Introducing Loihi

Mike Davies
Director, Neuromorphic Computing Lab | Intel Labs

Acknowledgement to the entire Loihi team: Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgois Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steve McCoy, Arnab Paul, Jon Tse, Guru Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang

Motivation: The Case for Neuromorphic Computing

Problem Statement:

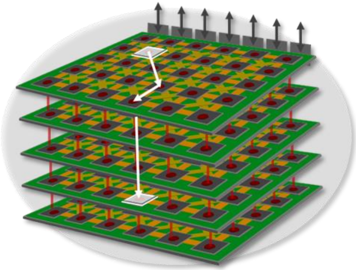
Emerging computing workloads demand *intelligent behaviors* that we do not know how to deliver *efficiently* with today's algorithms and computing architectures.

Examples:

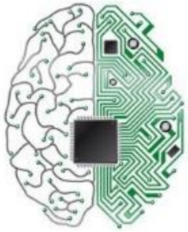
- Online and lifelong learning
- Learning without cloud assistance
- Learning with sparse supervision
- Understanding spatiotemporal data
- Probabilistic inference and learning
- Sparse coding/optimization
- Nonlinear adaptive control
- Pattern matching with high occlusion
- SLAM and path planning

Potential Future Product Applications

Robotics



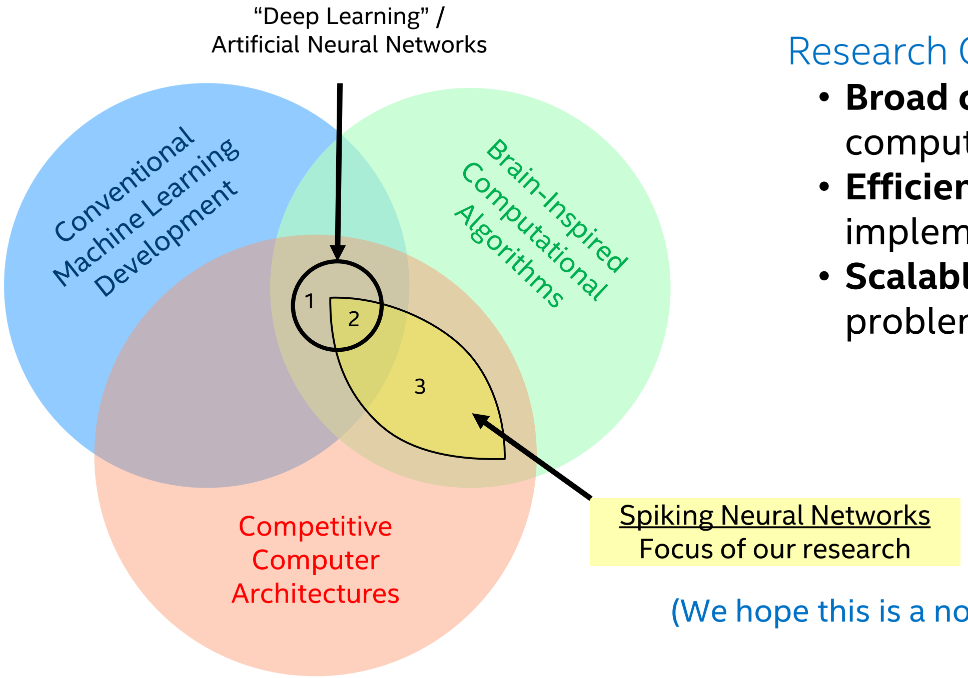
HPC Systems



Neuroprosthetics

Smart Glasses

Solution Exploration Space



Research Goals:

- **Broad class** of brain-inspired computation
- **Efficient** hardware implementations
- **Scalable** from small to large problems and systems

(We hope this is a non-empty class!)

The Engineering Perspective

- Nature has come up with something amazing. Let's copy it…
- Not so simple – very different design regimes
- Yet objectives and constraints are largely the same…
    - Energy minimization
    - Fast response time
    - Cheap to produce

Status today:

	Nature	Silicon	Ratio
Neuron density[1]	100k/mm^2	5k/mm^2	20x
Synaptic area[1]	0.001 um^2	0.4 um^2[2]	400x
Synaptic Op Energy	~2 fJ	~4 pJ	2000x

[1] Planar neocortex [2] ~5b SRAM

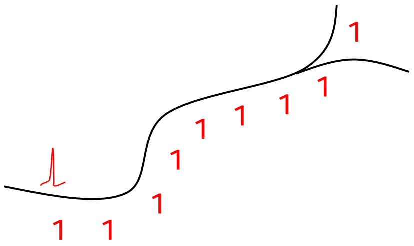
But…

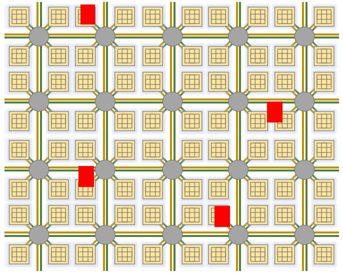
Max firing rate	100 Hz	1 GHz	10,000,000x
Synaptic error rate	75%	0%	∞

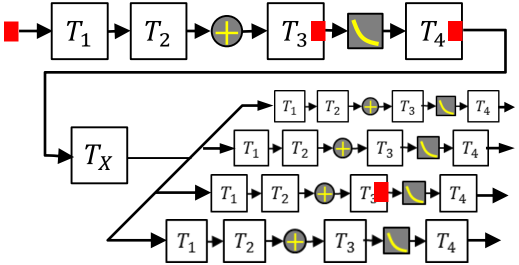
Nature	Silicon
Autonomous self-assembly	Fabricated manufacturing
Per-instance variability desired	Variability causes brittle failures
plasticity over lifetime	Must support rapid reprogramming
rministic operation	Deterministic operation desired

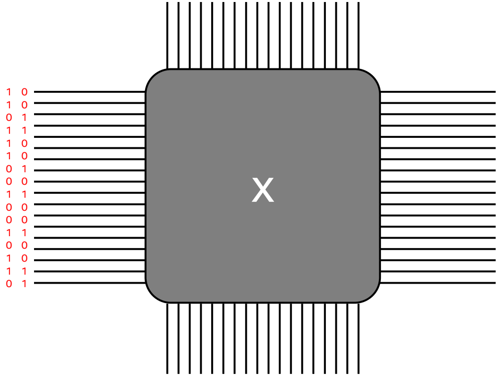
Need to understand and apply the basic principles, *adapting for differences*

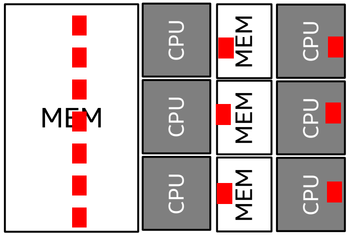
Are Spiking Architectures Efficient?

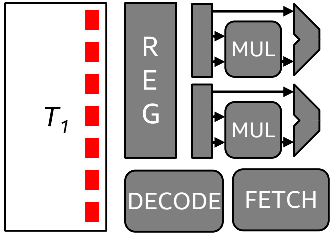












One Compelling Example: LASSO Sparse Coding

Problem

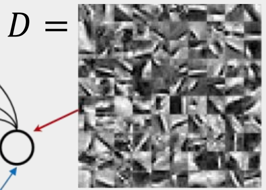
$$\min_z \frac{1}{2}\|x - Dz\|_2^2 + \lambda\|z\|_1$$

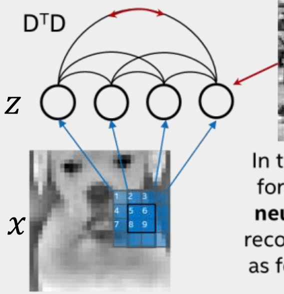
Input

Reconstruction

Sparse regularization

Implementation

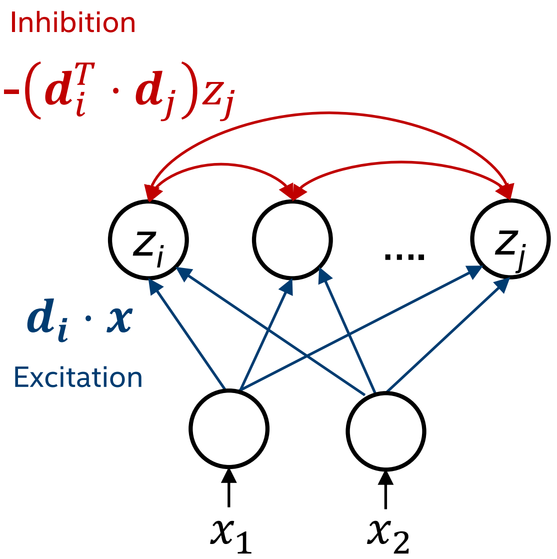


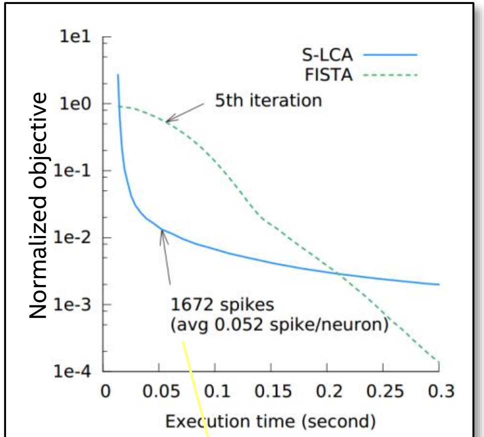


In the neural network formulation, **feature neurons compete** to reconstruct image with as few contributors as possible

Tang et al, arxiv: 1705:05475

LASSO Optimization Using the *Spiking Locally Competitive Algorithm*

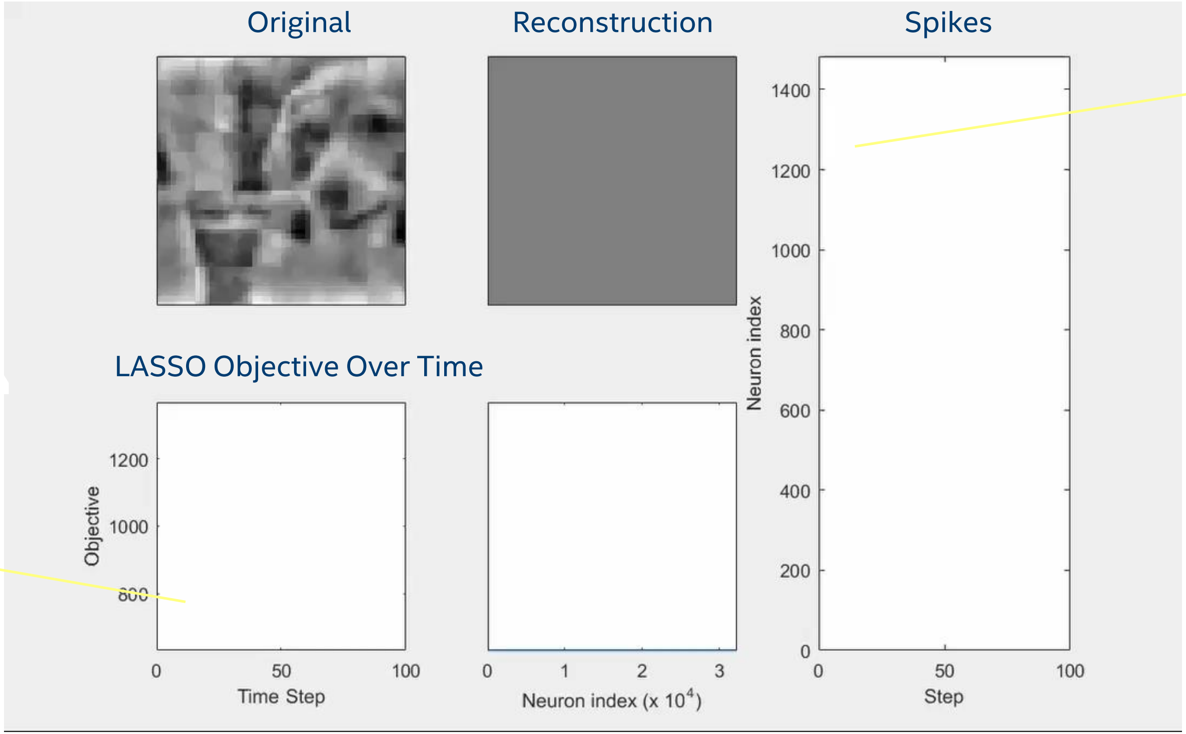




both S-LCA and FISTA running on a Xeon

Neuromorphic algorithm rapidly finds a near-optimal solution

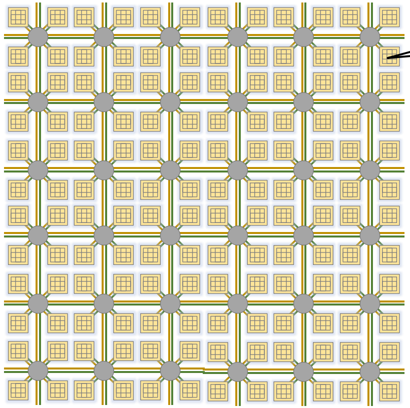
Spiking LCA dynamics on a Loihi predecessor



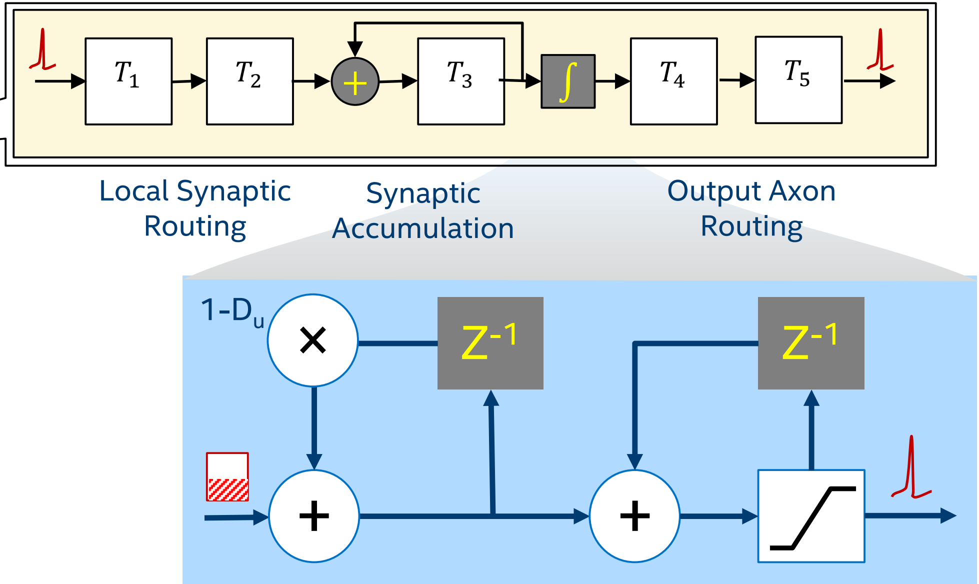
Much faster convergence on a neuromorphic architecture

Intense but very brief period of competition

What this gives us… a baseline SNN architecture



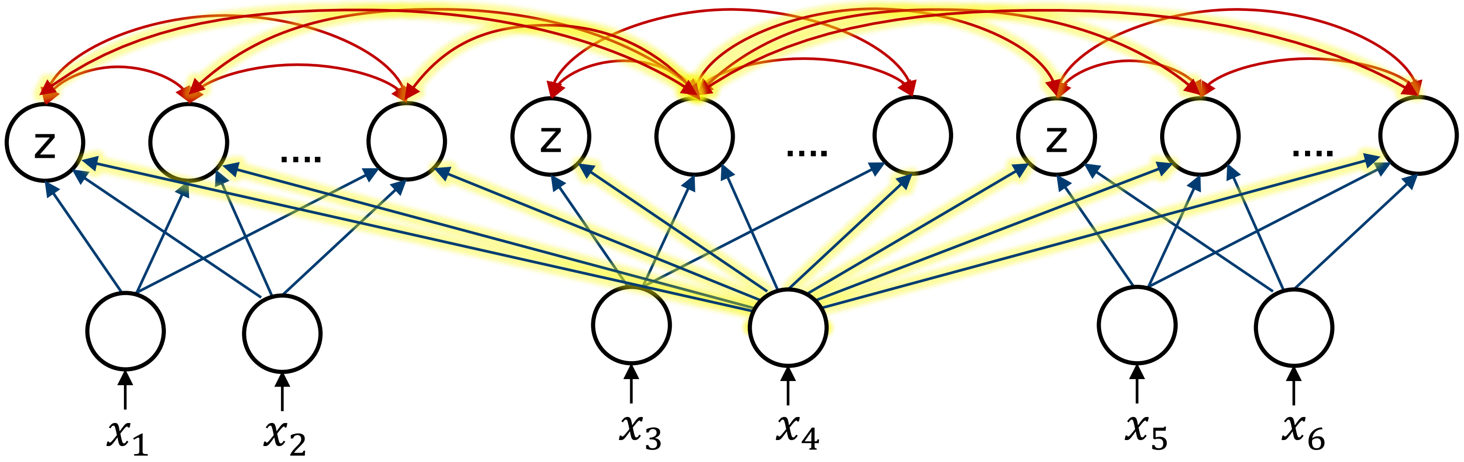
2D Mesh
Packetized spikes
High fanout required
Low overhead synchronization



Neuron Model (IF)

But how to scale to large LCA problems?

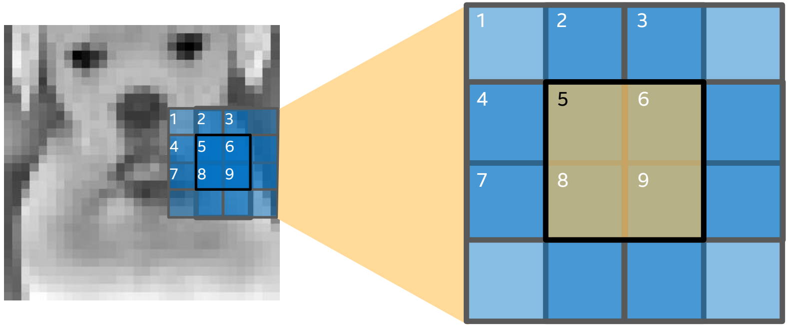
LCA is an all-to-all network...

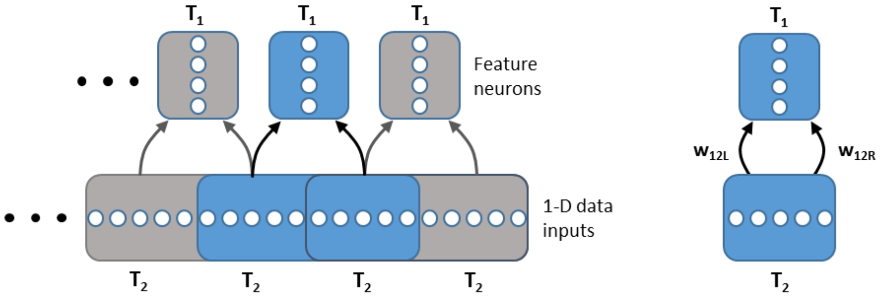


Just 1000 feature neurons requires 1000^2 = 1M synapses

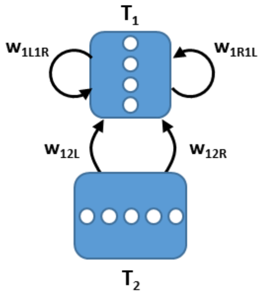
Answer: Patch-based Connectivity Reuse

Analogous to the “convolution” in ConvNets



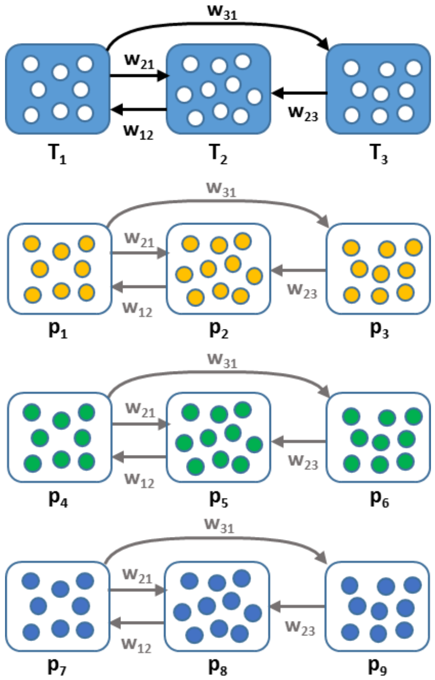


Conventional 1D convolution example

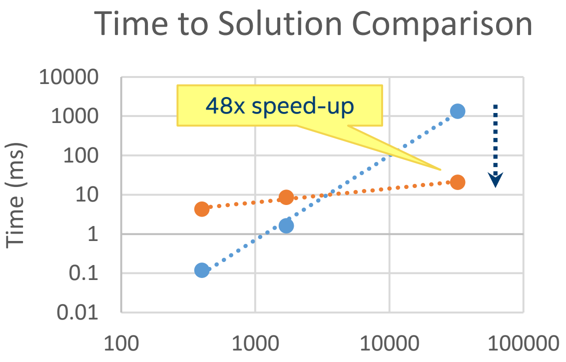


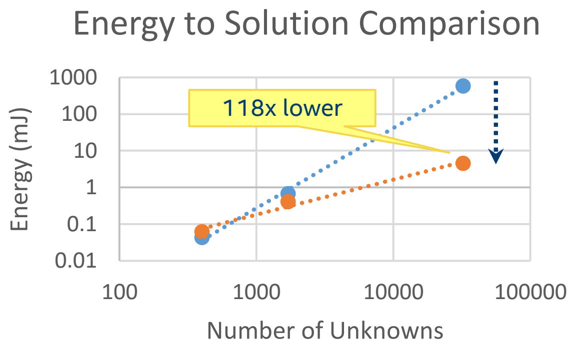
w/ Lateral inhibition

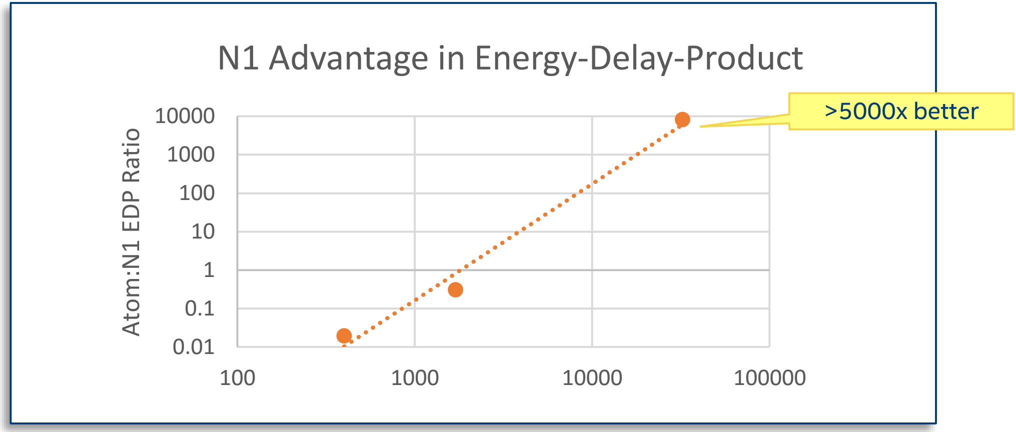
Generalized Hierarchical Connectivity Example



Sparse Coding Results: N1 vs Atom CPU

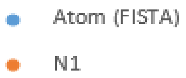






Comparison of sparse coding on N1 versus the FISTA* LASSO solver on an Atom CPU**

* Best conventional LASSO solver (LARS also evaluated)
** Iso-process, roughly iso-area (6-10mm^2)
PTPX-based measurements



Neuromorphic Core Architecture

All synaptic connections pooled
128KB shared memory

Sparse, *dense*, and *hierarchical*
Synaptic mapping representations

Synaptic delays

Synaptic eligibility traces

Flexible 3-tuple synaptic variables
(1-9b weight, 0-6b delay, 0-8b tag)

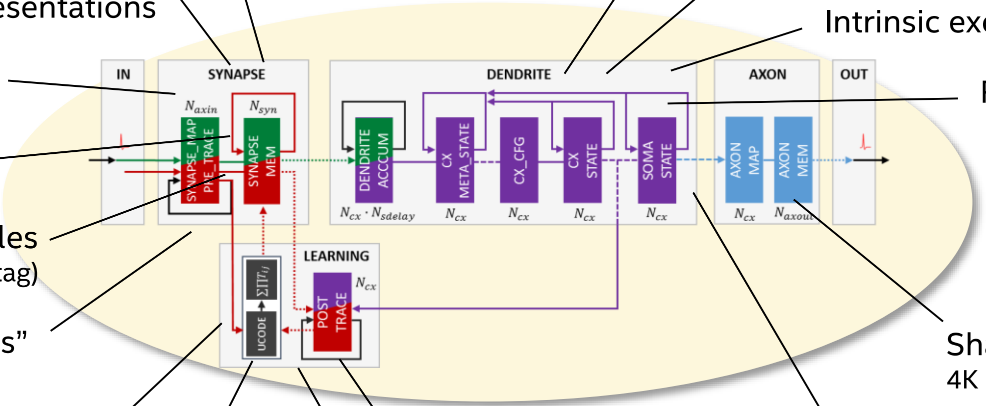
Graded “reward spikes”

Discrete time LIF neuron model (CUBA)

Multi-compartment dendritic trees
up to 1K compartments

Intrinsic excitability homeostasis

Random noise sources



Shared output routing table
4K axon routes

Flexible synaptic plasticity with microcode-programmable rules

Sum-of-products rule semantics

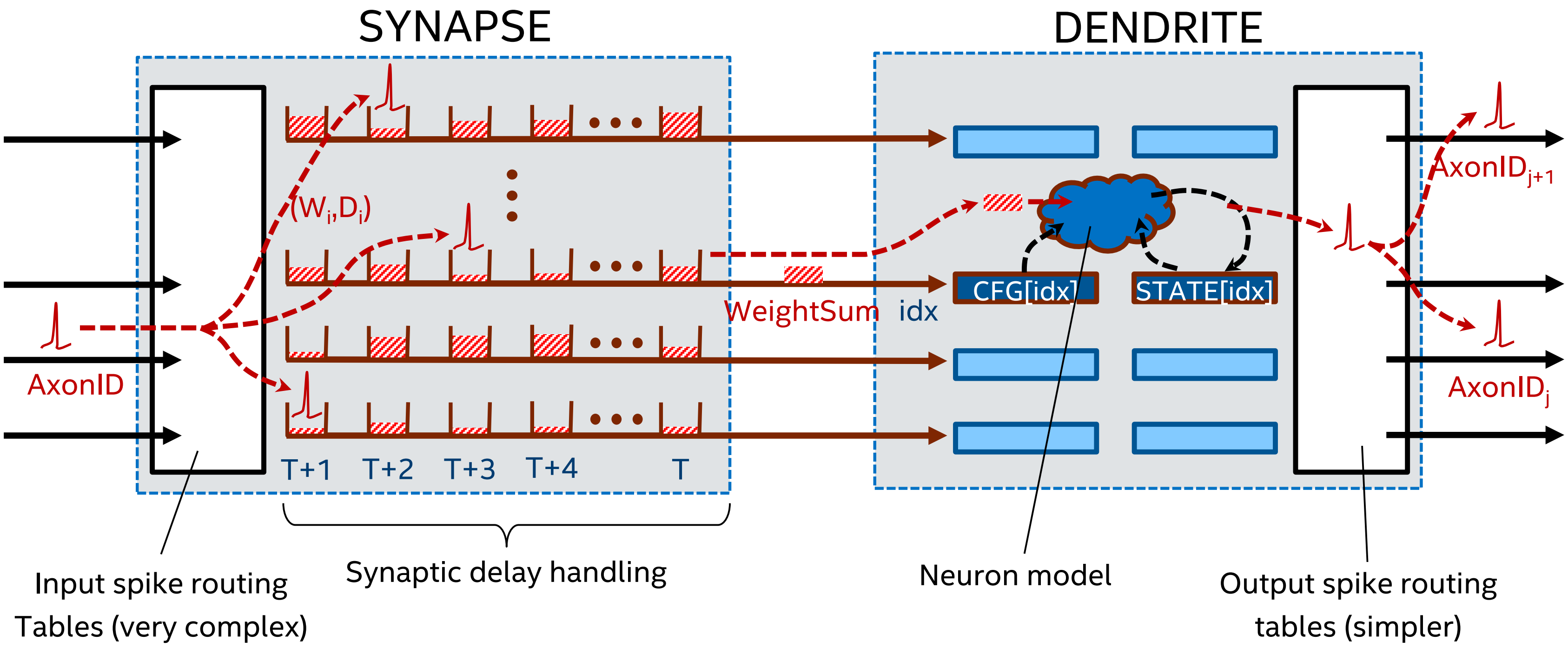
Filtered spike train traces

Plasticity rules target any synaptic variable

Axon delays
Refractory delays (+ random)

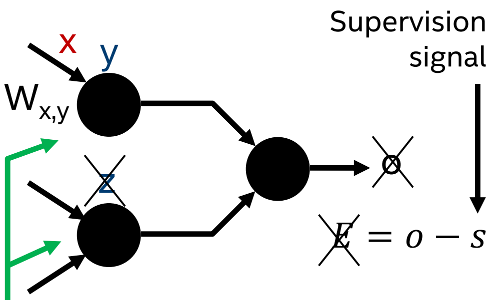
Basic Core Operation (Non-Learning)

(Time multiplexing illustrated unrolled in space)



Learning with Synaptic Plasticity

- **Local learning rules** – essential property for efficient scalability
  Compatible with biological plausibility
- Should be derived by **optimizing an emergent statistical objective**
  Too much directionless experimentation otherwise
- Plasticity on **wide range of time scales** is needed
  Delayed reward/punishment responses, eligibility traces

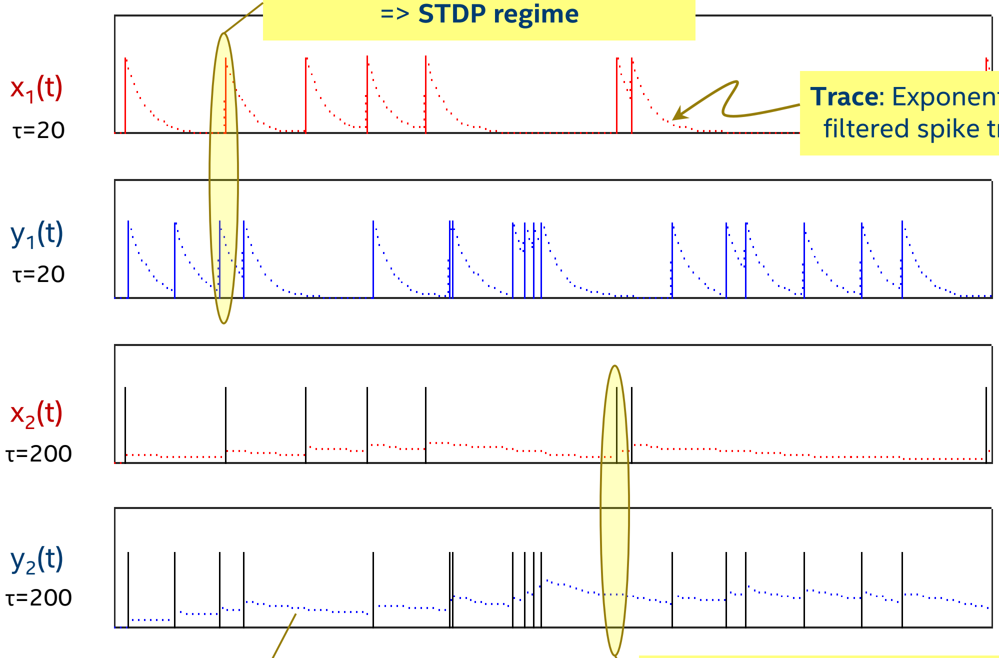


Learning rules for weight $W_{x,y}$ may *only* access presynaptic state x and postsynaptic state y

However *reward spikes* may be used to distribute graded reward/punishment values to a particular set of axon fanouts

Trace-Based Programmable Learning

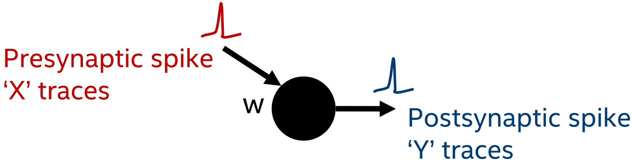
Short time scale trace correlations
=> **STDP regime**



Trace: Exponentially filtered spike train

Traces are **low precision** (7-9b) and may decay **stochastically** for implementation efficiency

Long time scale traces respond to correlations in activity rates



Weight, Delay, and Tag learning rules programmed as **sum-of-product equations**

$$w' = w + \sum_{i=1}^{N_P} S_i \prod_{j=1}^{n_i} (V_{i,j} + C_{i,j})$$

Synaptic Variables
Wgt, Delay, Tag
(variable precision)

Variable Dependencies
X_0, Y_0, X_1, Y_1, X_2, Y_2,
Wgt, Delay, Tag, etc.

Learning Rule Examples

Pairwise STDP:

$$W(t+1) = W(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t)$$

Triplet STDP with heterosynaptic decay:

$$W(t+1) = W(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t)y_2(t) - B \cdot W(t) \cdot y_3(t)$$

Delay STDP:

$$D(t+1) = D(t) - A_{-}x_0(t)(127 - y_1(t)) + A_{+}(127 - x_1(t))y_0(t)$$

Two-variable Learning Rule Examples

Distal Reward with Synaptic Tags:

$$T(t+1) = T(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t) - B \cdot T(t)$$

$$W(t+1) = W(t) + C \cdot r_1(t) \cdot T(t)$$

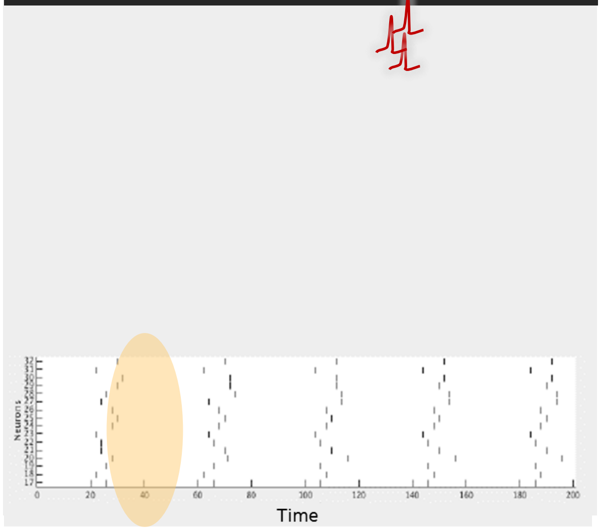
STDP with dynamic weight consolidation:

$$W(t+1) = W(t) - A_{-}x_0(t)y_1(t) + A_{+}x_1(t)y_0(t)y_2(t) - B_1(W-T)y_3(t)y_0(t)$$

$$T(t+1) = T(t) + \frac{1}{\tau_{cons}}(W-T) - B_2T(w_\theta - T)(w_{max} - T)$$

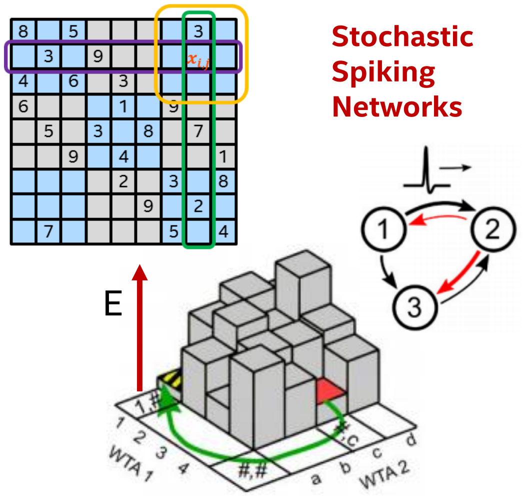
Example Novel Algorithms Supported by Loihi

Spatiotemporal Attractors



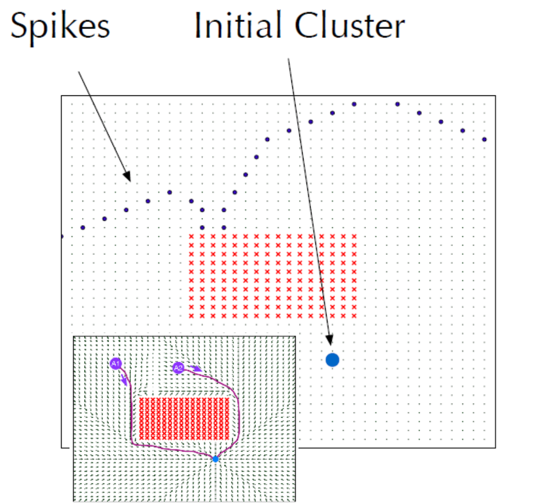
Artificial Olfaction

Constraint Satisfaction



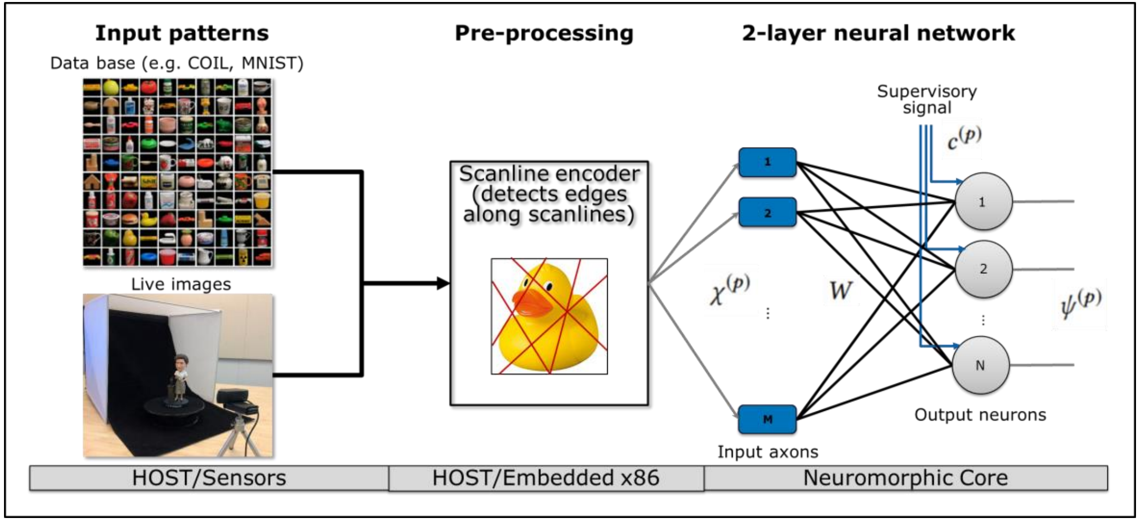
Sudoku

Graph Search

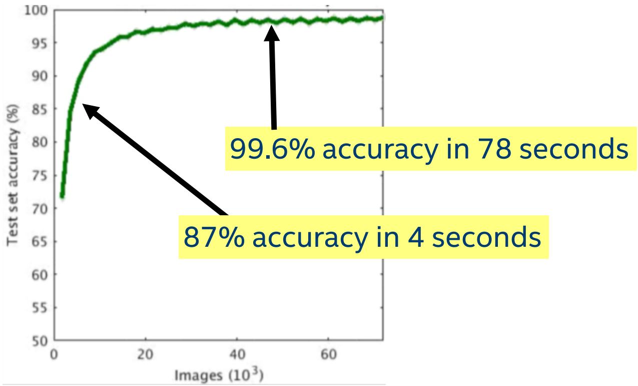


Path Planning

Our "Hello World" Application: Supervised Learning for Object Recognition



Performance on COIL20 data set



	Training	Inference
Active energy per image (total)	553 uJ	128 uJ
Neuromorphic energy	322 uJ	13 uJ
Processing time per image	7.5 ms	1.8 ms
Chip power	74 mW	73 mW
Neuromorphic power	43 mW	7.4 mW

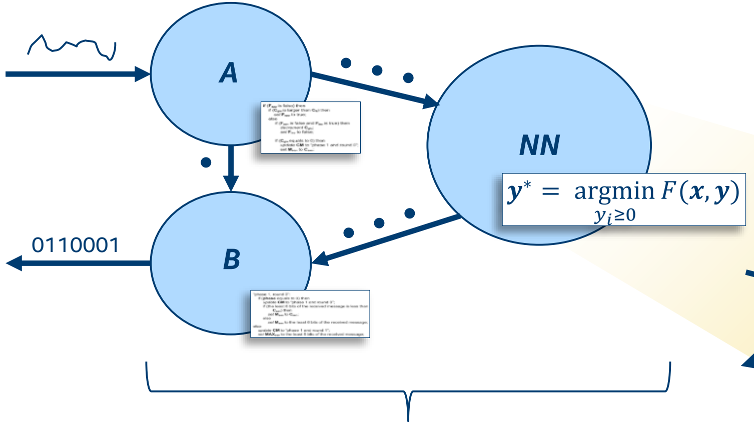
Resource Utilization	Count	Utilization
Neurons	20	0.02%
Synapses	38400	0.28%
SNN Cores	1	078%

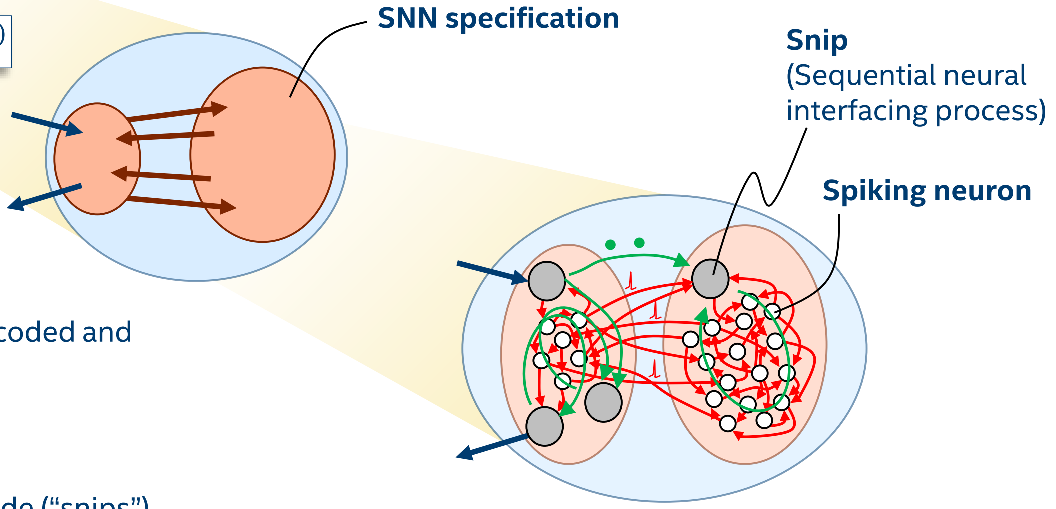
S-STDP rule:

$$W_{i,j}(t) = W_{i,j}(t-1) + \eta \cdot (u_\kappa \cdot \delta_{i,C(p)} - y_{i,0}) \cdot x_{j,1}$$

Up to the 10,000 foot view

The Nx System Framework
- Heterogeneous hierarchical parallel system
- Event-driven communication over channels
- Localized state
- Models describe *emergent behavior*



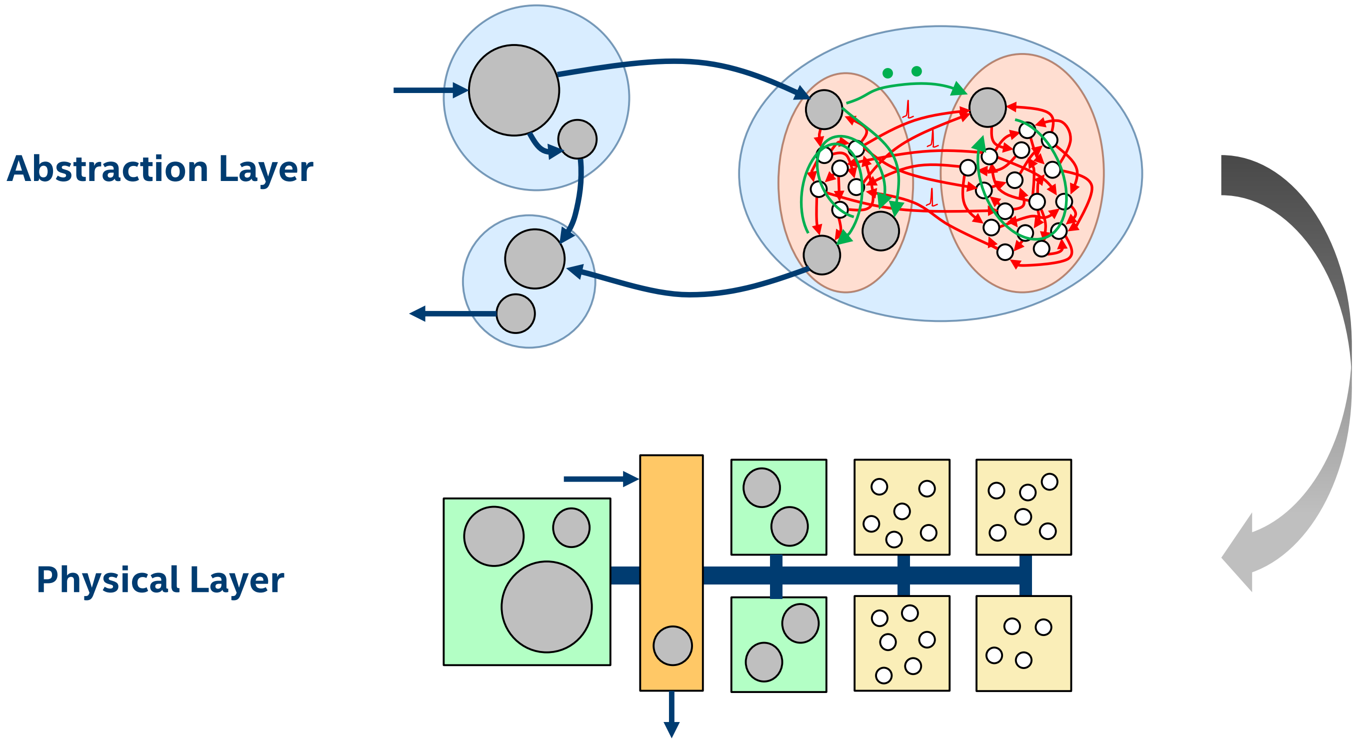


Modules w/ behavioral models

A, B: Sequential processes conventionally coded and run on conventional CPUs

NN: Neural network module
- Hierarchically specified
- Mathematical behavioral model
- May include conventional helper code (“snips”)

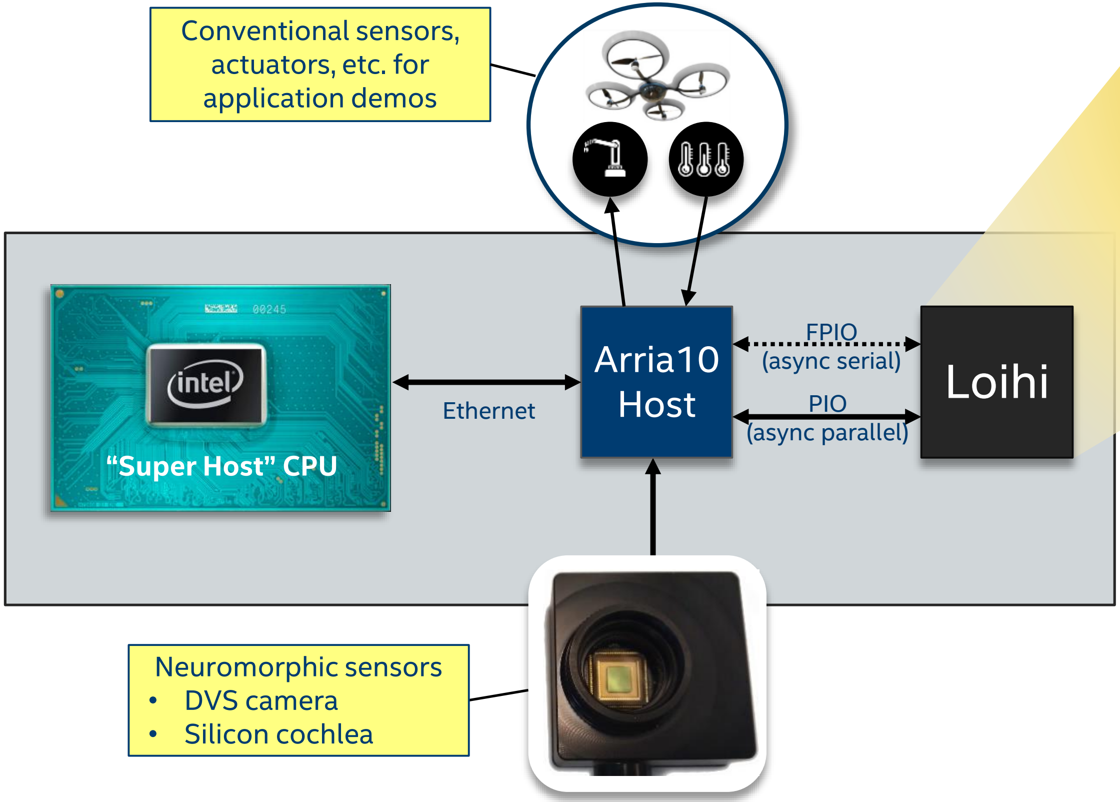
Mapping to the Physical Layer

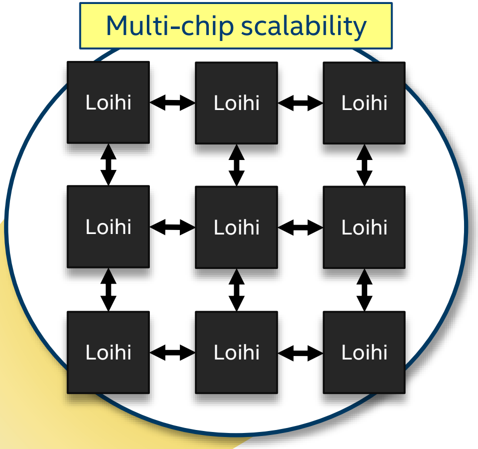


"Atoms" of the computational hierarchy are mapped to the system hardware resources

(Static scheduling)

System Architecture Today





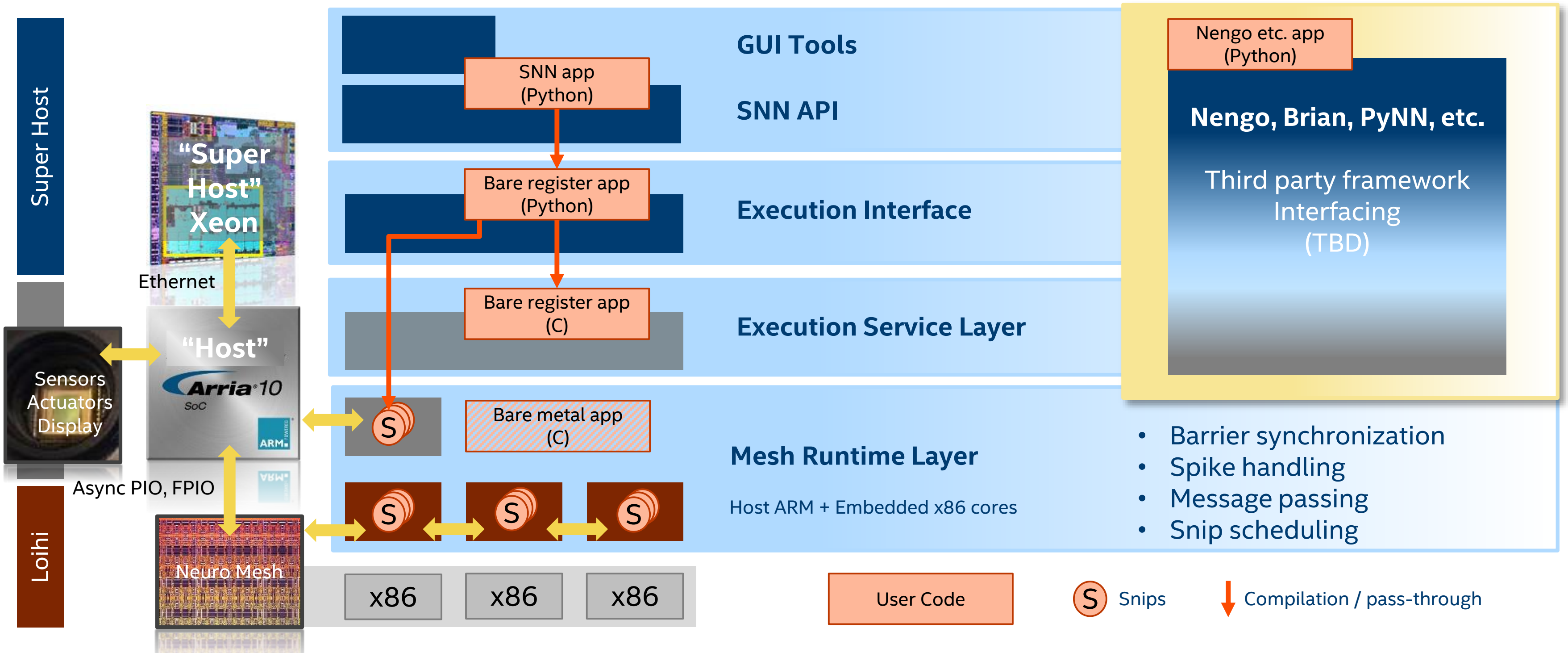
"Super Host" CPU
- Owns the high-level application
- Compilation, visualization, debug, UI

Arria10 Host
- Manages an entire mesh of Loihi chips
- Glue logic to Loihi interfaces
- Interface to real world/time data
- Spike encoding/decoding in some cases

Loihi
- Event-driven I/O model
- Participates in barrier synchronization

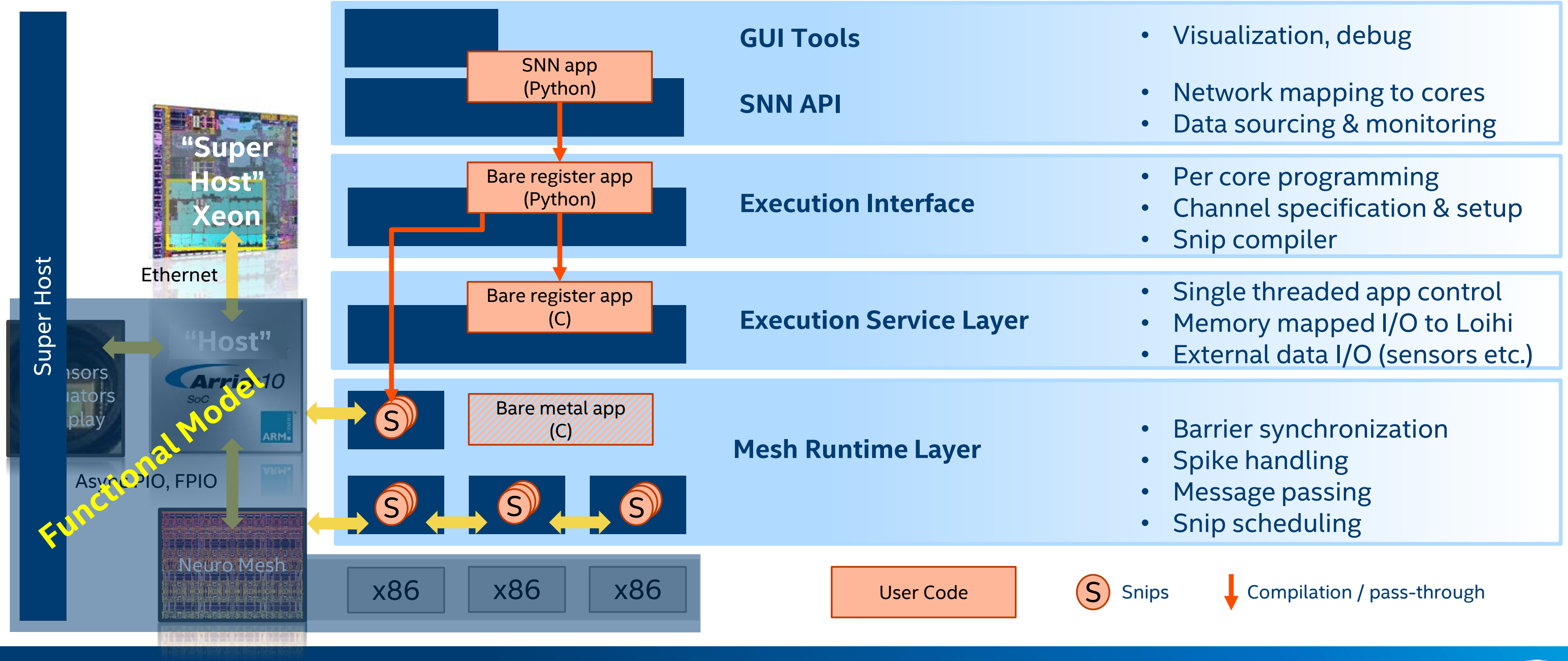
Current Software Development Kit

(work in progress)

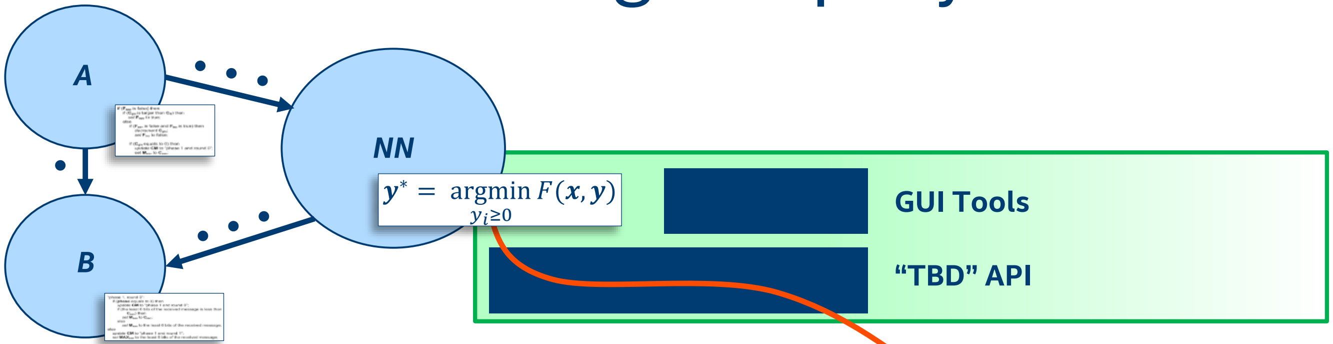


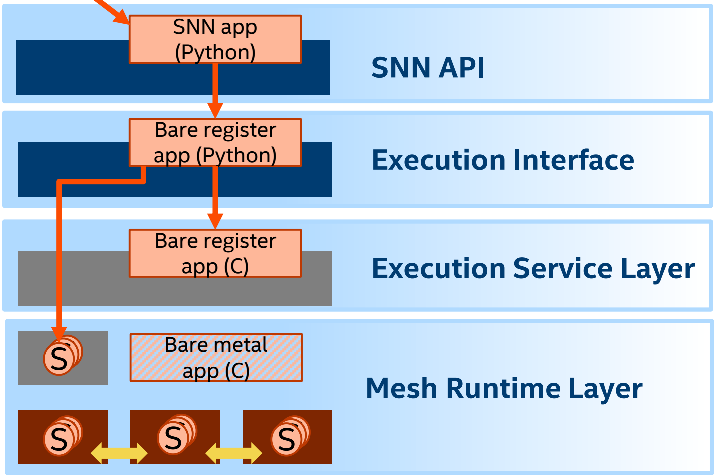
Current Software Development Kit

(work in progress)



What's the right top layer of the SDK?





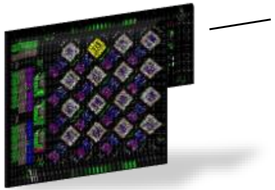
Not TensorFlow / other DL frameworks (wrong abstractions)

This is the unexplored frontier of neuromorphic software research

Loihi Systems Outlook

Today
Wolf Mountain
Remote Access
4 Loihi/Board

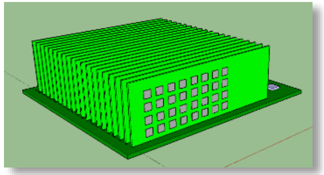
Early Q2 2018
Nahuku
Remote Access
8 Loihi/Board



Interfaces to standard Arria10 evaluation systems

Late Q2 2018
Nahuku
1st availability
for collaborators
8-32 Loihi/Board

Late Q3 2018
Pohoiki Springs
Remote Access
Very large system



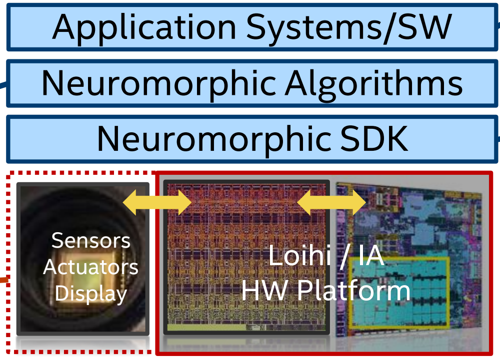
Future
Embedded System
1 Loihi

Intel Neuromorphic Research Community

RV1: Theory
- Abstract and quantify features of neuroscience to the context of systems engineering
- Computational complexity frameworks

RV2: Algorithms
- Principled derivations of SNN dynamics, features, and learning rules.

RV5: Sensors and Control
- Sparse, event-driven I/O for SNN systems



RV3: Applications
- Applications of Loihi and future Intel neuromorphic silicon / FPGA designs
- Benchmarks and value analysis may itself be research.

RV4: Programming Models
- New paradigms for conceptualizing and specifying SNN/neuromorphic algorithms

We wish to engage with collaborators in academic, government, industry research groups

INRC goals:
- Demonstrate value of Loihi vs conventional solutions
- Share code, results, algorithms
- Motivate improvements for future silicon iterations

What we offer to INRC collaborators
- Remote access to Loihi systems, SDK, SW
- Loaned Loihi systems and bare chips (limited)
- Opportunity for limited funding (RFP available late March)

Please Join Us! (at the right time)

You:
- Extensive experience with SNNs
- Extensive experience with other neuromorphic HW platforms

Us:
- Highly bandwidth limited

Email inrc_interest@intel.com for more information

Telluride 2018

Today

2018

2019

You:
- Vision for SNN application/algorithm research
- Can articulate the promise/value of project
- Can benchmark the result
- Interested in neuromorphic SW development

Us:
- More systems & documentation
- Complete SDK
- Scalable remote access system

You:
- Have a real-world problem not well solved now
- Prior SNN experience not necessary

Us:
- Mature, cross-framework SDK

Community:
- Critical mass, community forums, etc.
- Usable library of SDKs, tools, code, modules

Email inrc_interest@intel.com for more information

LEGAL INFORMATION

This presentation contains the general insights and opinions of Intel Corporation (“Intel”). The information in this presentation is provided for information only and is not to be relied upon for any other purpose than educational. Statements in this document that refer to Intel’s plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel’s results and plans is included in Intel’s SEC filings, including the annual report on Form 10-K.

Any forecasts of goods and services needed for Intel’s operations are provided for discussion purposes only. Intel will have no liability to make any purchase in connection with forecasts published in this document. Intel accepts no duty to update this presentation based on more current information. Intel is not liable for any damages, direct or indirect, consequential or otherwise, that may arise, directly or indirectly, from the use or misuse of the information in this presentation. Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.