MNSIM: A Simulation Platform for Memristor-based Neuromorphic Computing System

Lixue Xia¹, Boxun Li¹, Tianqi Tang¹, Peng Gu¹², Xiling Yin¹, Wenqin Huangfu¹, Pai-Yu Chen³, Shimeng Yu³, Yu Cao³, Yu Wang¹, Yuan Xie² and Huazhong Yang¹

¹ Dept. of E.E., Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing, China
² Department of Electrical and Computer Engineering, University of California at Santa Barbara, California, USA
³ School of Electrical, Computer and Energy Engineering, Arizona State University, Arizona, USA
Outline

• Background
  • Motivation
  • Related Work
  • Our Target

• Circuit Structure and Module Design
• Simulation Models
• Experimental Results
• Future Work
Energy Efficient Neuromorphic Accelerator

Training Data → Processing Data with Energy Efficient Neuromorphic Accelerators (FPGA, ASIC, Memristor/RRAM) → Visualization → Parameters → Big Data
CMOS Limitation

- Scale Up will not improve the energy efficiency
  - CPU: 1.5 GOPs/W, FPGA: 14.2 GOPs/W (our experiment, with same task)
  - DianNao [1]: 452 GOPs/W (peak)
  - Brain: 500,000 GOPs/W, still 1000X gap

![Diagram showing CMOS Scaling Down with ~10X, 6 orders of magnitude, 1,000,000X, Accelerator with ~100X, and Brain with 1000X.](image)
Memristor Neuromorphic

- Brain is **NOT** Boolean
- Emerging Memristor Devices, such as RRAM devices, provide a promising solution to realize better implementation of brain inspired circuits and systems

### Memristor

- **Merge Mem. & Compute**
- **Plasticity: Configure with Voltage/Current**
- **Non Volatile**

### High Density

- **O(n^2) → O(n^0)**

### Table

<table>
<thead>
<tr>
<th></th>
<th>Memristor</th>
<th>PCM</th>
<th>STT-RAM</th>
<th>DRAM</th>
<th>Flash</th>
<th>HD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chip area per bit (F^2)</td>
<td>4</td>
<td>8–16</td>
<td>14–64</td>
<td>6–8</td>
<td>4–8</td>
<td>n/a</td>
</tr>
<tr>
<td>Energy per bit (pJ)</td>
<td>0.1–3</td>
<td>2–100</td>
<td>0.1–1</td>
<td>2–4</td>
<td>10^8–10^14</td>
<td>10^8–10^7</td>
</tr>
<tr>
<td>Read time (ns)</td>
<td>&lt;10</td>
<td>20–70</td>
<td>10–30</td>
<td>10–50</td>
<td>25,000</td>
<td>5–8x10^6</td>
</tr>
<tr>
<td>Write time (ns)</td>
<td>20–30</td>
<td>50–500</td>
<td>13–95</td>
<td>10–50</td>
<td>200,000</td>
<td>5–8x10^6</td>
</tr>
<tr>
<td>Retention</td>
<td>&gt;10 years</td>
<td>&lt;10 years</td>
<td>Weeks</td>
<td>&lt;Second</td>
<td>^10 years</td>
<td>^10 years</td>
</tr>
<tr>
<td>Endurance (cycles)</td>
<td>~10^12</td>
<td>10^12–10^15</td>
<td>10^15</td>
<td>&gt;10^17</td>
<td>10^6–10^9</td>
<td>10^11–10^12</td>
</tr>
<tr>
<td>3D capability</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>n/a</td>
</tr>
</tbody>
</table>

[4] [5]

Fig. 1 Path Towards Ultra High Density NVM
(Sources: CM Workshop on Innovative Memory Technologies)
Memristor Crossbar

1 Memristor Cell

\[ \approx \]

1 $m$-bit Multiplier + 1 $m$-bit Adder + 1 $m$-bit Reg. (SRAM)

Non-Volatile Merge Mem. & Comp. >100X Efficiency Gains
Emerging Devices

- Emerging Devices are NOT just a prediction

**Intel 3D XPoint Unveiled—The Next Breakthrough in Memory Technology [2]**

**SAGE**: European Union funded **80 billion EUR** on high performance memory project “SAGE” leaded by Seagate Technology [3]
Gap between Application and Circuit

- Various NN Algorithms: ANN(DNN), CNN, and SNN
- Various Network Scale for Different Applications
- Crossbar Size Needs Optimization for Large Network
- Design of Peripheral Circuits Contains Multiple Parameters
- Various Technology Nodes for CMOS and Interconnect Lines

How to evaluate the computation accuracy, area, energy, and latency; and choose an optimal design?
Related Work

• NVSim [6] and NVMain [7] are powerful simulators for Non-Volatile Memories, but are not available for computing structure
  • The peripheral circuit structure is fixed for memory
  • They can not analyze computation accuracy corresponding trade-off relationships
Challenges of Simulation Platform

Target

Demand

Challenge

General and Flexible Tool

Early Simulation

Supporting Various Circuit Structure

High Simulation Speed

Structure Abstraction

Behavior-level Model
Outline

- Background
- Circuit Structure and Module Design
- Simulation Models
- Experimental Results
- Future Work
Hierarchical Structure for Large Network

- The hierarchical structure includes multiple levels: **Accelerator**, **Computing Bank**, and **Computing Unit**
- Adjusting detailed design parameters to explore the design space of implementation
Computing Unit

- A computing unit is the minimum structure that can individually process a write/read/compute task
- Consists of 4 main parts: memristor crossbar and address decoder, output peripheral circuit, input peripheral circuit, and control module
Difference with NVM

- **Computation-oriented modules**
  - Multilevel Cell: ADC/DAC
  - Signed Weight: Two crossbar and subtractor

- **All-open**: Computation-oriented Decoder
  - Decoders of Word Line (1T1R crossbar) and Source Line are different from memory oriented design because we need to simultaneously open all cells to process computation
  - MNSIM adds an “all-selected” signal and a NOR gate

![Diagram showing memory oriented and computation oriented decoders](image-url)
Computing Parallelism Degree

- Output peripheral circuits (especially ADCs) costs too much area and power [8]
  - User can choose sequential method that each read module works several times
  - MNSIM use computing parallelism degree to describe the amount of read circuits
  - This is a method to save area at the cost of more latency

\[
\begin{align*}
W_{1,1} & \quad W_{2,1} & \quad \cdots \\
W_{m,1} & \\
\end{align*}
\]
### Configurable Parameters

- **MNSIM** provides multiple configuration variables from different levels

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Level</th>
<th>Default Value</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>System</td>
<td>ANN</td>
<td>Algorithm Type</td>
</tr>
<tr>
<td>Application_Scale</td>
<td>System</td>
<td>-</td>
<td>Layers of Application</td>
</tr>
<tr>
<td>Weight_Bit</td>
<td>System</td>
<td>8</td>
<td>Weight Precision</td>
</tr>
<tr>
<td>Input_Bit</td>
<td>System</td>
<td>8</td>
<td>Signal Precision</td>
</tr>
<tr>
<td>Network_Scale</td>
<td>Bank</td>
<td>-</td>
<td>Scale of Each Layer</td>
</tr>
<tr>
<td>Weight_Polarity</td>
<td>Unit</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CMOS_Tech</td>
<td>Unit</td>
<td>90nm</td>
<td></td>
</tr>
<tr>
<td>Cell_Type</td>
<td>Unit</td>
<td>1T1R</td>
<td>1T1R/0T1R</td>
</tr>
<tr>
<td>Memristor_Model</td>
<td>Unit</td>
<td>RRAM</td>
<td></td>
</tr>
<tr>
<td>Interconnect_Tech</td>
<td>Unit</td>
<td>28nm</td>
<td></td>
</tr>
<tr>
<td>Crossbar_Size</td>
<td>Bank</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Parallelism_Degree</td>
<td>Unit</td>
<td>0</td>
<td>0 means All Parallel</td>
</tr>
<tr>
<td>Resistance_Range</td>
<td>Unit</td>
<td>[500 500k]</td>
<td>Min/Max Memristor Resistance</td>
</tr>
</tbody>
</table>

- If the user doesn’t determine all the configurations, MNSIM can explore the design space and give the optimal results with corresponding configurations.
Computing Bank

- A computing bank processes the function of a network layer, containing the function of synapse and neuron.

- The neuron function is non-linear (e.g. sigmoid), which must be performed after merging the results of multiple units.

\[
sigmoid\left(\sum_{n=1}^{N} W_{1,n} V_{i,n}\right) \neq \sum_{n=1}^{N} sigmoid(W_{1,n} V_{i,n})
\]

- MNSIM also provide a reconfigurable unit design: each unit contains an adder and a neuron function.
Outline

• Background
• Circuit Structure and Module Design
• Simulation Models
• Experimental Results
• Future Work
Estimation Model of Crossbar and Decoder

- **Memristor Crossbar**
  - The estimation model of area and latency is the same with memory-oriented design
  - The energy consumption of computation is larger than reading a single cell in memory
  - As an behavior-level simulator, MNSIM uses the average case to estimate the energy consumption of each cell
Estimation Model of Crossbar and Decoder

- **Decoder**
  - We design the reference transistor-level circuit of computation oriented decoder
  - MNSIM integrates both technology models and SPICE results to estimate the performance of decoder
To verify the area estimation model of crossbar and decoder, we design the layout of crossbar and decoder of different crossbar size.

- The figure shows a 32x32 1T1R crossbar and decoder supporting all-selected operation.
Estimation Model of Other Peripheral Modules

- **MNSIM** provides a reference transistor-level design of each peripheral module for scaling down
  - The technology parameters come from CACTI, NVSim, and PTM (Predictive Technology Model)
  - For area consumption, we use squared scaling method to estimate of other technology node that we have not integrated into MNSIM

- **Precision (Bit-level)**
  - The relationship between precision and performance (area, power, latency) is complex for ADC/DAC
  - MNSIM provides a reference ADC/DAC design for each precision
  - Users can also integrate the model of their own circuit
Estimation Model of Accuracy (Error Rate)

- MNSIM uses a *theoretic approximate model* to estimate accuracy, and the detailed parameters are verified by SPICE

- For a $M \times N$ memristor crossbar, the output voltage of each column is:
  \[
  V_o = \frac{R_S}{R_{parallel} + R_S} V_i
  \]

- If we take the crossbar interconnect line's resistance $r$ into account, the $R_{parallel}$ of the $n$th column from input is:
  \[
  \frac{1}{R_{parallel}} = \sum_{m=1}^{M} \frac{1}{R_{m,n} + mr + nr}
  \]

- Considering that $r$ is much less than $R$, the difference between denominators can be ignored. So in the worst case:
  \[
  \frac{1}{R_{parallel}} = \sum_{m=1}^{M} \frac{1}{R_{min} + Mr + Nr} = \frac{M}{R_{min} + Mr + Nr}
  \]
Estimation Model of Accuracy (Error Rate)

- So the different between actual output voltage and ideal output voltage is:

$$|\delta V_o| = |V_{o,\text{ideal}} - V_{o,\text{actual}}| = \left| \frac{R_S}{\frac{R_{\text{actual}}}{M} + m + N} + R_S - \frac{R_S}{\frac{R_{\text{ideal}}}{M} + R_S} \right| V_i$$

- where $R_{\text{actual}}$ is the practical resistance of cells. It is different from ideal resistance because of the non-linear V-I characteristic of memristor.

- The Error Rate of output voltage is:

$$\frac{|\delta V_o|}{V_{o,\text{ideal}}} = \frac{[(M + N)r + R_{\text{actual}} - R_{\text{ideal}}]R_S}{[(M + N)r + R_{\text{actual}} + R_SM](R_{\text{ideal}} + R_SM)}$$
We use the simulation results of SPICE as the real value to fit the parameters of our model.

- The technology node of interconnect line influence the resistance of line between neighbor cells.

\[
\text{Error Rate} = \frac{aNr + b}{cN^2r + dNr + eN + f}
\]
Outline

- Background
- Circuit Structure and Module Design
- Simulation Models
- Experimental Results
- Future Work
Speed-up compared with SPICE

- MNSIM can gain more than $7000 \times$ speed-up compared with SPICE
  - We use the simulation of a Computing Unit with $N \times N$ crossbar to evaluate the simulation time
  - The speed-up can be further large when simulating the whole neuromorphic system with neurons

<table>
<thead>
<tr>
<th>Crossbar Size</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPICE(s)</td>
<td>5.35</td>
<td>13.76</td>
<td>41.62</td>
<td>169.12</td>
<td>678.2</td>
</tr>
<tr>
<td>MNSIM(s)</td>
<td>0.0007</td>
<td>0.0011</td>
<td>0.0030</td>
<td>0.0192</td>
<td>0.0348</td>
</tr>
<tr>
<td>Speed-Up</td>
<td>$7642 \times$</td>
<td>$12509 \times$</td>
<td>$13873 \times$</td>
<td>$8088 \times$</td>
<td>$19489 \times$</td>
</tr>
</tbody>
</table>
Full Design Space Exploration

- Based on the high simulation speed, MNSIM can further explore the design space and obtain optimal designs
  - We use a $2048 \times 1024$ network as a case study to show the results of design space exploration
  - Each column of the table shows the performance factors and design details about an optimal implementation aimed at a specific optimization target

<table>
<thead>
<tr>
<th>Optimization Target</th>
<th>Area ($mm^2$)</th>
<th>Energy</th>
<th>Latency</th>
<th>Computation Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area ($mm^2$)</td>
<td>12.183</td>
<td>20.730</td>
<td>29.345</td>
<td>117.086</td>
</tr>
<tr>
<td>Energy per Operation ($\mu J$)</td>
<td>35.90</td>
<td>3.1923</td>
<td>3.748</td>
<td>10.35</td>
</tr>
<tr>
<td>Latency ($\mu s$)</td>
<td>43.43</td>
<td>0.5153</td>
<td>0.3470</td>
<td>10.35</td>
</tr>
<tr>
<td>Error Rate of Output (%)</td>
<td>17.98</td>
<td>17.98</td>
<td>17.98</td>
<td>1.09</td>
</tr>
<tr>
<td>Power ($W$)</td>
<td>0.8266</td>
<td>6.195</td>
<td>10.80</td>
<td>29.66</td>
</tr>
<tr>
<td>Crossbar Size</td>
<td>256</td>
<td>256</td>
<td>256</td>
<td>64</td>
</tr>
<tr>
<td>Line Tech Node</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>45</td>
</tr>
<tr>
<td>Parallelism Degree</td>
<td>1</td>
<td>128</td>
<td>256</td>
<td>64</td>
</tr>
</tbody>
</table>
MNSIM can help users to analyze trade-off between area, energy, and computation accuracy among different crossbar sizes.

- We use the 45nm technology as an example.
- The result shows that only when the crossbar size is larger than 64 can we get computing accuracy gain at the cost of area and power in this application.
- When the crossbar is too small, the cell can be influenced by the non-linear V-I characteristic of memristor.

<table>
<thead>
<tr>
<th>Crossbar Size</th>
<th>256</th>
<th>128</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error Rate(%)</td>
<td>7.71</td>
<td>2.07</td>
<td>1.09</td>
<td>1.46</td>
<td>2.38</td>
<td>3.50</td>
</tr>
<tr>
<td>Area(mm²)</td>
<td>29.34</td>
<td>58.59</td>
<td>117.11</td>
<td>234.10</td>
<td>468.32</td>
<td>936.81</td>
</tr>
<tr>
<td>Energy(μJ)</td>
<td>3.74</td>
<td>5.94</td>
<td>10.35</td>
<td>19.21</td>
<td>37.09</td>
<td>73.38</td>
</tr>
</tbody>
</table>
MNSIM can also provide the trade-off relationship between latency and area among different computation parallelism degrees.

Users can choose the reasonable design from their own preferences.
Outline

• Background
• Circuit Structure and Module Design
• Simulation Models
• Experimental Results
• Future Work
Our Service

- Online website service with multiple individual parts
  - Simulator: C++ based, compiled into an executable file on UNIX
  - Server: Python based, Flask web framework, supporting the website and executing the simulator
  - Input File (Configuration): Text file, generated by server
  - Output File (Result): Generated by Simulator


Thank you!

Q&A
For multi-layer network, the errors are accumulated through layers.

Given the quantization interval $V_{\text{interval}}$, the quantization boundaries are $(0.5V_{\text{interval}}, 1.5V_{\text{interval}}, \ldots, (k - 1 - 0.5) V_{\text{interval}})$.

In the worst case, the ideal computing result signal is just around the largest quantization boundary $(k - 1.5) V_{\text{interval}}$ and needs to be recognized as the maximum value $k - 1$.

$$\text{MaxErrorRate} = \frac{|(k-1.5)\varepsilon+0.5|}{k-1}$$

For average situation, the digital deviation of a specific quantization level $i$ can be represented by $[i\varepsilon + 0.5]$, so:

$$\text{AvgErrorRate} = \frac{\sum_{i=0}^{k-1}[i\varepsilon+0.5]}{k(k-1)}$$

Finally, we can estimate the error rate of next layer by:

$$(1 + \delta_1)(1 + \varepsilon_2)V_{\text{ideal}} \leq V_{\text{actual}} \leq (1 + \delta_1)(1 + \varepsilon_2)V_{\text{ideal}}$$