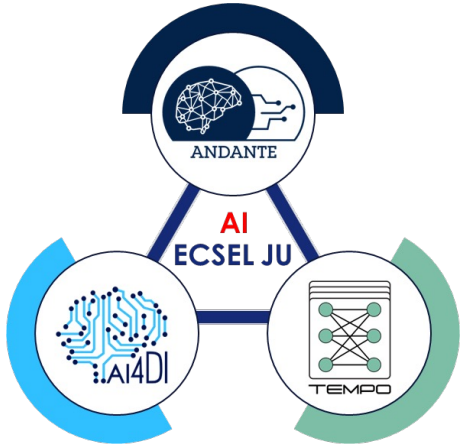
International Workshop on Embedded Artificial Intelligence Devices, Systems, and Industrial Applications (EAI)

MILAN
19-22 SEPTEMBER 2022
ESSDERC
ESSCIRC

Milan, Italy 19 September 2022

International Workshop on Embedded Artificial Intelligence Devices, Systems, and Industrial Applications (EAI)



Sub-mW audio processing with Xylo SNN

Hannah Bos, **Dylan Muir** | SynSense

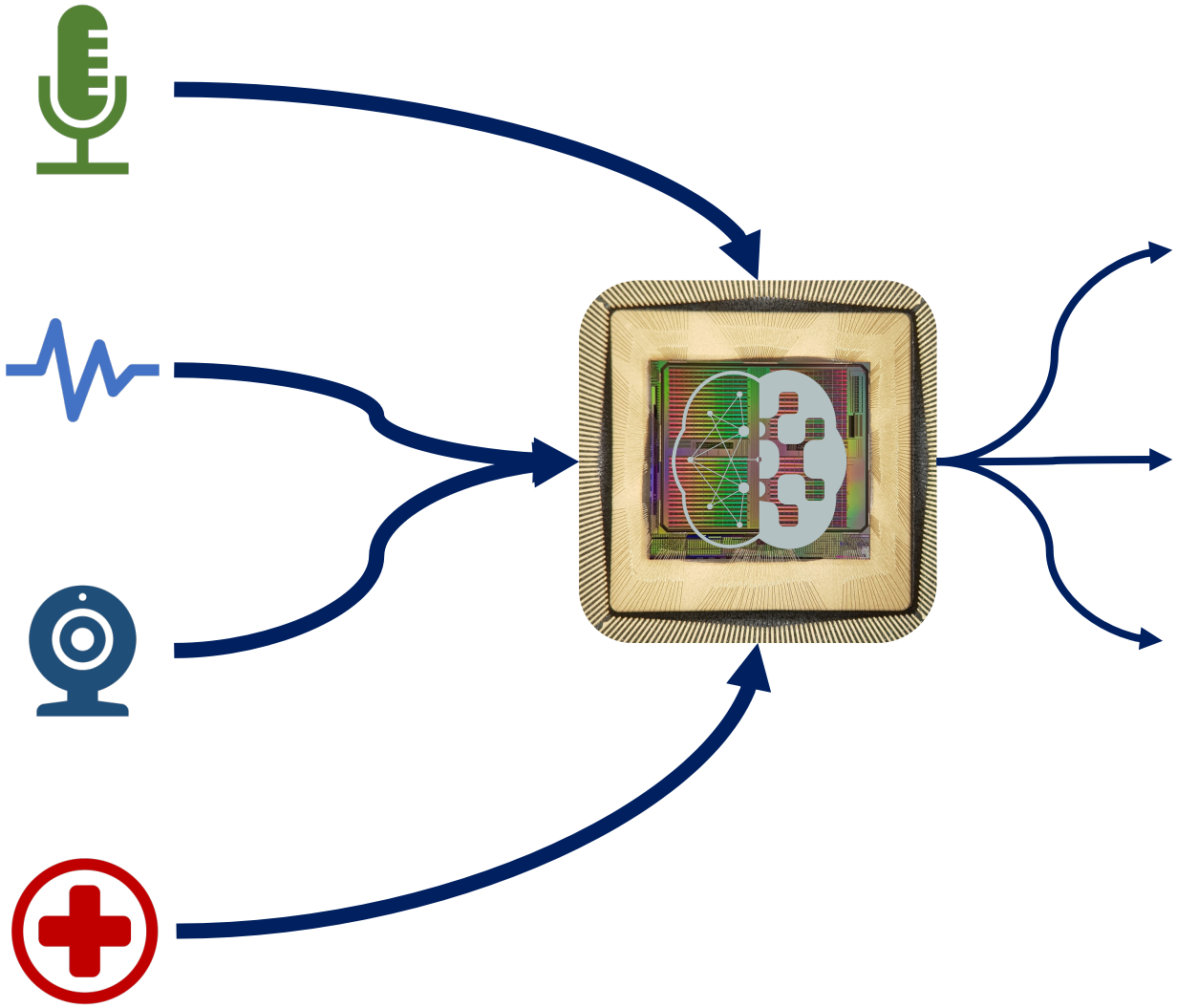
MILAN 19-22 SEPTEMBER 2022 ESSDERC ESSCIRC

19 September 2022 Milan, Italy

Copyright © 2022

Neuromorphic Smart Sensors

SynSense



- Highly informative output / low bandwidth output
- Smart condition detection
- Smart wake-up
- Continuous monitoring
- Low latency → <200 ms
- Low power → <5 mW

Audio processing applications

Ultra-low power audio processing

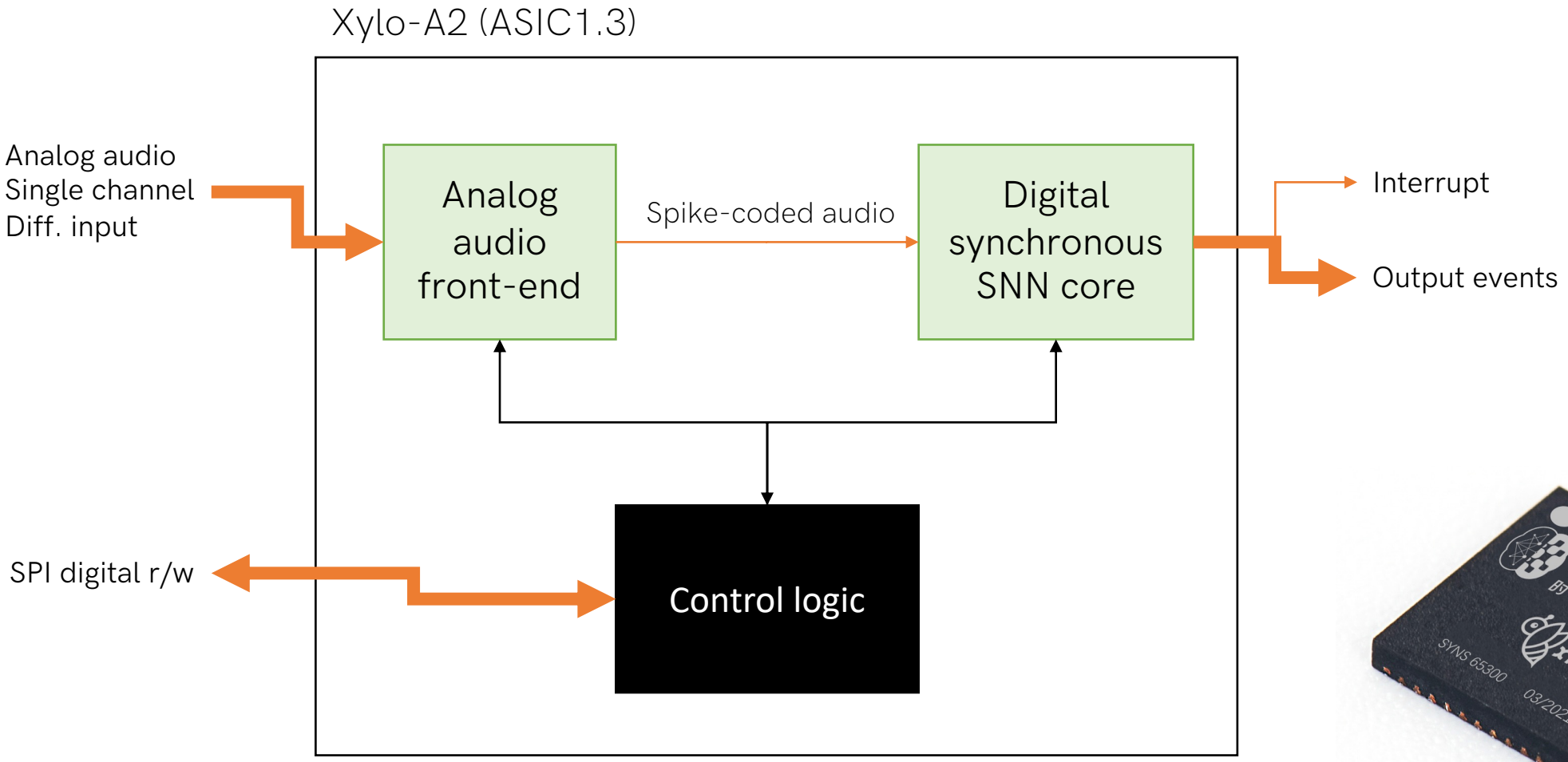
Keyword spotting / event detection

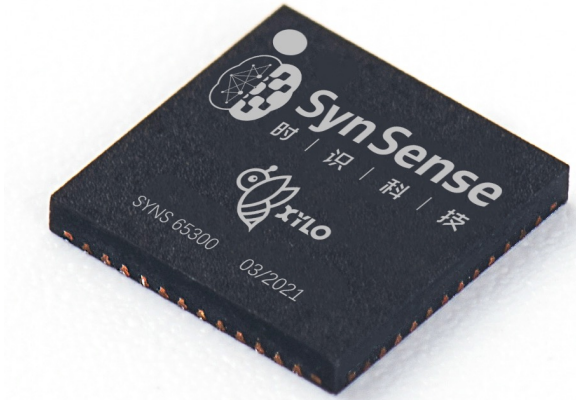
Acoustic scene recognition

Voice activity detection

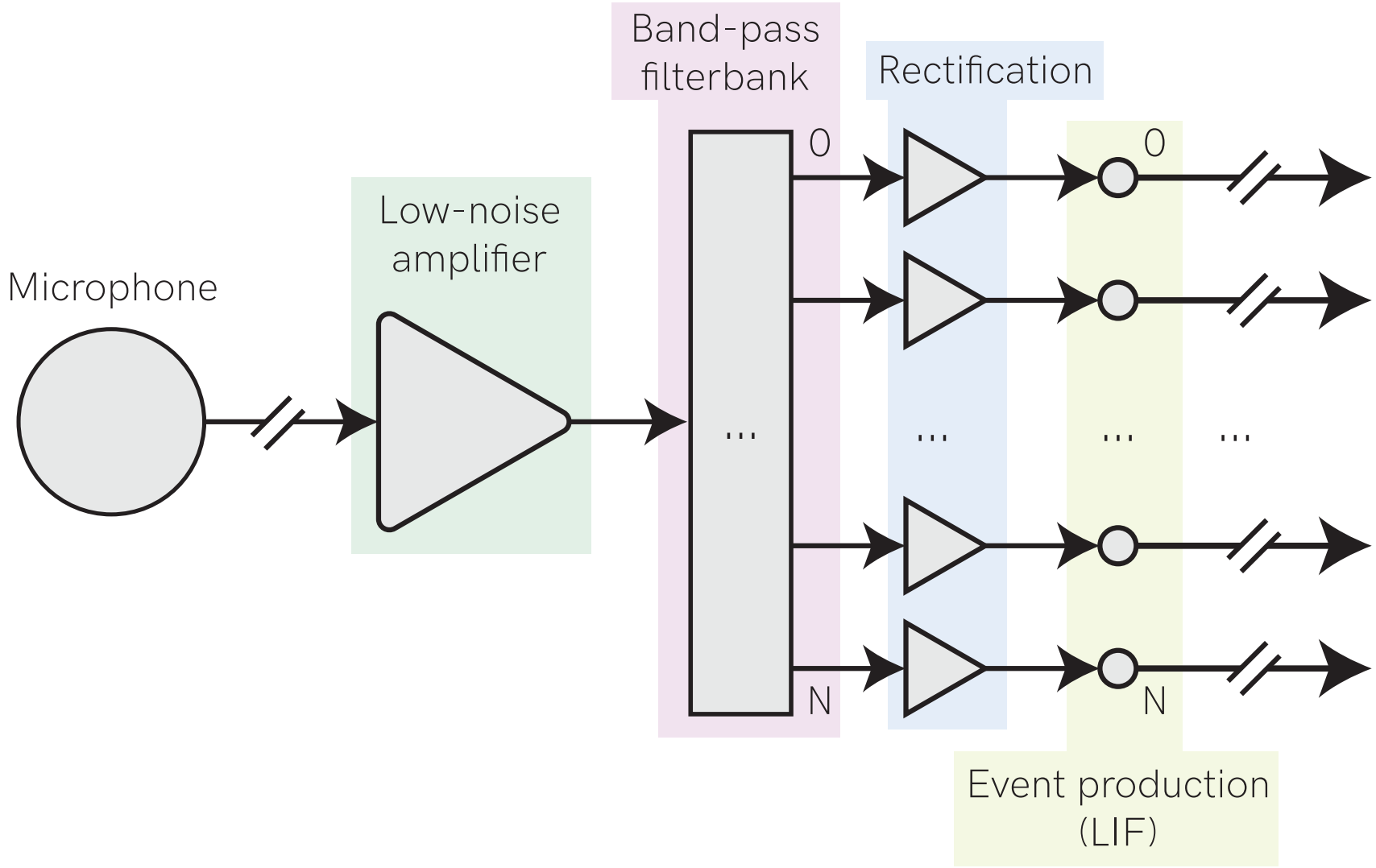
Industrial monitoring

Xylo audio processing — ASIC1.3





Audio input encoding



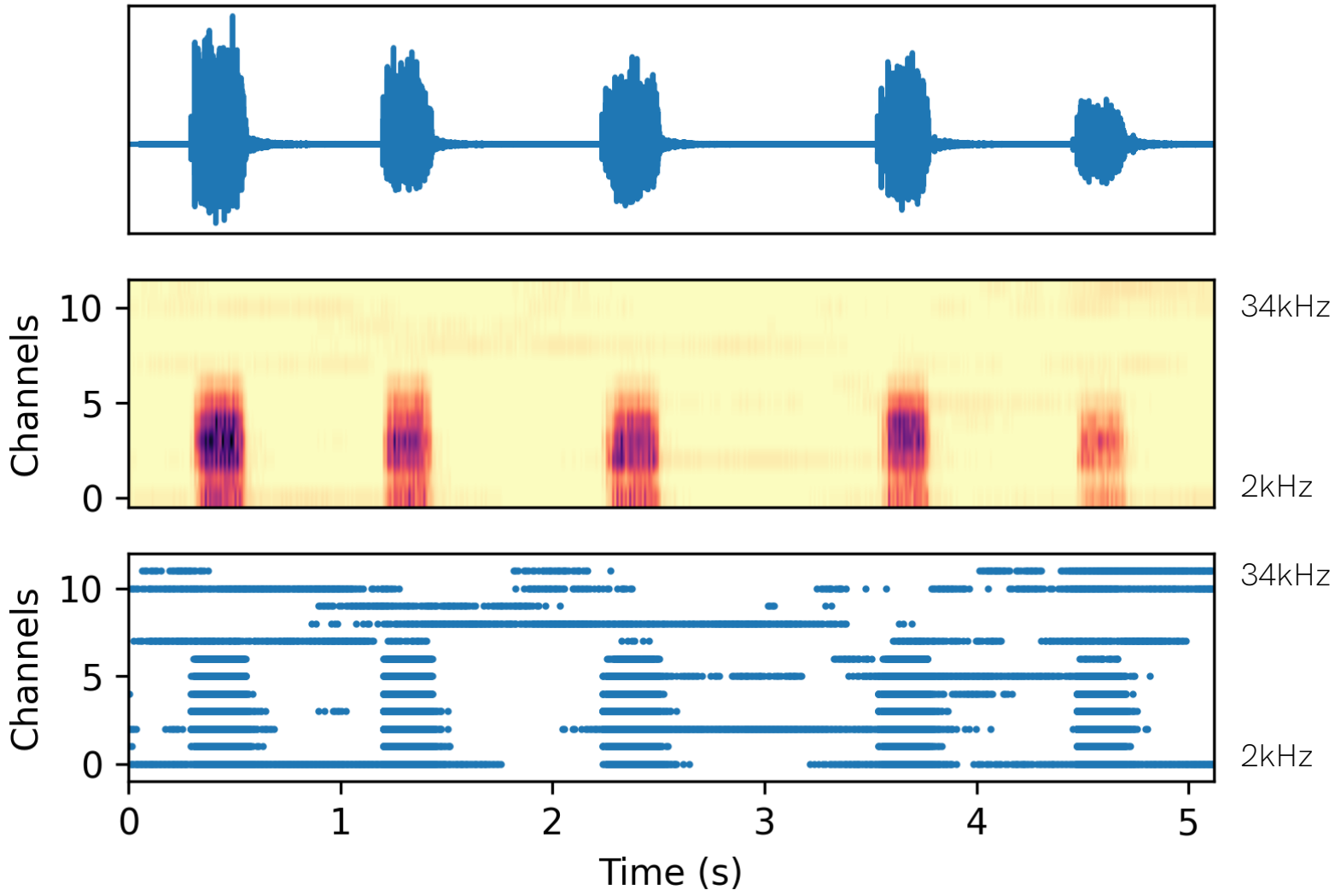
Audio input encoding

SynSense

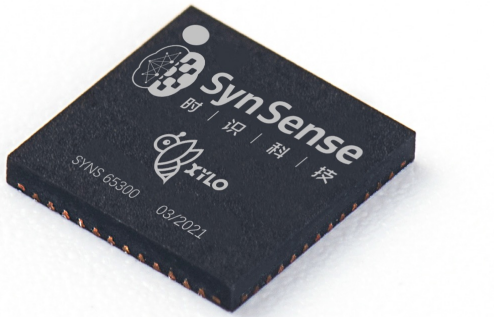
Raw audio

Spectrogram

Events



Xylo digital SNN architecture
Logical network architecture



LIF hidden neurons

W_{rec}

Recurrent weights

N_{hid}

LIF output neurons

N_{in}

Spiking Input channels

W_{in}

N_{hid}

N_{hid}

W_{out}

N_{out}

Readout weights

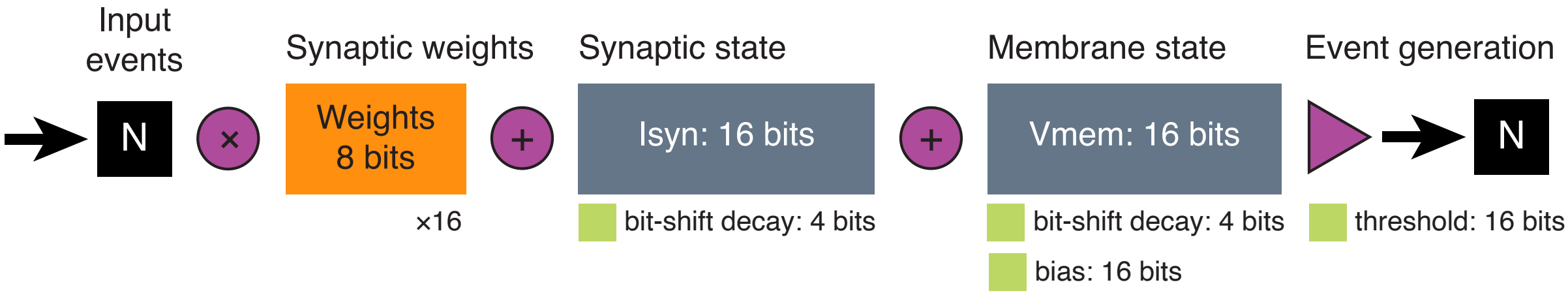
$N_{in} \leq 16$

Input weights

$N_{hid} \leq 1000$

$N_{out} \leq 8$

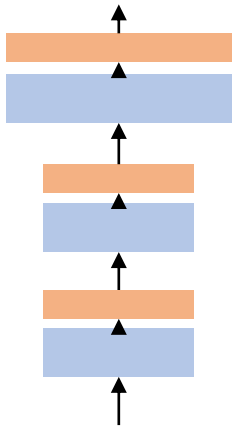
Xylo digital SNN architecture
Digital LIF Neuron



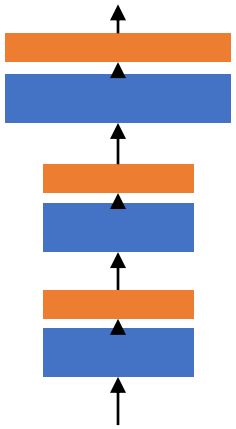
Application deployment pipeline

rockpool.ai

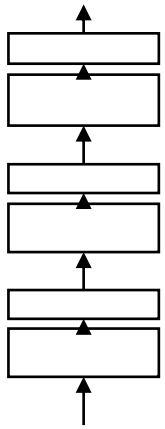
Simulatable network



Trained network



Computational graph



Mapped parameters

W_{in}
W_{rec}
W_{out}
τ_{mem}
τ_{syn}
Φ
...

Quantised parameters

$\widehat{W}_{in}$
$\widehat{W}_{rec}$
$\widehat{W}_{out}$
$\hat{\tau}_{mem}$
$\hat{\tau}_{syn}$
$\hat{\Phi}$
...

Bitstream HW configuration

```
0010010101
0001010100
1010010010
1001010101
0010101010
1110001110
0001110101
```

Define network → Train → Convert to graph → Map to HW architecture → Quantise parameters → Deploy to HW

```
rockpool.nn
 ↳ Linear...
 ↳ LIF...

rockpool.combinators
 ↳ Sequential
```

```
torch
jax
optax
```

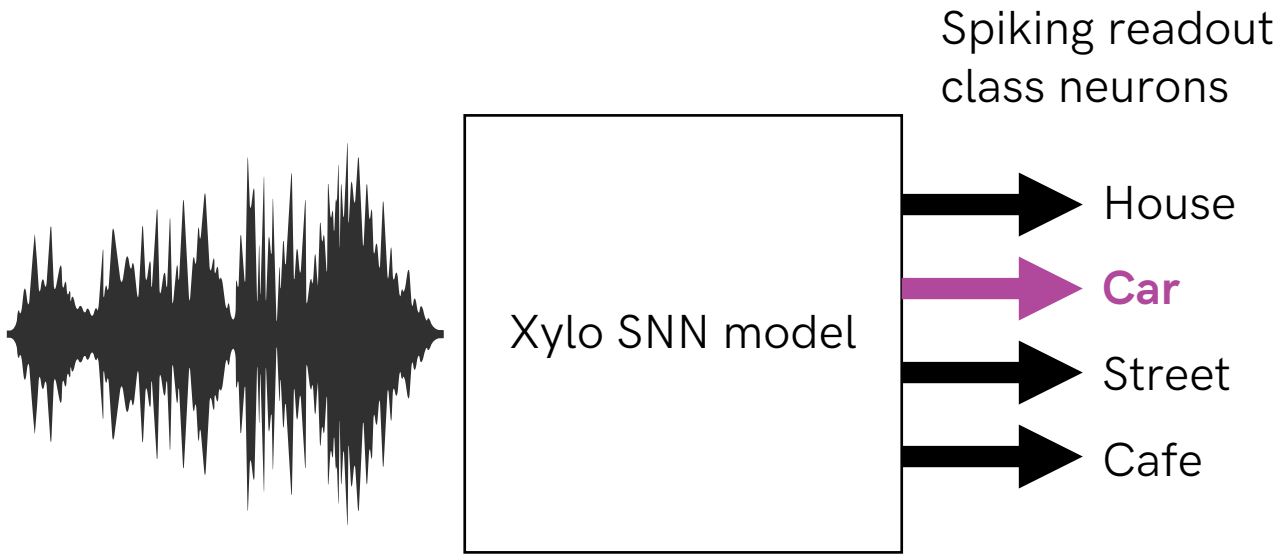
```
net.as_graph()
```

```
rockpool.devices.xylo
 ↳ mapper
```

```
rockpool.transform.quantize
```

```
rockpool.devices.xylo
 ↳ XyloSim
 ↳ XyloSamna
```

Example application
Ambient audio scene classification

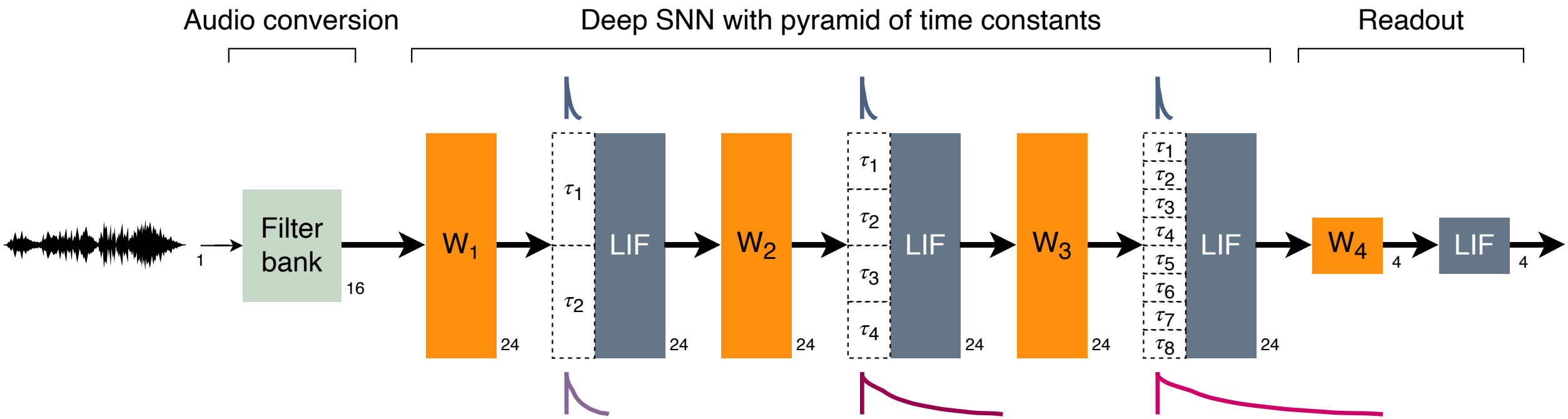


- Continuous low-power monitoring
- Embed in hearing aids
- Noise environment helps determine noise filtering settings

Network architecture

Ambient audio scene classification

SynSense



Rockpool code
Define network

Import modules and combinators from Rockpool

```
from rockpool.nn.combinators import Sequential
from rockpool.nn.modules import LinearTorch, LIFTorch
from rockpool.parameters import Constant
```

# of hidden neurons

```
Nh = 24 # - Hidden layer size
```

Define time constants
…

```
# - Define pyramid of time constants over SNN layers
tau_layer1, tau_layer2, tau_layer3 = …
```

Define network architecure
Sequential combinator
Linear weights (L1)
Spiking LIF neurons (L1)

```
# - Define the network as a sequential list of modules
net = Sequential(
        LinearTorch((16, Nh)),
        LIFTorch(Nh, tau_syn=Constant(tau_layer1)),
```

Layer 2
…

```
        LinearTorch((Nh, Nh)),
        LIFTorch(Nh, tau_syn=Constant(tau_layer2)),
```

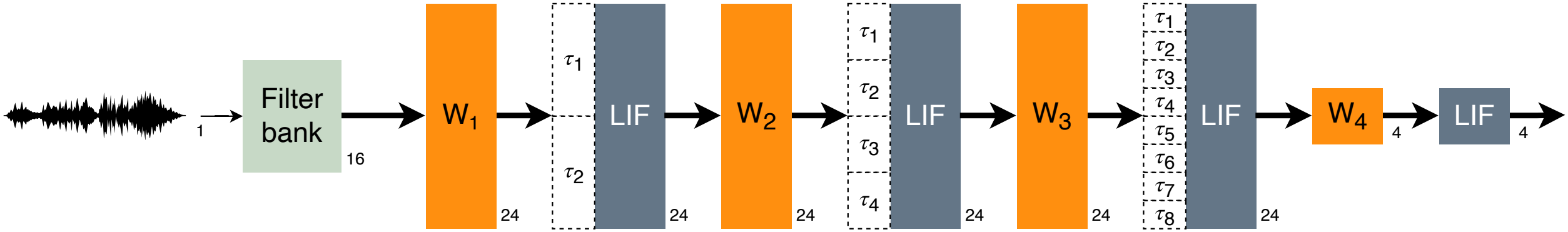
Layer 3
…

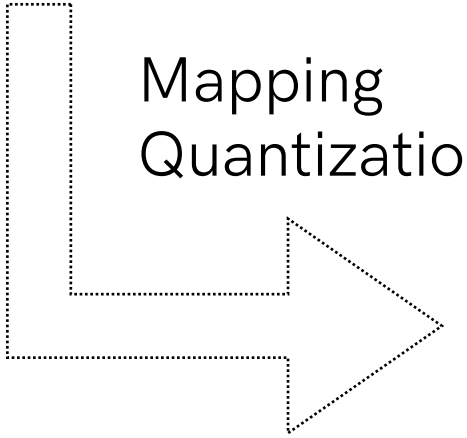
```
        LinearTorch((Nh, Nh)),
        LIFTorch(Nh, tau_syn=Constant(tau_layer3)),
```

Readout layer
…

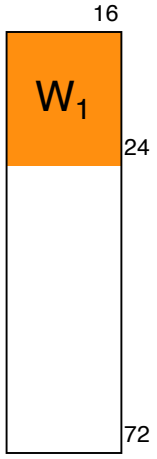
```
        LinearTorch((Nh, Nh)),
        LIFTorch(4))
```

Parameter translation to HW

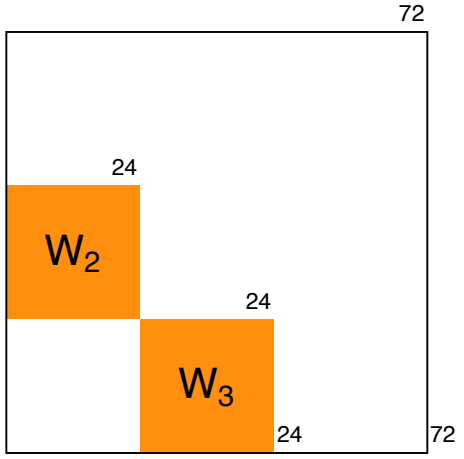




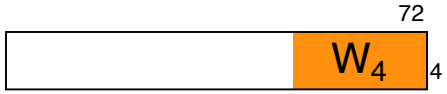
Input weights W_{in}



Recurrent weights W_{rec}



Output weights W_{out}



Rockpool code
Deploy network to Xylo HDK

Serialise the network

```
graph = net.as_graph()
```

Map the network to Xylo
Including DRC, HW assignment

```
from rockpool.devices import xylo
spec = xylo.mapper(graph)
```

Quantise the network weights
for the Xylo integer architecture

```
from rockpool.transform import quantize_methods as Q
spec.update(Q.channel_quantize(**spec)
```

Create a HW configurartion
Deploy to the HDK

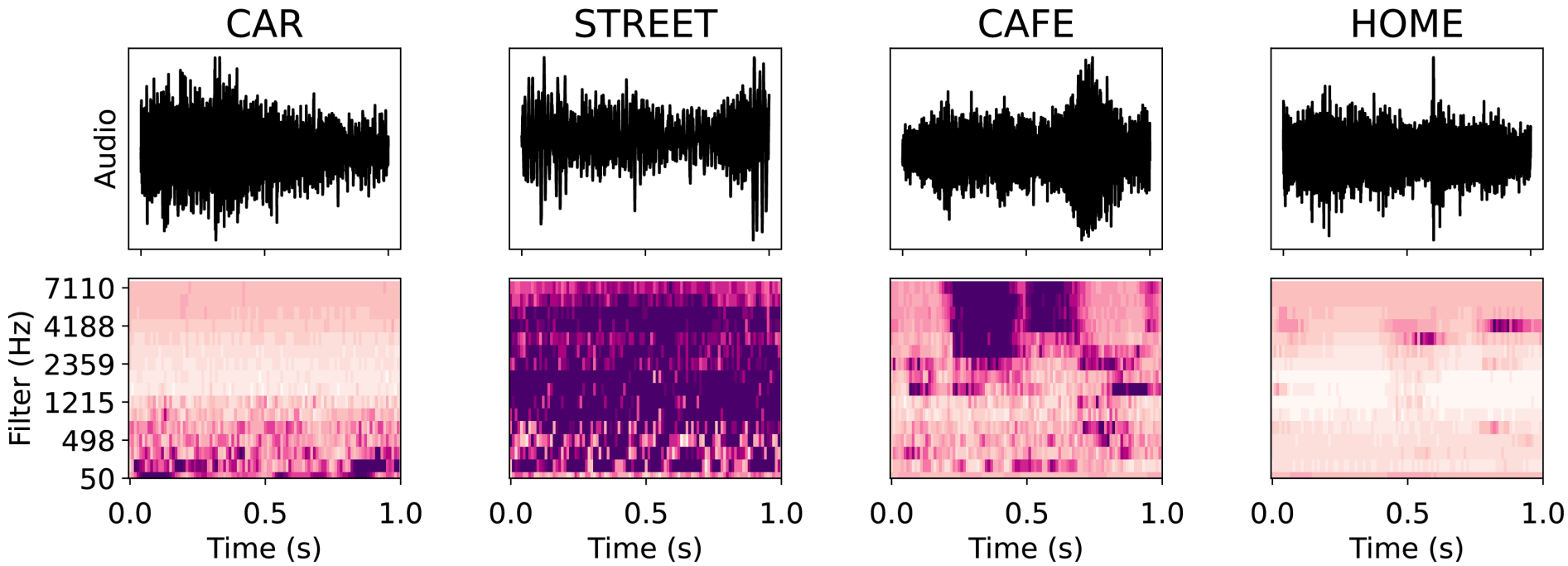
```
config = xylo.config_from_specification(**spec))
net_xylo = xylo.XyloSamna(hdk, config)
```

Run inference on the HDK

```
output, _, _ = net_xylo(inputs)
```

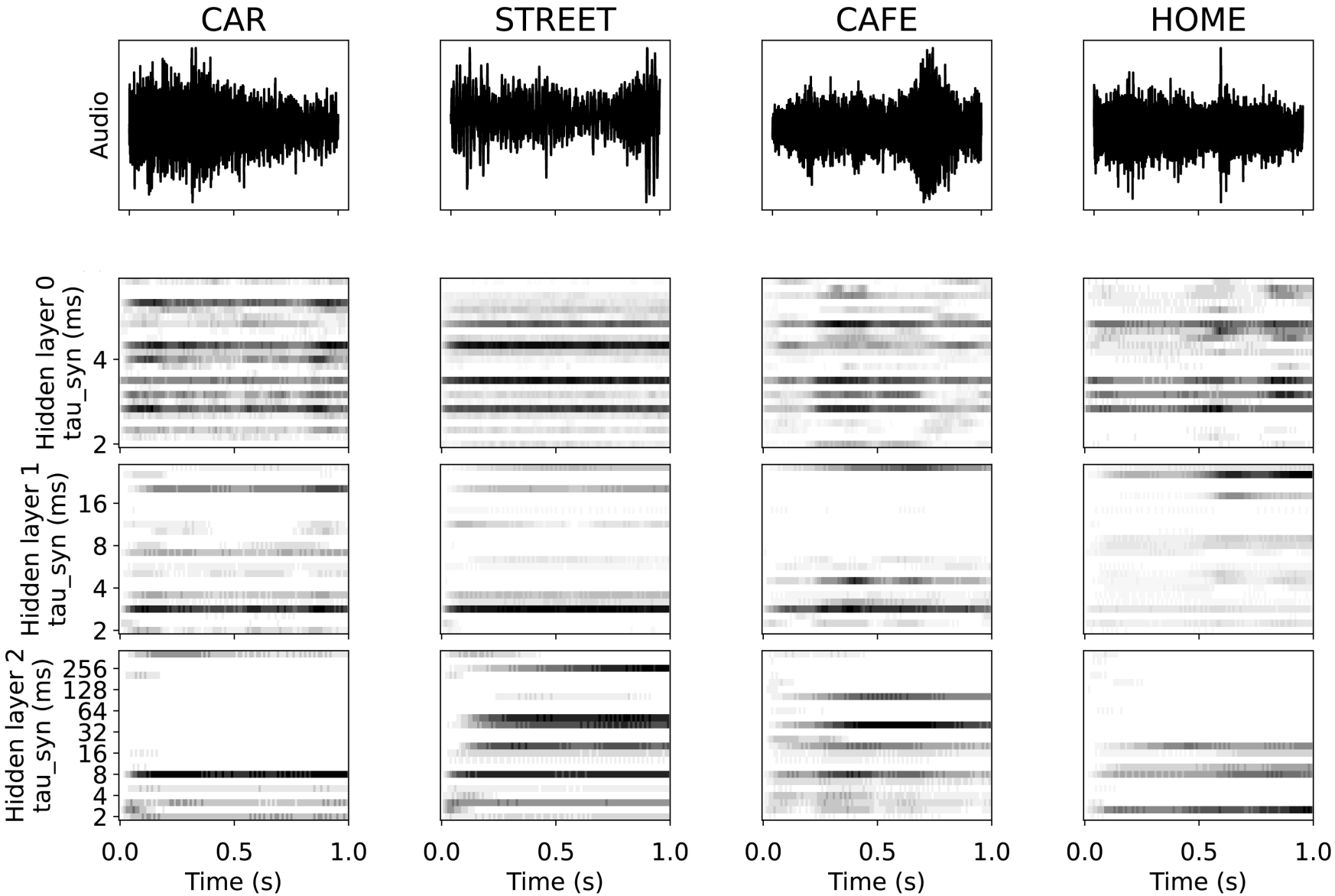
Network running on HDK

Ambient audio scene classification



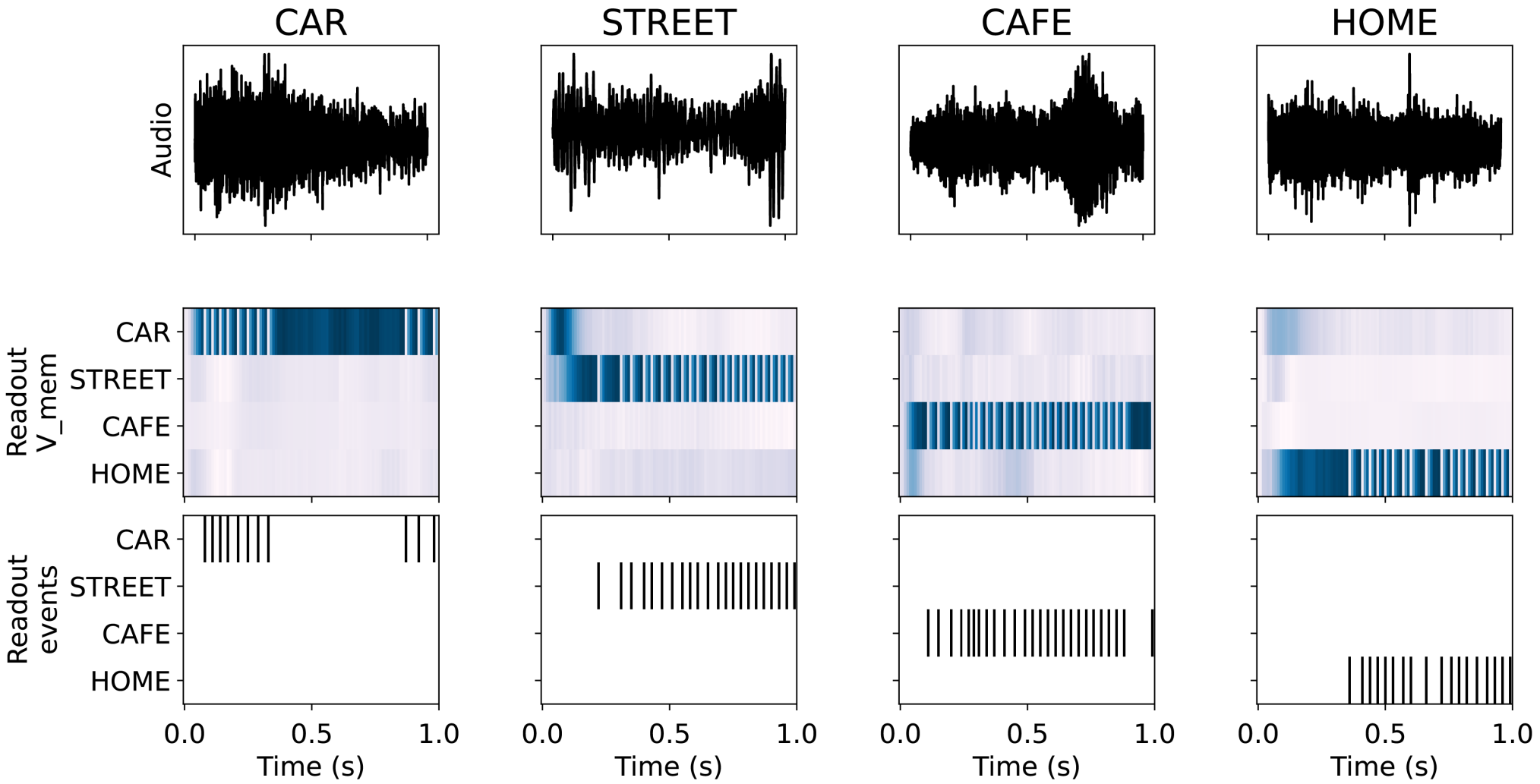
Network running on HDK

Ambient audio scene classification



Network running on HDK

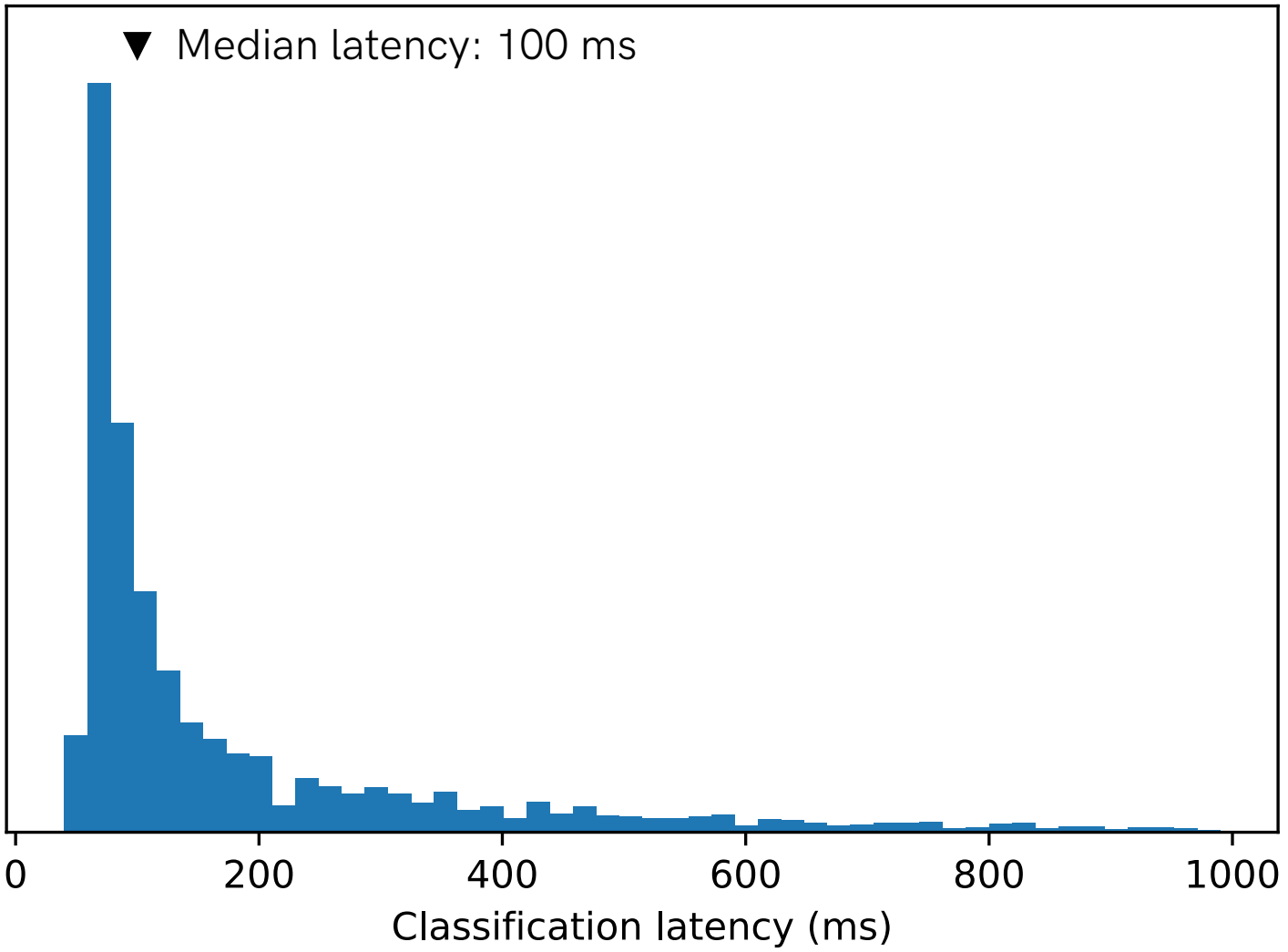
Ambient audio scene classification



Application performance
Accuracy

Four-class training accuracy (5s samples)	98.8%
Validation accuracy (5s samples; quantized)	98.7%
Test accuracy (Xylo HW; 60s samples; quantized)	98.0%

Application performance
Latency



Application performance
Power

SNN core idle power	219 µW
SNN core dynamic inference power	93 µW
SNN core total inference power	312 µW
IO power	230 µW

SynSense

Neuromorphic Intelligence

Dylan Muir | dylan.muir@synsense.ai