BrainScaleS Workshop

Visit by the SP10 TUM Roboy team

Sebastian Billaudelle & Korbinian Schreiber

September 25, 2017

Kirchhoff-Institute for Physics, Heidelberg University

Introduction

Information

WIFI:
ESSID: DemoBrainScaleS
password: BrainScaleS

Jupyter notebooks:
`192.168.124.1:8000`

GitHub:
`https://goo.gl/MzYqXX`

Spikey school:
`https://goo.gl/D6i3nA`

System overview

Host ⟷ FPGA ⟷ Spikey

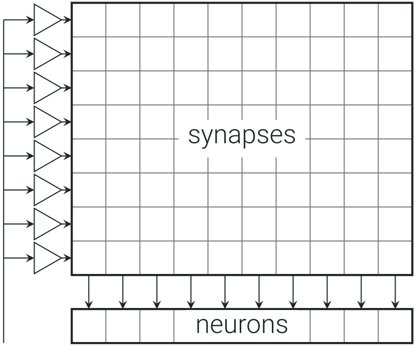
Field-programmable gate array:

- reconfigurable logic gates
- experiment control and communication

Spikey:

- 384 neurons, 384×256 synapses
- speedup of 10^4

The analog core



Synapses:

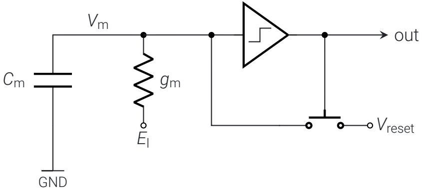
- 4 bit weights (0…15)
- STDP and STP

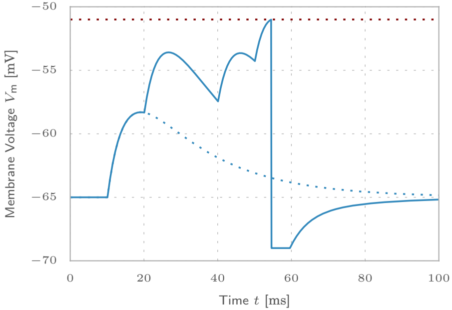
Neurons:

- Leaky-integrate-and-fire model (LIF)
- analog parameters can be configured freely

Leaky-integrate-and-fire neurons

$$C_\mathrm{m}\frac{\mathrm{d}\,V_\mathrm{m}}{\mathrm{d}\,t} = -g_\mathrm{l}(V_\mathrm{m} - E_\mathrm{l}) + I_\mathrm{syn} + I_\mathrm{ext}$$





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.minExcWeight()
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

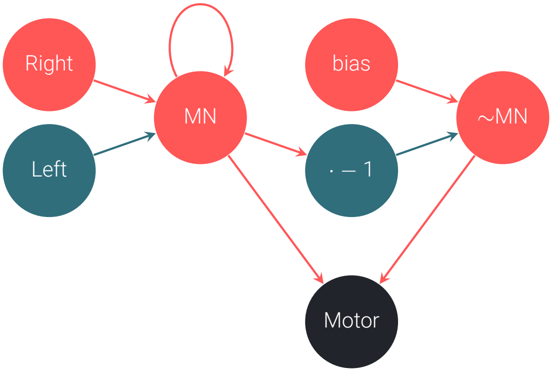
1. vary the synaptic weight and observe the membrane trace
2. play around with the inter-spike interval of the stimulating spike train
3. observe how the PSPs stack up and eventually cause the neuron to fire

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 3: bouncing vehicle



- transient right/left stimulus sets/resets the state of the motor neuron (MN) population
- MN and ∼MN determine the motor direction