MANCHESTER 1824
The University of Manchester

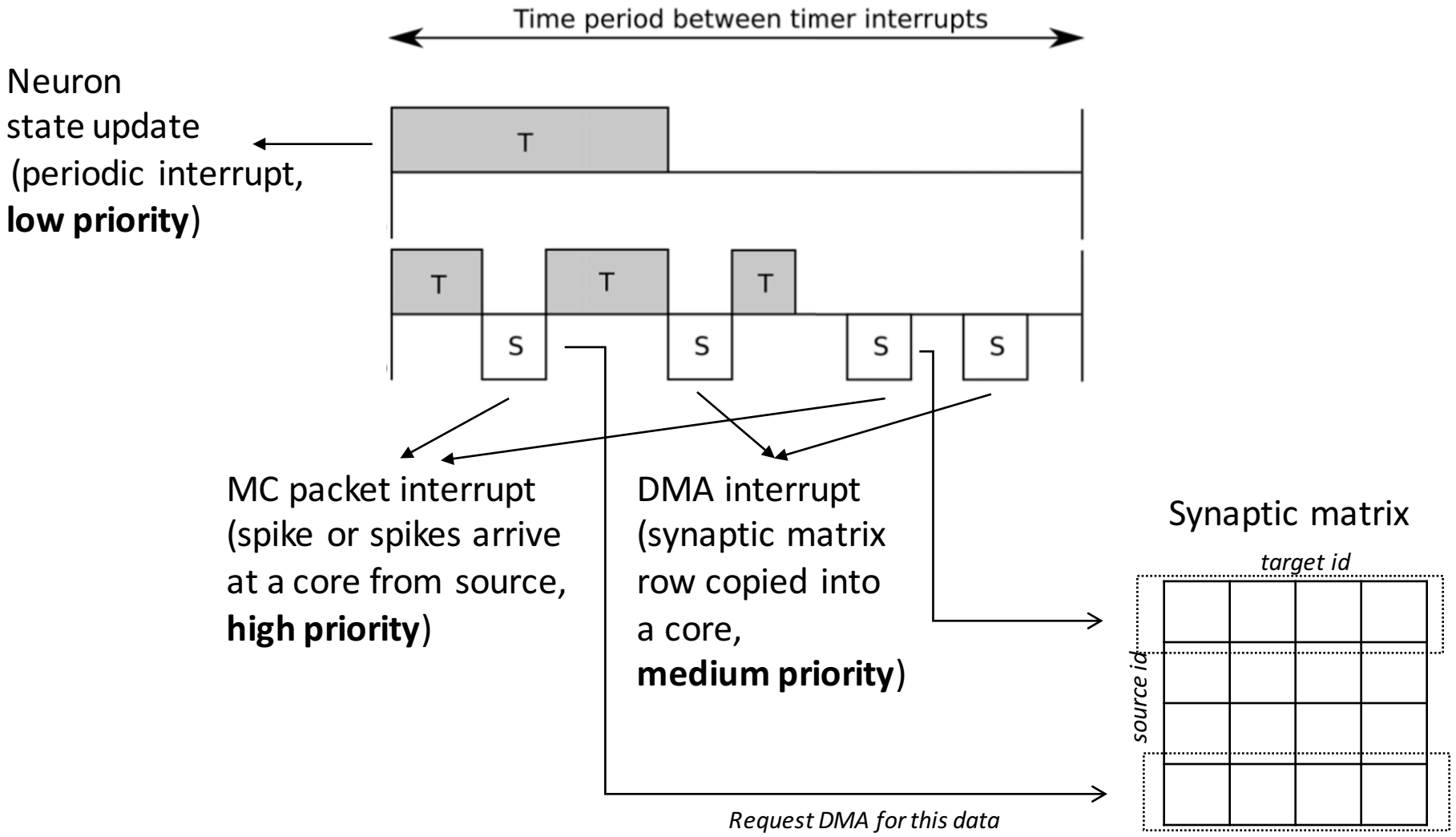
Results from profiling SpiNNaker-1 and early ideas on SpiNNaker-2 accelerators

Mantas Mikaitis

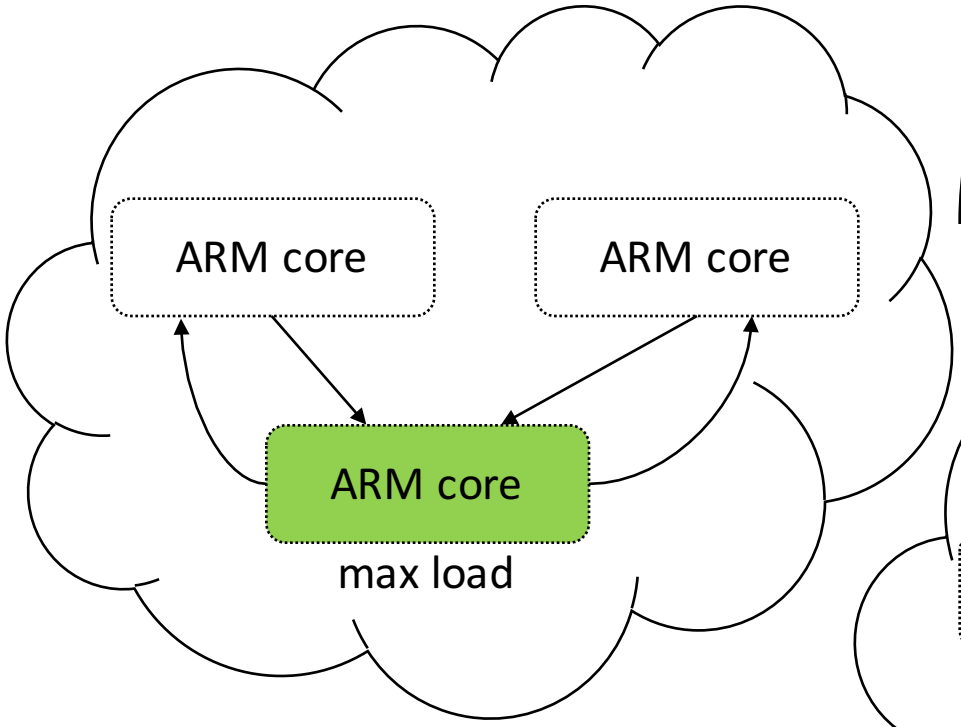
Contents

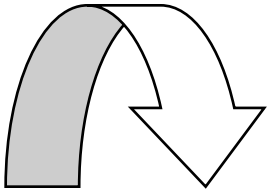
1. Performance of SpiNNaker-1 large scale networks
2. SpiNNaker-2 and Spiking-Neural-Networks 2020 onwards
3. Accelerators with limited numerical precision and errors

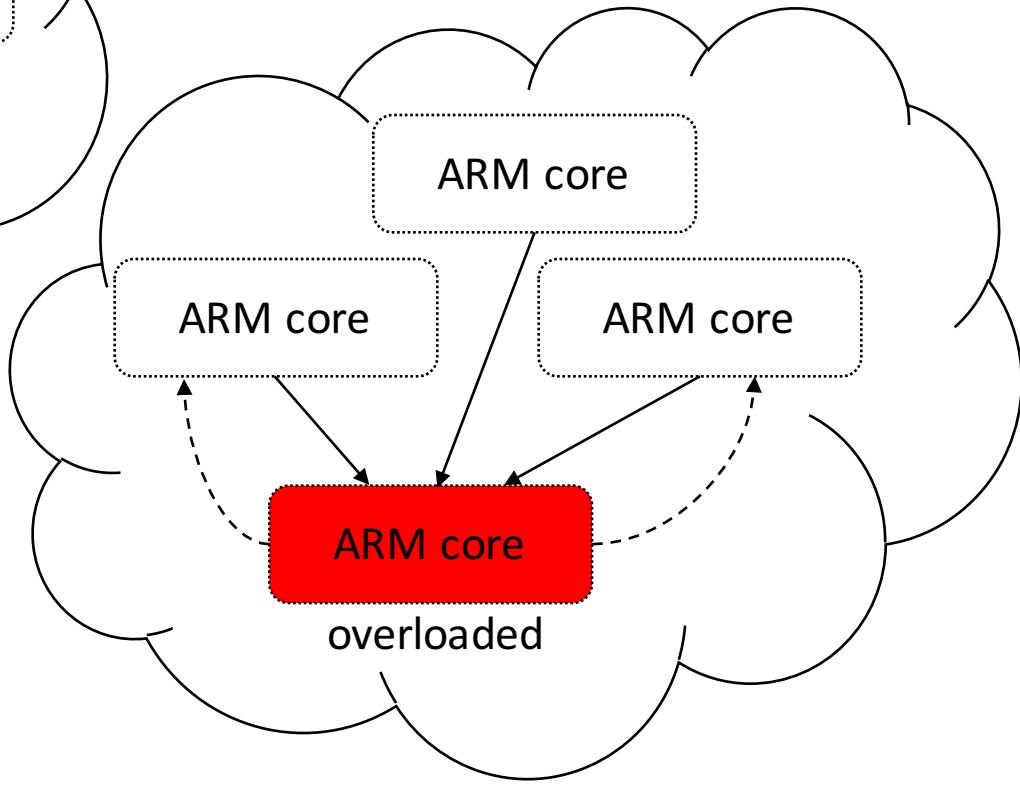
SpiNNaker simulation framework



Single Core Performance Profiling

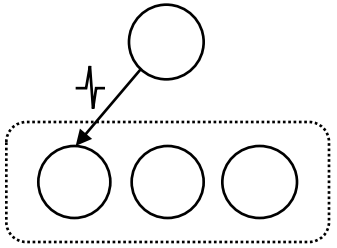
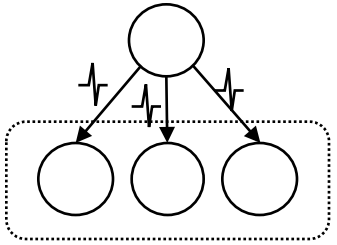
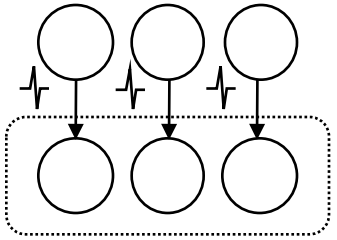
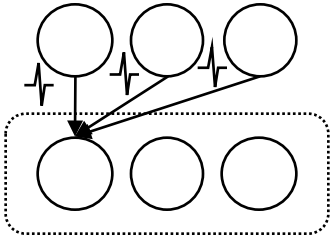




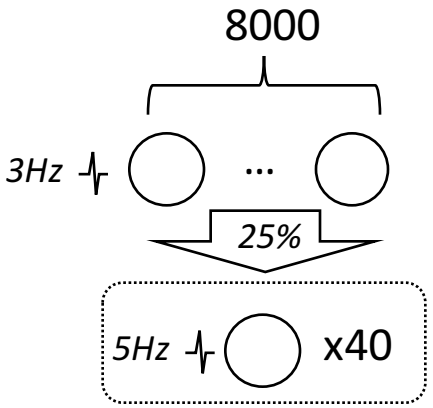


Building cost models of a single core can tell us a lot about the whole networks

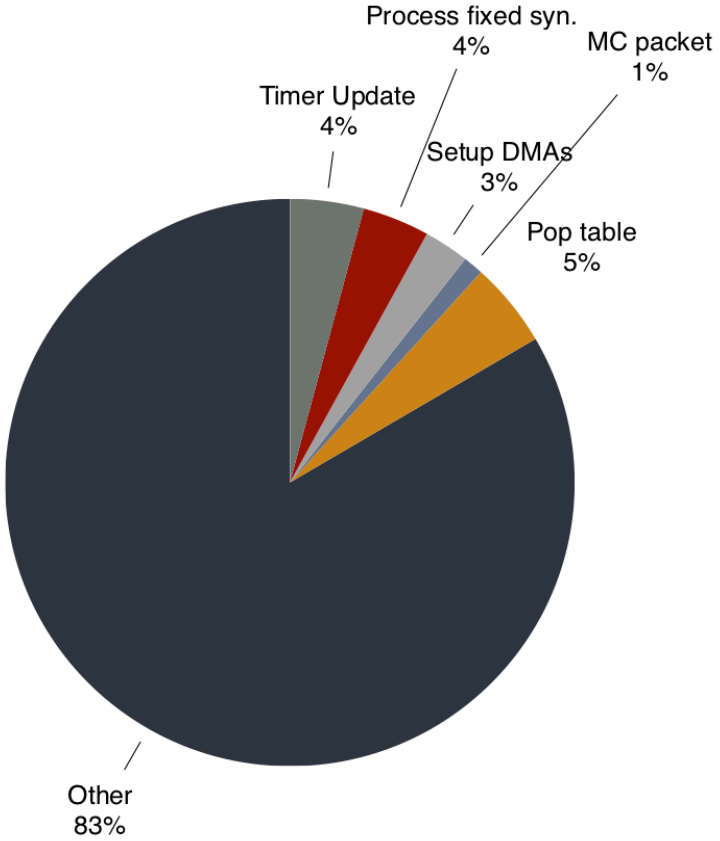
Effects of Scaling Networks on SpiNNaker

Connectivity	Scaling method	Connection density	Average indegree	Spikes	DMA invocations	Interrupts
	Original network	33%	1/3	1	1	2
	Increase connectivity density	100%	1	3	1	2
	Increase number of sources	33%	1	3	3	6
	Increase number of sources	33%	1	3	3	6

Profiling large scale fixed weight networks (Benchmark 1)

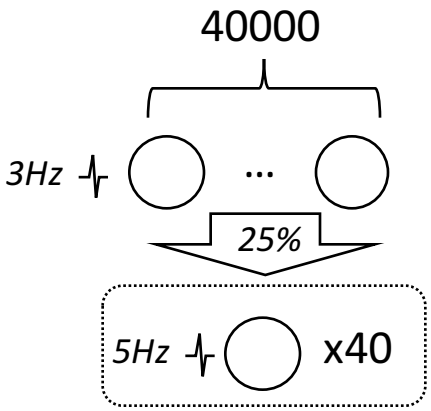


Activities of target core per 1s

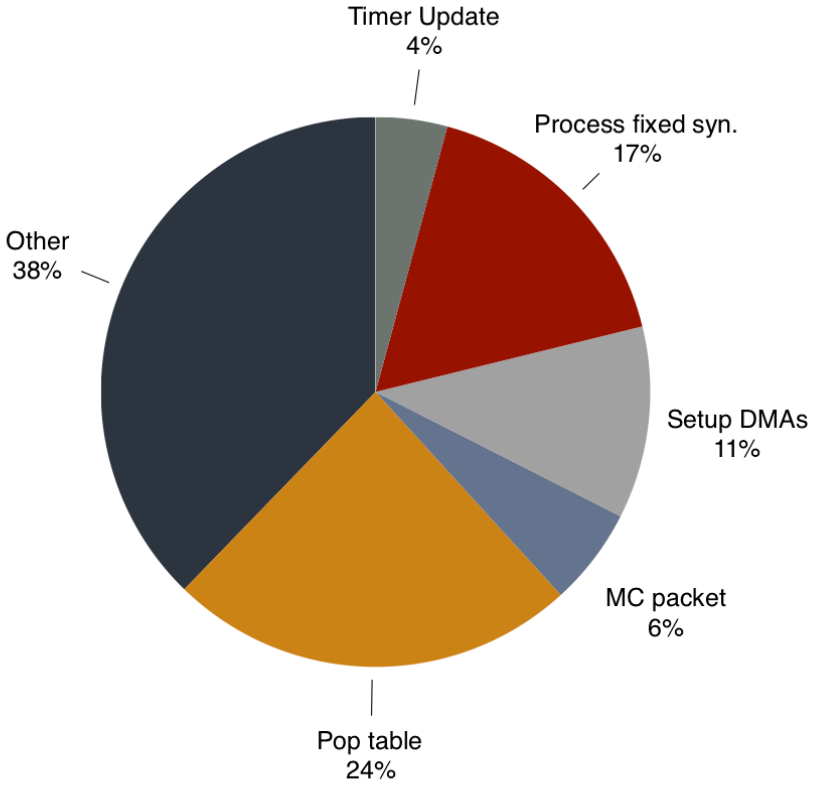


Synapses	80 000
Avg. target indegree	2000
Connections/s	240 000
MC interrupts	24 000
DMA interrupts	24 000

Profiling large scale fixed weight networks (Benchmark 2)



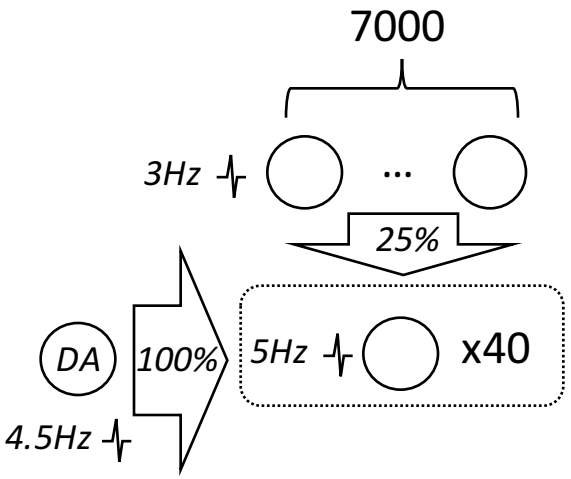
Activities of target core per 1s



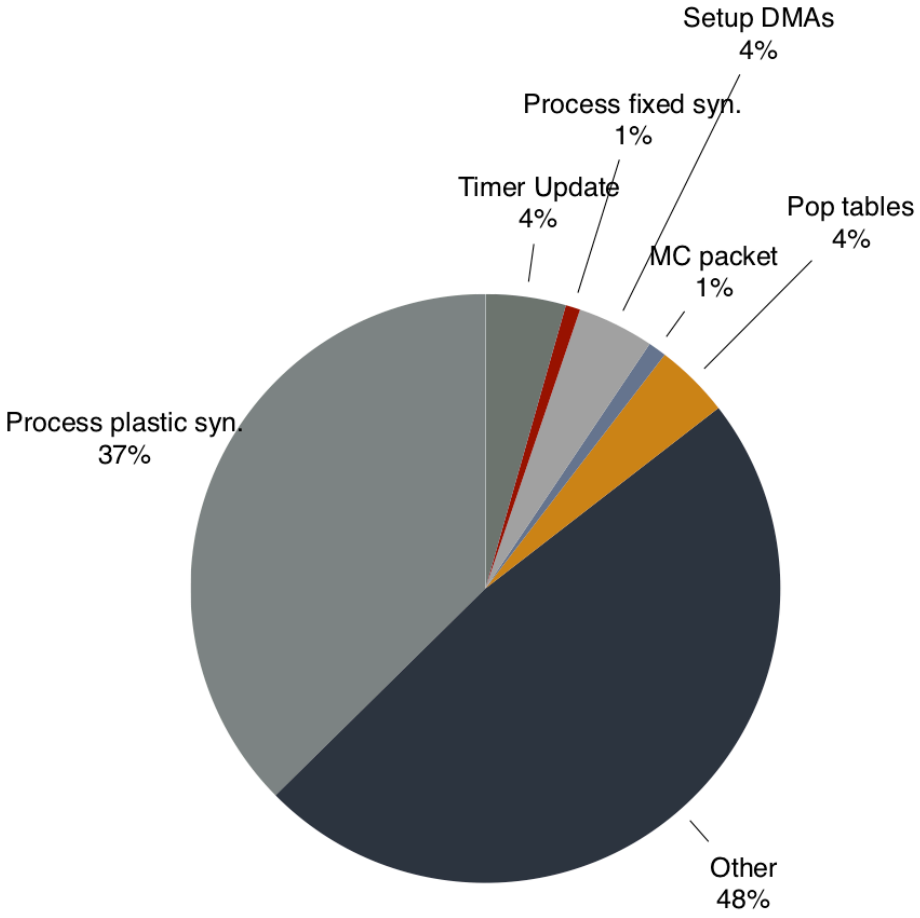
Synapses	400 000
Avg. target indegree	10 000
Connections/s	1.2 Million
MC interrupts	120 000
DMA interrupts	120 000

Interrupt callback queue overloaded

Profiling large scale plastic weight networks (Benchmark 3)

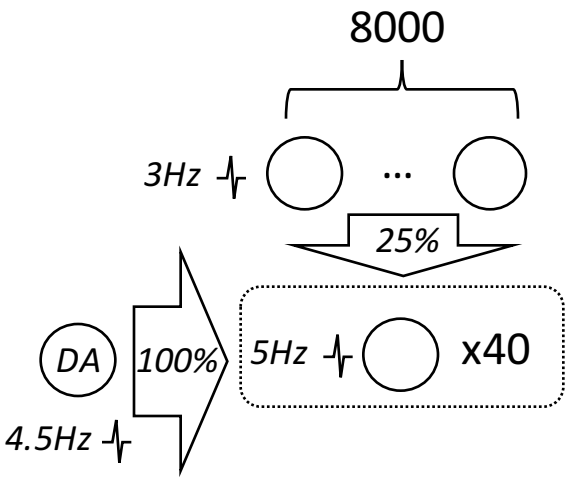


Activities of target core per 1s

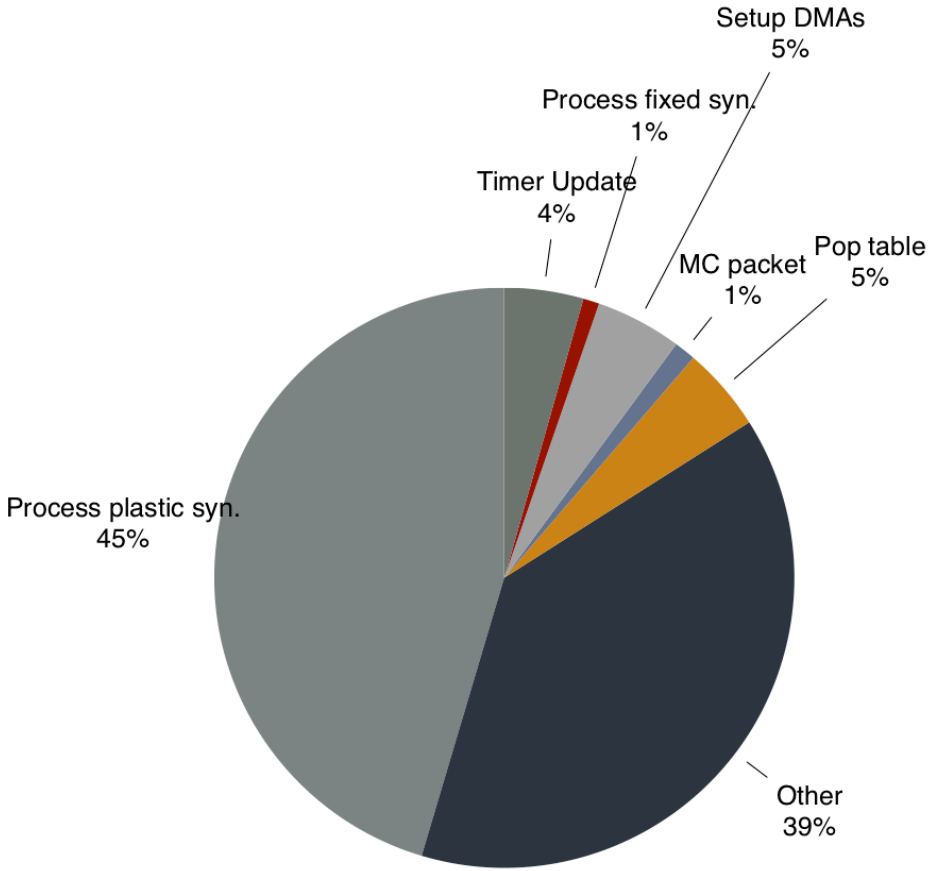


Synapses	70 000
Avg. target indegree	1750
Connections/s	210 000
MC interrupts	21 000
DMA interrupts	21 000

Profiling large scale plastic weight networks (Benchmark 4)



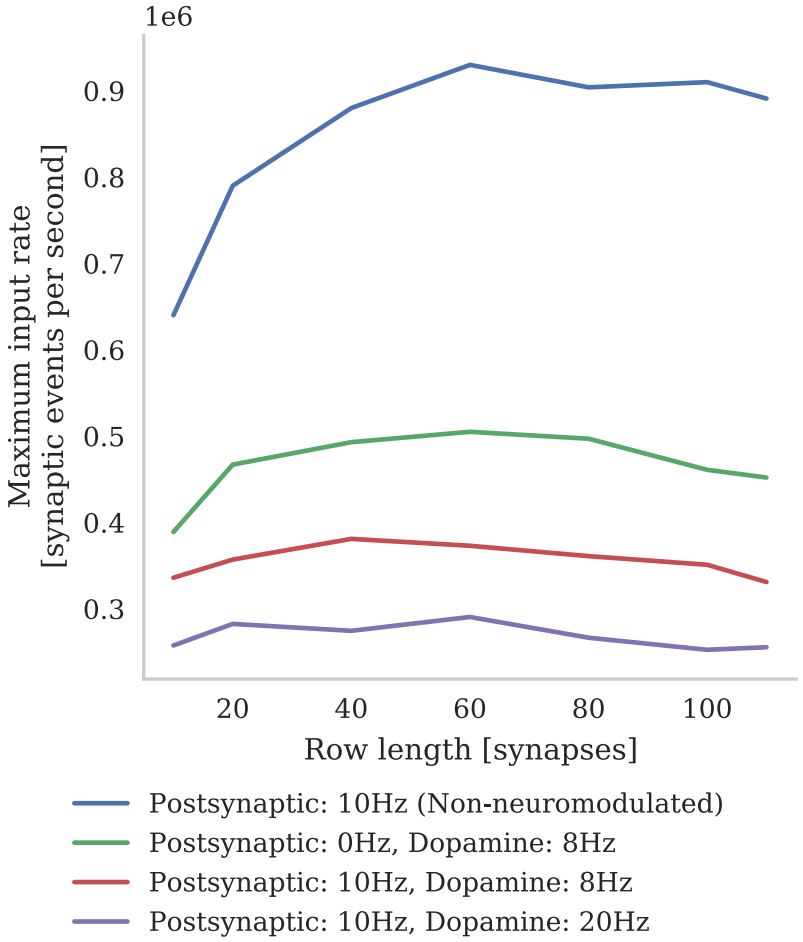
Activities of target core per 1s



Synapses	80 000
Avg. target indegree	2000
Connections/s	240 000
MC interrupts	24 000
DMA interrupts	24 000

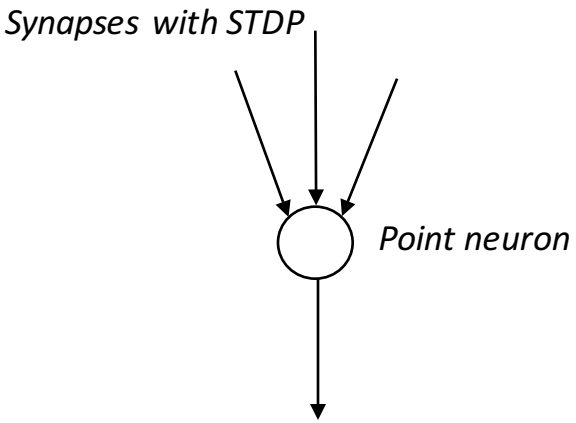
Interrupt callback queue overloaded

Maximum performance (No profiler overhead)



Spiking Neural Networks 2020 and beyond

Current models:



SpiNNaker-2 might need to support:

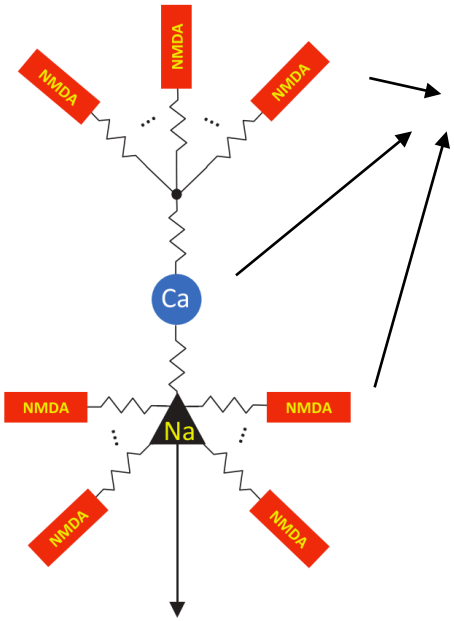


Image from Schemmel et al 2017 and Larkum et al 2009

- Multi compartment neuron models, each with its own set of incoming synapses
- Multiple-factor synaptic plasticity
- Structural plasticity
- Intrinsic Hodgkin-Huxley type currents
- Neuromodulation of synaptic plasticity and intrinsic neuron properties (Volume transmission at high rates)

SpiNNaker-2 chip pathway

- Prototype chip 1 (codename Santos) was already tested in 2016/2017.
- Prototype chip 2 (codename JIB1) is about to be manufactured
- Prototype chip 3 will be produced Q1 2019
- Final SpiNNaker-2 chip will be produced Q2 2020

Feature comparison

SpiNNaker-1

- 18 ARM968 cores
- 96K memory per core
- 128MB Off-chip memory
- 1W power

SpiNNaker-2

- 144 ARM M4F cores
- 128K memory per core (With capability to use other core's memories)
- 2GB Off-chip memory
- Single precision floating point operations
- Random Number Generators
- Machine Learning Accelerator
- Elementary Functions
- 1W power

ARM M4F Floating-Point-Unit

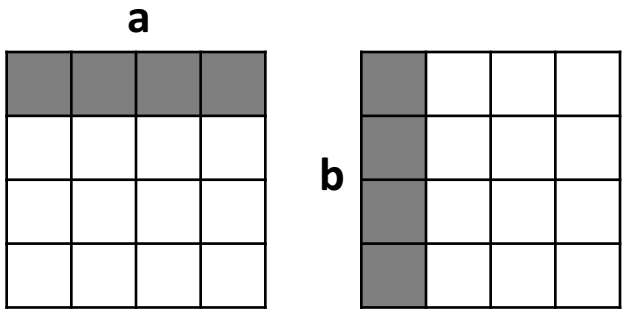
- SpiNNaker-2 ARM core has a hardware unit for 32bit floating point arithmetic
- Apart from standard single-cycle ADD, MOVE, COMPARE, MULTIPLY operations, it also has some special operations:

Operation	Cycles
Convert between fixed and floating point	1
Multiply-accumulate/subtract (Fused)	3
Square root	14
Divide	14

Deep Learning with Limited Numerical Precision

- Deep learning is a machine learning method that given digital data can find patterns and classify data based on them
- Most common introductory example: handwritten digit recognition (MNIST dataset)
- Most common operation is inner product of two vectors (Matrix multiplication)
- Inner product requires doing many MAC (Multiply-accumulate) operations
- All cells of the result matrix can be computed in parallel

Deep Learning with Limited Numerical Precision



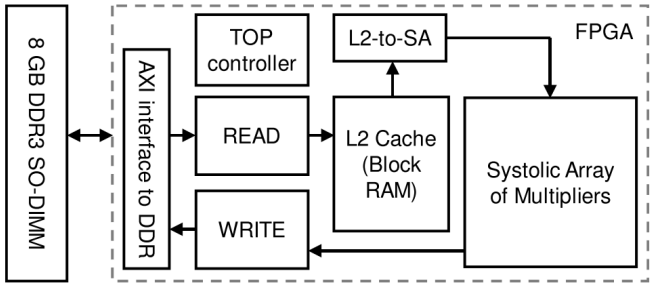
a and **b** are vectors with fixed-point elements of format *<IL, FL>* where IL – number of bits in the integer part and FL – number of bits in the fractional part.

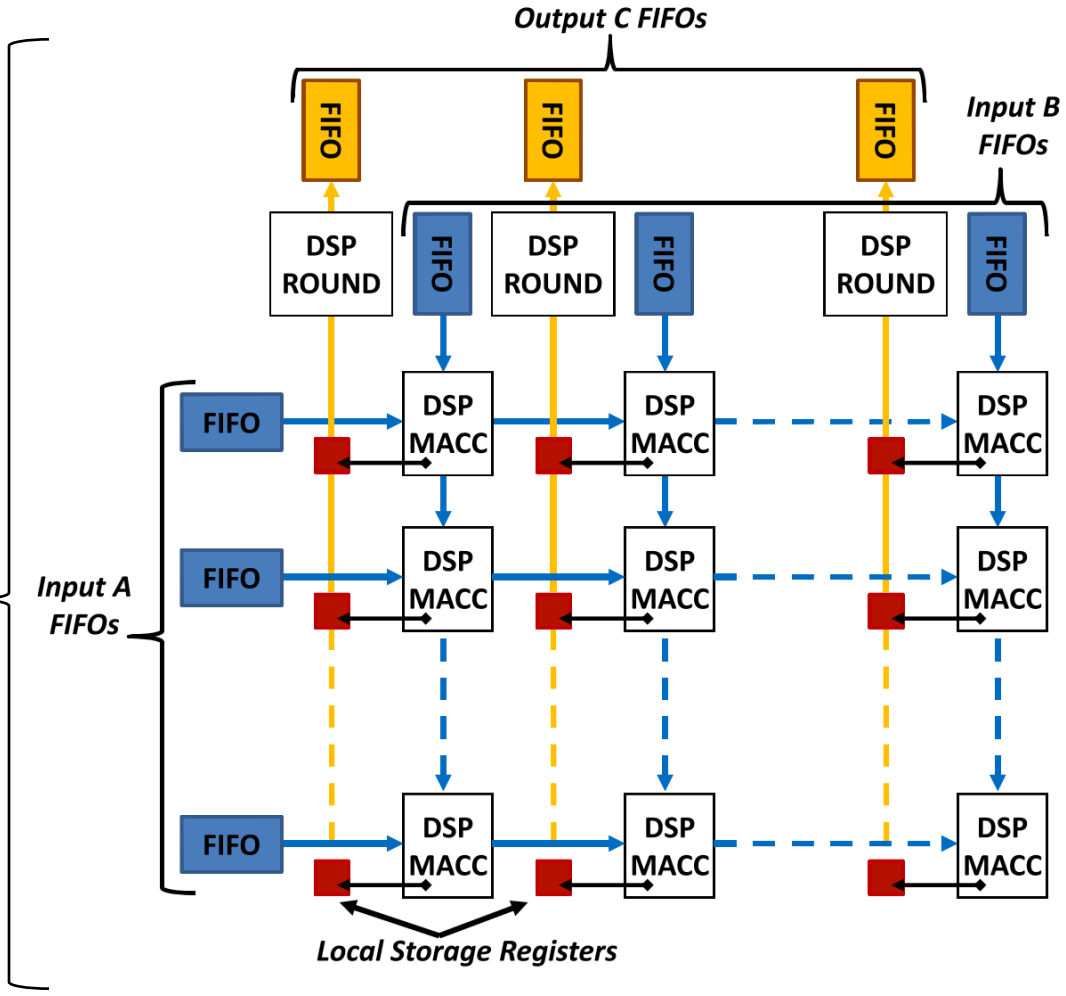
- Inner product: $\boldsymbol{a} \cdot \boldsymbol{b} = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + \cdots + a_n b_n$
- Result of a product $a_i b_i$ will have a fixed-point format *<2*IL, 2*FL>*
- Keep all MAC operations in full precision and round the number to *<IL, FL>* at the end
- Round-to-nearest: $Round(x, < IL, FL >) = \begin{cases} \lfloor x \rfloor & if\ \lfloor x \rfloor \leq x \leq \lfloor x \rfloor + \frac{\epsilon}{2} \\ \lfloor x \rfloor + \epsilon & if\ \lfloor x \rfloor + \frac{\epsilon}{2} < x \leq \lfloor x \rfloor + \epsilon \end{cases}$
- Stochastic rounding: $Round(x, < IL, FL >) = \begin{cases} \lfloor x \rfloor & w.p\ 1 - \frac{x - \lfloor x \rfloor}{\epsilon} \\ \lfloor x \rfloor + \epsilon & w.p\ \frac{x - \lfloor x \rfloor}{\epsilon} \end{cases}$

where $\lfloor x \rfloor$ is a largest integer less than or equal to x representable by <IL, FL> and $\epsilon = 2^{-FL}$

Gupta et al, 2015

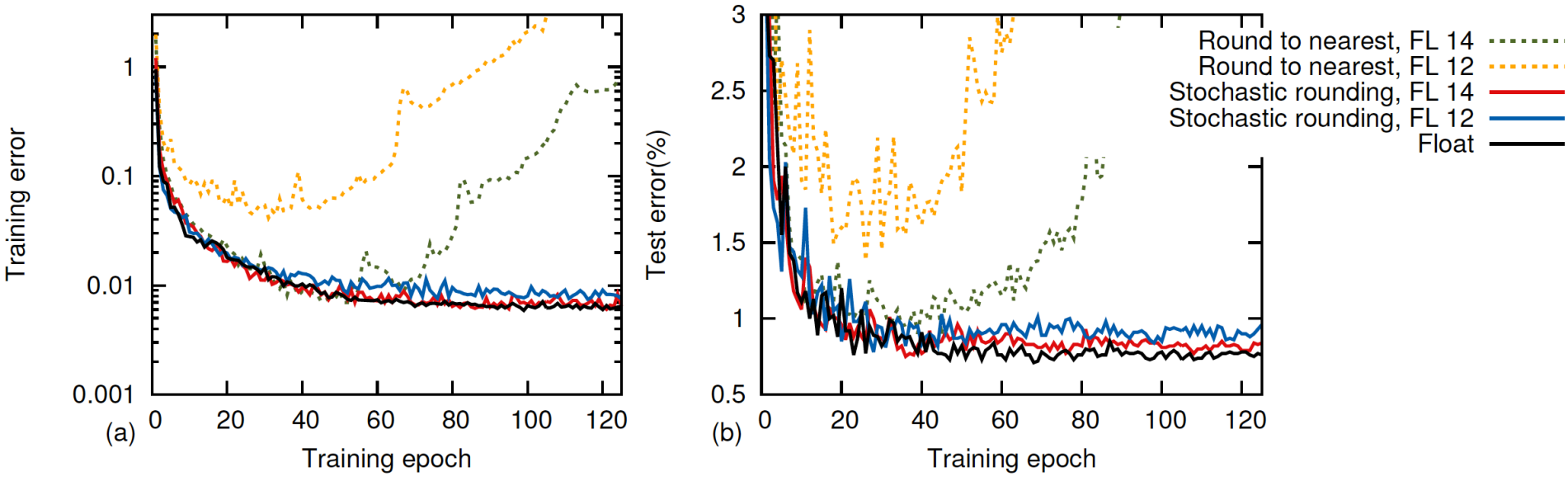
Deep Learning with Limited Numerical Precision





Gupta et al, 2015

Deep Learning with Limited Numerical Precision



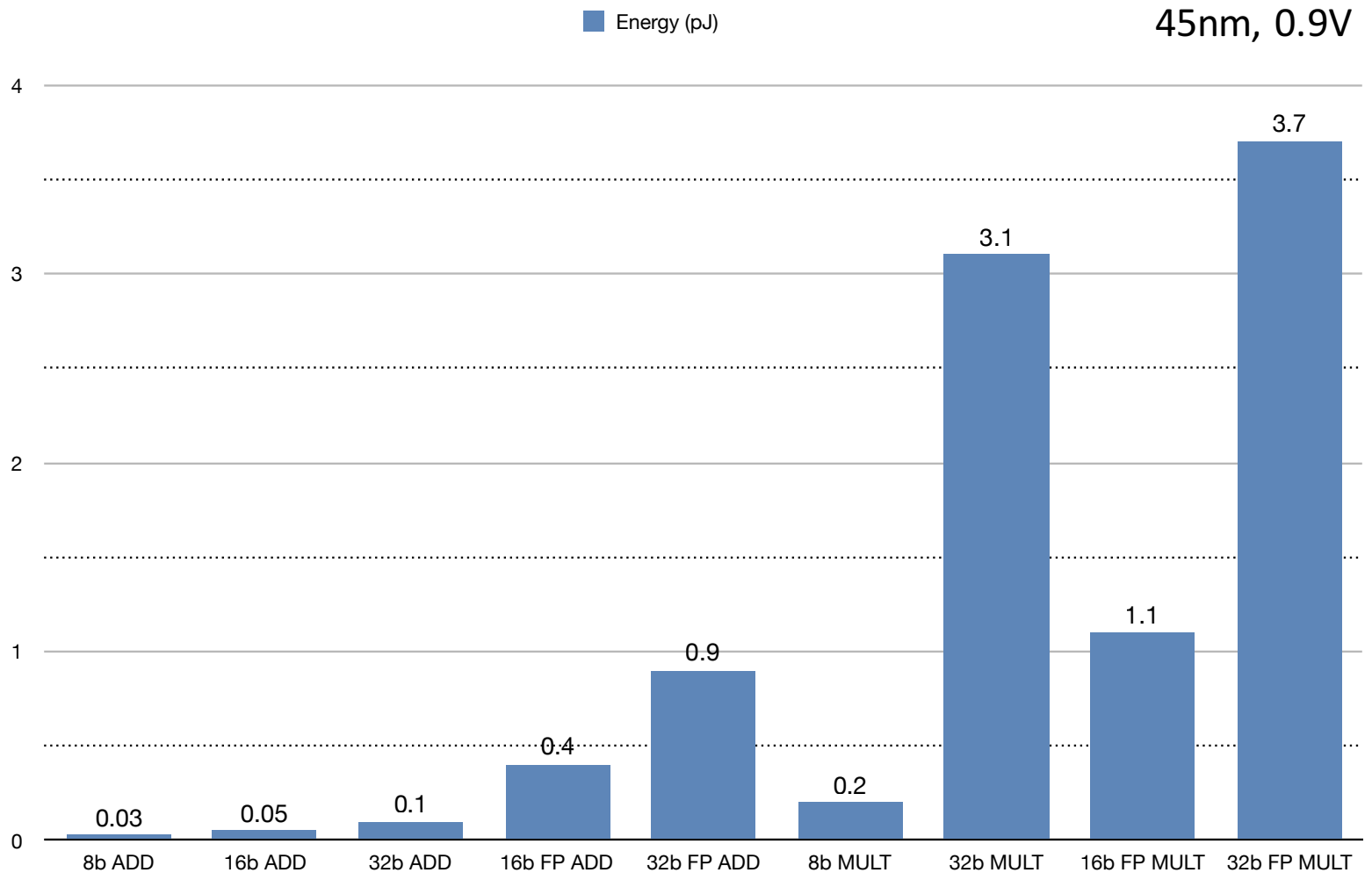
Gupta et al, 2015

Deep Learning with Limited Numerical Precision

- "*Stochastic rounding […] possesses the desirable property that the expected rounding error is zero*"
- *"It is well appreciated that in the presence of statistical approximation and estimation errors, high-precision computation in the context of learning is rather unnecessary"*
- *"Moreover, the addition of noise during training has been shown to improve the neural network's performance"*
- *"This work is built upon the idea that algorithm-level noise tolerance can be leveraged to simplify underlying hard-ware requirements"*

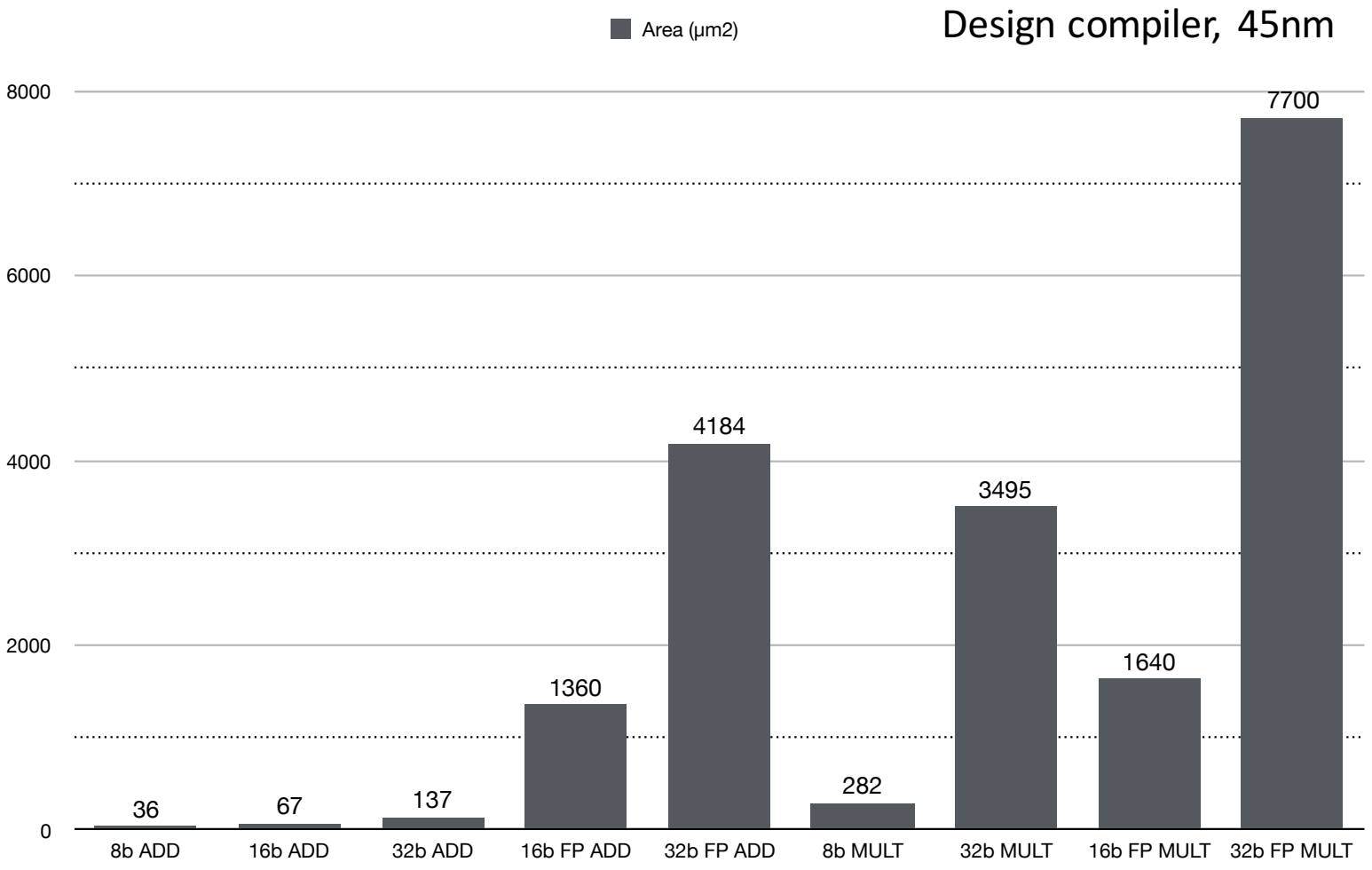
Gupta et al, 2015

Floating vs Fixed Point Arithmetic in Hardware



Data from: "High-Performance Hardware for Machine Learning", W. Dally, U.C. Berkeley, 2016

Floating vs Fixed Point Arithmetic in Hardware

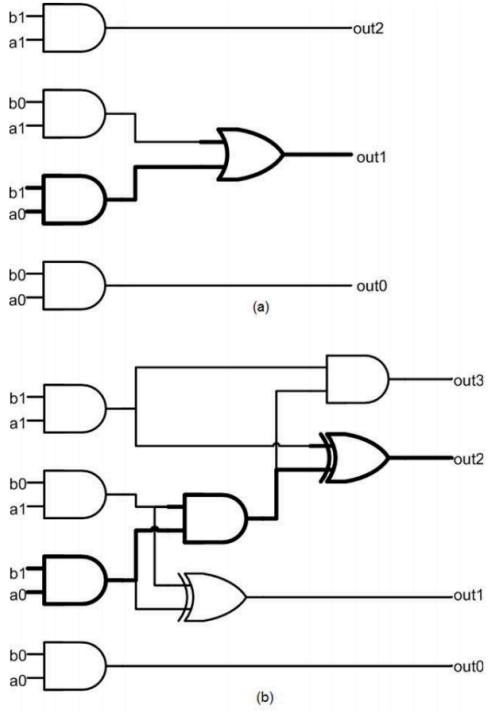


Data from: "High-Performance Hardware for Machine Learning", W. Dally, U.C. Berkeley, 2016

Approximate Computing

Introduce some level of error in the circuit to reduce area, power and delay. Application must be error tolerant.

- Probability of error 1/16
- Error magnitude 9-7=2
- When building larger multipliers, magnitude of error remains constant but probability of error rises
- Mean error saturates at 3.35%
- Mix accurate and inaccurate 2x2 multipliers to control error
- Power saving of up to 45%



a) approximate 2x2 multiplier
b) standard 2x2 multiplier

$a_1a_0b_1b_0$	out (approx)
0000	0000
0001	0000
0010	0000
0011	0000
0100	0000
0101	0001
0110	0010
0111	0011
1000	0000
1001	0010
1010	0100
1011	0110
1100	0000
1101	0011
1110	0110
1111	0111

Kulkarni et al, 2011; Ercegovac 2013.

Image sharpening with approximate multiplier (Gaussian smoothing)

(a) (b) (c)

a) original picture; b) smoothing with accurate multiplier; c) smoothing with inaccurate multiplier (41.5% power reduction).

Kulkarni et al, 2011; Ercegovac 2013.

Summary

- From profiling results, plasticity computation seems to be the main area to accelerate
- Future models will involve more and more differential equations per timer/spike interrupt
- Fastest accelerators can be implemented using fixed-point operations
- Rounding methods play crucial part in statistical results of the learning algorithms
- Approximate computing can be used to further reduce circuit sizes and delays
- Some SNN algorithms might need floating-point accuracy but others might get away with small reduced precision integers
- **Which parts of SNNs can tolerate arithmetic errors and how much?**

Questions