Lohi ("low-EE-hee") Architecture in detail

-First fully integrated SNN chip (60mm^2, 14nm process)
- Can solve LASSO optimization problems with 1000 times better energy-delay product

$\sigma(t) = \sum_k \delta(t - t_k)$ Spike train where t_k is the kth spike

Utilizes CUBA leaky-integrate and fire model:
$u_i(t)$: synaptic response current
$v_i(t)$: membrane potential

$$u_i(t) = \sum_{j \neq i} w_{ij} (\alpha_u * \sigma_j)(t) + b_i$$

w_{ij}: synaptic weight from neuron j to I

$\alpha_u(t) = \tau_u^{-1} \exp(-t/\tau_u) H(t)$: synaptic filter impulse response

Time constant

Unit step function

b_i: constant bias

Synaptic current is integrated until neuron outputs a spike when reaching the firing threshold θ_i

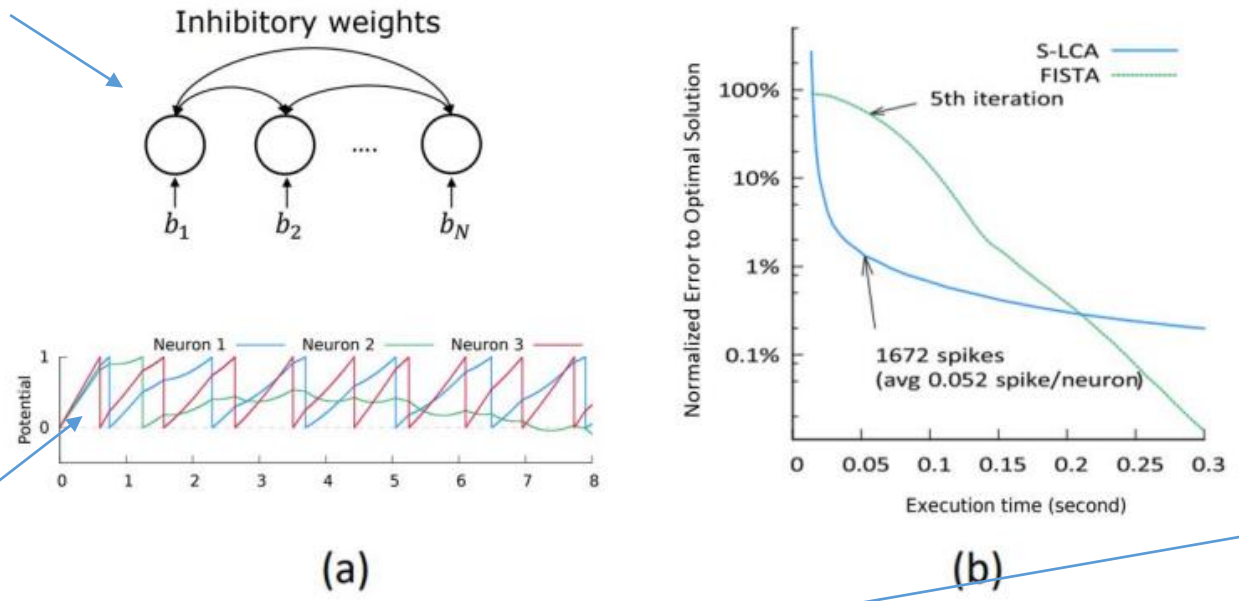
$$\dot{v}_i(t) = -\frac{1}{\tau_v} v_i(t) + u_i(t) - \theta_i \sigma_i(t)$$

Leaky time constant

v_i is initialized to a value less than θ_i and reset to 0 when spiking event occurs

Computation with spikes and fine-grained parallelism

Spiking locally competitive algorithm to determine sparse set of coefficients that best represents a given input as the linear combination of features from a feature dictionary



$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

$$RSS = \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$$

$$RSS + \lambda \sum_{j=1}^{p} |\beta_j|$$

Minimizing this makes many coefficients to converge to 0 and makes the equation sparce

L1 minimizing sparce coding

Spiking activity can represent the coefficients

Figure 1. (a) The network topology for solving LASSO. Each neuron receives the correlation b_i between the input data and a predefined feature vector as its input. The bottom figure shows the evolution of membrane potential in a three-neuron example; the spike rates of the neurons stabilize to fixed values. (b) An algorithmic efficiency comparison of a solution based on spiking network (S-LCA) and a conventional optimization method (FISTA). Both algorithms are implemented on a CPU with single thread. The *y*-axis is the normalized difference to the optimal objective function value. The figures are taken from P.T.P. Tang, T.H. Lin, and M. Davies.[2]

This means solution to the optimization problem found

This shows SNN can be very fast but not as accurate

- SNN’s advantage is using inhibition to perform winner take all. It significantly reduces computational complexity (However, may loose some accuracy since a lot of information is thrown away.)

- SNN's another advantage is the inherent parallelism: evolution of individual neuron states within a time step can be performed in parallel.

Learning with local information

- Each weight can only be accessed and modified by the destination neuron
- Decentralized learning algorithm
- Loihi tries to provide additional rules to further minimize the loss function
  a. spike traces corresponding to filtered presynaptic and postsynaptic spike trains with configurable time constraints. (short time constraint emphasize the precise spike timing while long time constraint emphasize the information contained in the spike rates)
  b. multiple spike traces for a given spike train filtered with different time constants
  c. two additional state variables per synapse besides the normal weight.
  d. reward traces that correspond to special reward spikes carrying signed impulse values to represent reward or punishment signals for reinforcement learning (reward spikes are broadcast to defined sets of synapses in the network)

Other computational primitives

- Stochastic noise insertion
- Configurable and adaptable synaptic, axon, and refractory delay
- Configurable dendritic tree processing. (neurons can become synapses to allow more inputs in a tree like structure, only one neuron is the soma)
- Neuron threshold adaptation in support of intrinsic excitability homeostasis
- Scaling and saturation of synaptic weights in support of permanence levels that exceed the range of weights used during inference.

Lohi (“low-EE-hee”) Architecture

-128 neuromorphic cores
-3 embedded x86 processor cores
-off-chip communication interfaces to extend the mesh in 4 directions
-Asynchronous network on chip (NoC): all communications between cores are through packet messages.
-all messages can be sourced externally by a host CPU or on-chip by the x86 cores
-each neuromorphic core has 1024 primitive spiking neural units (compartments) grouped into sets of trees constituting neurons.

connectivity

- Sparse network compression: supports 3 sparse matrix compression models that uses index state in each synapse to compute fan-out neuron indicies
- Core-to-core multicast: ability to send a spike from any neuron to any number of neurons
- Variable synaptic formats: any weight precision between 1~9bits, can be mixed
- Population-based hierarchical connectivity: connectivity templates may be defined to reduce connectivity resources significantly

learning

- Each core includes a programmable learning engine
- Operates on filtered spike traces
- Pairwise STDP, triplet STDP, reinforcement learning with synaptic tag
- Can utilize both rate averaged and spike-timing traces

Mesh operation example:

Step t
initial

Step t
Two cores
generate spikes

Step t
Let all firing neurons
in the two cores
distribute

Step t+1
Let all cores know we
are at the next
algorithmic step (all in
flight spikes are
flushed)

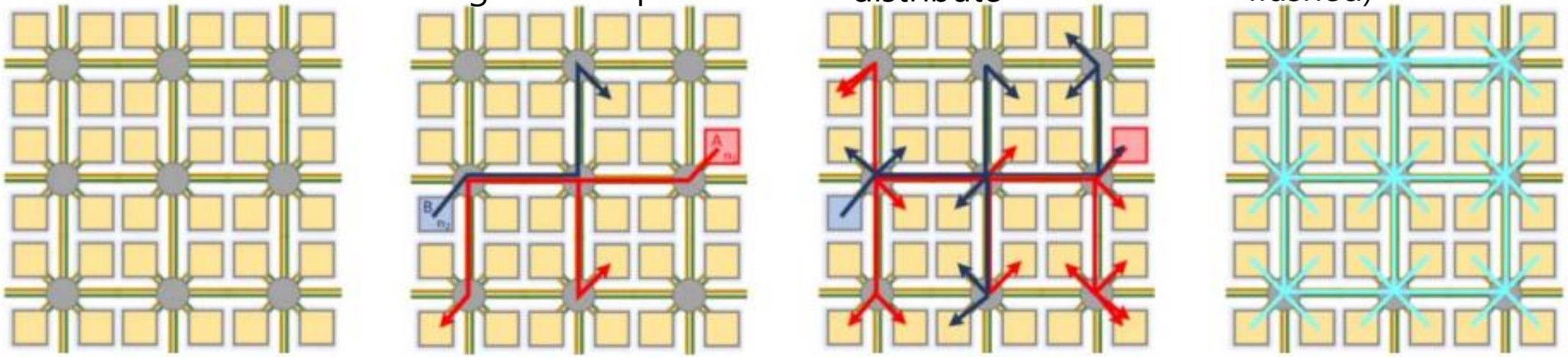


Figure 2. Mesh Operation: first box, initial idle state for time-step *t* (each square represents a core in the mesh containing multiple neurons); second box, neurons n_1 and n_2 in cores *A* and *B* fire and generate spike messages; third box, spikes from all other neurons firing on time-step *t* in cores *A* and *B* are distributed to their destination cores; and fourth box, each core advances its algorithmic time-step to $t + 1$ as it handshakes with its neighbors through barrier synchronization messages.

Loihi has two physical networks to avoid deadlock. Also, messages are sent alternately (shown in blue and red: not simultaneous)

Network Connectivity Architecture

Example of 7 neuron network mapped to 3 cores

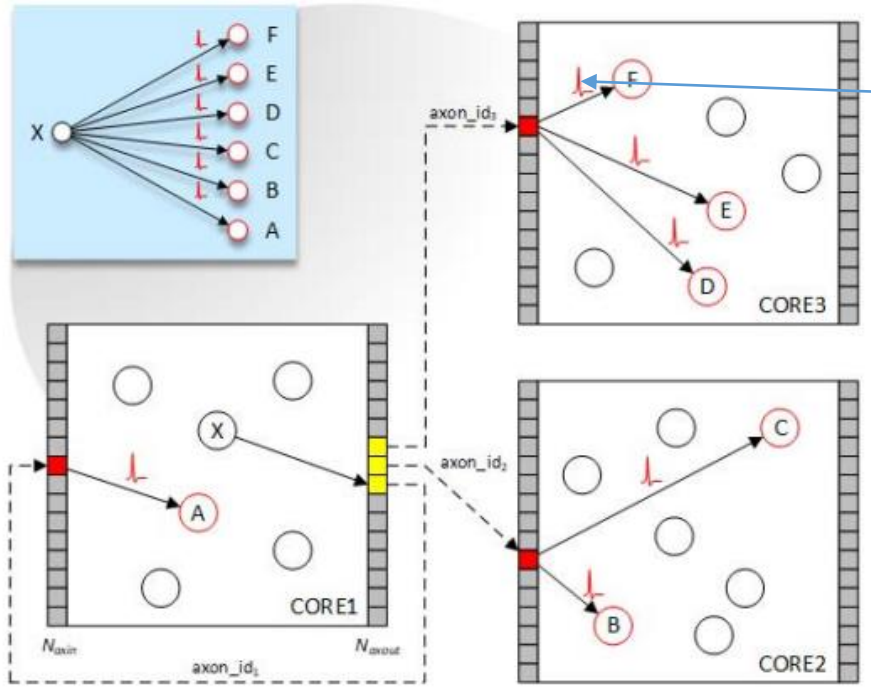


Figure 3. Neuron-to-neuron mesh routing model.

Each synapse can be described with
(i,j,wgt,dly,tag)

source destination

Synaptic fan-in states wgt, dly, tag are stored in synaptic memory

Each yellow fan-out edge containing (j,wgt,dly,tag)
Is assigned to unique (j) red axon id identifier

Connectivity constraints

- less 1024 neurons per core can be assigned
- Total synaptic fan-in state for a core is less than 128KB
- Max core output destination is 4096

Learning Engine

Updates all synaptic state after a periodic learning epoch time T_{epoch}

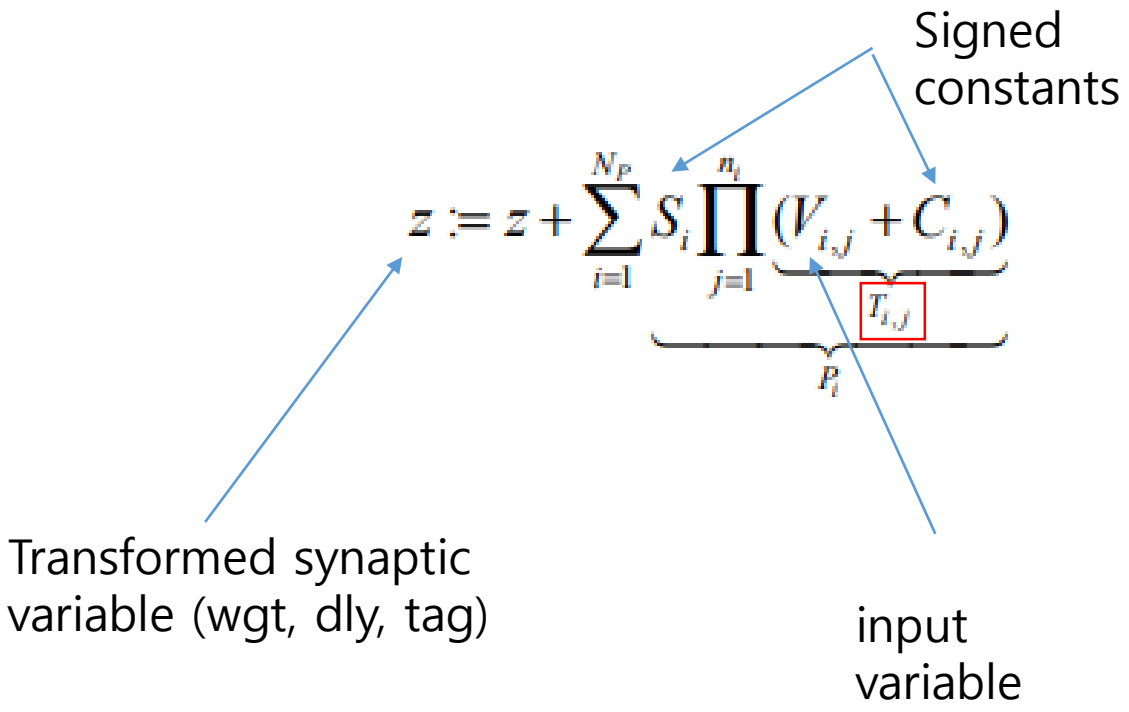
Pairwise STDP:

$$\Delta w_{i,j} = \begin{cases} A_{-}\mathcal{F}(t - t_i^{post}), & \text{On presynaptic spike} \\ A_{+}\mathcal{F}(t - t_j^{pre}), & \text{On postsynaptic spike} \end{cases}$$

Approximation of $e^{-x/\tau} \cdot H(x)$

Learning rule functional form

-for each epoch, a synapse will be updated if pre or post synaptic conditions are satisfied



To ensure only one spike is received in a given epoch it is typically set to minimum refractory delay of all neurons in the network

Trace evaluation

Decay factor

Spike arrival sequence: value can be 1 or 0

$$x[t] = \alpha \cdot x[t - 1] + \delta \cdot s[t].$$

Trace sequence over time

impulse amount, if set to 1 emulates the baseline STDP dependent on pre/post spike time separations

If set less than 1, it can emulate average spike rate

Table 1. Learning rule product terms.

Encoding	$Term\,(T_{i,j})$	Bits	Description
0	$x_0 + C$	5b (U)	Presynaptic spike count
1	$x_1 + C$	7b (U)	1st presynaptic trace
2	$x_2 + C$	7b (U)	2nd presynaptic trace
3	$y_0 + C$	5b (U)	Postsynaptic spike count
4	$y_1 + C$	7b (U)	1st postsynaptic trace
5	$y_2 + C$	7b (U)	2nd postsynaptic trace
6	$y_3 + C$	7b (U)	3rd postsynaptic trace
7	$r_0 + C$	1b (U)	Reward spike
8	$r_1 + C$	8b (S)	Reward trace
9	wgt+C	9b (S)	Synaptic weight
10	dly+C	6b (U)	Synaptic delay
11	tag+C	9b (S)	Synaptic tag
12	sgn(wgt+C)	1b (S)	Sign of case 9 (± 1)
13	sgn(dly+C)	1b (S)	Sign of case 10 (± 1)
14	sgn(tag+C)	1b (S)	Sign of case 11 (± 1)
15	C	8b (S)	Constant term. (Variant 1)
15	$S_m \cdot 2^{S_e}$	4b (S)	Scaling term. 4b mantissa, 4b exponent. (Variant 2)

Core microarchitecture

Color boxes represent major memories that store connectivity, configuration, dynamic state of all neurons mapped to the core

2Mb SRAM

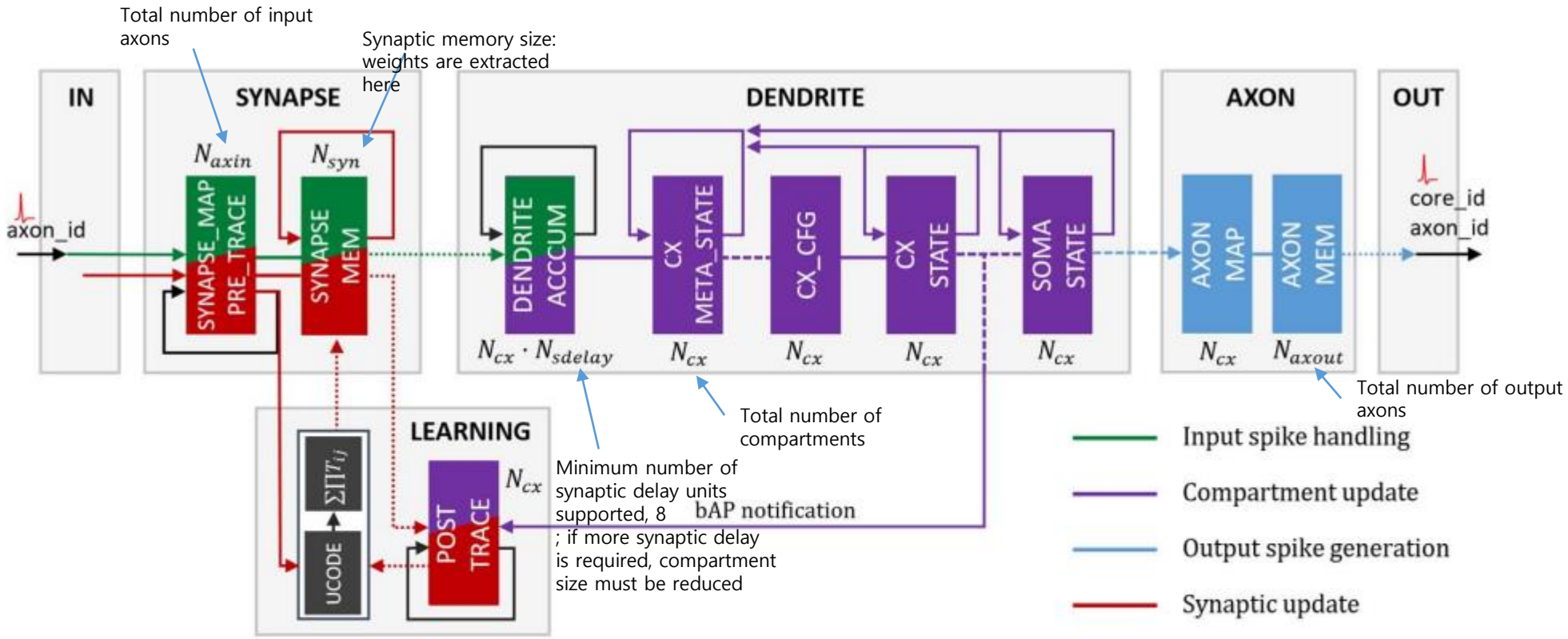


Figure 4. Core Top-Level Microarchitecture. The SYNAPSE unit processes all incoming spikes and reads out the associated synaptic weights from the memory. The DENDRITE unit updates the state variables *u* and *v* of all neurons in the core. The AXON unit generates spike messages for all fan-out cores of each firing neuron. The LEARNING unit updates synaptic weights using the programmed learning rules at epoch boundaries.

Asynchronous design

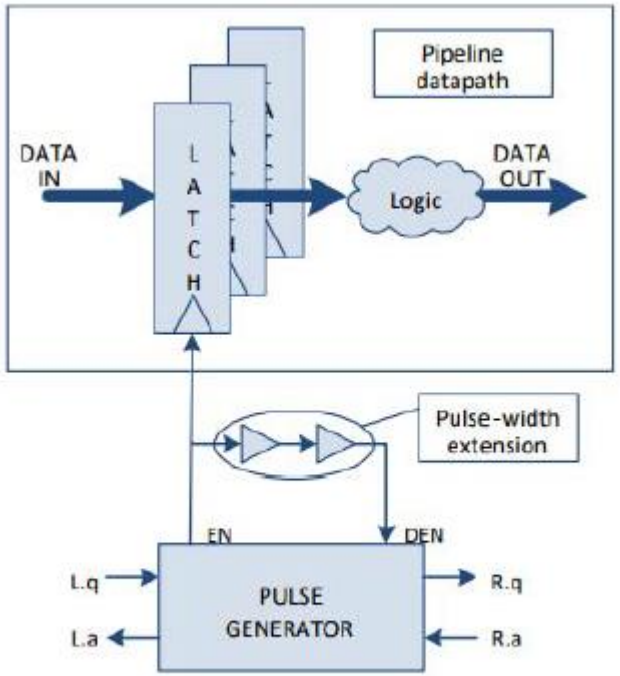


Figure 5: Bundled data pipeline stage.

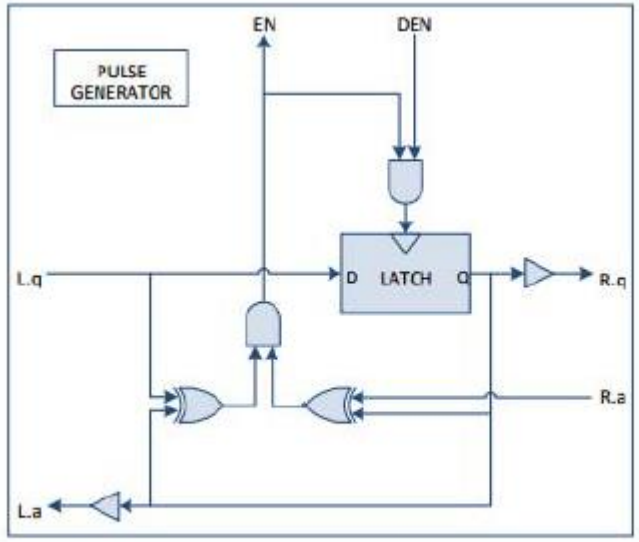


Figure 6. Bundled data pulse generator circuit.

Each pipeline stage has at least one pulse generator that implements two-phase (request, acknowledge) handshake and latch sequencing

Summary

- 14 nm FinFET process
- 2.07 x 10^9 TRs
- 33 MB SRAM
- 128 cores
- 3 x86 cores
- 60 mm^2
- 16MB of synaptic memory

Table 2. Loihi pre-silicon performance and energy measurements.

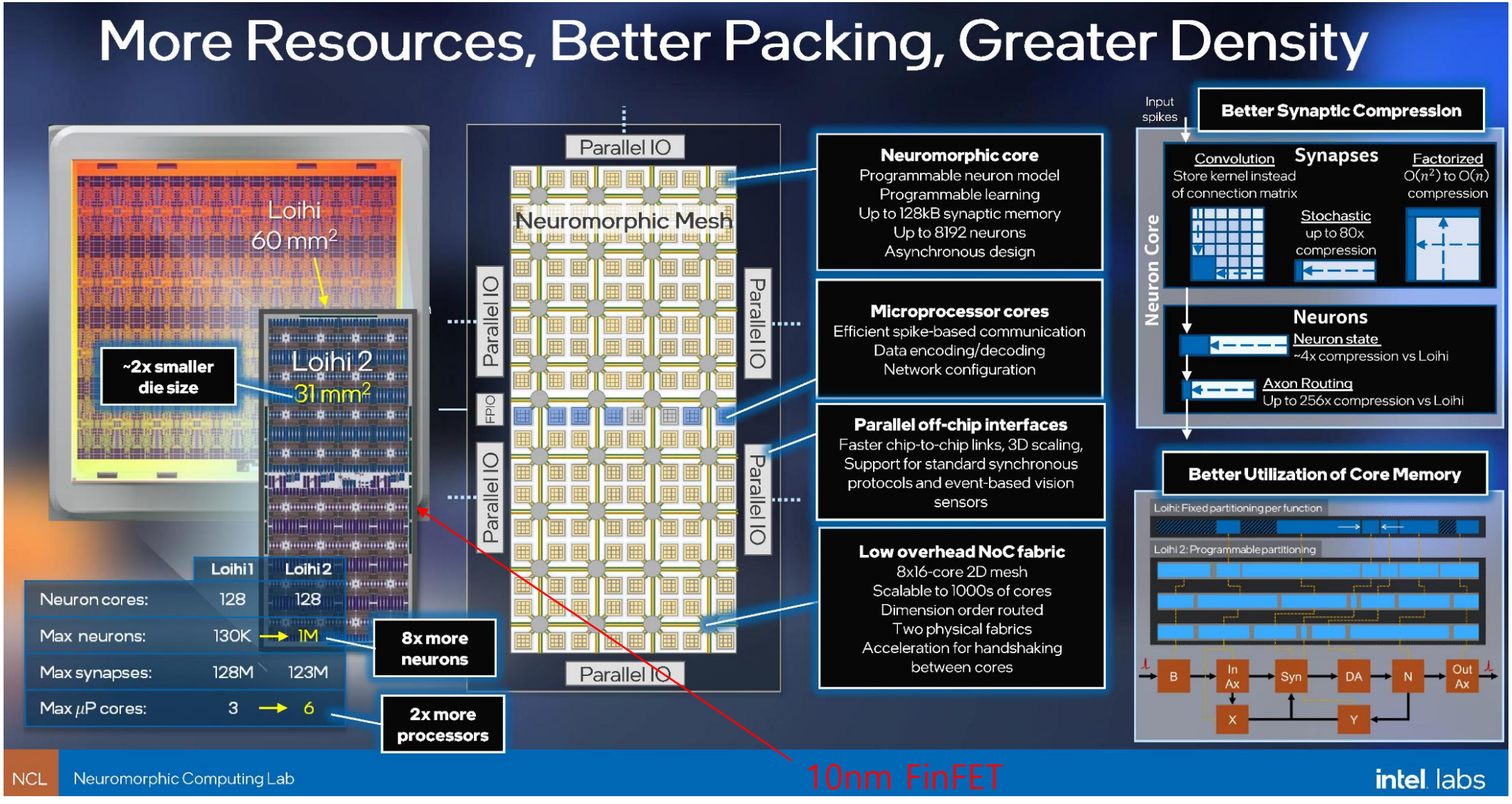
Measured parameter	**Value at 0.75 V**
Cross-sectional spike bandwidth per tile	3.44 Gspike/s
Within-tile spike energy	1.7 pJ
Within-tile spike latency	2.1 ns
Energy per tile hop (E-W / N-S)	3.0 pJ / 4.0 pJ
Latency per tile hop (E-W / N-S)	4.1 ns / 6.5 ns
Energy per synaptic spike op (min)	23.6 pJ
Time per synaptic spike op (max)	3.5 ns
Energy per synaptic update (pairwise STDP)	120 pJ
Time per synaptic update (pairwise STDP)	6.1 ns
Energy per neuron update (active / inactive)	81 pJ / 52 pJ
Time per neuron update (active / inactive)	8.4 ns / 5.3 ns
Mesh-wide barrier sync time (1-32 tiles)	113-465 ns

Table 3. Comparison of solving ℓ_1 minimization on Loihi and Atom.

Number of Unknowns	400	1,700	32,256
Number of non-zeros in solutions	≈10	≈30	≈420
Energy	2.58x	8.08x	48.74x
Delay	0.27x	2.76x	118.18x
EDP	0.7x	22.33x	5760x

Loihi 2

Much more compact than Loihi, due to hardware optimization. Technology node advancement helped a little too.



Loihi2 related research paper -> A 4096-neuron 1M-synapse 3.8-pJ/SOP spiking neural network with on chip STDP learning and sparse weights in 10-nm FinFET CMOS

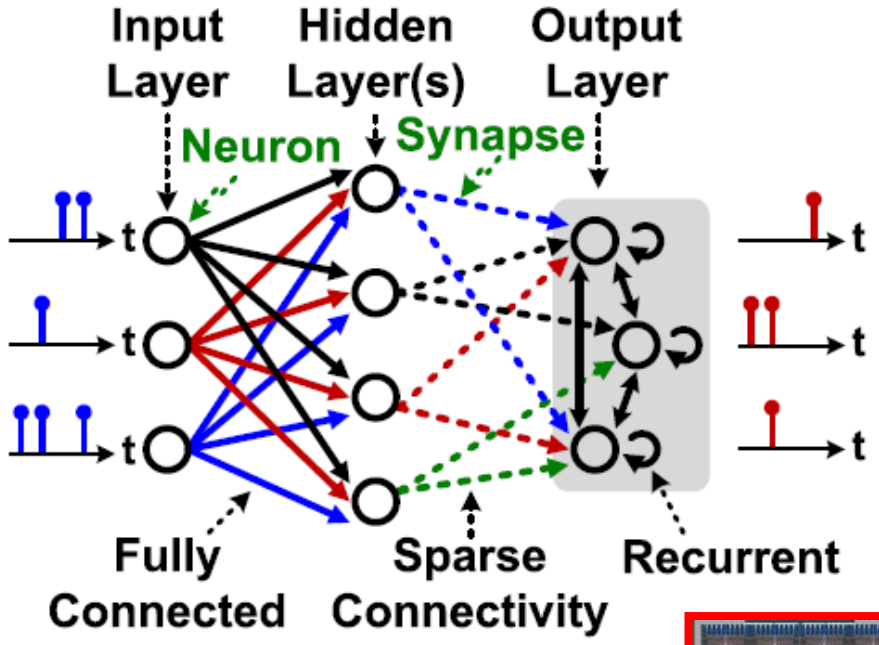
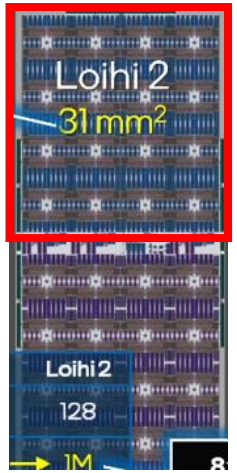


Fig. 1. SNN topology and terminology.



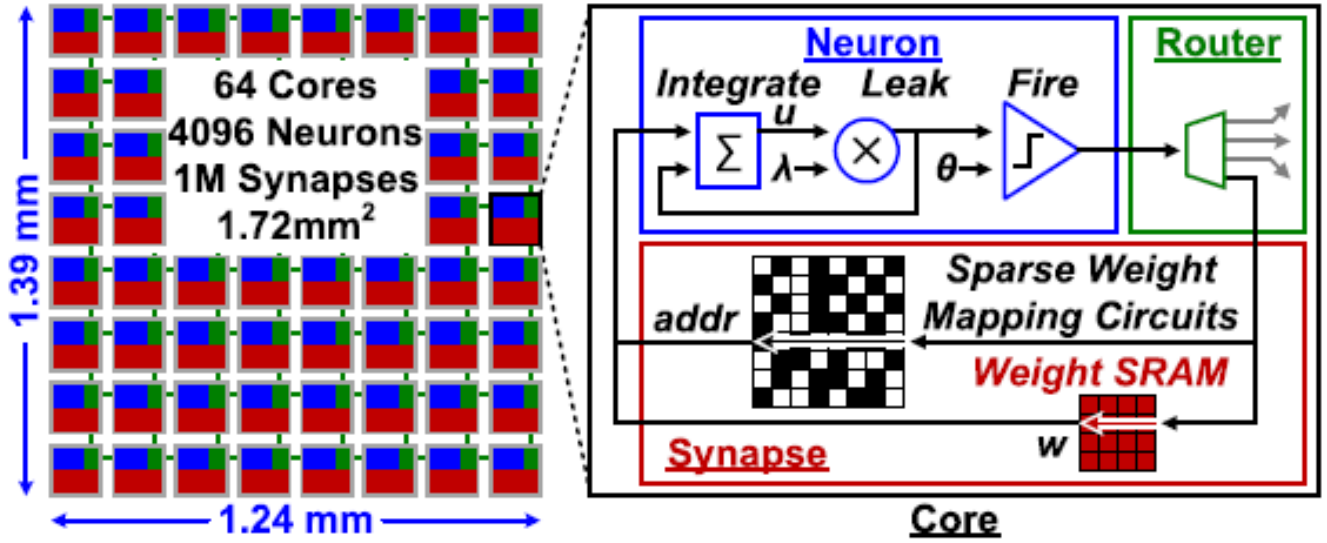
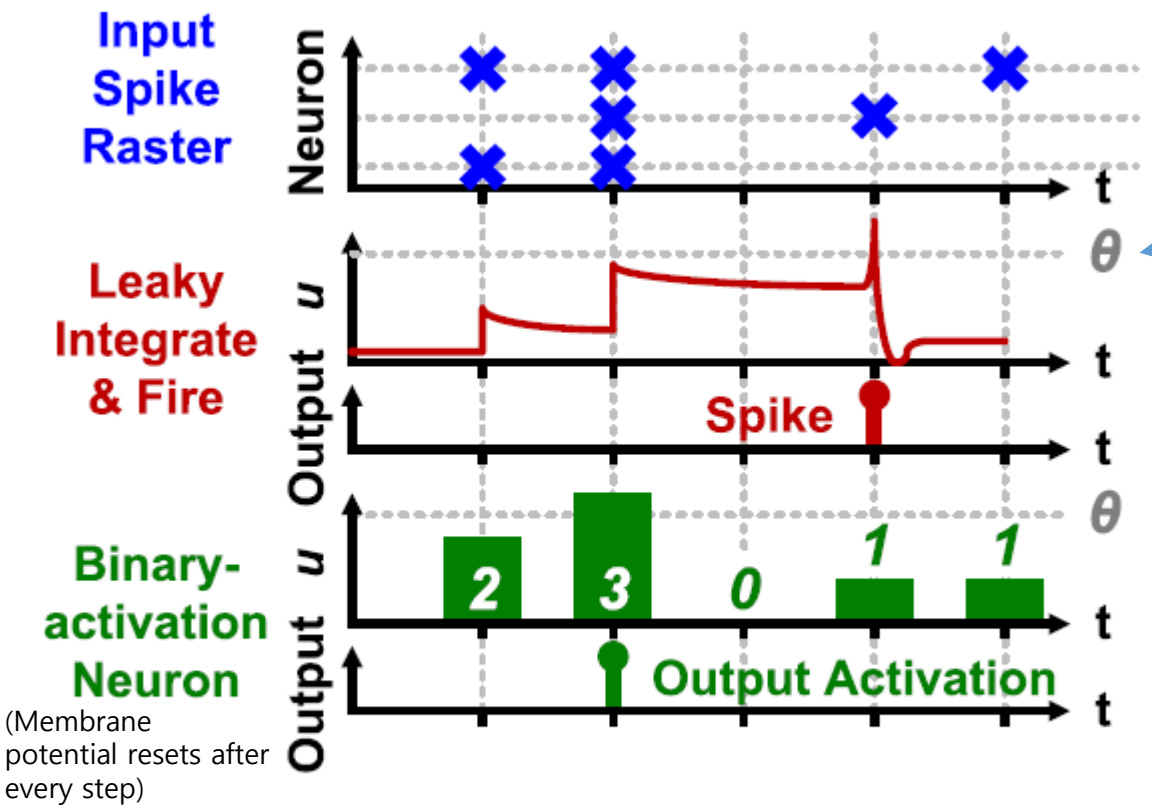


Fig. 2. SNN chip overview.

LIF



(Membrane potential resets after every step)

Fig. 3. LIF neurons integrate weighted spikes onto the membrane potential and generate an output spike when the net effect of inputs over time exceeds a threshold.

Threshold

$$u(t+1) = \lambda u(t) + \sum \mathrm{sw} + B + I$$

Membrane potential

Leak factor

spike

weight

bias

Primary input

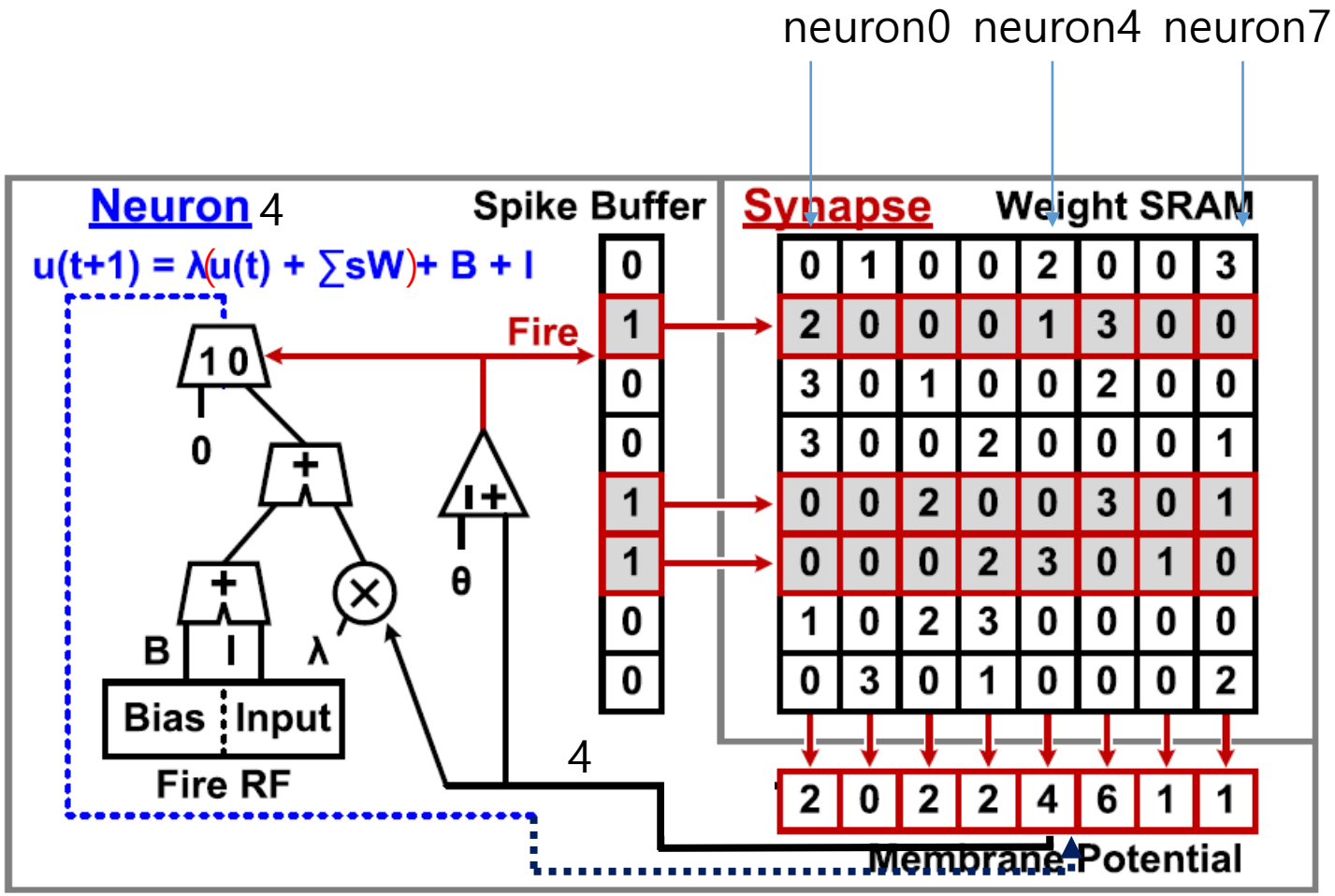


Fig. 4. Neuron circuits multiply the weight matrix with the sparse spike vector and for integrate the result into the membrane potential RF.

(Register file)

Spike rate is adjusted to a target value by changing the bias

- Higher bias allows more frequent threshold crossings and more spikes

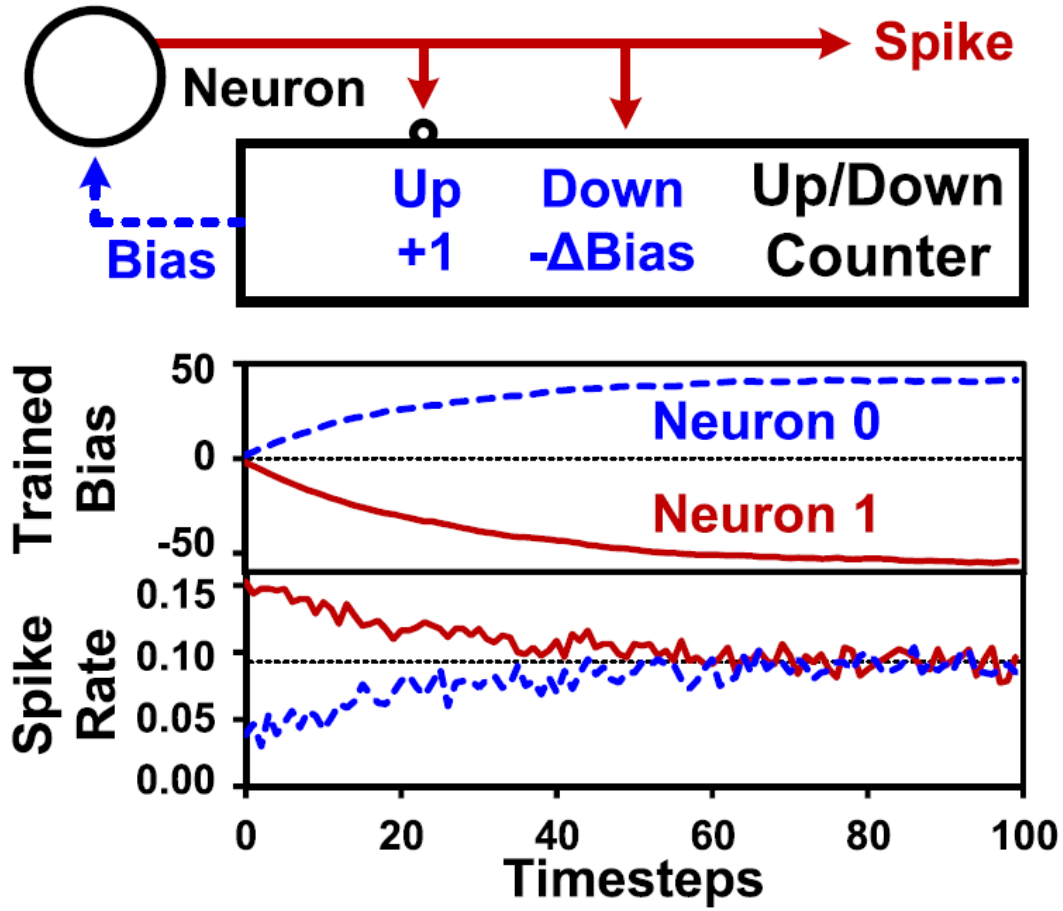


Fig. 5. Homeostasis sets the average spike rate with negative bias feedback.

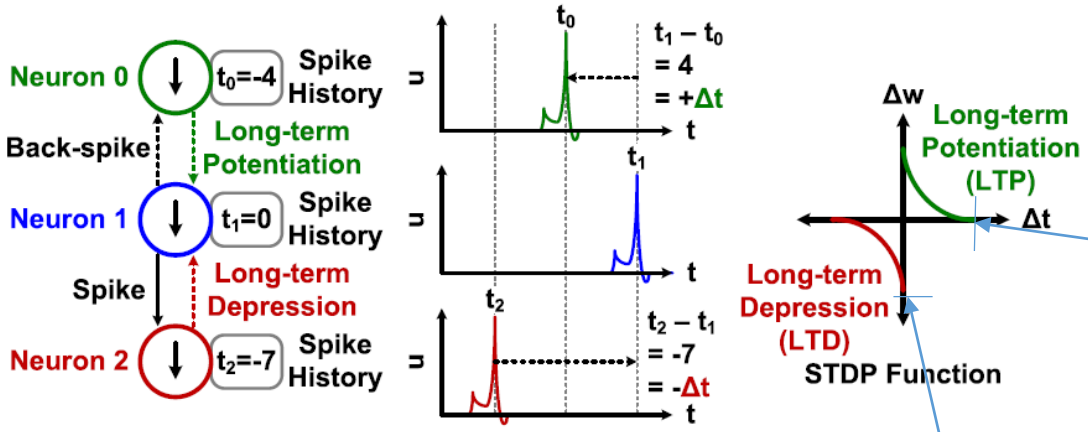
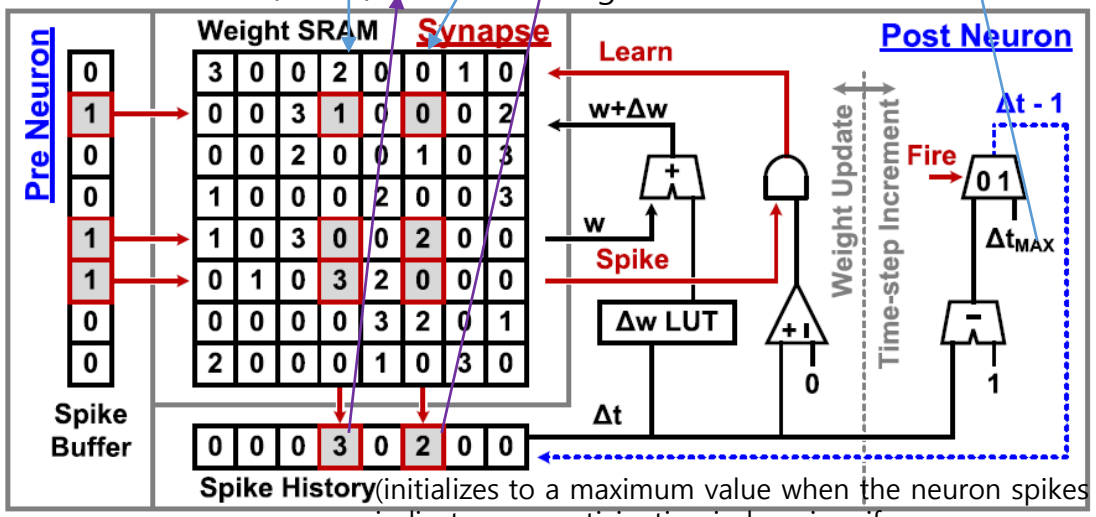
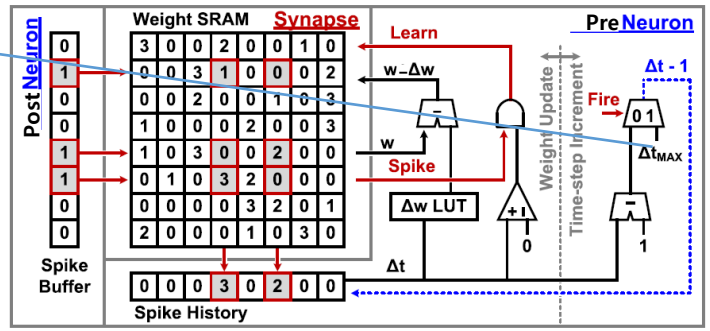


Fig. 6. STDP overview.



(initializes to a maximum value when the neuron spikes and counts down each step till saturation at 0 which indicate non participation in learning; if non zero upon receiving the next spike, it uses the history value to look up appropriate **Δw**)

Fig. 7. STDP circuits, spike history counters, and weight LUT for event-driven update of synapses connecting neurons that fire in close succession.



Example of LTD where post neuron spike history is used

LTP learning example

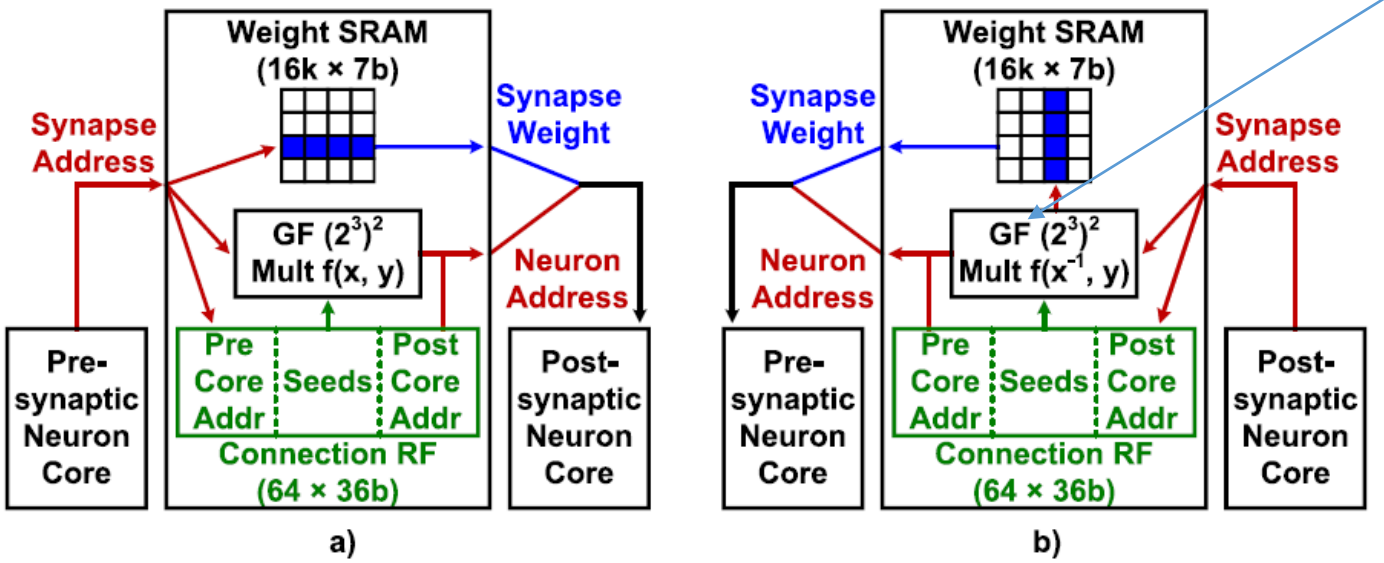


Fig. 8. Structured connectivity enables a low-memory method of generating sparse weights. A permutation function calculates a target neuron for each synapse. (a) Forward and (b) reverse pathways are implemented with an invertible permutation function to perform on-chip STDP learning.

Galois field permutation function

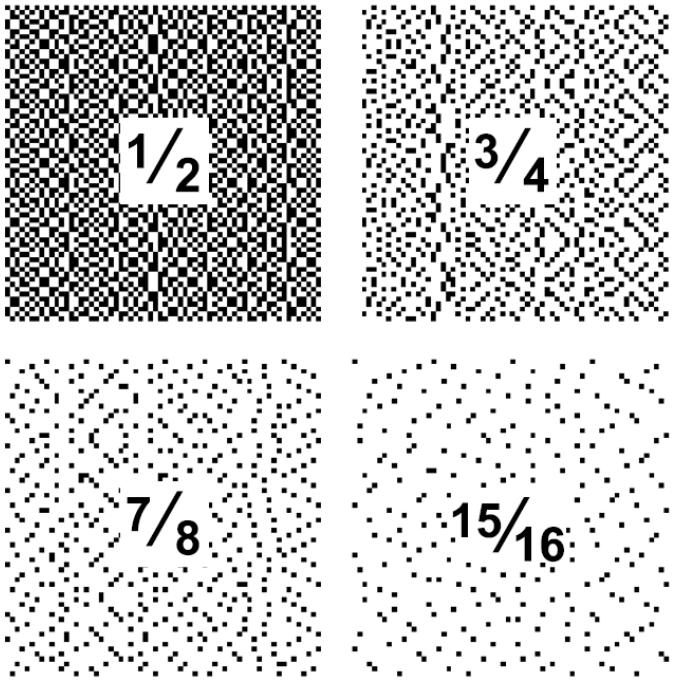


Fig. 9. Example 64 × 64 connectivity maps with user-defined sparsity.

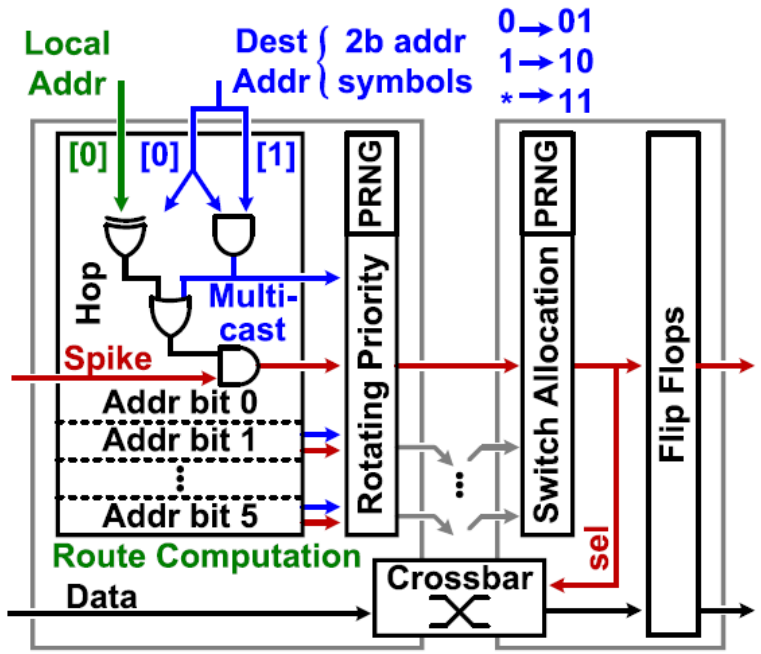


Fig. 13. NoC router schematic including multicast route computation circuits.

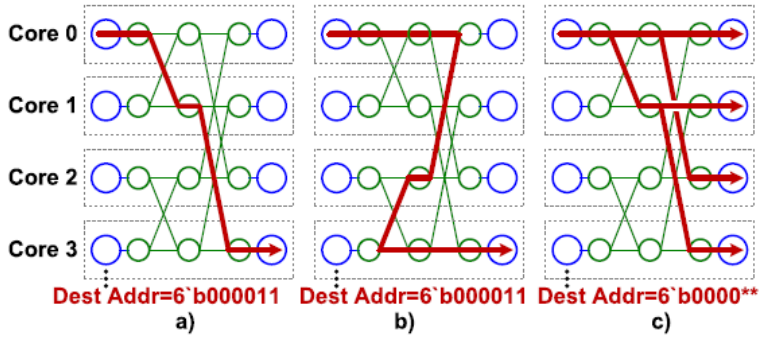
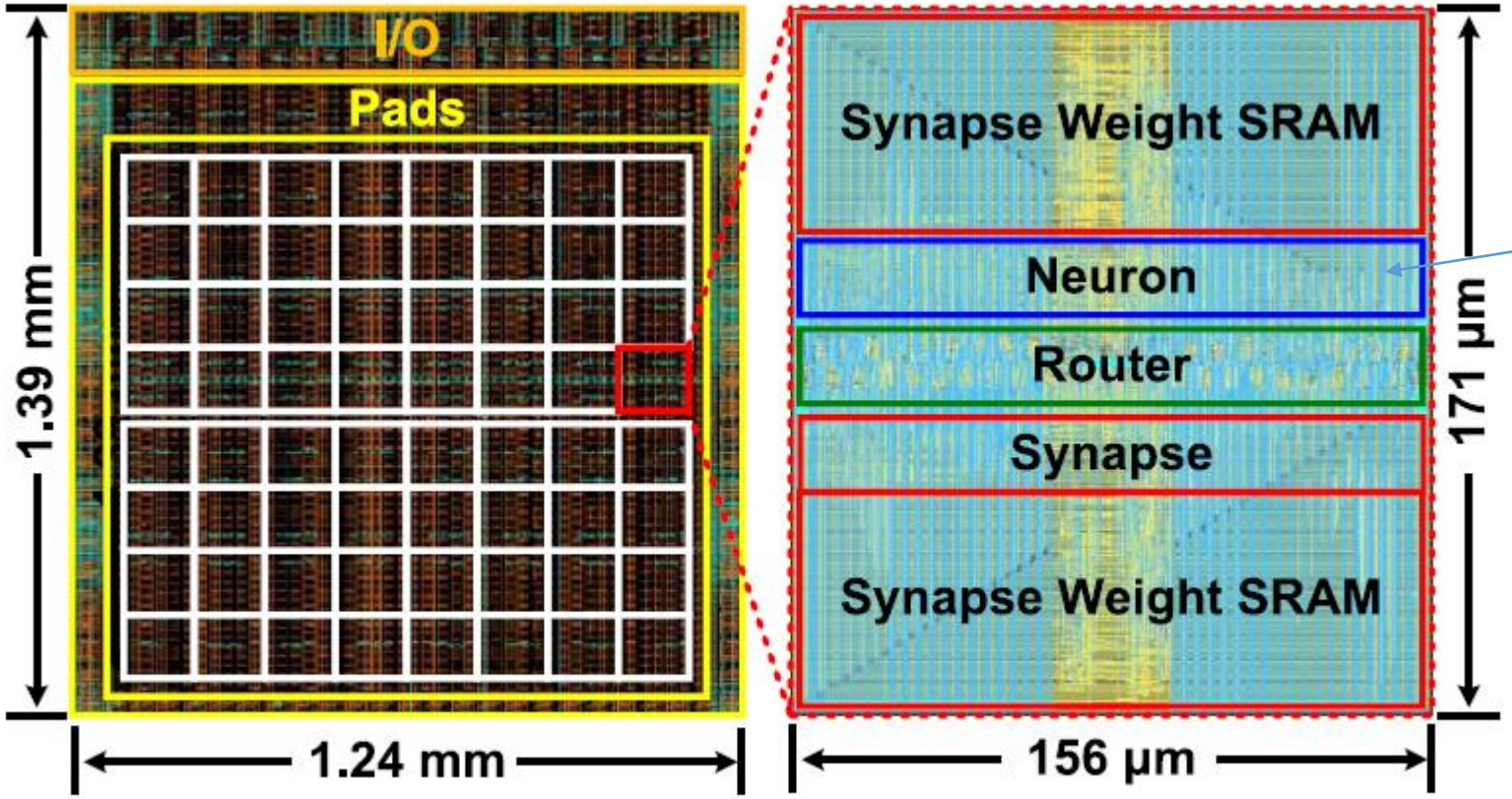


Fig. 14. Example spike delivery paths for randomized routing and multicast. (a) Fewer hops. (b) Path diversity. (c) Multicast.

Pseudo Random Number Generator is used to randomly select from candidate hops to take different routes between the same source and destination cores to improve noise characteristics.
(due to asynchronous operation, some path may not work as well as others?)



4k/64 = 64 neurons

Fig. 16. 10-nm FinFET chip micrograph and core layout detail.

Provide reward using supervised learning

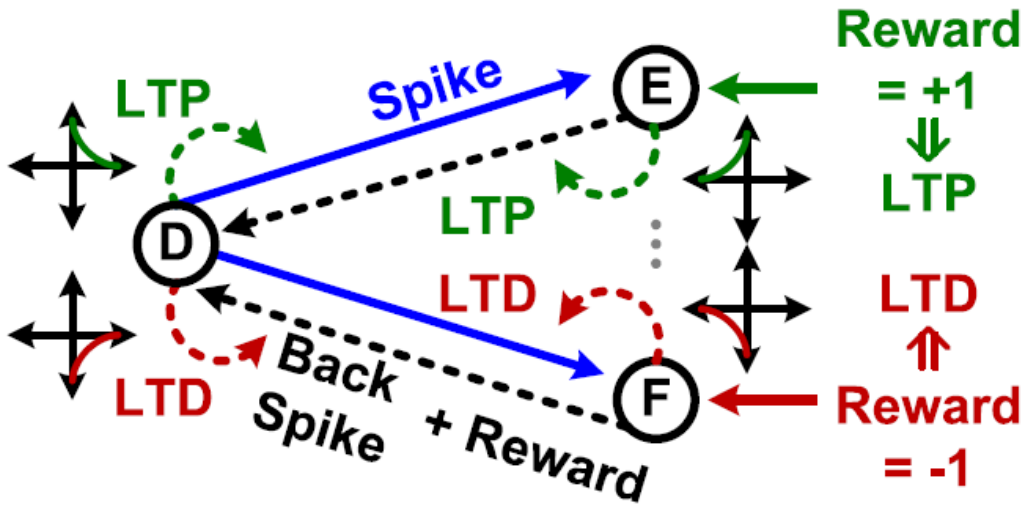
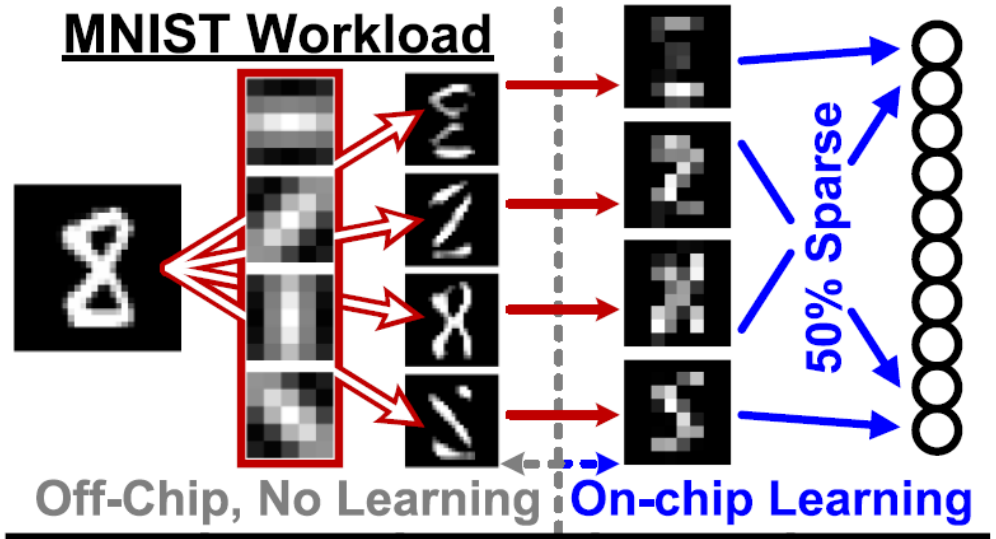
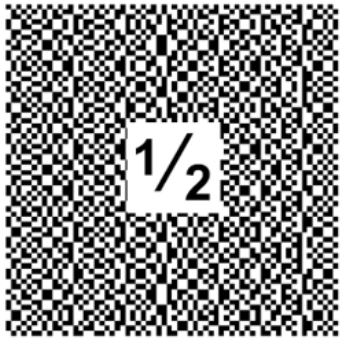


Fig. 24. Supervised training by modulating STDP polarity.



MNIST	Gabor	Pool	Input	Output
28x28	4@5×5	3x3	236	20

Mapped to 4 cores



89% accuracy for fully connected,
88% for 50% weight sparsity

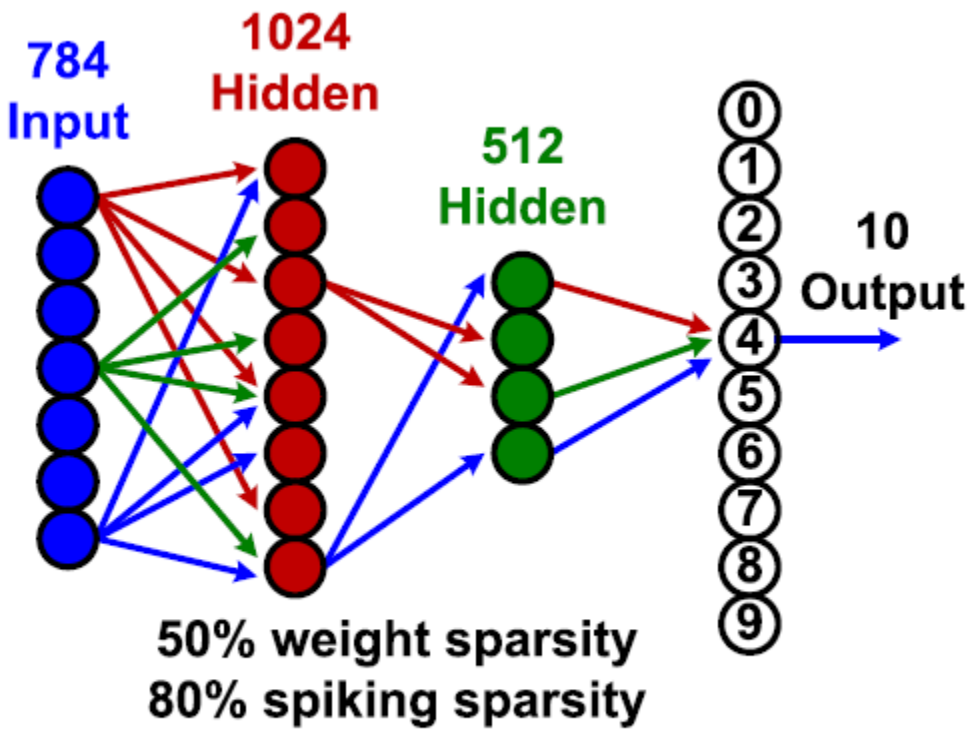


Fig. 27. Binary-activation MLP with 50% sparse weights.

using

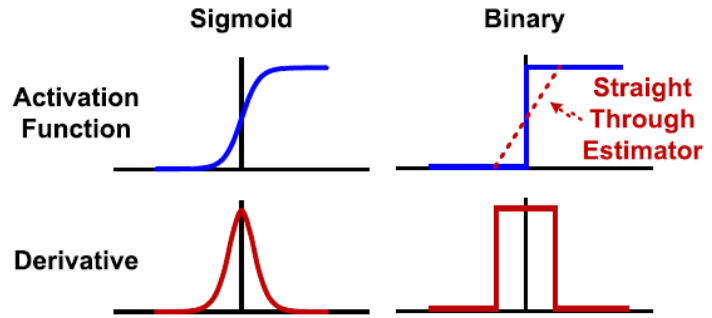


Fig. 28. Backpropagation with binary activations and derivative estimation.

Offline backpropagation, can be uploaded to the chip to perform the inference with high accuracy: 98.15% at 50% sparsity

Spiking neurons with spatiotemporal dynamics and gain modulation for monolithically integrated memristive neural networks

Dendrites: receive information.
Axons: transmit information

Axon->synapse->Dendrite

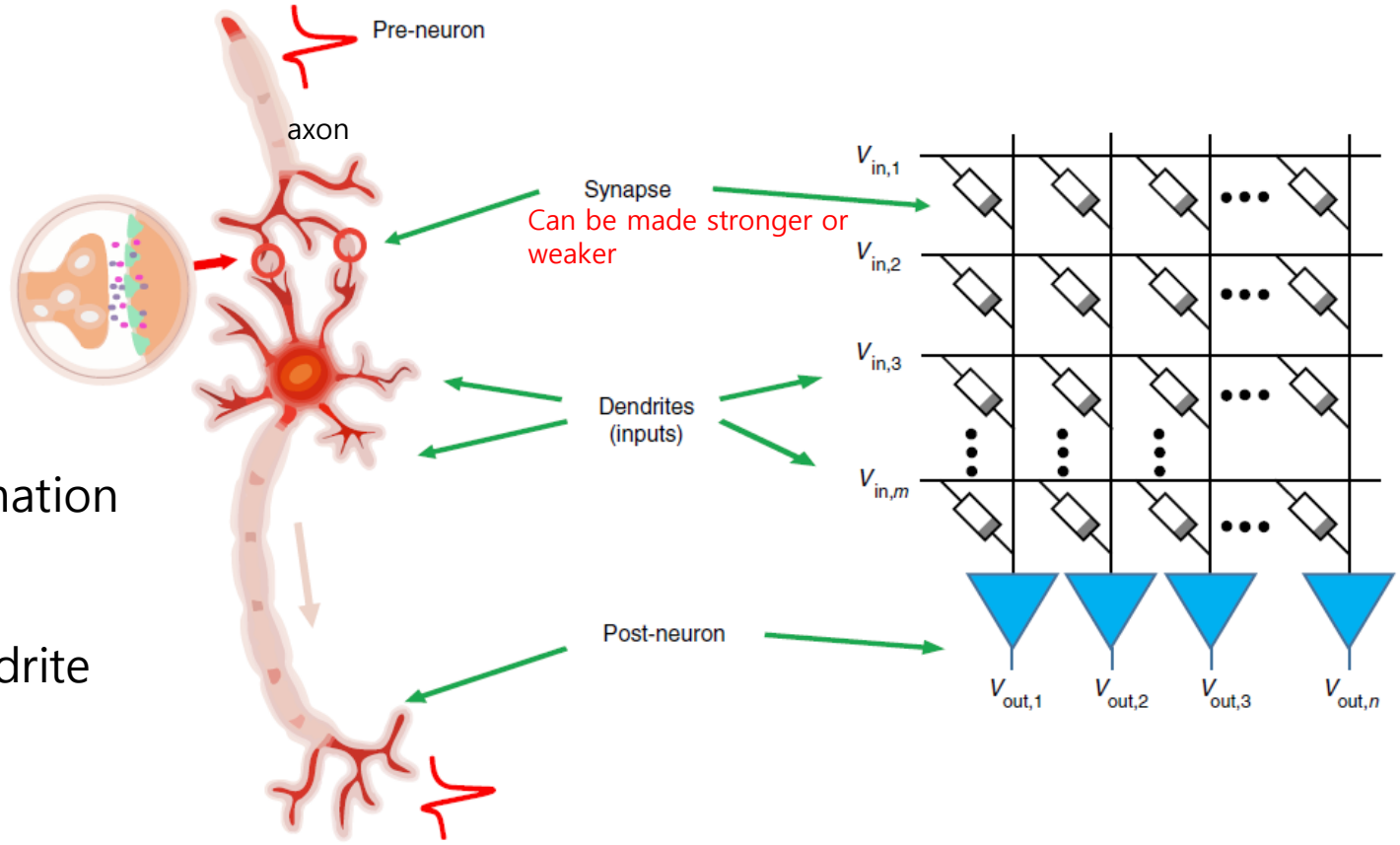


Fig. 1 Comparison of biological and artificial neurons. Schematic of biological neurons and synapses (left) compared with an artificial neural network (right).

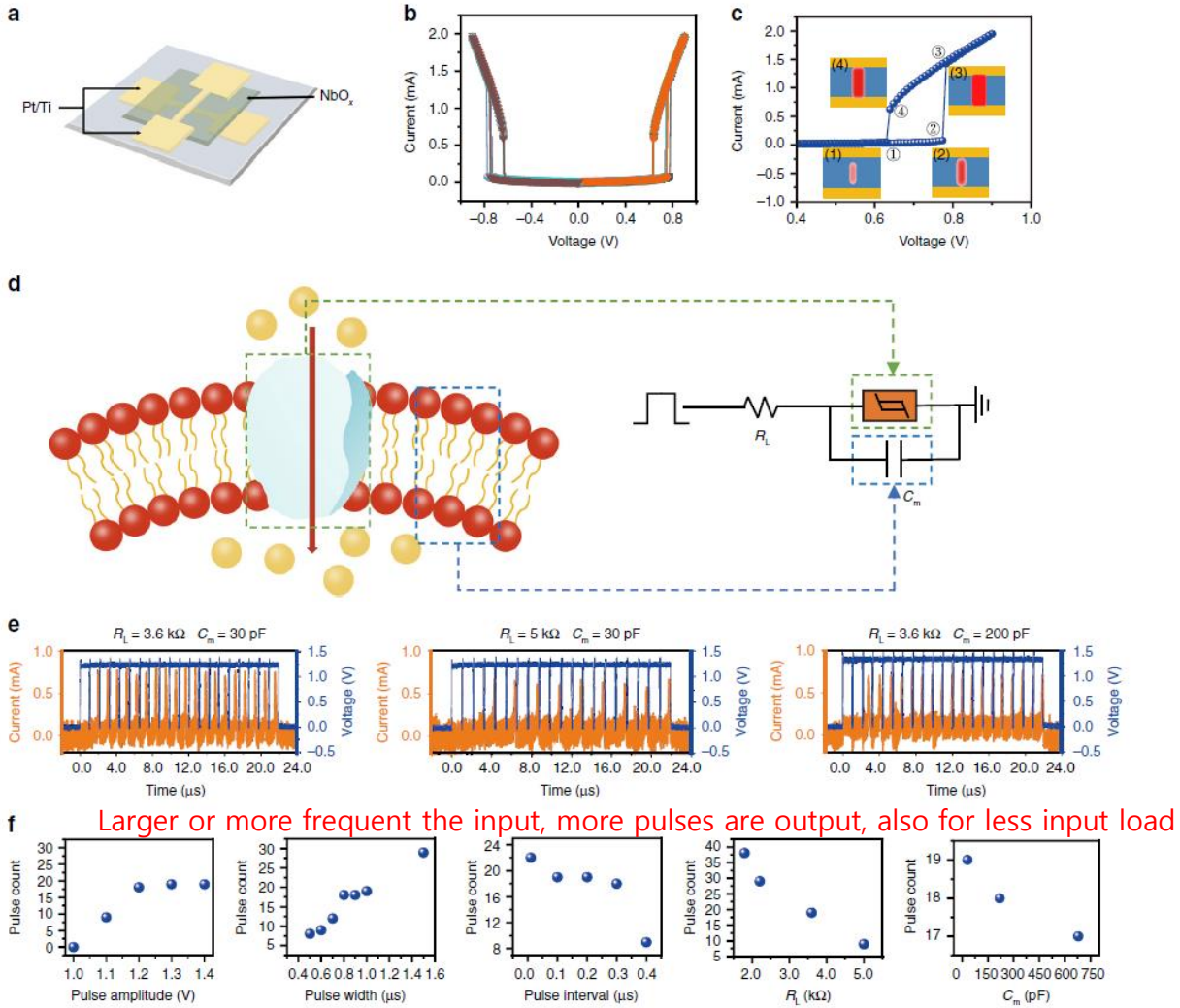


Fig. 2 Spiking neuron based on NbO_x volatile memristors. **a** Schematic diagram of the memristive device, which consists of a NbO_x layer sandwiched between two Pt/Ti electrodes. **b** Current-voltage characteristics of the device repeated for 10 cycles. **c** Schematic illustration of the mechanism responsible for threshold switching in the NbO_x devices. **d** Illustration of an ion channel embedded in cell membrane of a biological neuron, with corresponding circuit based on NbO_x devices for implementation of a spiking neuron. **e** Characterization of the LIF neuron under a continuous pulse train and the influence of varying capacitance (C_m) and resistance (R_L). **f** The firing response of the spiking neuron under different input and circuit conditions. When changing the input parameters such as the pulse width, pulse interval and circuit parameters such as C_m and R_L, the firing pulse count has an obvious change with 20 input pulse cycles.

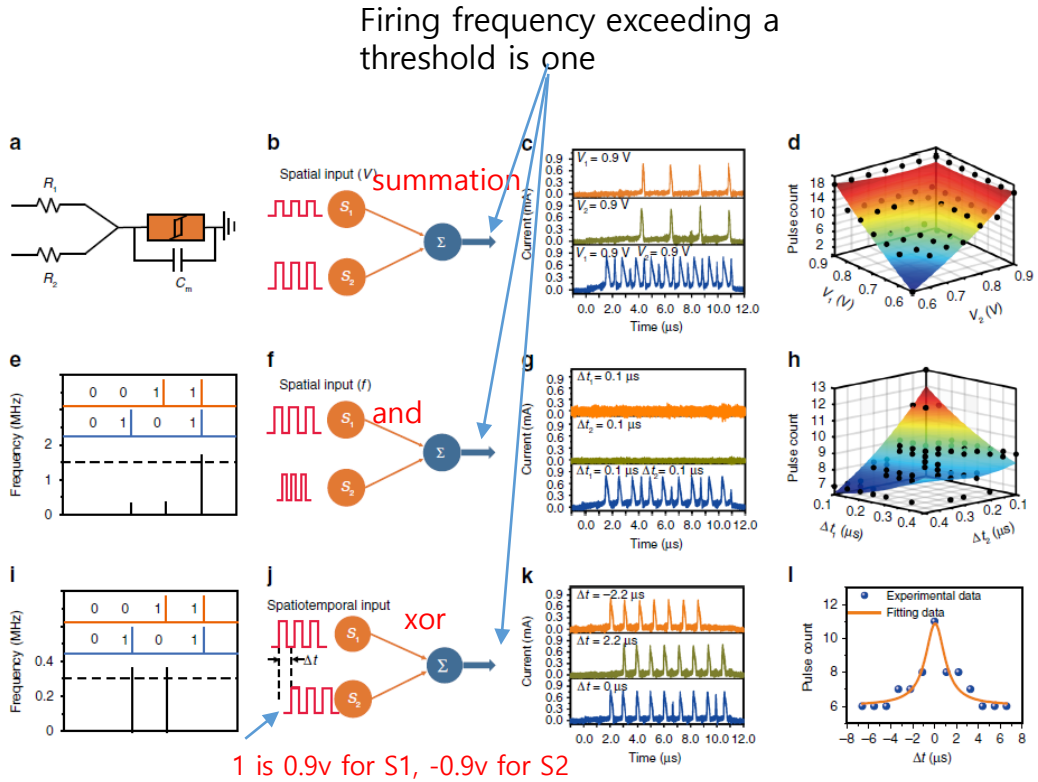
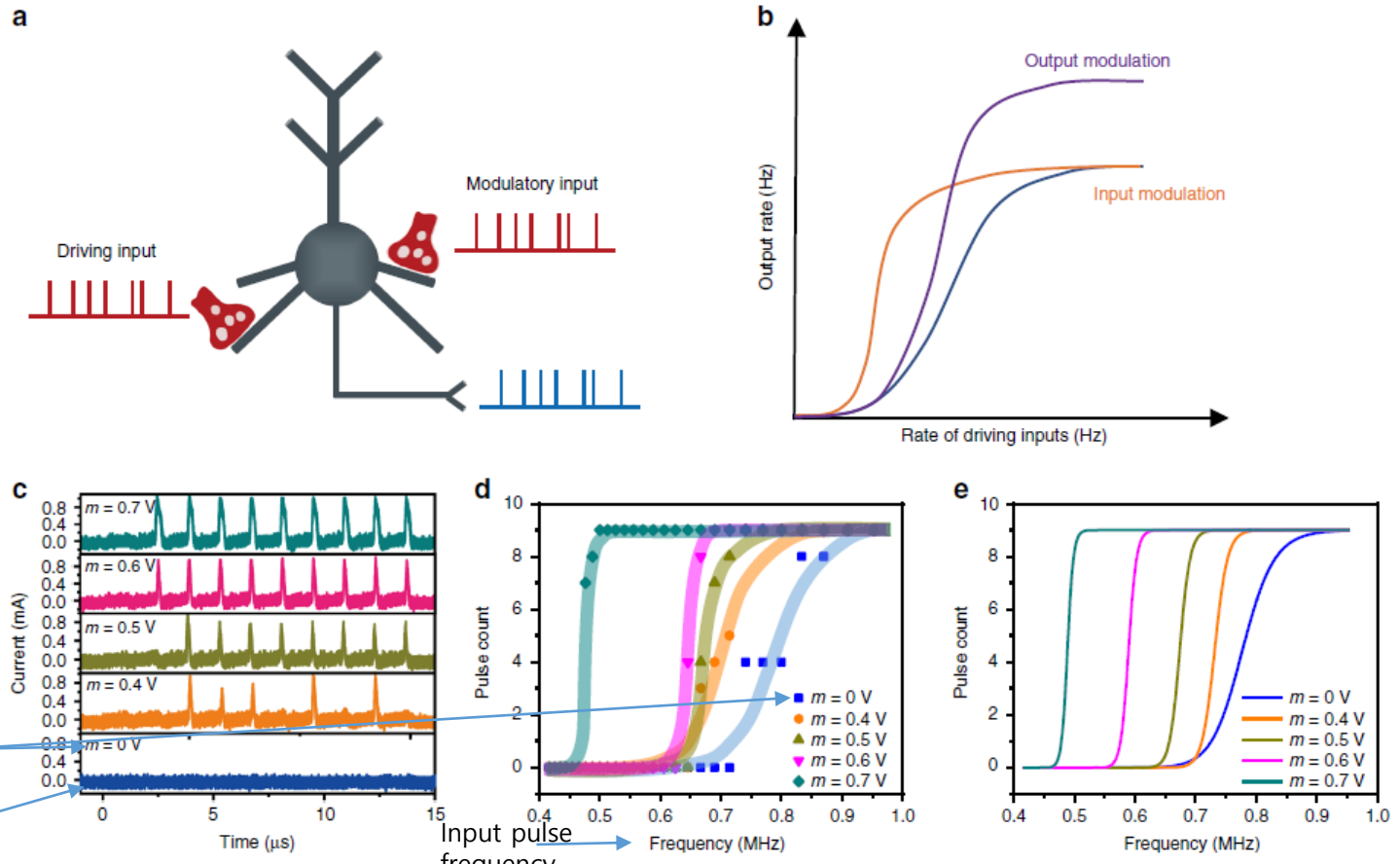


Fig. 3 Spatiotemporal integration and dynamic logic in NbO_x based neuron. **a** The circuit diagram of the spiking neuron. **b** Schematic diagram of spatial summation with different input pulse amplitudes. **c** Neuronal response triggered by two input pulses (0.9 V amplitude, 1 μs width, interval 0.1 μs, 10 pulse cycles) applied on S_1 and S_2 individually and simultaneously. **d** The spatial summation at varied conditions and corresponding fitting results. The amplitudes of the two input spikes (V_1 and V_2) were systematically changed from 0.6 to 0.9 V, and applied simultaneously to the respective nodes. **e** The input-output characteristics for "AND" logic with two input pulses applied on S_1 and S_2. **f** Schematic diagram of spatial summation with different input pulse intervals. **g** Neuronal response triggered by two input pulses (0.8 V amplitude, 1 μs width, interval 0.1 μs, 10 pulse cycles) applied on S_1 and S_2 individually and simultaneously. **h** The spatial summation at varied conditions and corresponding fitting results. The intervals of the two input spikes were systematically changed from 0.05 to 0.4 μs. **i** The input-output characteristics for "XOR" logic with two input pulses applied on S_1 and S_2. **j** Schematic diagram of spatiotemporal summation with different time intervals of input pulses. **k** Neuronal response triggered by two input pulses (0.9 V amplitude, 1 μs width, 0.1 μs interval) applied on S_1 and S_2 with different time intervals of 0, 2.2 and −2.2 μs. **l** Spatiotemporal summation results when the time interval of two input spikes is changed from −6.6 to 6.6 μs. The solid line is fitting result by Eq. (1).



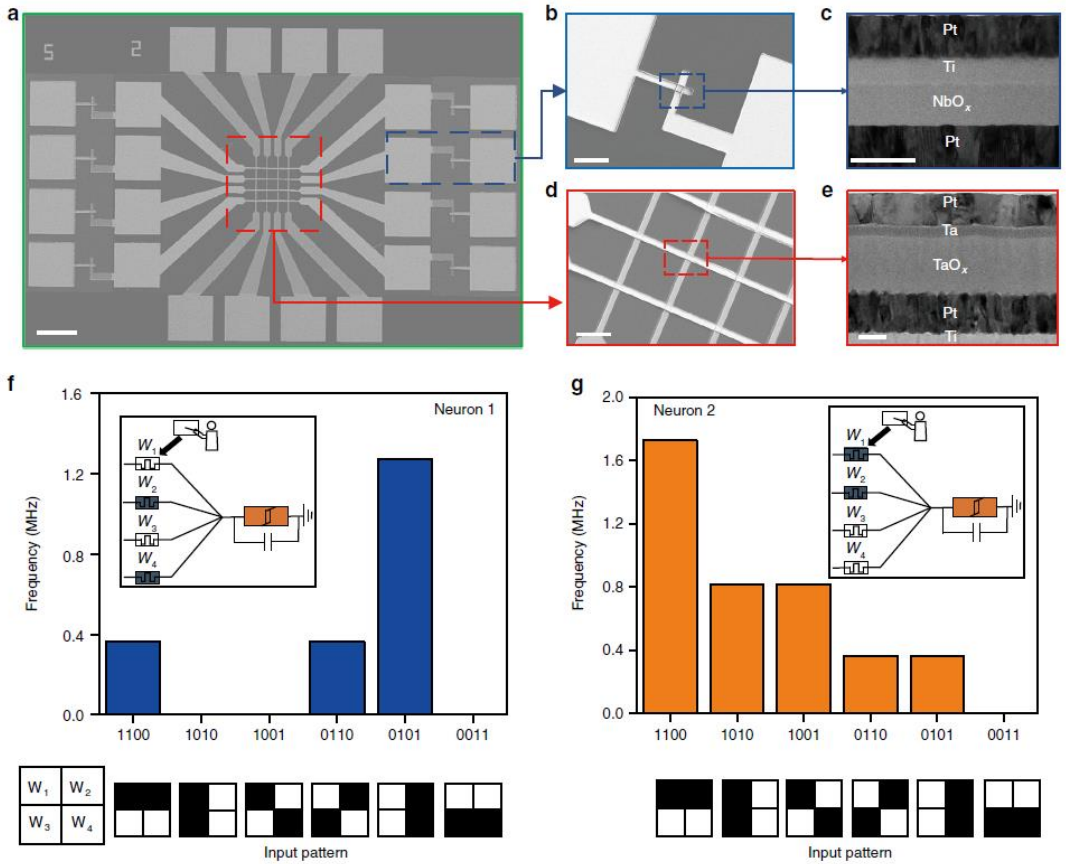
Voltage of modulatory input pulses with the same period as the input

Corresponding output when 10 input pulses of 1.2v are applied

Input pulse frequency

Fig. 4 Neuronal gain modulation of NbO_x based neuron. a Schematic diagram of the rate-coded neuronal signaling. The driving input and modulatory input affect output firing rate jointly. **b** Schematic diagram of neuronal gain modulation by modulatory inputs, including input modulation (orange line) and output modulation (purple line). The blue line is the *I*–*O* curve with only the driving input. **c** Neuronal responses triggered by only the driving input (1.2 V amplitude, 1 μs width, 0.4 μs interval) applied on S_1 (bottom), and triggered by the driving input (1.2 V amplitude) and modulatory input (1 μs width, 0.4 μs interval) with different amplitudes of 0.4 V, 0.5 V, 0.6 V and 0.7 V, applied simultaneously to nodes S_1 and S_2, as shown in the middle and top of the panel. **d** Experimental neuron responses showing multiplicative gain modulation when applying different modulatory inputs ($m = 0$, 0.4, 0.5, 0.6 and 0.7 V). The background lines indicate the overall trend in data. **e** Fitting lines calculated by Eq. (2).

Prototype test array



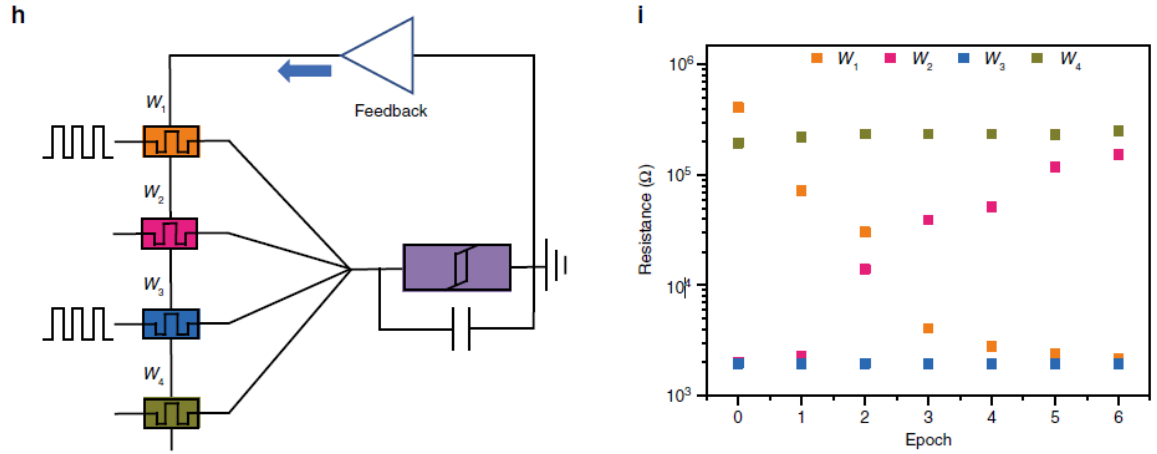
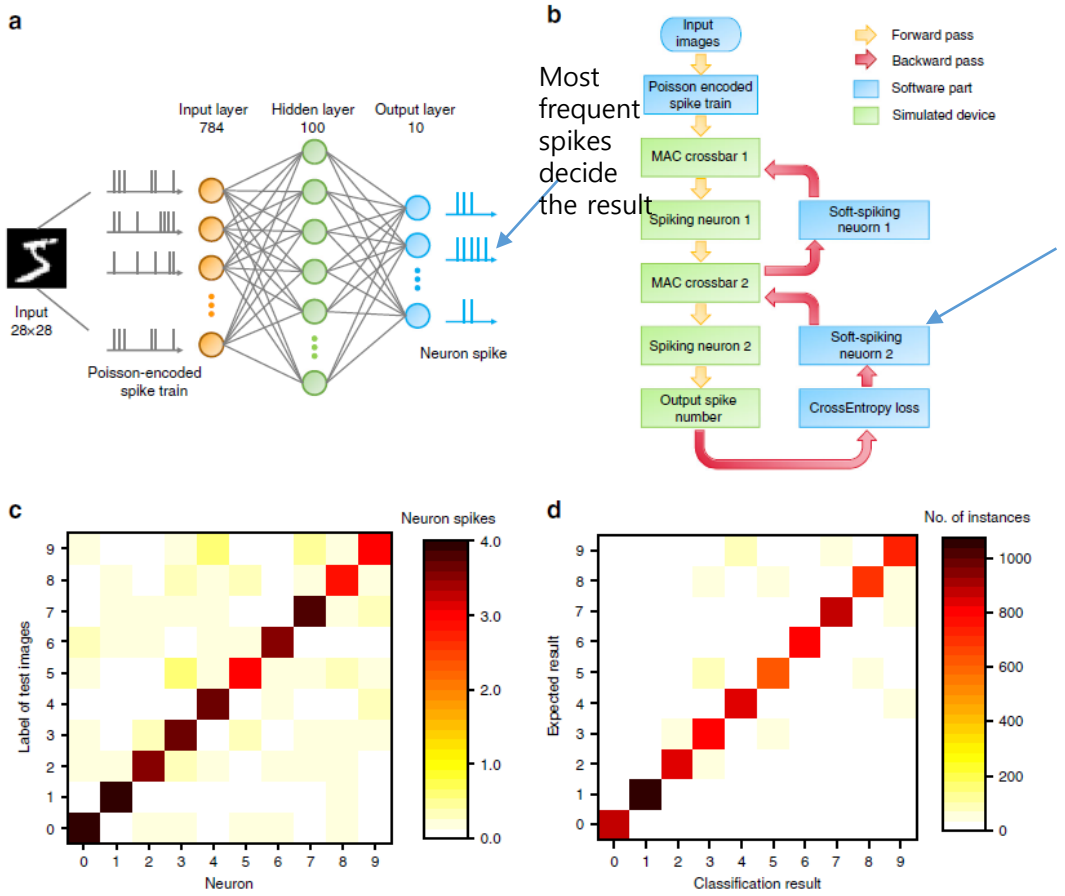


Fig. 5 Fully memristive spiking neural network. a Scanning electron microscopy (SEM) image of the monolithically integrated memristive neural network. Scale bar, 100 μm. **b** SEM image of the Pt/Ti/NbO_x/Pt threshold switching device. Scale bar, 20 μm. **c** Cross-sectional transmission electron microscopy (TEM) image of the Pt/Ti/NbO_x/Pt threshold switching device. Scale bar, 50 nm. **d** SEM image of a Pt/Ta/TaO_x/Pt synapse device. Scale bar, 10 μm. **e** Cross-sectional TEM image of the Pt/Ta/TaO_x/Pt device. Scale bar, 20 nm. **f**, **g** Neuronal outputs when presented with different input patterns. **f** Neuron 1 learnt to recognize pattern "0101", while neuron 2 learnt to recognize pattern "1100" **g**. The input pulses are 1 V in amplitude, 1 μs in width, 0.1 μs in interval and repeated for 10 cycles. **h** Schematic illustration of supervised learning in a fully memristive neural network. **i** Evolutions of synaptic weights over time during online learning, where the pattern "1010" is repeatedly applied, with an initial state of "0110".

Supervised online learning: the output is compared against the expected output spikes until the difference is minimized.

Inferencing: the weights are learned offline

Simulations using experimental data obtained from the small test array



Most frequent spikes decide the result

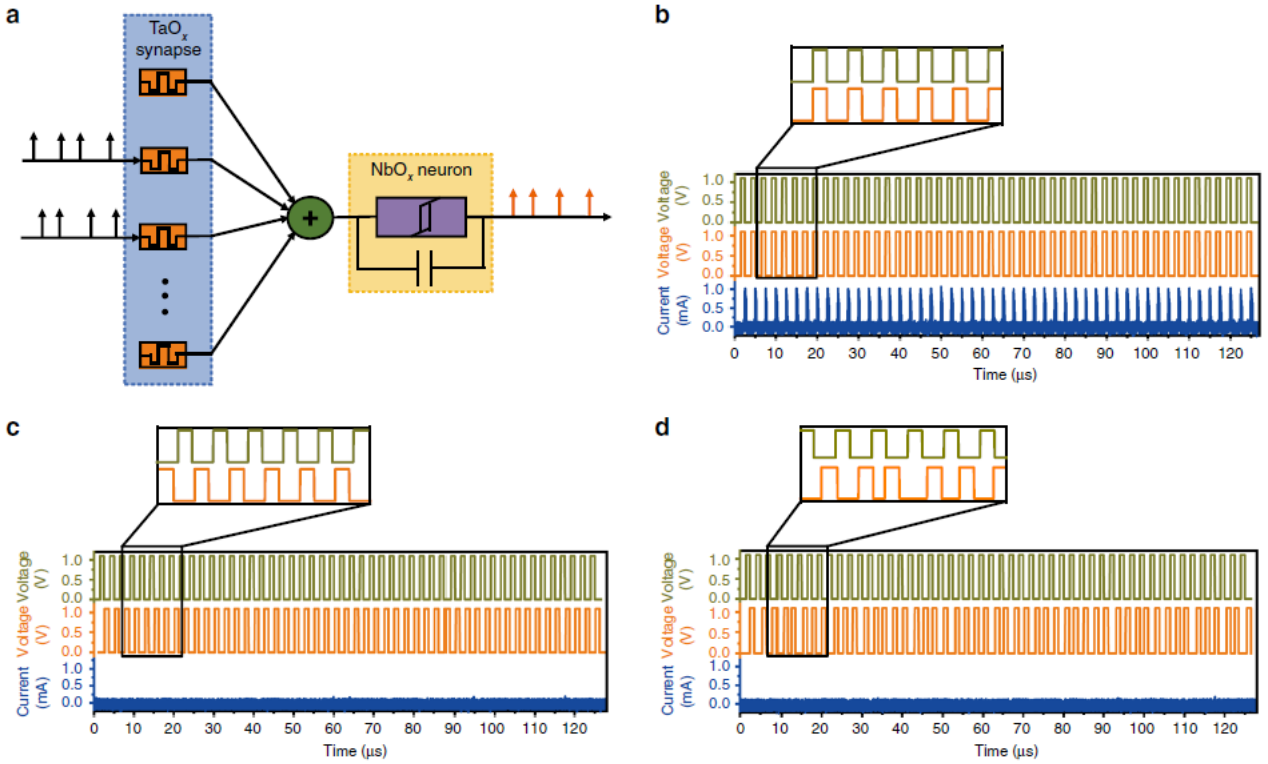
Used backpropagation for online learning

Activation function is approximated by sigmoid for back propagation (it is in reality a step function)

83.24% accuracy achieved for MNIST test dataset

Fig. 6 A 3-layer spiking neural network by simulation. a Schematic of the spiking neural network for MNIST classification. Input images are first converted to Poisson spike trains, where the spiking rates are proportional to pixel values. The spike trains are then input to the network, weighted and integrated by the neurons. At last, we count the spiking rates of output neurons to get the prediction results. **b** Flow chart of the simulation process. In the forward phase (yellow arrow), input spike trains are weighted by the memristor crossbar and then integrated on neurons, and the spiking rates of output neurons are used in loss computing. In the backward phase (purple arrow), a soft function like sigmoid is used as an alternative to neuron spike function (step function) in gradient computing. The computing units in green boxes are simulated based on experimental data, while the units in blue boxes are implemented by software. **c** In the case where the input picture is correctly identified, the statistics of the firing numbers issued by ten neurons in the case of input pictures corresponding to the MNIST numbers themselves. Statistics of firing numbers of the category neurons in response to different inputs, showing that the inputs are classified correctly in most cases. **d** Averaged confusion matrix of the testing results, showing that the test inputs are well classified after training.

Spike train methods are good for detecting coincidence



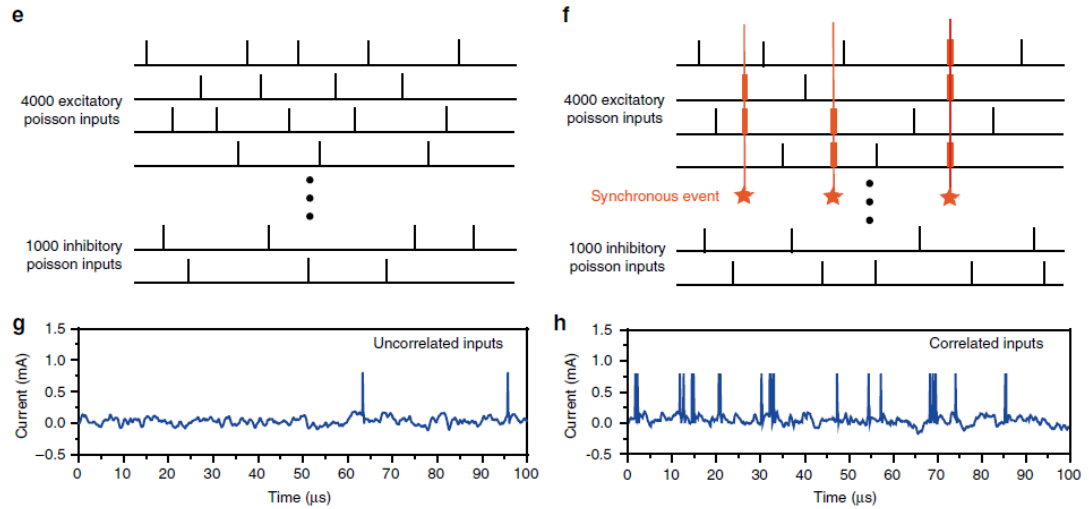


Fig. 7 Coincidence detection based on spatiotemporal dynamics. a Schematic diagram of coincidence detection in the fully memristive neural network. **b** The neuronal response by two synchronous input pulse trains (1.1 V in amplitude, 1 μs in width, 1.5 μs in interval, for 50 cycles) applied on S_1 and S_2. **c** The neuronal response by two asynchronous input pulse trains (1.1 V in amplitude, 1 μs in width, 1.5 μs in interval, for 50 cycles) applied on S_1 and S_2, where S_2 is behind S_1 by 1.2 μs. **d** The neuronal response by two asynchronous input pulse trains (1.1 V in amplitude, 1 μs in width, for 50 cycles) applied on S_1 and S_2, where S_1 has and interval of 1.5 μs and S_2 is random relative to S_2 in timing. **e** Input pulse trains of the neuron from independent 4000 excitatory and 1000 inhibitory random spike trains following Poisson statistics. **f** Introduction of synchronous events following Poisson statistics into the excitatory inputs, where the input rates are unchanged and the proportion of synchronous events is 0.3%. **g** Simulated neuronal response as a result of the inputs shown in **e**, where the artificial neuron only fires 2 spikes. **h** Simulated neuronal response as a result of the inputs shown in **f**, where the artificial neuron fires 21 spikes.

Modulation signal allows transmission of pulses: They try to say that this is similar to receptive field remapping

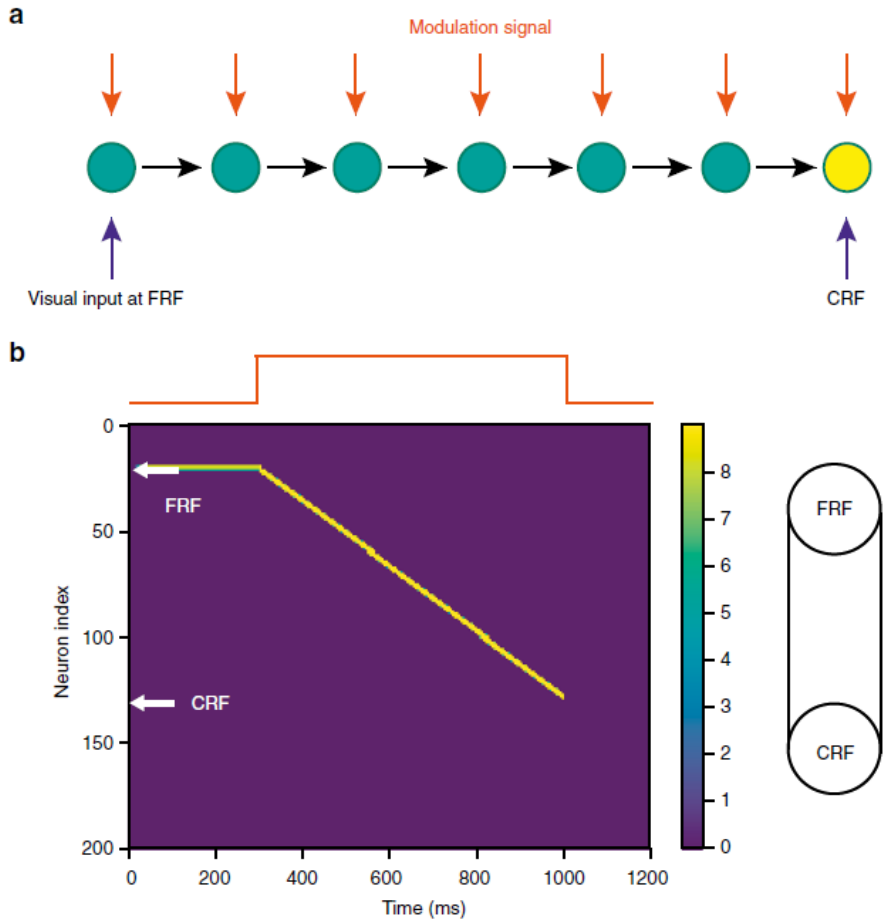
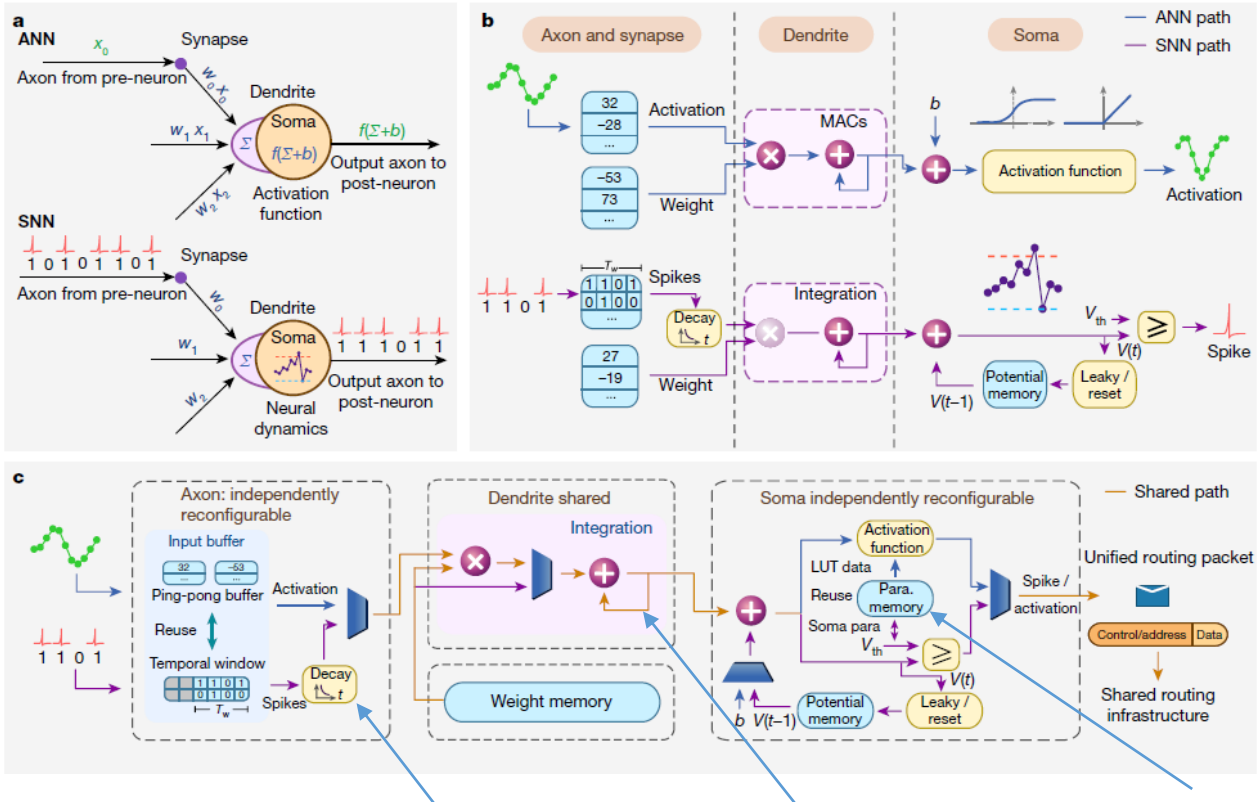


Fig. 8 A network of memristive neurons with gain modulation for receptive field remapping. a A one-dimensional network where neurons are connected in an uni-direction. Each neuron receives a modulatory signal (orange arrow). A visual input is applied at the future receptive field (FRF) of a neuron (the yellow one). **b** The visual input at FRF and the modulatory signal (top, red line) triggers a propagation of neural activity from FRF to current receptive field of the yellow neuron, as if the receptive field of the yellow neuron is temporally expanded when the modulatory signal is applied, achieving RF remapping (right panel). The distance of remapping is controlled by the duration of the modulatory signal. In the simulation, the visual input is applied during 0–300 ms, while the modulatory signal $m = 0.8$ is applied during 300–1000 ms. $\theta_{thr} = 0.41$, $W_{i,i-1} = 0.095$, $\tau = 10$ ms, $dt = 1$ ms, $I_{ext,0} = 0.09$.

Towards artificial general intelligence with hybrid Tianjic chip architecture

Interneuron cooperation, interFCore hierarchical accumulation, intraFCore/interFCore neuron copy, interFCore multicast routing

Allows diverse fan-in, fan-out capability



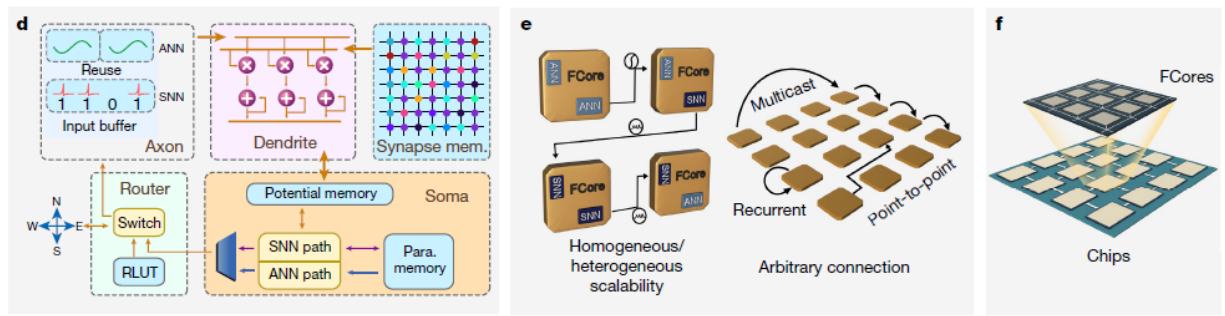


Fig. 2 | Design of the Tianjic chip. **a**, Computational model of an ANN or biologically inspired (for example, SNN) neuron. w_0, w_1, w_2 are synaptic weights; x_0, x_1, x_2 are input activations; Σ is the dendritic integration; f is the activation function; and b is the bias. **b**, Implementation diagrams for an ANN or SNN neuron. $V(t)$ is the neuronal membrane potential at time step t, and V_{th} is the firing threshold. Numbers in blue boxes are examples of input activation/spike and weight values. The faded purple multiplier in the SNN path indicates that the dendrite can possibly bypass the multiplication (for example, in the case that the time window length equals one). **c**, Diagram of a hybrid circuit, showing a cross-paradigm neuron with fused ANN and SNN components. Para. memory, parameter memory. **d**, Diagram of a unified functional core (FCore). Each FCore includes axon, synapse, dendrite, soma and router building blocks. Synapse mem., synapse memory. **e**, Flexible modelling configuration and connection topology of FCores. The coding schemes can be freely transformed between ANN and SNN modes, enabling heterogeneous neural networks. The scheme also allows flexible connections for the implementation of arbitrary network topology. **f**, Illustration of the hierarchy of 2D mesh architecture at the core and chip levels, demonstrating the ability to scale up the technique. RLUT, routing look-up table.

Used for CANN/

16 MACs per dendrite

LUT for activation function and threshold

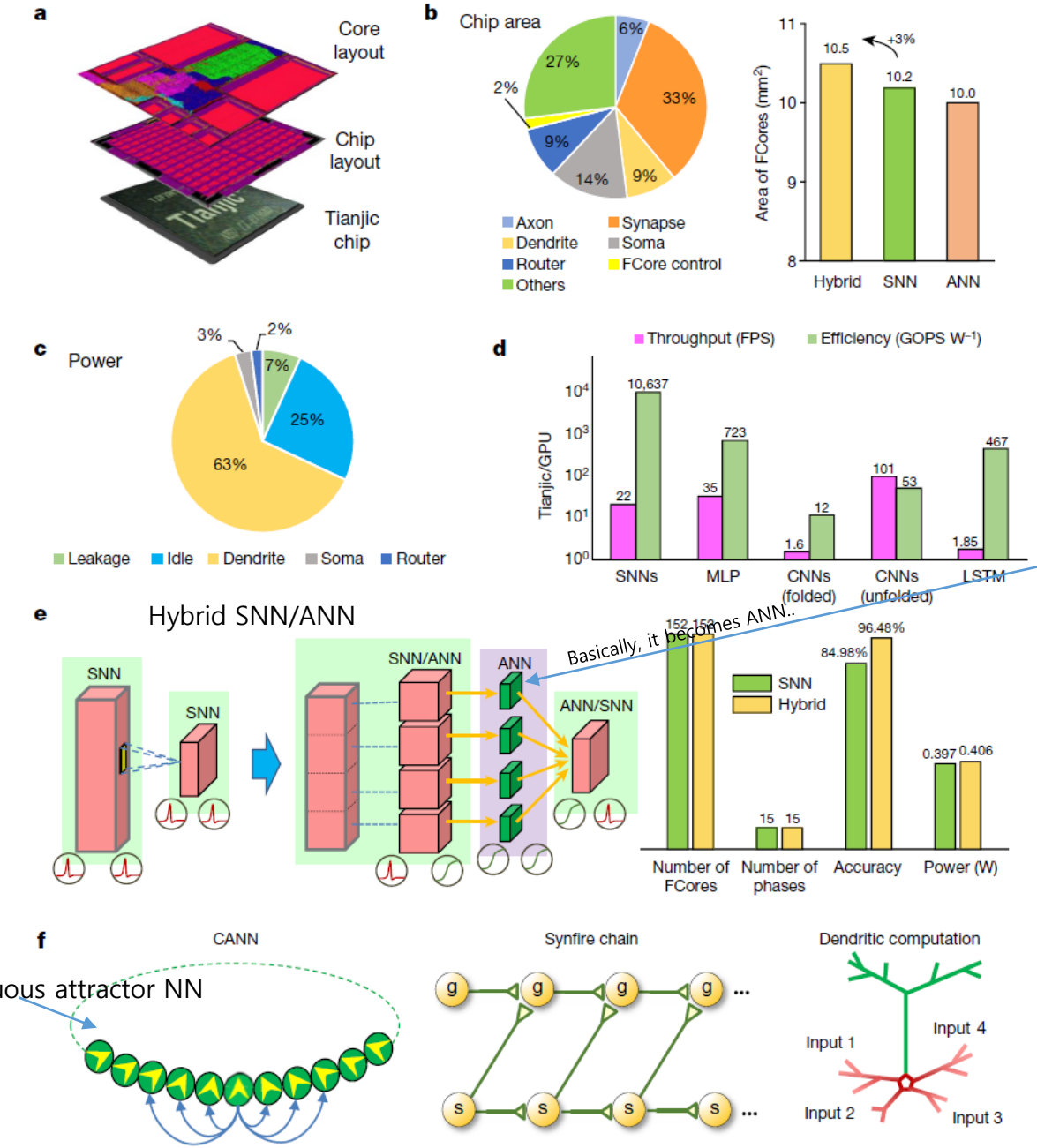
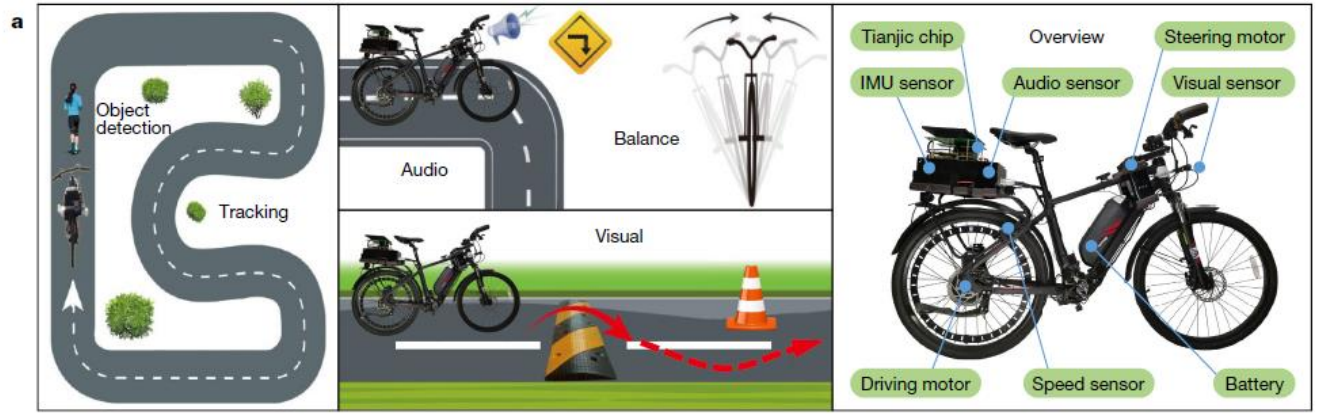


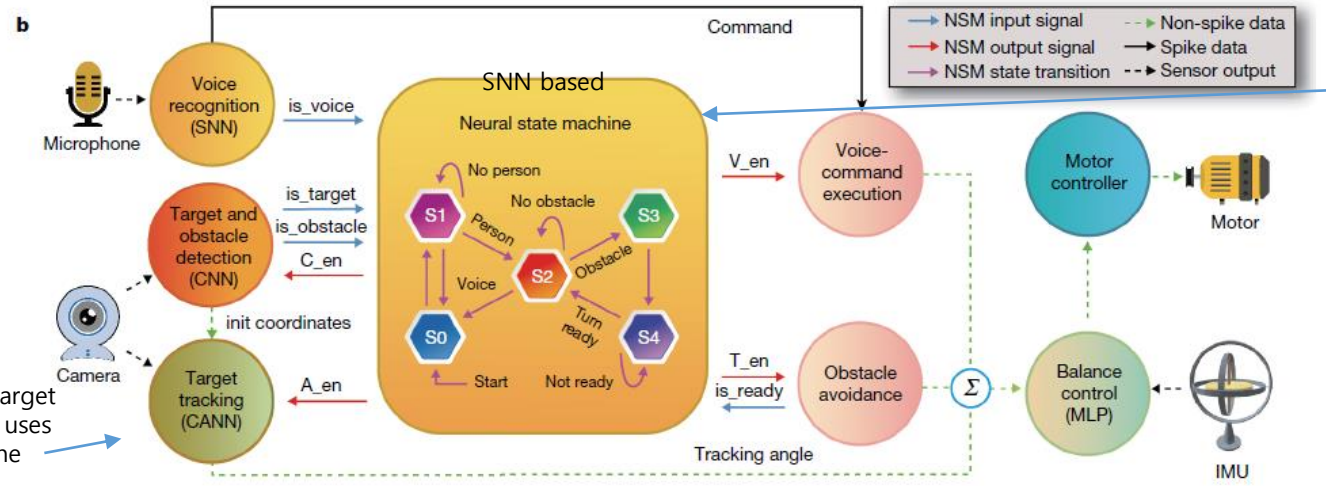
Fig. 3 | Summary of chip evaluation and modelling. **a**, Integrated layout and packaging of the Tianjic chip. **b**, Left, percentage of the chip area that is occupied by different features (axons, dendrites, routers and so on). Right, owing to the high level of resource sharing and reconfigurability, only a small area increase (roughly 3%) is needed to fuse the two paradigms. **c**, Power breakdown for FCore. **d**, Evaluation of FCore performance in various single-paradigm models, including SNNs, MLPs, CNNs (under folded or unfolded mapping) and long short-term memory networks (LSTMs). GOPS, giga operations per second; FPS, frames per second. **e**, Left, example of the implementation of a large-scale SNN with ANN dendritic relay. Right, with the help of ANN relays to transfer intermediate membrane potentials with high precision, a hybrid device was able to achieve higher recognition accuracy than an SNN alone, with negligible hardware overheads. **f**, The Tianjic chip can also support more biologically plausible neural network models (for example, CANN; a temporal-coding-based synchronous firing (synfire) neural chain; and dendritic multicompartment models). g, graded; s, synfire.

3.8x3.8 mm^2
28nm Technology
156 Fcores
40,000 Neurons
10,000,000 synapses



CNN, CANN, SNN and MLP networks are pretrained

Neural state machine outputs enabling signals and action signals



Human target tracking: uses membrane potential normalization

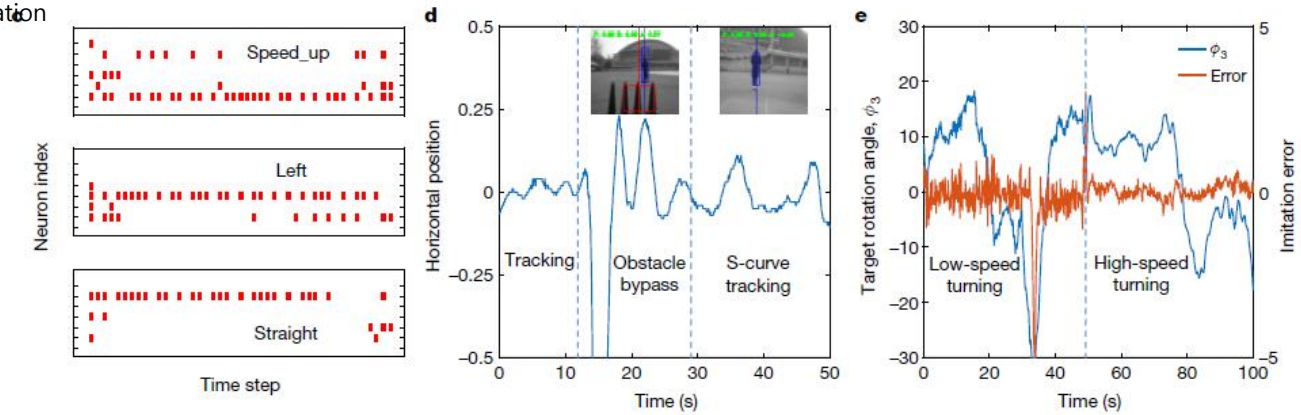


Fig. 4 | Demonstration of multimodal integration on a Tianjic chip for an unmanned bicycle. **a**, Left and centre, illustration of the tasks conducted in the bicycle experiment, including real-time object detection, tracking, voice perception, riding over a speed bump, automatic obstacle avoidance and balance of attitude. Right, the bicycle was equipped with a camera, gyroscope, speedometer, motors and a Tianjic chip. IMU, inertial measurement unit. **b**, Diagram of the multiple neural networks used in the unmanned bicycle experiment. The states inside the NSM diagram were defined as: voice command execution (S0), human detection (S1), human tracking (S2), a start of obstacle avoidance (S3), and a wait of avoidance completion (S4). An init coordinate is an initialization coordinate. C_en, A_en, V_en and T_en denote enabling signals for CNN, CANN, voice control and turning control, respectively. **c**, SNN voice-command-recognition test. The neuron producing the most spikes indicates the resulting classification. **d**, Tracking test. The *y* axis shows the relative horizontal position of the human in the frame. The bicycle automatically avoided an obstacle, then followed an instructor who ran in an S-curve route. **e**, Balance control and turning by an MLP network, which was trained by imitating the outputs of several well tuned controllers using the proportional-integral-derivative algorithm at different speeds (low to high).