Open-Source Neuromorphic Computing

Jason K. Eshraghian
Assistant Professor, ECE, UC Santa Cruz

13 December 2023
Power bill of training AI in data centers: >$1 million

Power budget of brains: ~12 watts

At the UCSC NCG, our goal is to bridge this gap by applying principles from neuroscience to build better technology.
Circuits from 1D→ 2D → 3D → ...4D?

**Process Technology**
- Planar FET: 1 gate on channel
- FinFET: 3 gates on channel
- Gate-All-Around: 4 gates on channel

**Memory Circuits**
- Bit Line BL Contact
- 64MB L3 Cache Extension

**Microprocessors**
- AMD 3D V-Cache

**Image Sensors**
- Sony & Prophesee, 2021.

**Neuromorphic Computing**
- Take the computational principles that underpin the brain and integrate them with silicon
  - TinyML
  - Compressed Models
  - Quantization
  - Spike-based processing
  - Temporal coding
  - In-memory computing
  - Dataflow
In-Memory Processing Using RRAM Crossbars

Valence Change Mechanism

BEOL metal-insulator-metal structure

Bit-line current summation

Chip micrograph: 65-nm mixed-signal 16x16 RRAM Crossbar
In-Memory Processing Using RRAM Crossbars

Neuron Weights

\[
\begin{bmatrix}
  a_0 & b_0 & c_0 \\
  a_1 & b_1 & c_1 \\
  \vdots & \vdots & \vdots \\
  a_n & b_n & c_n
\end{bmatrix}
\]

Output Equations

\[
\begin{align*}
y_a &= x_0a_0 + x_1a_1 + x_2a_2 \\
y_b &= x_0b_0 + x_1b_1 + x_2b_2 \\
y_c &= x_0c_0 + x_1c_1 + x_2c_2
\end{align*}
\]

Input Data

\[
\begin{bmatrix}
x_0 & x_1 & \cdots & x_n
\end{bmatrix}
\]

RRAM Conductance

Voltage Input

\[
\begin{bmatrix}
V_0 \\
V_1 \\
\vdots \\
V_n
\end{bmatrix}
\]

Column Current

\[
\begin{bmatrix}
I_A &= V_0G_{A0} + V_1G_{A1} + V_2G_{A2} \\
I_B &= V_0G_{B0} + V_1G_{B1} + V_2G_{B2} \\
I_C &= V_0G_{C0} + V_1G_{C1} + V_2G_{C2}
\end{bmatrix}
\]

Chip micrograph: 65-nm mixed-signal 16x16 RRAM Crossbar
In-Memory Processing Using RRAM Crossbars

<table>
<thead>
<tr>
<th></th>
<th>Our Work</th>
<th>Mythic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technology</strong></td>
<td>65 nm</td>
<td>40 nm</td>
</tr>
<tr>
<td><strong>CLK Freq (MHz)</strong></td>
<td>40 - ADC</td>
<td>170</td>
</tr>
<tr>
<td><strong>Power (W)</strong></td>
<td>1.89</td>
<td>5</td>
</tr>
<tr>
<td><strong>TOPS/W</strong></td>
<td>5.90</td>
<td>4</td>
</tr>
<tr>
<td><strong>Quantization</strong></td>
<td>Reconfigurable</td>
<td>8-bit</td>
</tr>
</tbody>
</table>

Challenges

- 1.5-3x improvements are insufficient to justify startup costs for a new process
- Scaling challenges: advanced processes are not optimized for analog operation: thermal noise
  - ADCs still dominate latency, and ~50% energy consumption
  - Removal of ADCs requires algorithm-level optimization

spike-based processing?
**Spikes**: Biological neurons interact via single-bit spikes

**Sparsity**: Biological neurons spend most of their time at rest, setting most activations to zero at any given time

\[ [0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 5] \]

“7 at position 10; 5 at position 20”

**Static Suppression (aka Event-driven Processing)**: The sensory periphery only processes information when there is new information to process
The Leaky Integrate-and-Fire Neuron

- Bilipid thin-film membrane surrounded by ions: **capacitive**
- Ion-leakage/transfer: **resistive**
- The leaky integrate-and-fire neuron is just a 1st-order low-pass filter, i.e., an **RC circuit**

The neuroscientists stole from electrical engineers so it’s time to steal back from them
A Recurrent Representation

The leaky integrate-and-fire neuron is now compatible with all the tricks and hacks that go with training deep learning models.
How do we train models that change over time?

Error Backpropagation …Through Time

Unrolled Computational Graph

Spatial Credit Assignment
How does a loss assign ‘blame’ to a weight that is spatially far?

Temporal Credit Assignment
All states throughout history must be stored to run the BPTT algorithm i.e., the entire graph must be stored
Memory complexity: $O(nT)$
Gradient-based Learning with Spiking Neural Networks

Python package for gradient-based optimization of SNNs

real-time online learning

seamless integration with PyTorch

CUDA + IPU accelerated

neuromorphic HW compatible

github.com/jeshraghian/snntorch
Python package for gradient-based optimization of SNNs

real-time online learning

seamless integration with PyTorch

CUDA + IPU accelerated

neuromorphic HW compatible

github.com/jeshraghian/snntorch
RRAM-based SNN Processing

Communicating noise-prone analog signals down long lengths of wire: noise-prone

Communicating noise-robust digital signals down long lengths of wire: noise-tolerant

An RRAM Approach to Spiking Neuron Dynamics

- Charge-based integration on the bit-line capacitance
- State decay using heterogeneous RRAM time constants

Communicating noise-prone analog signals down long lengths of wire: noise-prone

Communicating noise-robust digital signals down long lengths of wire: noise-tolerant

An RRAM Approach to Spiking Neuron Dynamics

- Charge-based integration on the bit-line capacitance
- State decay using heterogeneous RRAM time constants
- Pre-charged sense amplifier for spiking dynamics (replace ADC)

RRAM-based SNN Processing

Classical pain points of IMC RRAM are largely addressed by spikes

- Variability
- ADC overhead
- Bandlimited at scale
- Endurance

Communicating noise-prone analog signals down long lengths of wire: noise-prone

Communicating noise-tolerant robust digital signals down long lengths of wire: noise-tolerant

RRAM-based SNN Processing

Classical pain points of IMC RRAM are largely addressed by spikes

Variability
ADC overhead
Bandlimited at scale
Endurance

Heterogeneity
Single-bit Spikes
Sparse data movement

Weight updates only occur at spike times

Communicating noise-prone analog signals down long lengths of wire: noise-prone

Communicating noise-robust digital signals down long lengths of wire: noise-tolerant

SkyWater 130 Neuromorphic Accelerator

Open-Source Neuromorphic IP
- Google-sponsored tape-outs using SkyWater 130nm process
- Deep learning success was in part due to open-source; let’s port this hype train over to silicon
- 2x successful tape-outs + 1x tape-in...

MPW6 BSNN Streaming Accelerator
- Online event streaming accelerator
- HD Event Cameras: ~10 MHz spike rate
- Die area: 0.09 mm²
- Average Power: 11 nW
- GPIO: 50 MHz peak, Core: 210 MHz

MPW8 SNN Accelerator
- 154 kB of dual-power 2 kB SRAM macros
- 9 mm² core, 21.89 mm² memory
- SRAM: 40 MHz, Core: 20 MHz
- Custom precision (up to 8-bits)

The next generation of neuromorphic engineers

31 first-time chip designers at UCSC taking the “Brain-Inspired Machine Learning” class defined their own design and submitted a chip.

It is currently being manufactured and will ship back in April 2023.

www.tinytapeout.com
Hodgkin-Huxley neuron in silicon
Designed in ECE 183/293
How many of you have converted your garage to a semiconductor manufacturing plant?

I didn’t think so.
How can we port dynamical, time-varying neural networks across different systems?

Solutions already exist for vanilla deep learning.

Nothing but pain exists for spiking neural networks.

... until now.
Neuromorphic Intermediate Representation

A Unified Instruction Set for Interoperable Brain-Inspired Computing

- Provides a common standard to enable continuous-time, dynamical neural networks to interface with each other
- Allows the broader community to tap into commercial & exotic hardware with their software of choice – e.g., Intel Loihi

>> model.to_nir()

We wrote 1000s of lines of code so you only have to write 1.

Neuromorphic Intermediate Representation

A Unified Instruction Set for Interoperable Brain-Inspired Computing

- Provides a common standard to enable continuous-time, dynamical neural networks to interface with each other
- Allows the broader community to tap into commercial & exotic hardware with their software of choice – e.g., Intel Loihi

```
>> model.to_nir()
```

We wrote 1000s of lines of code so you only have to write 1.


Server-scale neuromorphic HW: Loihi, Intel Labs

Edge neuromorphic hardware: SynSense, BrainChip
SNN Evaluation – Multimodal Data

### Accuracy

<table>
<thead>
<tr>
<th>SNN:</th>
<th>NVIDIA Jetson Nano</th>
<th>MorphIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNN: Loihi</td>
<td>96.0</td>
<td></td>
</tr>
<tr>
<td>ANN:</td>
<td>95.4</td>
<td></td>
</tr>
<tr>
<td>SNN: MorphIC</td>
<td>89.4</td>
<td></td>
</tr>
</tbody>
</table>

### Energy-Delay Product (uJ*s)

<table>
<thead>
<tr>
<th>SNN:</th>
<th>NVIDIA Jetson Nano</th>
<th>MorphIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNN: Loihi</td>
<td>8.6</td>
<td></td>
</tr>
<tr>
<td>ANN:</td>
<td>221.1</td>
<td></td>
</tr>
<tr>
<td>SNN: MorphIC</td>
<td>0.42</td>
<td></td>
</tr>
</tbody>
</table>

Open-Source Neuromorphic Computing

Jason K. Eshraghian
Assistant Professor, ECE, UC Santa Cruz

13 December 2023