Simulation and Visualization of Custom Neuromorphic Hardware using NeMo
NeMo:

- Uses Parallel Discrete Event Simulation (PDES)
- Built on top of ROSS (Rensselaer Optimistic Simulation System)
- Able to simulate existing and novel neuromorphic hardware models
- Validated against TrueNorth
NeMo and NeMo 2

NeMo 1

<table>
<thead>
<tr>
<th>Active Message Chain</th>
<th>Inactive Chain</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Axons</strong></td>
<td></td>
</tr>
<tr>
<td>AX₀</td>
<td></td>
</tr>
<tr>
<td>S₀,₀</td>
<td>S₁,₀</td>
</tr>
<tr>
<td>S₀,₁</td>
<td>S₁,₁</td>
</tr>
<tr>
<td>S₀,₂</td>
<td>S₁,₂</td>
</tr>
<tr>
<td><strong>Synapses</strong></td>
<td></td>
</tr>
<tr>
<td>N₀</td>
<td>N₁</td>
</tr>
<tr>
<td>N₁</td>
<td>N₂</td>
</tr>
</tbody>
</table>

NeMo 2

- **Neuron₁**
- **Neuron₂**
- **Neuronₙ**

**Virtual Synaptic Grid**

- **SuperSynapse**
- **Axon₁**
- **Axon₂**
- **Axonₙ**
IO Details

- Embedded Lua interpreter
- Network configuration files are valid Lua source
  - Allows very flexible and expressive configuration
  - Potential dynamic behaviors
  - Fine grained error checking
IO Details

- Embedded Lua interpreter

```lua
-- NeMo Configuration
cores = 256
neuronsPerCore = 256
neuronWeights = 4

-- Neuron Definitions
neurons = {
  TN_2_0 = {type = "TN", coreID=2, localID =0, alpha=1, beta=2,...},
  TN_2_1 = {type = "TN", coreID=2, localID =1, alpha=3, beta=0,...},
  TN_2_2 = {type = "TN", coreID=2, localID =2, alpha=4, beta=0,...},
  TN_2_3 = {type = "TN", coreID=2, localID =3, alpha=4, beta=0,...}
}
```
NeMo Instrumentation

- New ROSS / CODES feature
  - Low model overhead instrumentation
- Implemented in NeMo:
  - Event Tracing
  - Tracks Neuron Activity:
    - Spikes & Integration Per Tick
    - Sent Messages
MNIST Letter Identification - Core Activity
MNIST Letter Identification - Core Activity
MNIST Letter Identification - Core Activity
MNIST Letter Identification - Core Activity
MNIST Letter Identification - Core Activity
Performance

- Excellent performance when simulating large systems and smaller systems
- Able to simulate many types of hardware
- Small enough memory footprint to run simple models on desktop hardware
Instrumentation

- Adds ability to trace and debug models
- Currently provides way to view core activity over time
- Potential applications include
  - Power use estimates
  - Core-wise debugging
Future Work

- Integration with supercomputing simulation framework
- Further performance and usability improvements
Open Source Neuromorphic Hardware Modeling
https://github.com/markplagge/NeMo
plaggm@rpi.edu
Input File Def / Status

- Still WIP - Currently 2 Files:
  - CSV for input spikes
    - (time, destCore, destAxon)
  - Lua for network config
    - List of defined variables
    - Dictionary (key/value pair) per neuron
- Subject to change – Final def. will be uploaded to NeMo GitHub Wiki
Weak Scaling
Wall-Clock Time
Performance Results

- Weak Scaling on the Blue Gene/Q
- Simulation ran for 1,000 Ticks
  - 1 second real-time

NeMo Run

- NeMo–ES
- NeMo–SS 512 Neurons
- NeMo–SS Real-Time 512 Neurons
- NeMo–SS 256 Neurons
- NeMo–SS Real-Time 256 Neurons
COMPASS Comparison

- COMPASS results were reported as neurosynaptic events per second
- NeMo’s results were total events per second
  - Includes all axon, synapse, and neuron events
- To compare, we calculated equivalent COMPASS events per second
COMPASS Comparison

- For every neuron spike in NeMo-ES
  - 256 events generated with one axon event
- Assumed that COMPASS has 50/50 remote to local events

\[ e_{\text{total}} = s_{\text{reported}} \times 2 \times f \]

\[ e_{\text{second}} = \frac{e_{\text{total}}}{t} \]

\[ e_{\text{second/rack}} = \frac{e_{\text{second}}}{16} \]
Validation

- Validated NeMo against two of Izhikevich’s *Biologically Relevant Behaviors* as seen in Cassidy, et al. 2013