

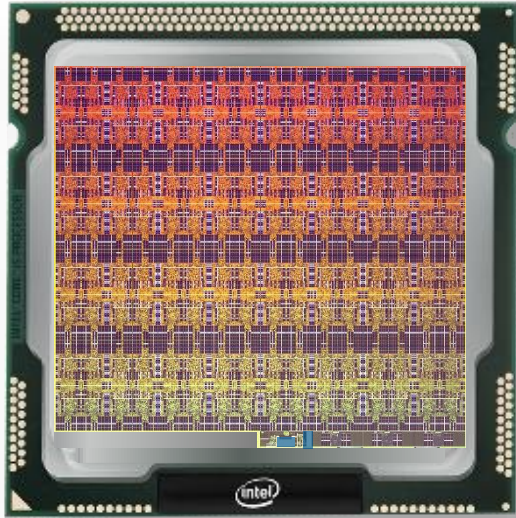


Loihi – a brief introduction

Mike Davies

Director, Neuromorphic Computing Lab | Intel Labs

Loihi at a Glance



**Integrated
Memory + Compute
Neuromorphic Architecture**

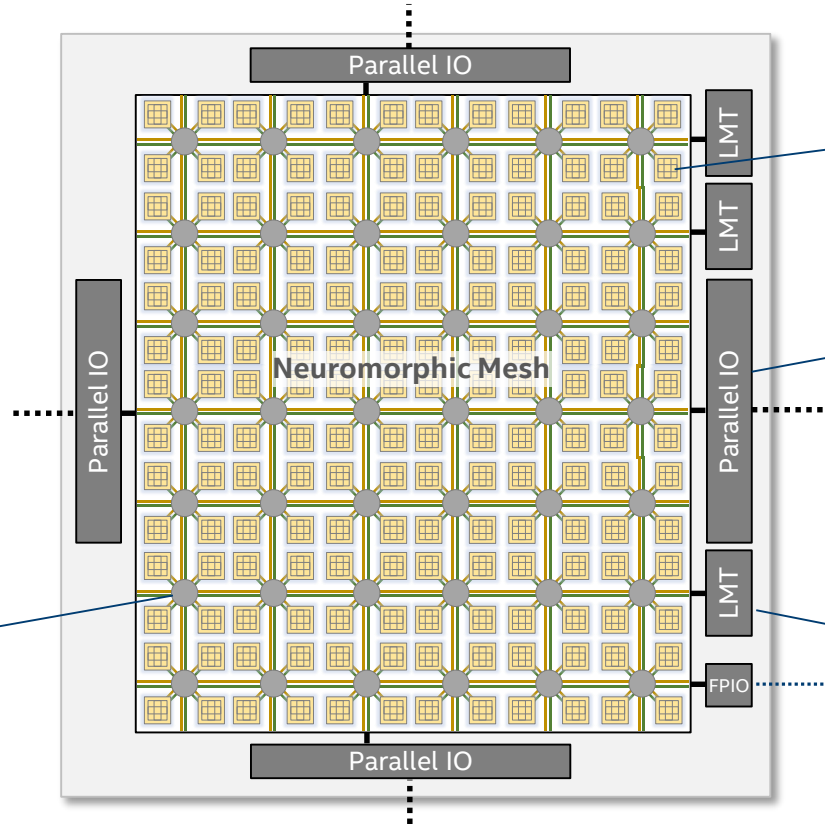
Key Properties

- 128 neuromorphic cores supporting up to 128k neurons and 128M synapses with an **advanced SNN feature set**.
- **Scalable on-chip learning** capabilities to support a range of learning paradigms (unsupervised, supervised, reinforcement-based, and others)
- Supports **highly complex neural network topologies** (up to 2000-way fan-out between neurons)
- Fully digital **asynchronous** implementation
- Fabricated in Intel's **14nm FinFET process** technology

Chip Architecture

Technology:	14nm
Die Area:	64 mm ²
Core area:	0.41 mm ²
NmC cores:	128 cores
x86 cores:	3 LMT cores
Max # neurons:	128K neurons
Max # synapses:	128M synapses
Transistors:	2.07 billion

- Low-overhead NoC fabric**
- 8x16-core 2D mesh
 - Scalable to 1000's cores
 - Dimension order routed
 - Two physical fabrics
 - 8 GB/s per hop

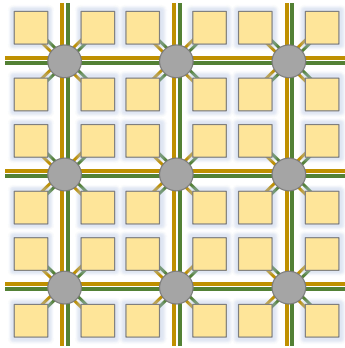


- Neuromorphic core**
- LIF neuron model
 - Programmable learning
 - 128 KB synaptic memory
 - Up to 1,024 neurons
 - Asynchronous design

- Parallel off-chip interfaces**
- Two-phase asynchronous
 - Single-ended signaling
 - 100-200 MB/s BW

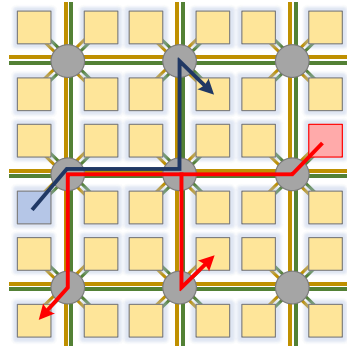
- Embedded x86 processors**
- Efficient spike-based communication with neuromorphic cores
 - Data encoding/decoding
 - Network configuration
 - Synchronous design

Mesh Operation



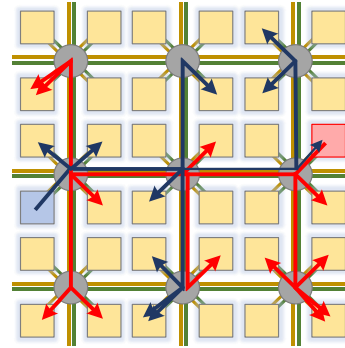
Time step T begins.

Cores update dynamic neuron state and evaluate firing thresholds

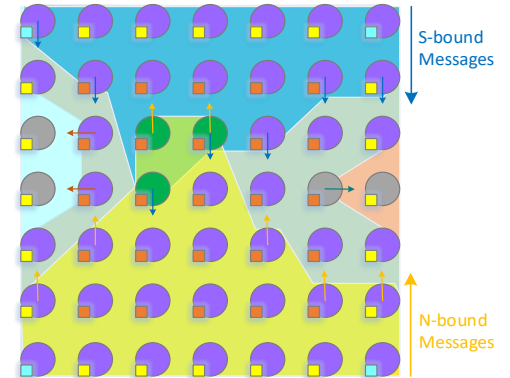
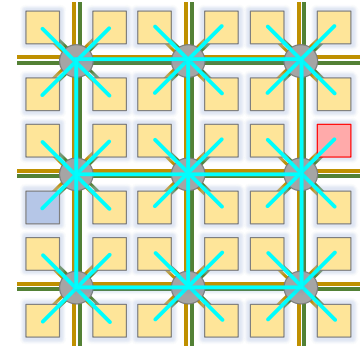


Above-threshold neurons send spike messages to fanout cores

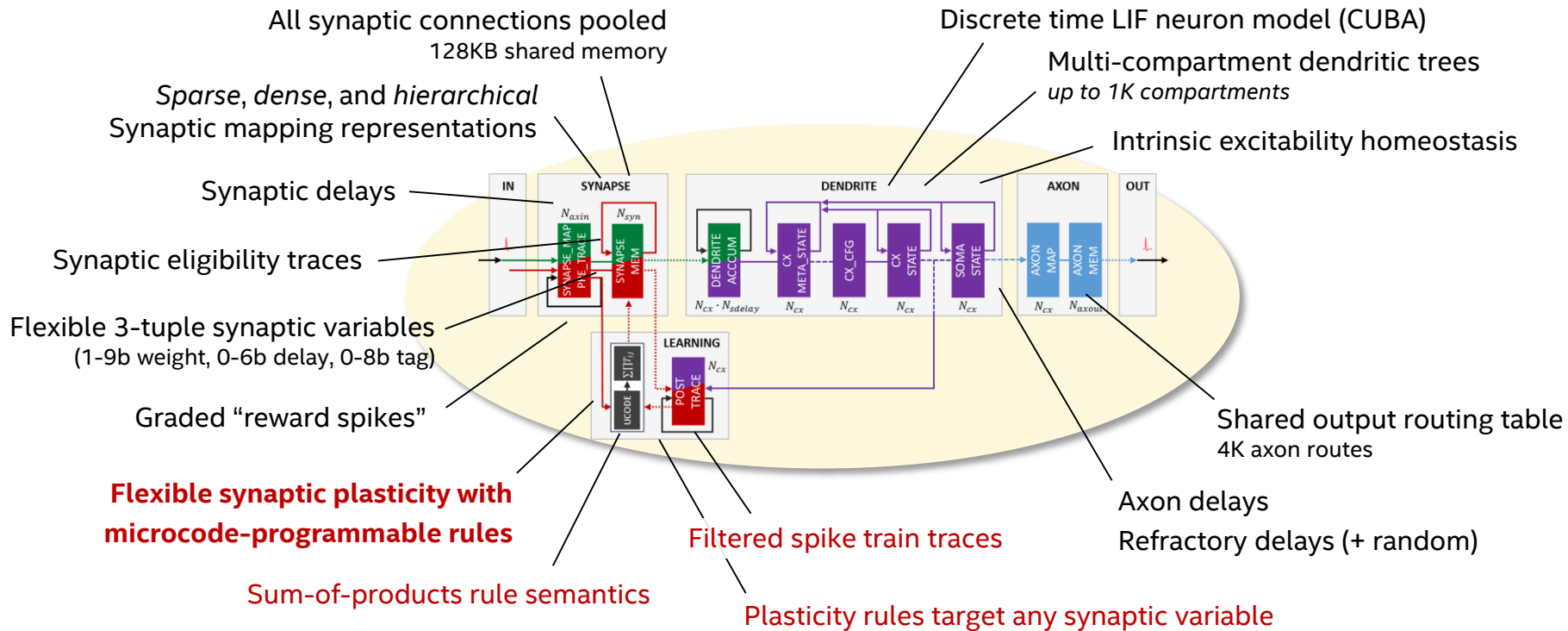
(Two neuron firings shown.)



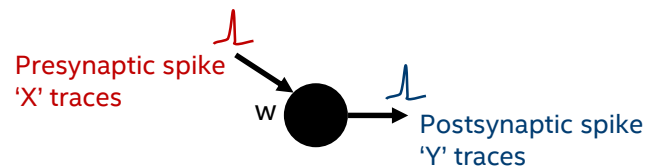
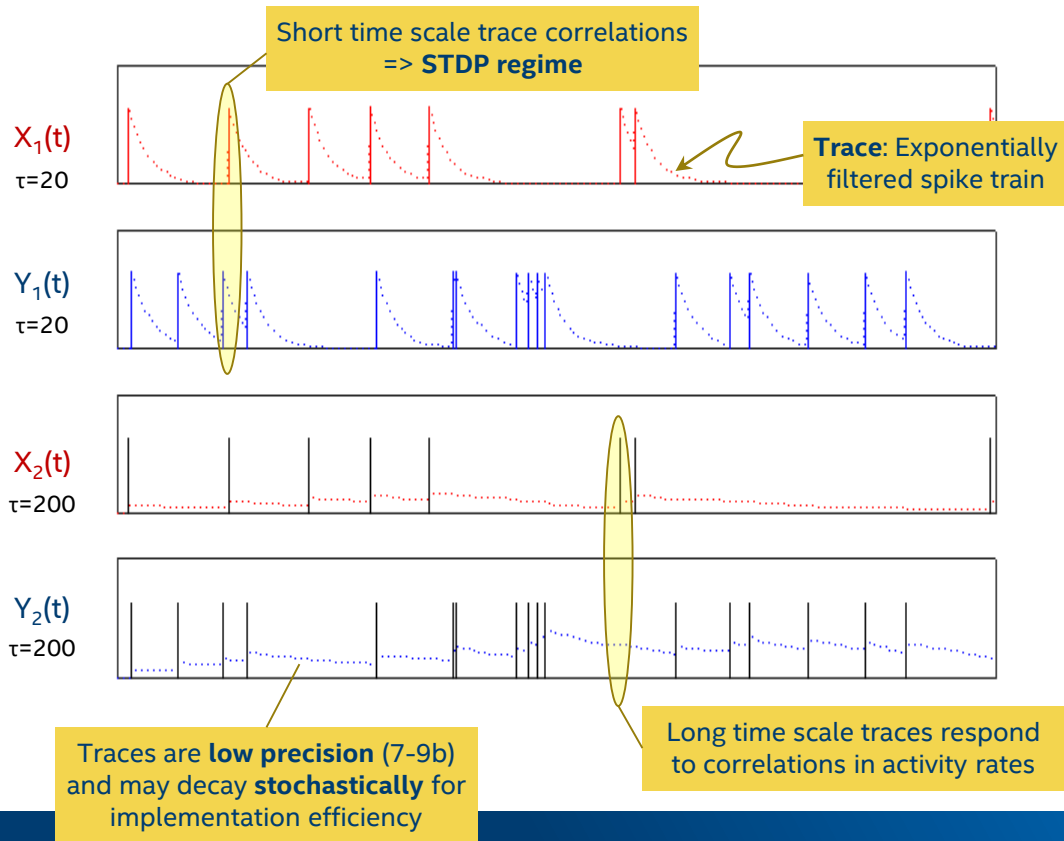
All neurons that fire in time T route their spike messages to all destination cores.



Neuromorphic Core Architecture



Trace-Based Programmable Learning Rules



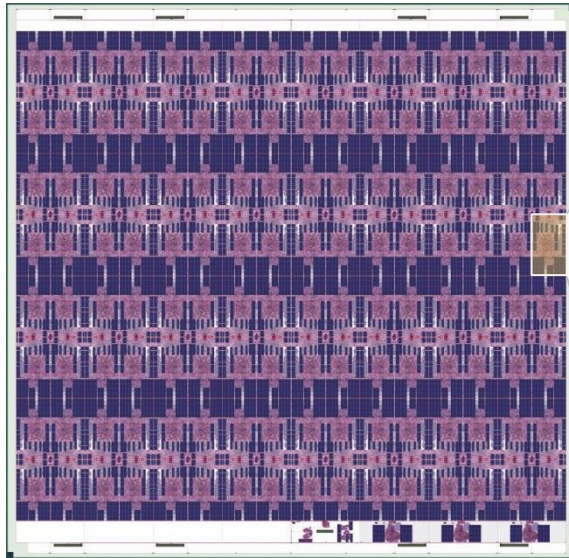
Weight, Delay, and Tag learning rules
programmed as **sum-of-product equations**

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

Synaptic Variables
Wgt, Delay, Tag
(variable precision)

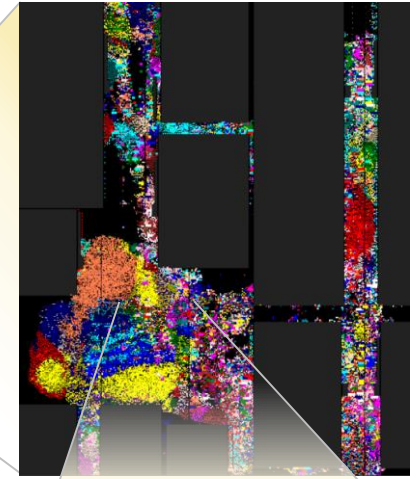
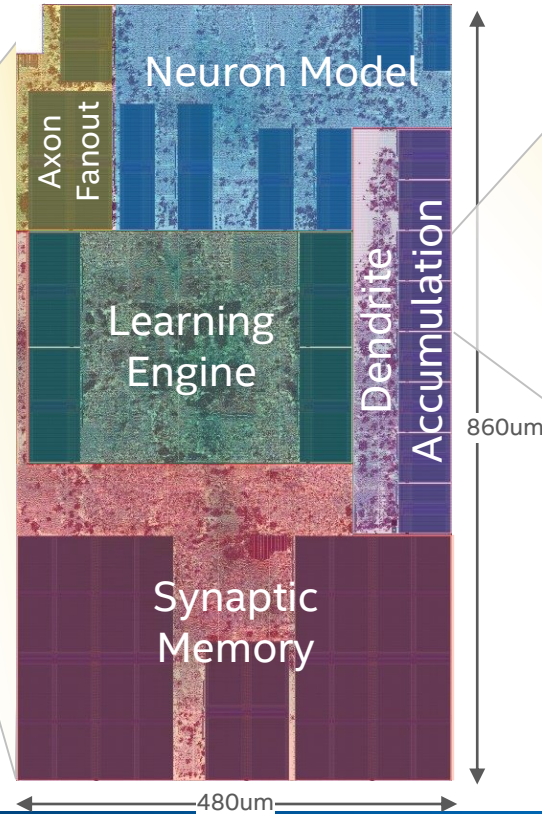
Variable Dependencies
 $X_0, Y_0, X_1, Y_1, X_2, Y_2,$
Wgt, Delay, Tag, etc.

Physical Implementation



Bundled Data Asynchronous Implementation

- Event-driven with integrated flow control
- Fully automated design flow from CSP
- Supports FPGA emulation
- Integrates with synchronous x86 CPUs



One asynchronous controller's associated pipeline logic

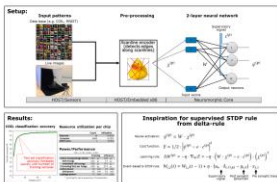
Application Results



SUPERVISED LEARNING ON LOIHI

Spiking neural network rapidly learns to recognize labeled real-world objects (Our "Hello World" network)

- Efficient supervised learning is achieved with spiking neural network using supervised form of spike timing dependent plasticity (STDP) rule.
- Heterogeneous compute platform: Demo showcases data acquisition on host, pre-processing on host/embedded x86 core and training & inference on Loihi's neuromorphic cores.
- Input images are converted to spatiotemporal spike patterns encoded by scan lines.

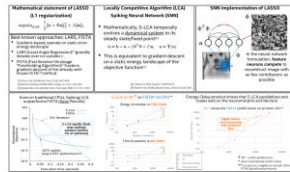


Intel Neuromorphic Computing Lab

EFFICIENT SPARSE CODING ON LOIHI

Solve LASSO optimization with orders of magnitude lower energy and runtime

- Solving LASSO optimization is foundational for many sparse coding problems in ML/AI. Signal Processing, Statistics, etc.
- Loihi solves LASSO within 1% of optimal solution with significantly lower energy and shorter runtime than classical solvers on CPU (great for real-time applications)
- Scaling advantage: Superiority in energy/runtime grows with problem size due to integrated compute & memory and event based encoding by scan lines.

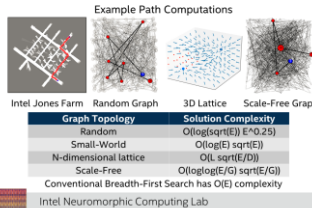


Intel Neuromorphic Computing Lab

PARALLEL PATH SEARCH WITH PROPAGATING SPIKES AND STDP

A Loihi network efficiently computes shortest paths in arbitrary graph topologies

- Neurons represent nodes of the graph and synapses represent their interconnections.
- Target node(s) is stimulated resulting in a propagating chain of spikes.
- STDP acts on the propagating spikes to encode paths to the target(s) in the synaptic weights of the network. Paths are decoded by reading out the synaptic weights or by subsequent stimulation of the network.
- The solution generalizes to arbitrary graph topologies.



Intel Neuromorphic Computing Lab

Reinforcement learning with spiking neurons

Abstract: Reinforcement learning (RL) has seen wide application in contemporary machine learning. Few RL algorithms have been shown to work efficiently on a spiking neuron implementation, and have never been implemented in neuromorphic hardware. We demonstrate one such algorithm and apply it to a simple game, Tic Tac Toe.

Methods: We developed an algorithm to train a Monte Carlo reinforcement learning agent on the Tic Tac Toe game. We implemented it on a Loihi chip and a Mezzio II neuromorphic computer.

Neuromorphic keyword detection

Abstract: Keyword detection systems are an essential component in most voice-controlled electronic devices. To function effectively, a keyword detector must continuously monitor a stream of audio input to identify the most likely words that occur within a sentence phrase in a "wake-up" state. Due to the "wake-up" requirement of keyword detection, they can consume significant energy, especially on mobile and embedded devices.

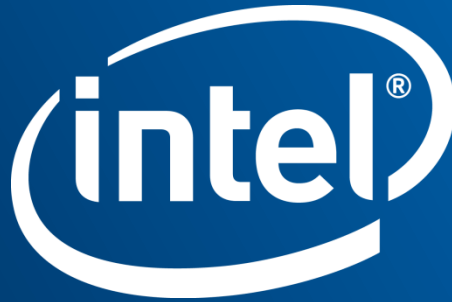
Methods: To train a neuromorphic keyword detector, we first collect a small corpus of speech samples including both the target keyword, "John", and related decoder phrases, for the sake of a balanced keyword search space. We then train a neuromorphic keyword detector on these samples using the Intel Loihi II chip and the Mezzio II neuromorphic computer.

Adaptive control of a robot arm with Loihi

Abstract: Force control of robotic systems provides several highly desirable qualities, most valued among them being safe human interaction. To provide effective force control, a controller must have an accurate model of the robot's dynamics. These dynamics are difficult to model for complex systems, and subject to change over the lifetime of the machine.

Methods: Adaptive control methods provide a means of learning dynamics online, addressing these issues. We have implemented an adaptive neuromorphic controller for the 1-degree-of-freedom (DOF) KUKA Junior II robot, using the Intel Loihi II chip and Mezzio II computer.

More to come – Thursday morning
Also see our posters and demos for more



M. Davies *et al.*, "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning,"
in *IEEE Micro*, vol. 38, no. 1, pp. 82-99, January/February 2018

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.

Copyright © 2018 Intel Corporation.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others