BrainScaleS Workshop

4th HBP School

Korbinian Schreiber & Sebastian Billaudelle

June 14, 2017

Kirchhoff-Institute for Physics, Heidelberg University

Introduction

Analog Neuromorphic Hardware

1

observations

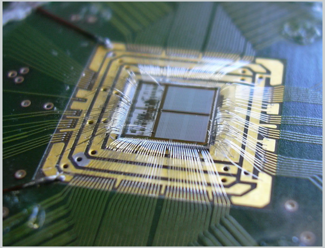
2

mathematical model

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_\mathrm{L}(V - E_\mathrm{L}) + I_\mathrm{syn}(t)$$

3

hardware realization



Roadmap

2004

Spikey

- single chip system
- 384 LIF neurons

2010

HICANN

- 180 nm CMOS
- 512 AdEx neurons

2015

20 Wafer System

- 4 million neurons
- 0.9 billion synapses

2017

HICANN DLS

- 65 nm CMOS
- PPU: integrated processing unit for advanced plasticity

2022

500 Wafer System

- 500 million neurons
- 130 billion synapses

Working with Spikey

PyNN API documentation

`https://neuralensemble.org/docs/PyNN/0.7/api/api-0.7.html`

Look out for:

- `pynn.Population`
- `pynn.Projection`
- `pynn.*Connector`

Creating (groups of) neurons

Create *populations* of neurons:

```
params = {
    "v_thresh": -60.0
    }
neurons = pynn.Population(42, pynn.IF_facets_hardware1, cellparams=params)
```

Get a list of default neuron parameters:

```
print pynn.IF_facets_hardware1.default_parameters
```

Generating stimuli

Create a stimulus from a spike train:

```
spike_train = np.arange(10.0, 101.0, 10.0)
stimulus = pynn.Population(1, pynn.SpikeSourceArray, {"spike_times": spike_train})
```

There is also a Poisson spike source:

```
poisson_params = {
    "start": 10.0,
    "duration": 100.0,
    "rate": 5.0
    }
stimulus = pynn.Population(1, pynn.SpikeSourcePoisson, poisson_params)
```

Synaptic connections

Connect all pre-synaptic to all post-synaptic neurons:

```
weight = 15 * pynn.getMinExcWeight()
conn = pynn.AllToAllConnector(weights=weight)
proj = pynn.Projection(pre, post, conn)
```

Specify connections in a list:

```
conn = pynn.FromListConnector([(7, 13, w, d), (42, 0, w, d)])
```

Other connectors (look at specification):

`FixedNumberPreConnector` `FixedNumberPostConnector` `FixedProbabilityConnector`

Recording observables

Spike times:

```
neurons.record()
...
spikes = neurons.getSpikes()
```

Analog membrane traces:

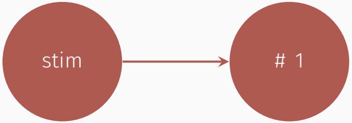
```
pynn.record_v(neurons[0], "")
```

- only *one* analog-to-digital converter (ADC)

→ one can record a single neuron at a time

Tasks

Task 1: a single neuron



- create a spike source
- create a single LIF neuron
- connect these two populations with maximum weight
- record spikes and the membrane trace of the stimulated neuron

1. vary the synaptic weight and observe the membrane trace
2. play around with the inter-spike interval of the stimulating spike train
3. observe how the PSPs stack up and eventually cause the neuron to fire

Task 2: passing spikes

- extend the network by adding another neuron
- record and plot the spikes of both neurons

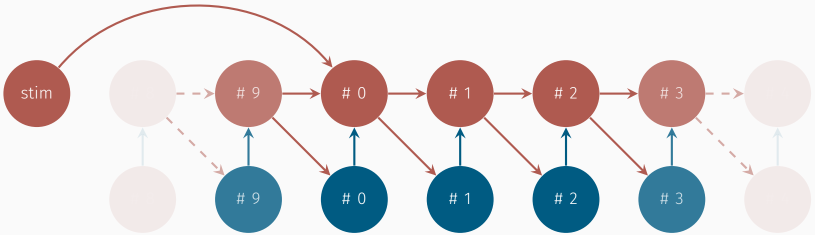
1. think about different possibilities of creating and connecting the neurons
2. check that the stimulation is passed to the second neuron

Task 2: passing spikes

- extend the network by adding another neuron
- record and plot the spikes of both neurons

1. think about different possibilities of creating and connecting the neurons
2. check that the stimulation is passed to the second neuron

Task 3: a closed synfire chain



- create ten excitatory and ten inhibitory populations of neurons and connect them as depicted
- create a transient stimulus to the first neuron
- record and plot the spikes of the excitatory neurons

1. What happens if you disconnect the inhibitory neurons?