The FACETS Demonstrator

M. A. Petrovici
University of Heidelberg

Code Jam 3
Freiburg, 09.10.2009
The FACETS Hardware

Stage 1: The Spikey Chip

- 384 neurons
- $10^5$ synapses

Stage 2: Waferscale integration

- 16 $\cdot 10^4$ neurons
- 4 $\cdot 10^7$ synapses

www.facets-project.org
The FACETS Demonstrator is...

... a collection of cortical neural network models emulated with the FACETS wafer-scale hardware *

* as long as not yet available: dedicated executable system specification

...exhibiting functionality that can be demonstrated

...written in PyNN

...computable with established software simulators
(for verification, performance evaluation etc.)
Why bother?

Reviewers & community are interested in:

**Ability of the wafer-scale hardware system to serve as a flexible research tool in modeling neuroscience:**

- despite hardware constraints, relevant experiments possible
- crucial interplay between hardware and software
- massive complexity of hardware configuration space can be handled by intelligent software
- operability by non-hardware experts

**Ability of FACETS to fruitfully cooperate:**

- Demonstrator is actively developed by UHEI, TUD, KTH, UNIC, ALUF, INCM
- hardware neuron and synapse models: result of discussions. FACETS has participated in their original development, e.g. EPFL.
Why not wait for the hardware?

Testing and evaluation of all involved software layers

Virtual hardware allows to

- test software before hardware is available
- test without possible hardware-specific problems
- provide a preliminary PyNN module for off-line testing of experiments

Verification of possible hardware changes

- e.g. optionally insert detailed HICANN model

⇒ Important test-bench for preparation work
Simulating the emulator
Simulating the emulator

- Instantiates and connects submodules
- Gets configuration via access to Graph Model
- Configures: DNC_IF, Layer1net, Anncore

HICANN

DNC_IF (DNC Interface)

Layer1net

Anncore

- Config
- Port (RTL)
- Interface (TLM)
- Channel

to DNC

to neighbour Layer1nets

to neighbour Anncores
Experiments and demonstrations (to-do list…)

Evaluation of the involved software layers

- show full support of low-level and object-oriented PyNN API

- quantify performance of the mapping and routing algorithms
  - Execution time, memory consumption etc as function of network size, parameter heterogeneity, applied optimization strategies etc.
  - Neuron and synapse loss as function of …
  - Histogram of layer1 hops as function of …

Demonstrate correct functionality of the virtual hardware

- Unit tests
Experiments and demonstrations (to-do list…)

Model performance: Influence of hardware-specific imperfections or inhomogeneities

- neuron and synapse parameters (e.g. 4-bit weights, limited precision), signal transmission delays, …
  - comparison software vs. (virtual) hardware
  - incorporate hardware-specific distortions into software model

- automatically extracted: “post-mapping” PyNN scripts that reflect mapping distortions
The Demonstrator models (so far)

- A layer 2/3 attractor memory
  (by KTH, Krishnamurty / Lansner)

- A synfire chain model
  (by INCM and ALUF, Kremkow / Aertsen / Masson)

- A model of self-sustaining cortical AI states
  (by UNIC, Davison / Destexhe)

- Upcoming: Two-layer model by UNIC

All written in PyNN, all scalable, basic versions can be mapped to hardware without synapse loss
Attractor memory schematic
Attractor memory (NEST): 2 attractors
Attractor memory (NEST): pyramidal spike rate
Attractor memory (NEST): RSNP spike rate
Attractor memory (NEST): average membrane potential

attractor 1
Attractor memory (NEST): average membrane potential

attractor 2
Attractor memory (NEST): phase diagram
Attractor memory (NEST): 5 attractors
Attractor memory (NEST): “advanced” phase diagram
Attractor memory (NEST): 0% synaptic loss
Attractor memory (NEST): 20% synaptic loss
Attractor memory (NEST): 40% synaptic loss
Attractor memory (NEST): phase diagrams

0% loss

10% loss

25% loss

40% loss
Attractor memory (NEST): firing rates

![Graph showing average firing rate vs relative synapse loss]

- **ON state**
- **OFF state**

The graph illustrates the average firing rate in different states of synapse loss.
Attractor memory (NEST): attractor dwell times

![Graph showing attractor dwell times vs. relative synaptic loss. The graph indicates a decrease in dwell time as the relative synaptic loss increases.](image-url)
Attractor memory (NEST): threshold voltage

![Graph showing the relationship between threshold voltage and relative synaptic loss. The x-axis represents the relative synaptic loss, ranging from 0.0 to 0.4, while the y-axis represents the threshold voltage in mV, ranging from -64.4 to -63.0. The graph displays a trend of increasing threshold voltage with increasing relative synaptic loss.](image-url)
Attractor memory (NEST): pattern completion
Synfire chain schematic
Synfire chain simulations (NEST)
Technical infrastructure and visibility

- Wiki documentation (internal)
- Full code available in SVN repositories
- Experimental results in BSCW repository
- Tele-conferences
- Publications