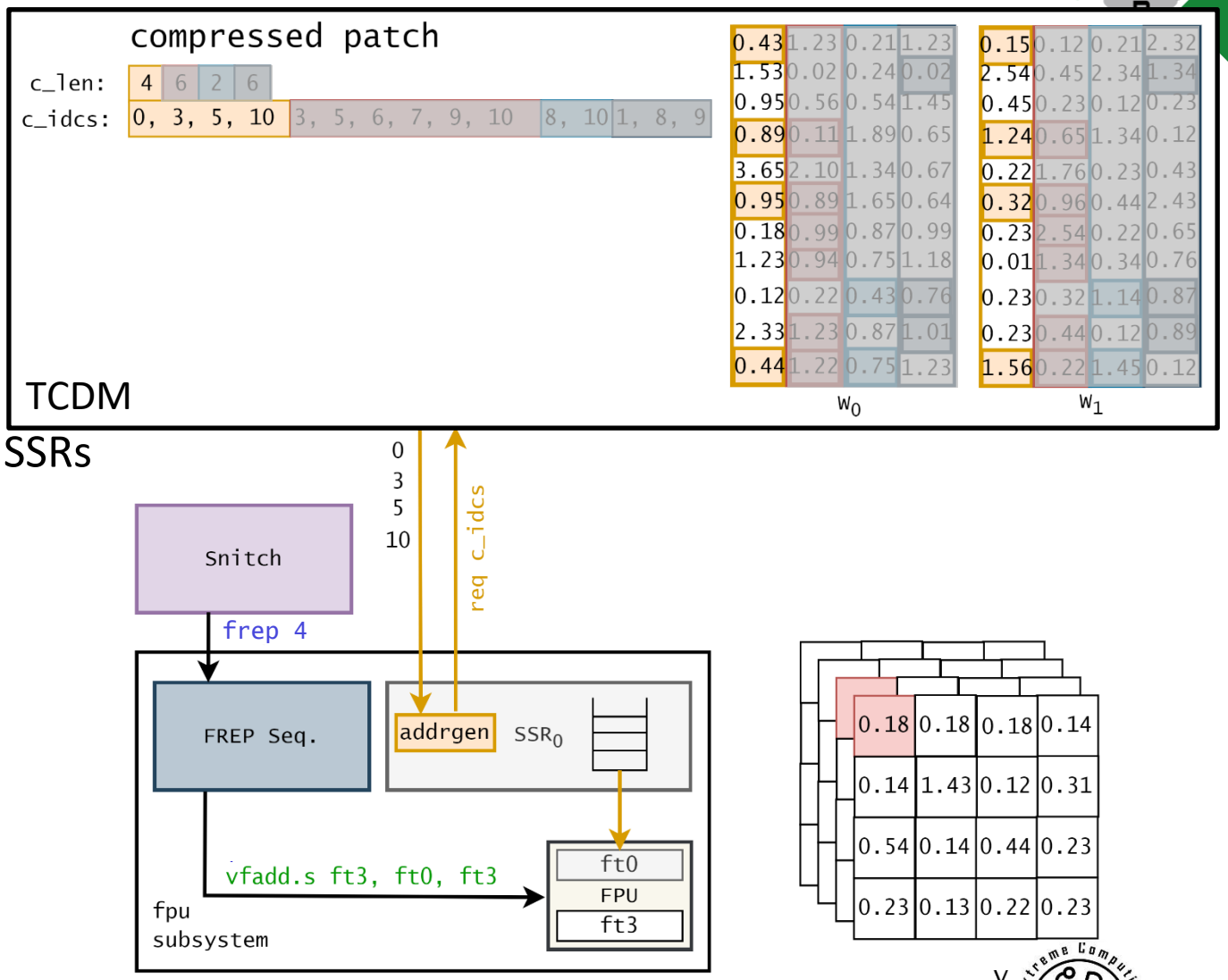
Example Layer: Sparse Convolution – Dataflow in Snitch

```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        fadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

Sparse baseline kernel

```
load v          { fld ft3, 0(%v)
L0:
setup streamer  { call ssr_setup_ft0
SIMD acc
on streamed     { frep %c_len:
weigths             vfadd.s ft3, ft0, ft3
```

Snitch optimized kernel



- Load the membrane potential
- Setup the streamer: Provide W base addr to SSRs
- Repeat vfadd instruction c_len times
- SIMD over different weights

Example Layer: Sparse Convolution – Dataflow in Snitch

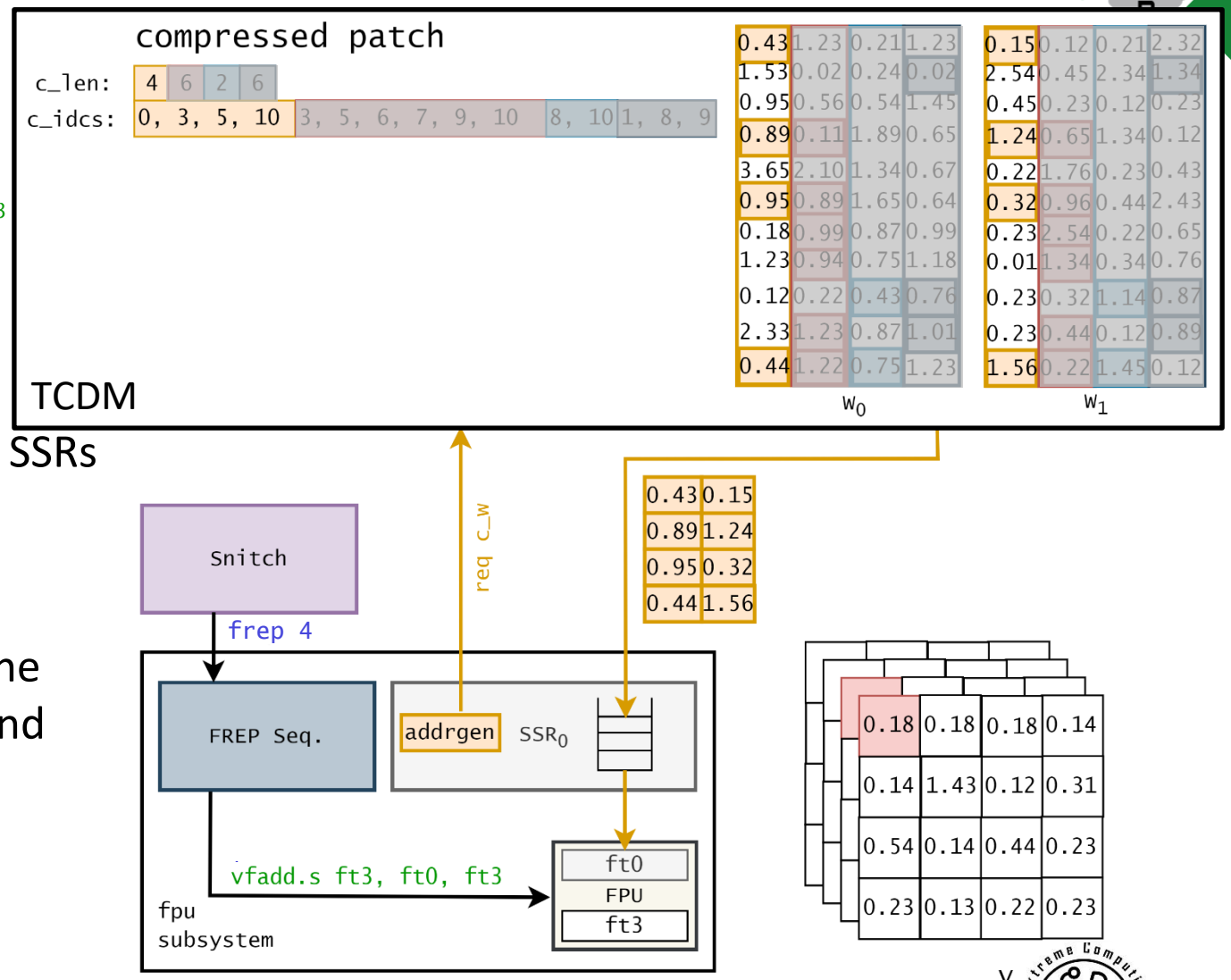
```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        fadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

Sparse baseline kernel

```
load v          { fld ft3, 0(%v)
L0:
setup streamer  { call ssr_setup_ft0
SIMD acc
on streamed     { frep %c_len:
weigths             vfadd.s ft3, ft0, ft3
```

Snitch optimized kernel

- Load the membrane potential
- Setup the streamer: Provide W base addr to SSRs
- Repeat `vfadd` instruction `c_len` times
- SIMD over different weights
- SSRs + FREP lead to only one instruction in the hot loop by removing explicit issue of load and branch instructions



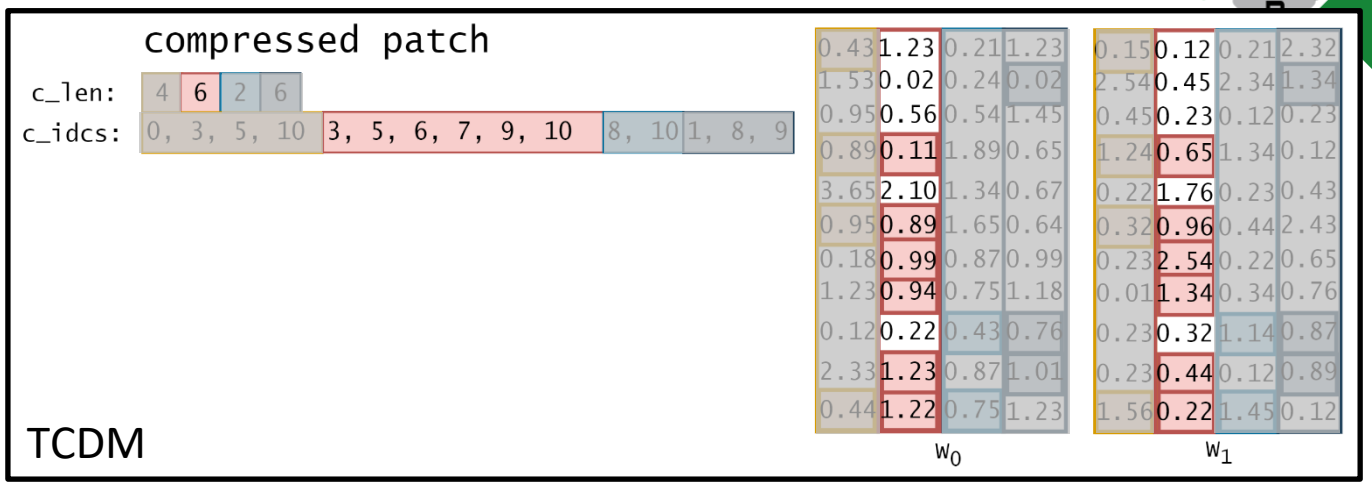
Example Layer: Sparse Convolution – Dataflow in Snitch

```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        fadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

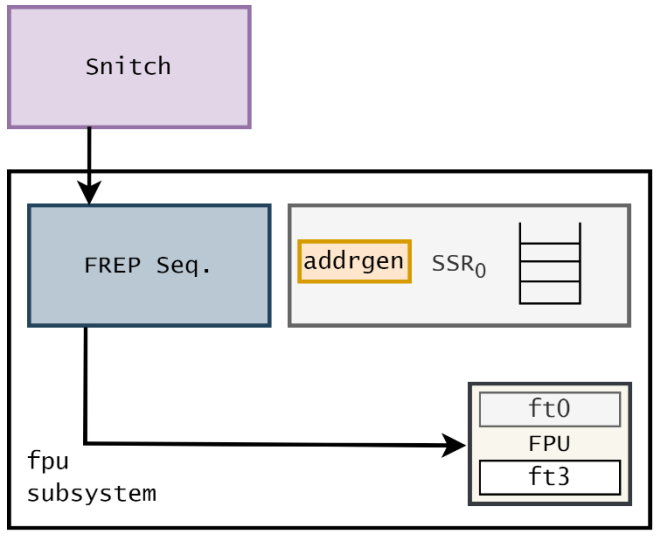
Sparse baseline kernel

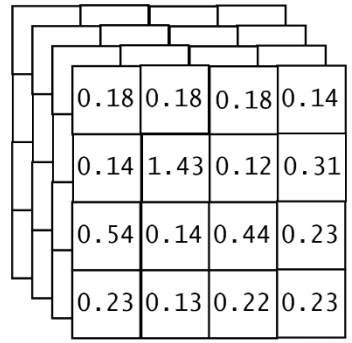
```
load v          { fld ft3, 0(%v)
                L0:
setup streamer  { call ssr_setup_ft0
SIMD acc
on streamed     { frep %c_len:
weigths             vfadd.s ft3, ft0, ft3
                sub %iter, %iter, 1
                bnez %iter, L0
```

Snitch optimized kernel



- Load the membrane potential
- Setup the streamer: Provide W base addr to SSRs
- Repeat `vfadd` instruction `c_len` times
- SIMD over different weights
- SSRs + FREP lead to only one instruction in the hot loop by removing explicit issue of load and branch instructions
- Iterate on the next subpatch





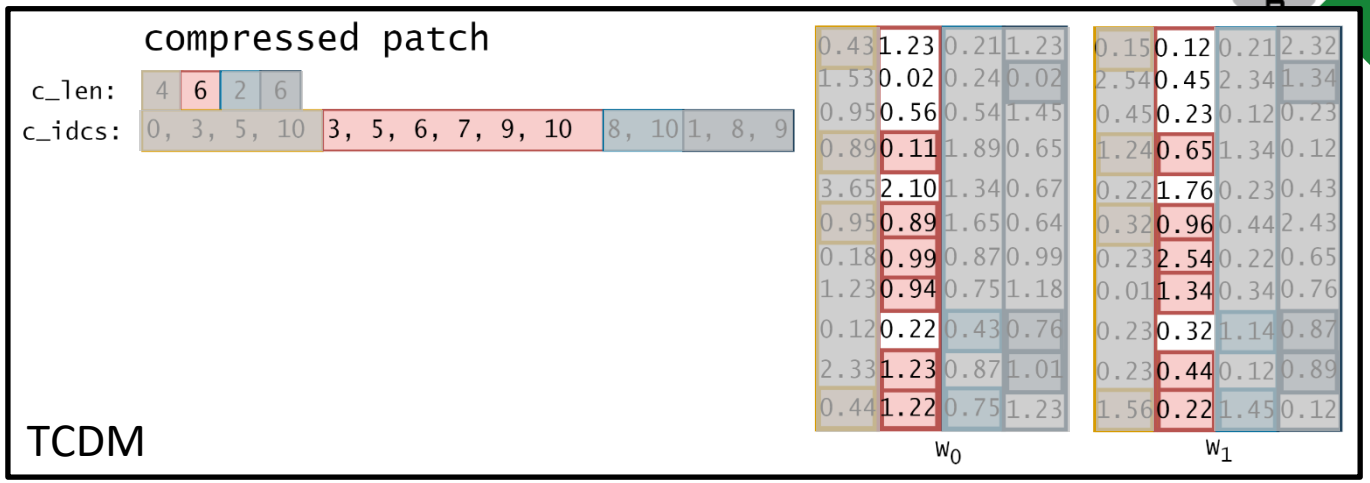
Example Layer: Sparse Convolution – Dataflow in Snitch

```
spvacc: lw t0, 0(%c_idcs)
        slli t0, t0, 3
        add t0, t0, %weights
        fld ft0, 0(t0)
        fadd.s ft3, ft0,
        addi %c_idcs, %c_idcs, 4
        bne %c_vals, %c_end, spvacc
```

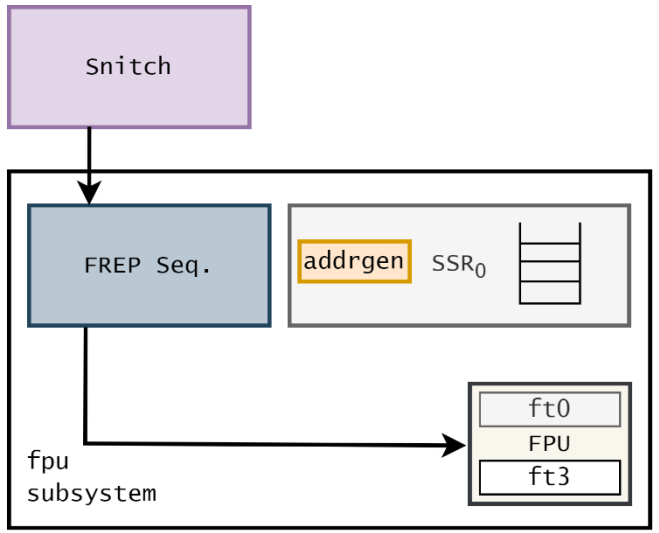
Sparse baseline kernel

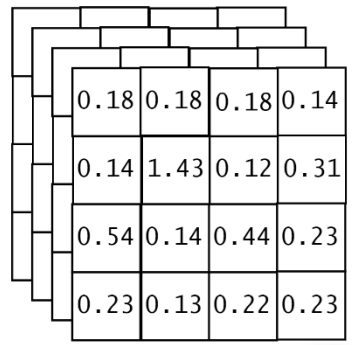
```
load v          { fld ft3, 0(%v)
                L0:
setup streamer  { call ssr_setup_ft0
SIMD acc
on streamed     { frep %c_len:
weigths             vfadd.s ft3, ft0, ft3
                sub %iter, %iter, 1
                bnez %iter, L0
```

Snitch optimized kernel



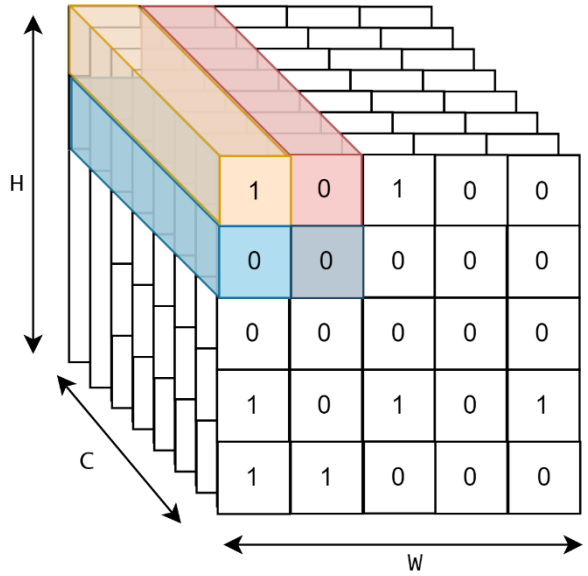
- Load the membrane potential
- Setup the streamer: Provide W base addr to SSRs
- Repeat `vfadd` instruction `c_len` times
- SIMD over different weights
- SSRs + FREP lead to only one instruction in the hot loop by removing explicit issue of load and branch instructions
- Iterate on the next subpatch
- At the end of the patch: decay and threshold

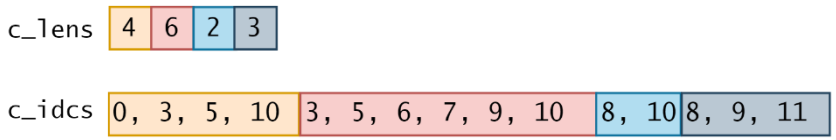





```
vfmul.r.h ft3, ft3, %[alpha]
vfle.h    t3, ft3, %[vth]
```

Example Layer: Sparse Convolution – Parallelization

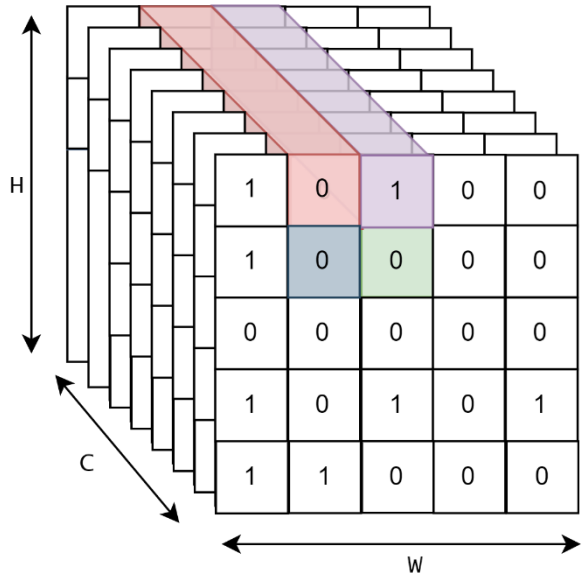


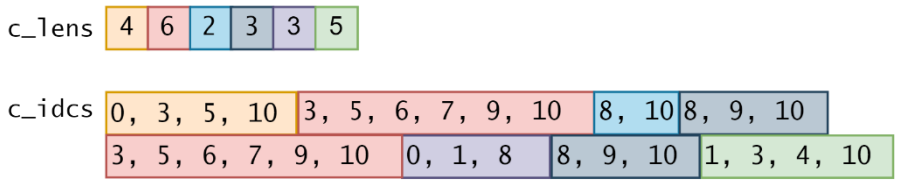


TCDM data (coming from ext. DRAM)

async DMA copy in comp patch

Example Layer: Sparse Convolution – Parallelization



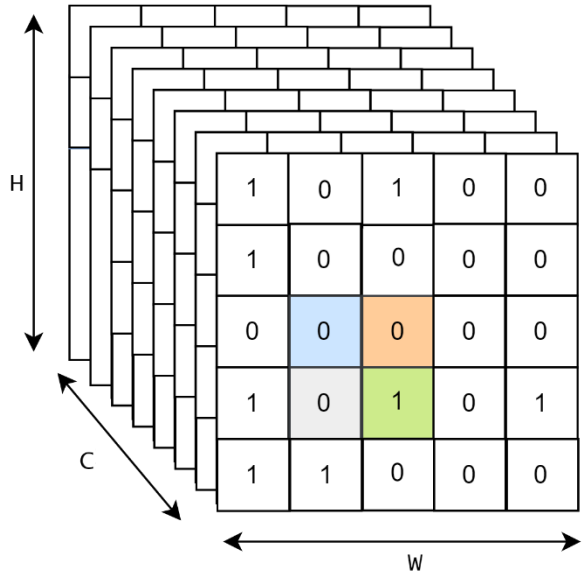


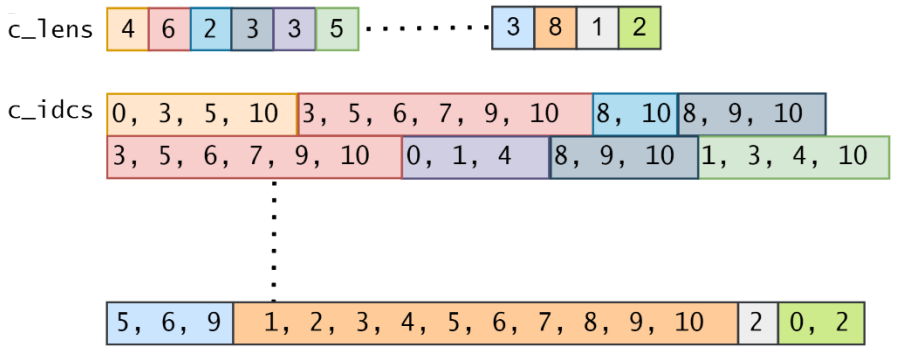
TCDM data (coming from ext. DRAM)

async DMA copy in comp patch

async DMA copy in comp patch

Example Layer: Sparse Convolution – Parallelization





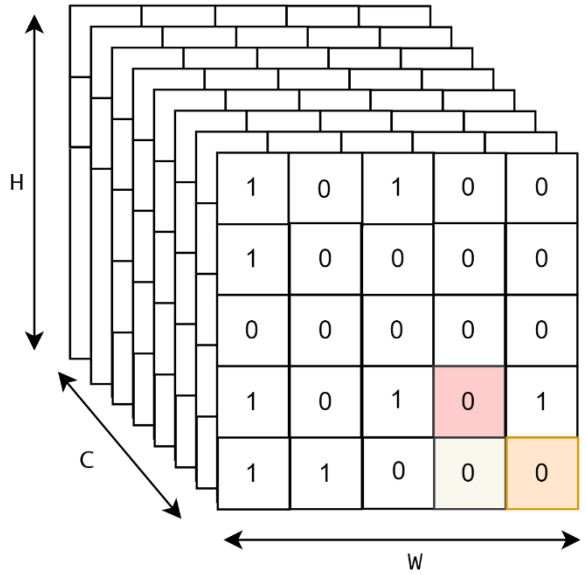
TCDM data (coming from ext. DRAM)

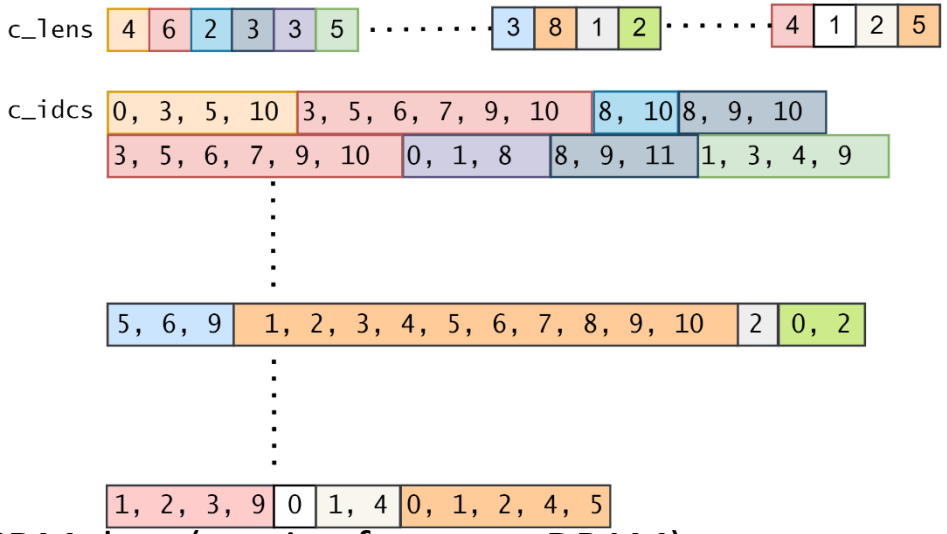
async DMA copy in comp patch

async DMA copy in comp patch

async DMA copy in comp patch

Example Layer: Sparse Convolution – Parallelization





TCDM data (coming from ext. DRAM)

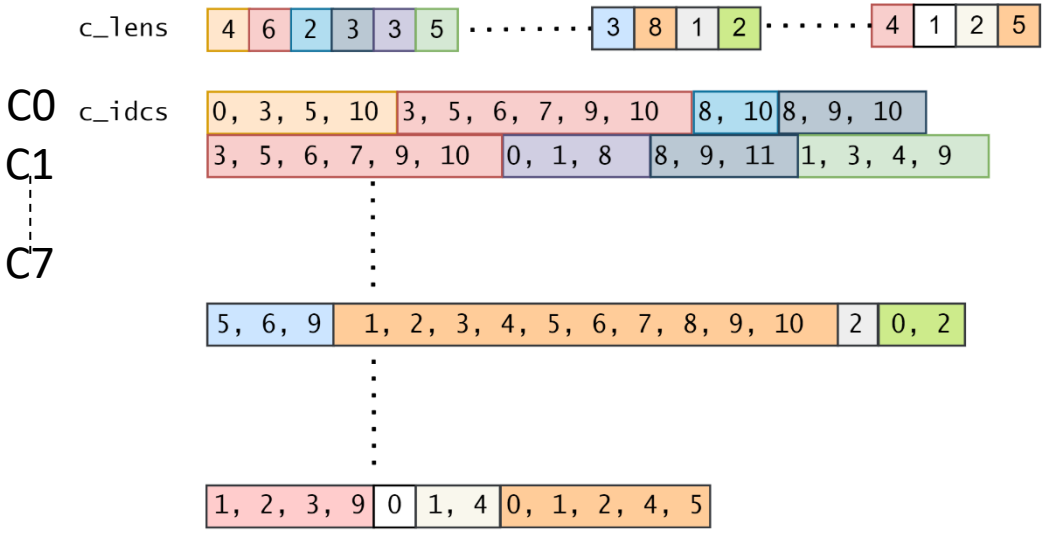
async DMA copy in comp patch

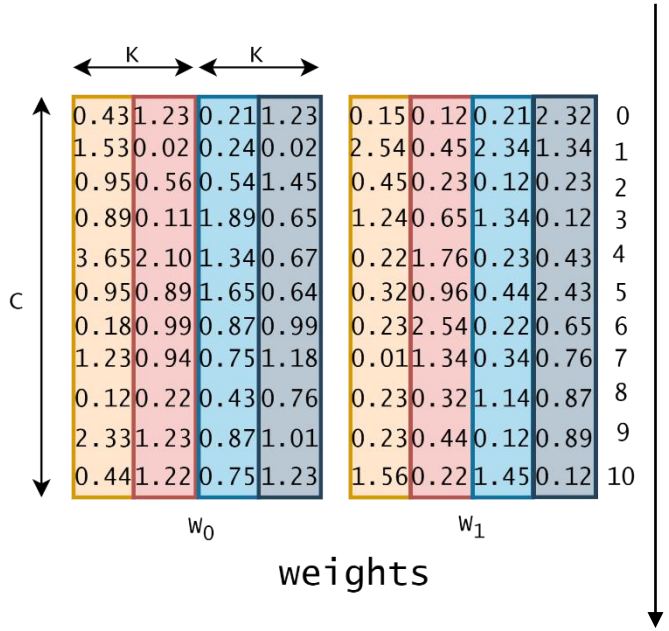
async DMA copy in comp patch

async DMA copy in comp patch

async DMA copy in comp patch

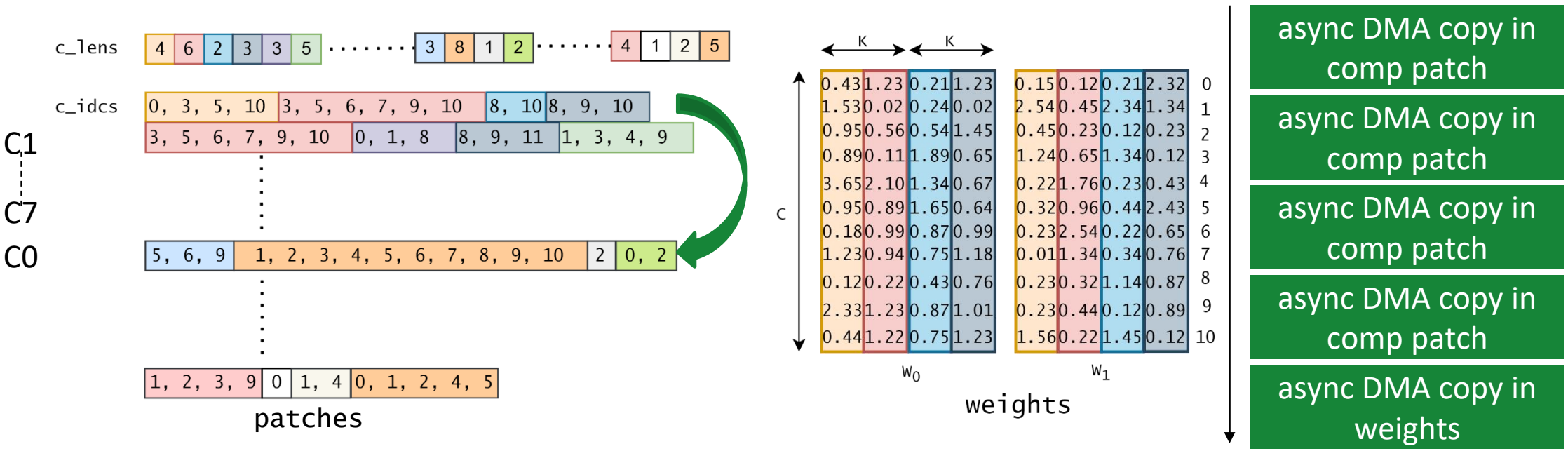
Example Layer: Sparse Convolution – Parallelization





- async DMA copy in comp patch
- async DMA copy in comp patch
- async DMA copy in comp patch
- async DMA copy in comp patch
- async DMA copy in weights

Example Layer: Sparse Convolution – Parallelization



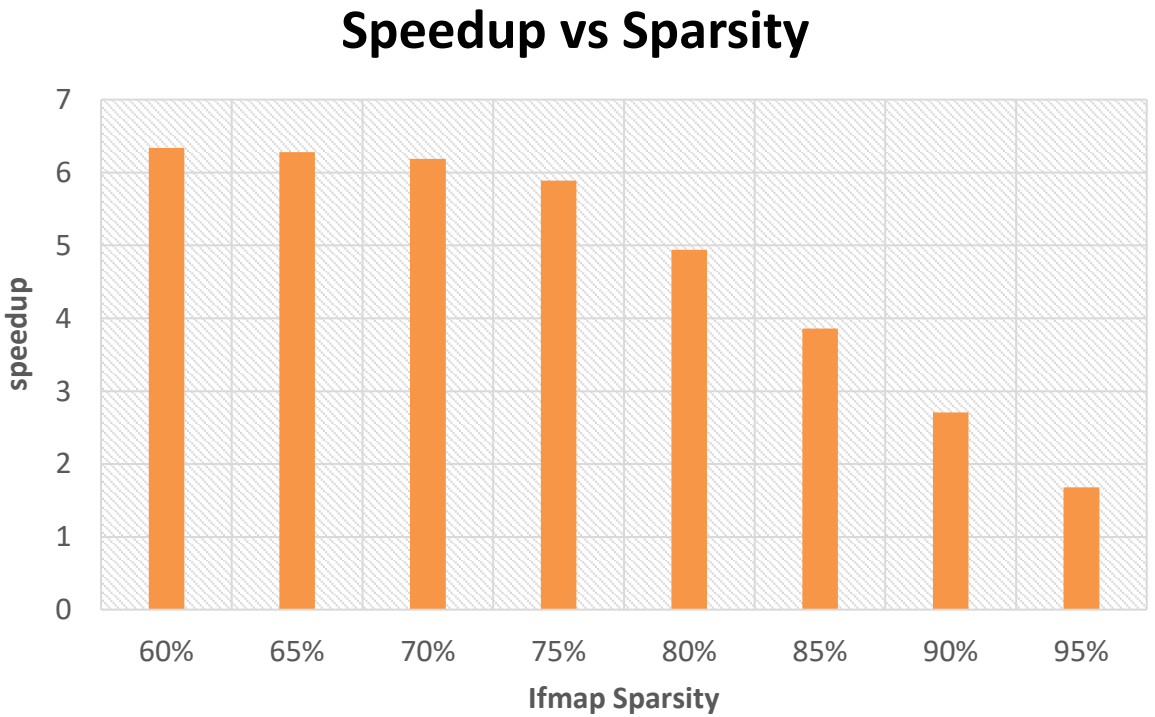
- Weights are stationary in TCDM “broadcasted” to all the cores
- To mitigate workload imbalance, whenever a core processed a patch, jumps to the first “free” patch
- DMA memory transfers are performed in a double-buffered fashion to hide latency

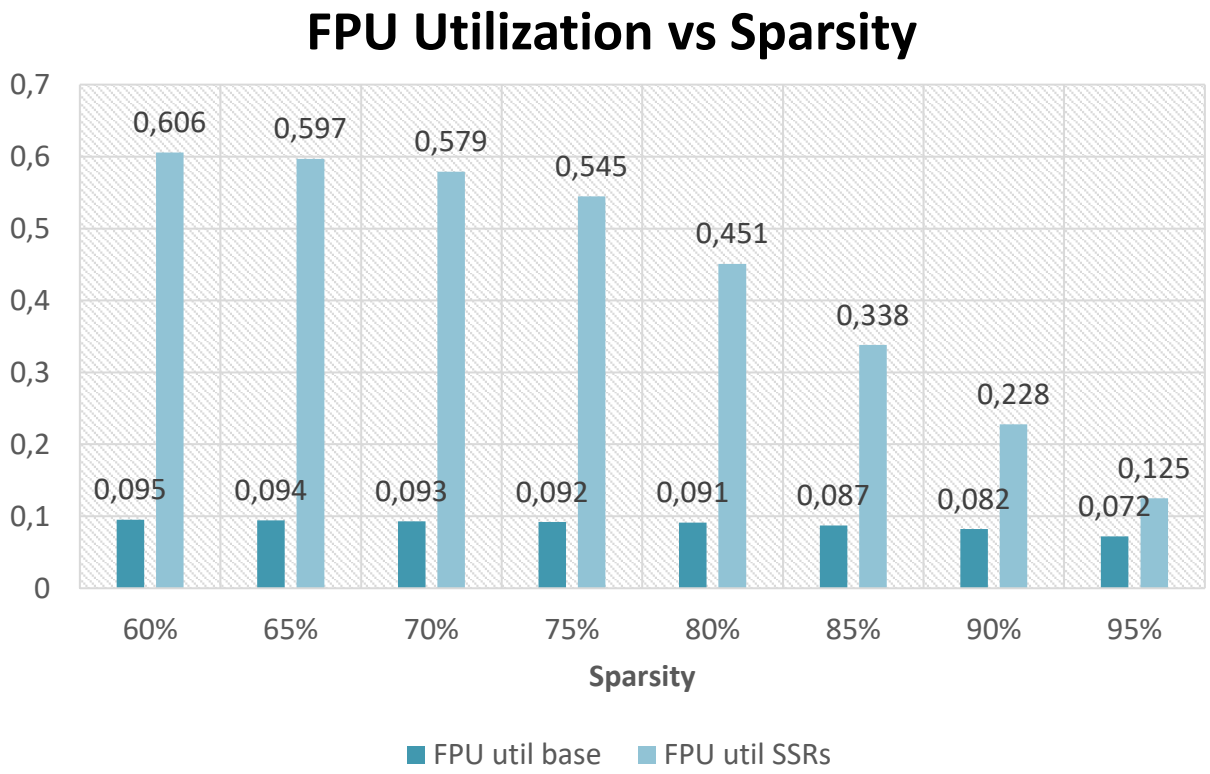
Outline

- Neuromorphic Computing/SNNs
- Neuromorphic/SNNs Accelerators
- Snitch Cluster architecture
- Contributions & Sparse conv layer example:
  - Compressed format
  - Dataflow in Snitch
  - Parallelization in Snitch cluster
- **Results & Conclusions**

Preliminar Results: kernel 14x14x256

- Run test using Questa cycle accurate simulator
- Speedup and FPU utilization of SSRs acceleration w.r.t. the sparse baseline
- With sparsity above 80% the kernel starts to be bounded by the DMA programming





Conclusions and future developements

- In this work we proposed a kernel library for Spiking Neural Networks on an open-source RV32G core cluster enhanced with FREP, Streaming Semantic Registers and smallFloat ISA extensions
- We preliminarly evaluated our method kernel size of 14x14x256 with different sparsity degrees
- Our optimized kernels always outperform the sparse baseline. Above 80%, it starts to be bounded by the frequent DMA programming related to the transfer of many input tensors
- Power simulations, end-to-end deployment and comparison with neuromorphic processors is ongoing

Conclusions and future developements

- In this work we proposed a kernel library for Spiking Neural Networks on an open-source RV32G core cluster enhanced with FREP, Streaming Semantic Registers and smallFloat ISA extensions
- We preliminarly evaluated our method kernel size of 14x14x256 with different sparsity degrees
- Our optimized kernels always outperform the sparse baseline. Above 80%, it starts to be bounded by the frequent DMA programming related to the transfer of many input tensors
- Power simulations, end-to-end deployment and comparison with neuromorphic processors is ongoing
- This research was supported by the HE EU Graph-Massivizer project (g.a. 101093202) and EdgeAI (g.a. 101097300)