# In-memory computing: Memory devices and applications

**Abu Sebastian**

Distinguished Research Staff Member

IBM Research Europe



# IBM Research – 75 Years

**3,000**
Researchers

**19**
Locations

**6**
Continents




**6**
Nobel Laureates

**10**
Medals of Technology

**5**
National Medals of Science

**6**
Turing Awards

# Outline

- Introduction
- Memory devices and computational primitives
  - ✓Charge-based memory devices & Computational primitives
  - ✓Resistance-based memory devices & Computational primitives
  - ✓Phase change memory: A prototypical resistance-based memory
- Applications
  - ✓Exploiting non-volatile binary storage
  - ✓Scientific computing
  - ✓Signal processing & Optimization
  - ✓Deep learning
  - ✓Stochastic computing and security
- Discussion
  - ✓Increasing the precision of in-memory computing
  - ✓Photonic in-memory computing
  - ✓Summary

# Outline

- Introduction
- Memory devices and computational primitives
  - ✓Charge-based memory devices & Computational primitives
  - ✓Resistance-based memory devices & Computational primitives
  - ✓Phase change memory: A prototypical resistance-based memory
- Applications
  - ✓Exploiting non-volatile binary storage
  - ✓Scientific computing
  - ✓Signal processing & Optimization
  - ✓Deep learning
  - ✓Stochastic computing and security
- Discussion
  - ✓Increasing the precision of in-memory computing
  - ✓Photonic in-memory computing
  - ✓Summary

# Computer systems: Trends and opportunity

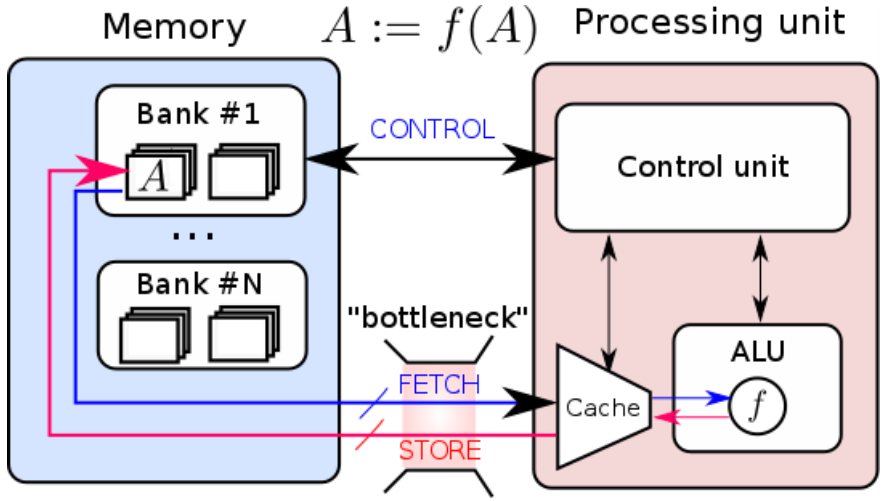
- Three key trends
  - ✓Data access is a major bottleneck
  - ✓Energy consumption is a key limiter
  - ✓Energy to move data dominates compute energy

- Opportunity
  - ✓Minimize data movement by performing computation directly (near) where the data resides
  - ✓Processing in memory (PIM)
    - In-memory computing
    - Near-memory computing/near data processing

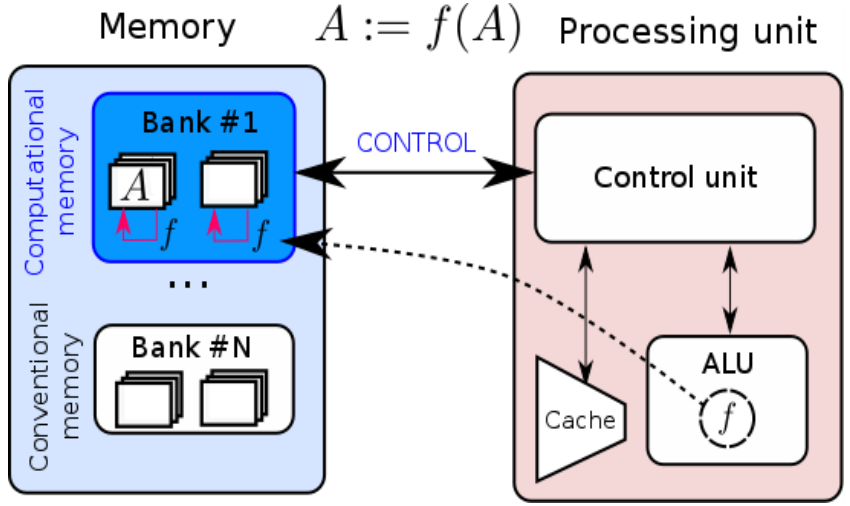
***Mutlu et al., Microprocess. Microsyst. (2019)***

# In-memory computing

## Processing unit & Conventional memory



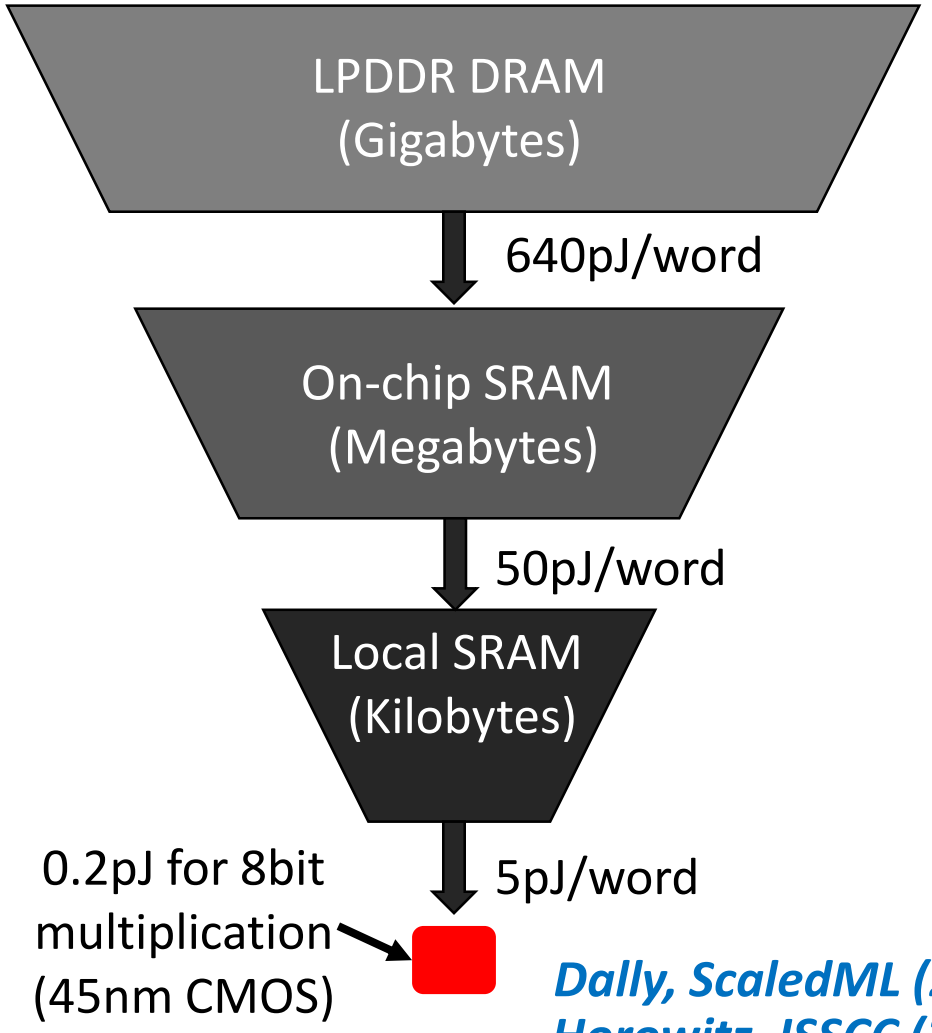

## Processing unit & Computational memory




- Perform “certain” computational tasks **in place in memory**
- Achieved by exploiting **the physical attributes of the memory devices**, their **array level organization**, the **peripheral circuitry** as well as the **control logic**
- At **no point during computation**, the memory content is read back and **processed at the granularity of a single memory element**

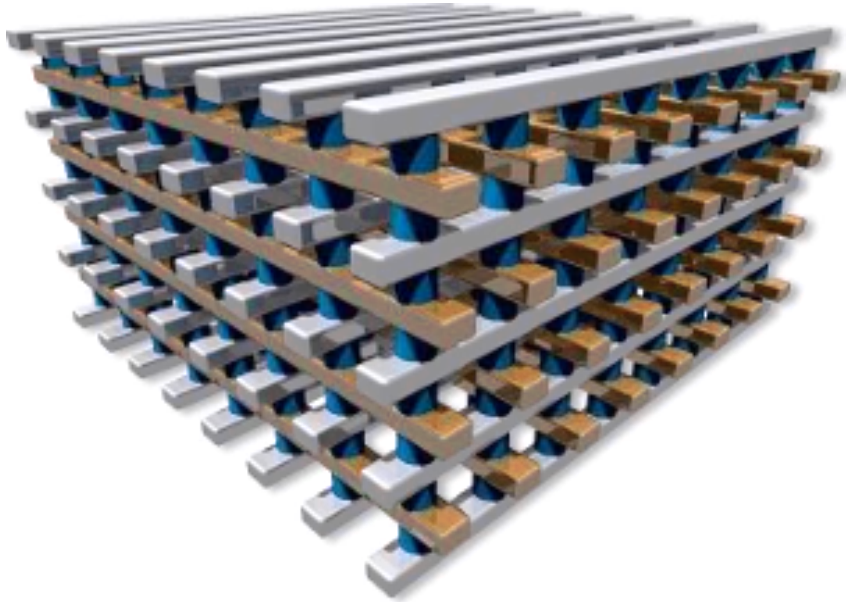
# Why in-memory computing?

## Reduce the cost of data motion




*Dally, ScaledML (2019)*
*Horowitz, ISSCC (2014)*

## Reduce computational time complexity



**Mostly from massive parallelism and analog way of computing**

**Additional complexity reduction from physical coupling**

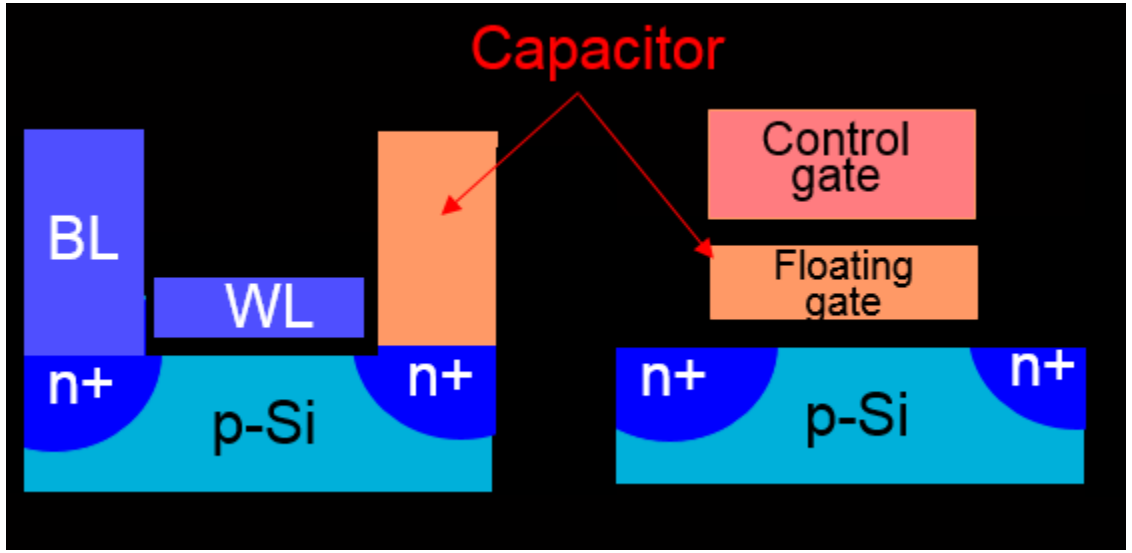
*Sebastian et al., Nature Comm. (2017)*
*Di Ventra, Nature Phys. (2013)*

# Outline

- Introduction
- Memory devices and computational primitives
  - ✓Charge-based memory devices & Computational primitives
  - ✓Resistance-based memory devices & Computational primitives
  - ✓Phase change memory: A prototypical resistance-based memory
- Applications
  - ✓Exploiting non-volatile binary storage
  - ✓Scientific computing
  - ✓Signal processing & Optimization
  - ✓Deep learning
  - ✓Stochastic computing and security
- Discussion
  - ✓Increasing the precision of in-memory computing
  - ✓Photonic in-memory computing
  - ✓Summary

# Constituent elements

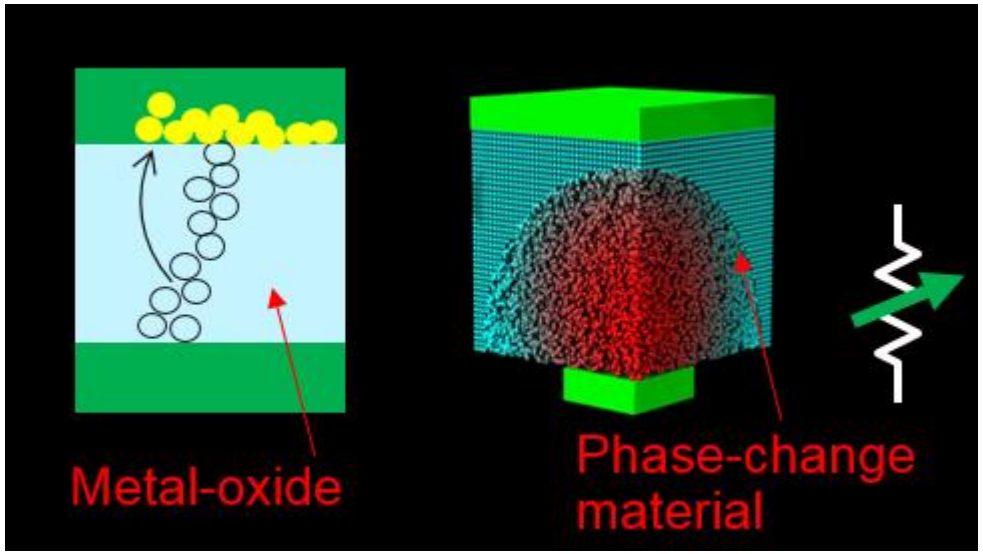
## Charge-based memory




**Coulomb's law**

*Aga et al., HPCA (2017) (SRAM)*
*Seshadri et al., MICRO (2017) (DRAM)*
*Merrikh-Bayat, IEEE TNNLS (2018) (Flash)*

## Resistance-based memory



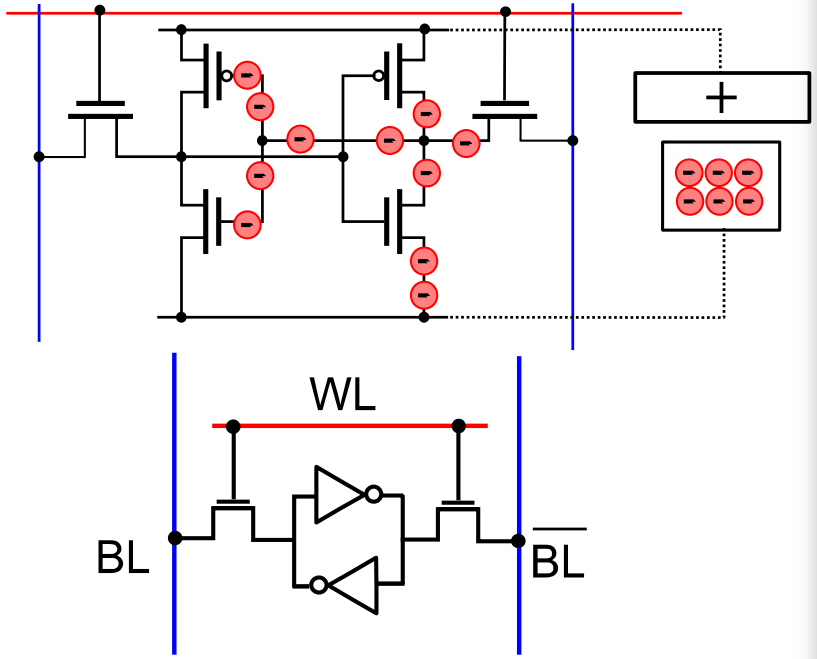

**Ohm's law**

*Burr et al., Adv. Phys. X (2017)*
*Sebastian et al., J. Appl. Phys. (2018)*
*Ielmini and Wong, Nature Electr. (2018)*

- **Charge-based memory:** Presence or absence of charge (eg. DRAM, SRAM, Flash)
- **Resistance-based memory:** Differences in atomic arrangements or orientation of ferromagnetic metal layers (eg. PCM, metal-oxide RRAM, STT-MRAM)
- Several **computational primitives realized by both types of memory**

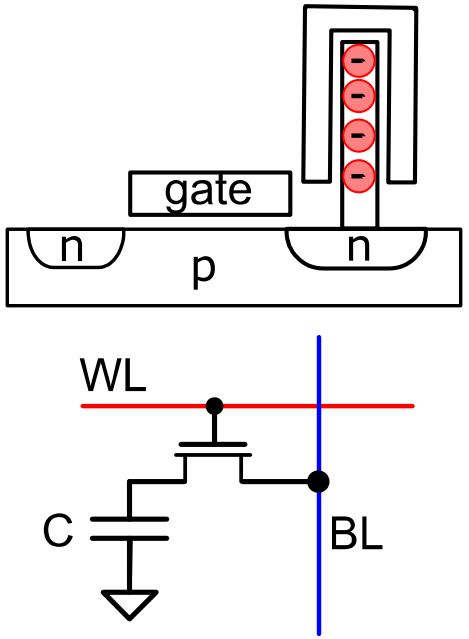
# Charge-based memory devices

## SRAM



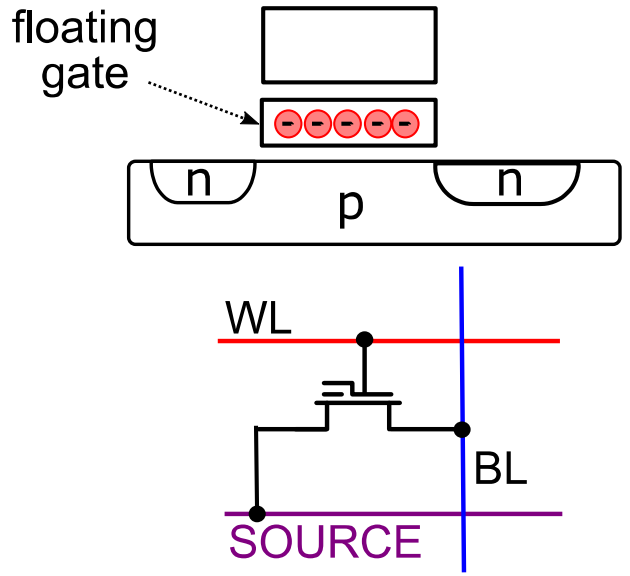

~ $10^3$ electrons
Access time < 1ns
Endurance > $10^{16}$

## DRAM




~ $10^5$ electrons
Access time < 10ns
Endurance > $10^{16}$

## Flash memory

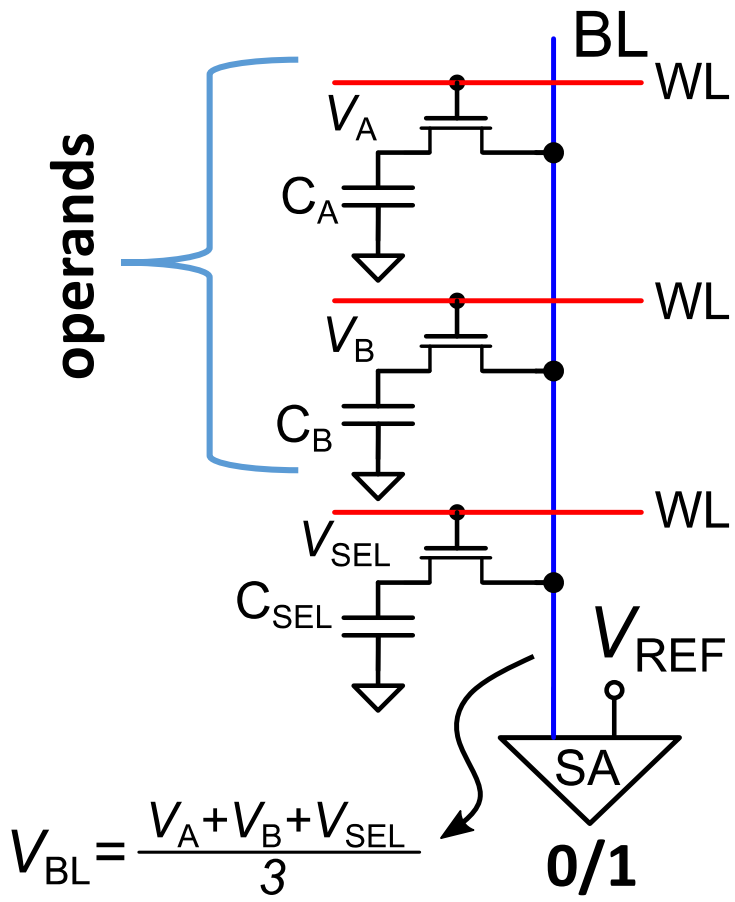


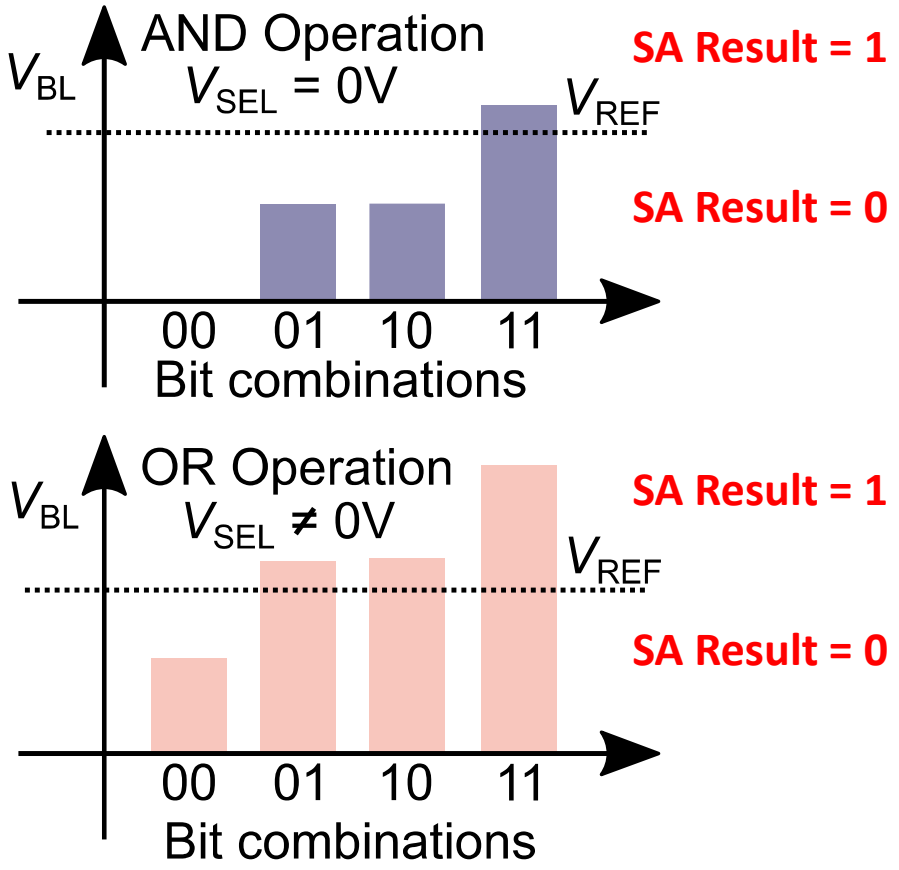

~ 100 electrons
Access time (read) < 100ns
Endurance < $10^5$

- **SRAM:** Two CMOS inverters connected back to back. The charge is confined within the barriers formed by FET channels and by gate insulators
- **DRAM:** Capacitor connected in series to a FET
- **Flash:** The storage node is coupled to the gate of a FET

***Emerging nanoelectronics devices, John Wiley & Sons (2015)***

# Logical operations using DRAM

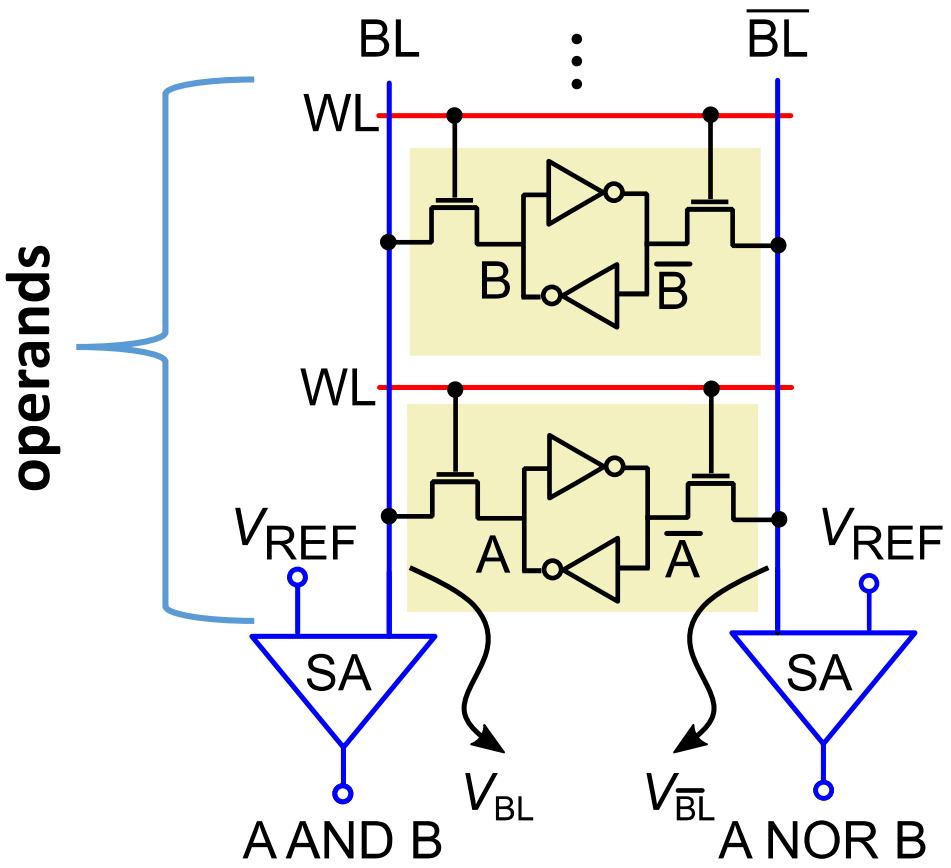


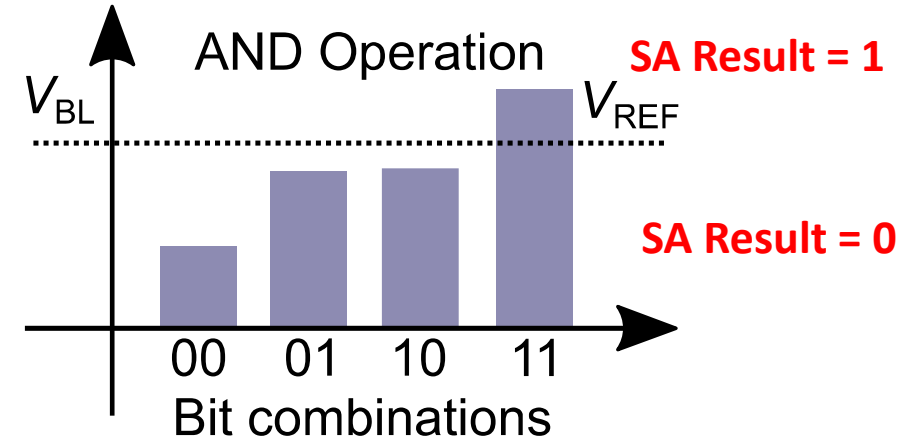



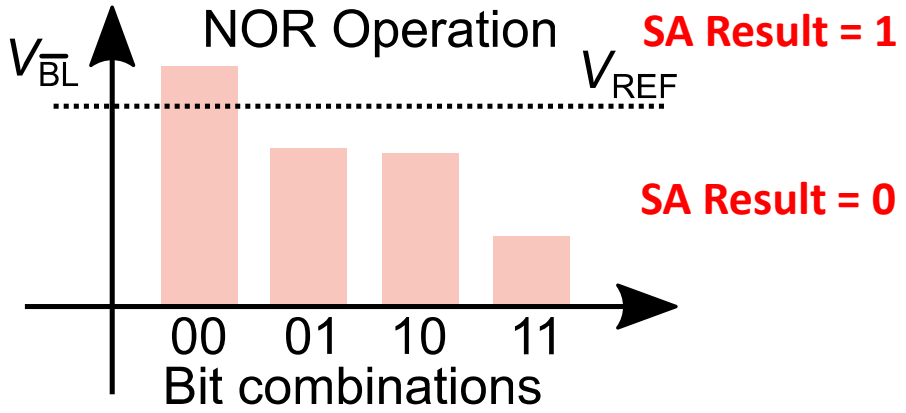

- Bitwise logical operations performed by **simultaneously activating WLs**
- Operands in cells A and B, **SEL is used to dictate whether AND or OR is realized**

*Sheshadri et al., MICRO (2017), Li et al., MICRO (2017)*

# Logical operations using SRAM




- BL and $\overline{BL}$ are **pre-charged to the supply voltage**
- **Both the WLs are activated** so that both BL and $\overline{BL}$ are discharged at different rates that depend on the data stored in the bit-cells

*Aga et al., HPCA (2017), Jeloka et al., JSSC (2016)*

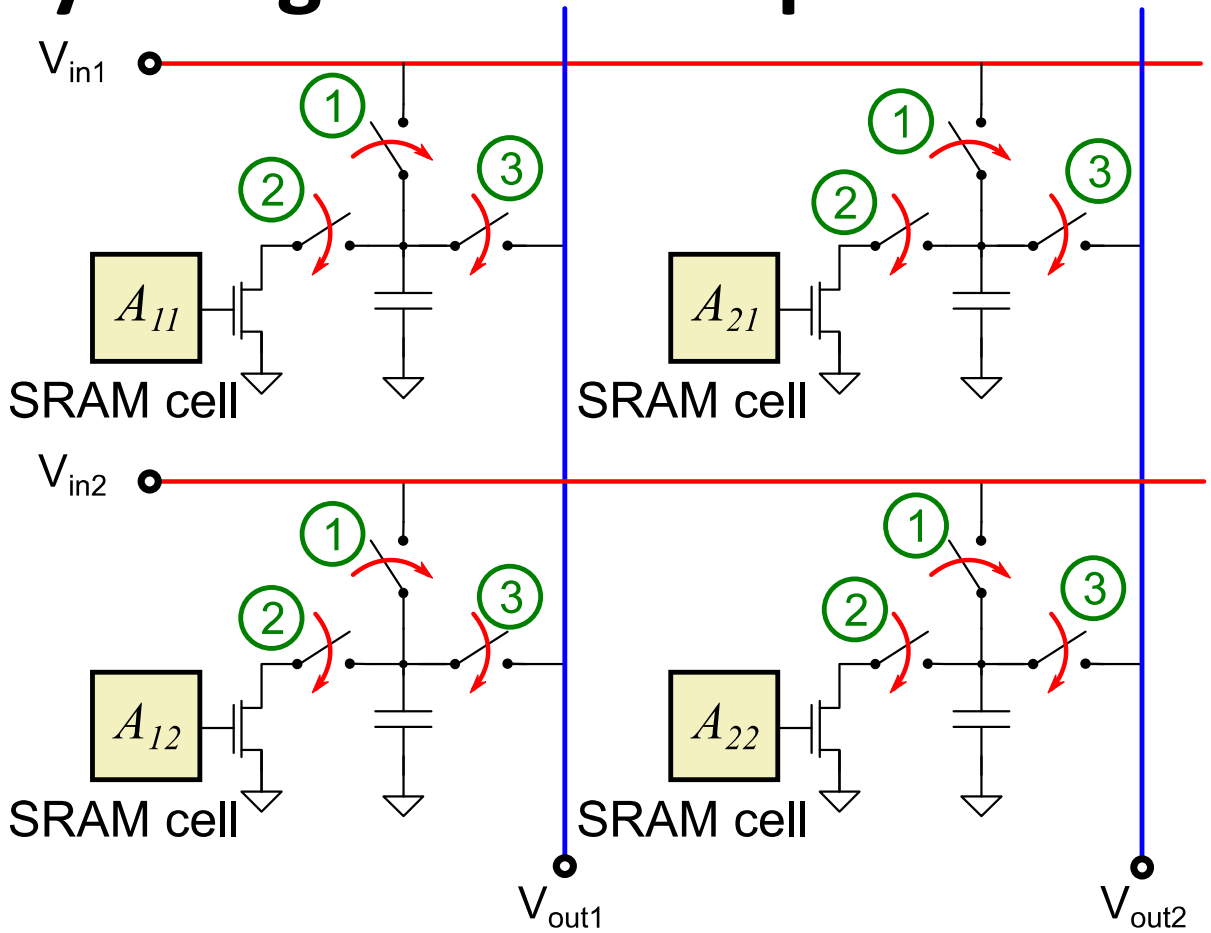
# Matrix-Vector Multiply using SRAM + Capacitor

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

MAP to SRAM content

MAP to cap voltage

DECIPHER from voltage along the BL




- SRAM cells used to store **the elements of a binary matrix**
- **Step 1:** Capacitors charged to input values
- **Step 2:** Capacitors associated with value 0 are discharged
- **Step 3:** Capacitors shorted along the columns

*Biswas et al., ISSCC (2018)*
*Valavi et al., JSSC (2019)*
*Khaddam-Aljameh, TVLSI (2020)*

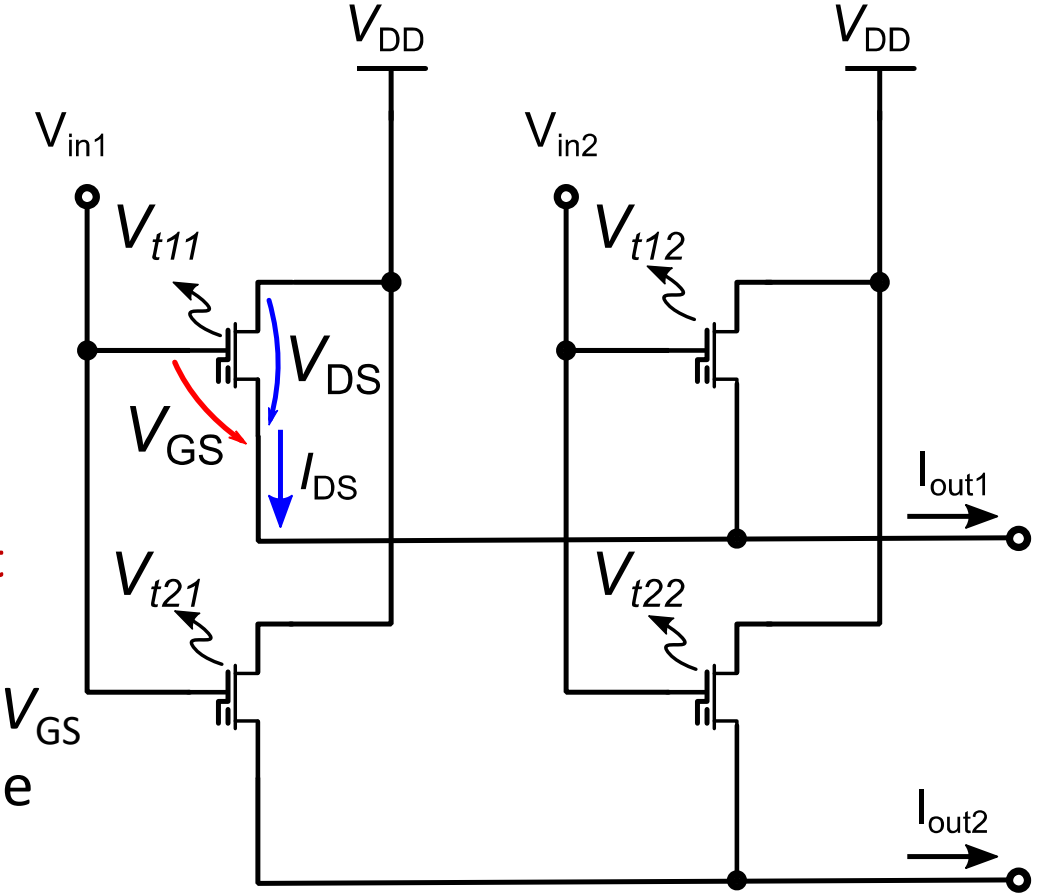
# MVM using Flash memory

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

MAP to charge on floating gate

MAP to gate voltage (binary)

DECIPHER from current



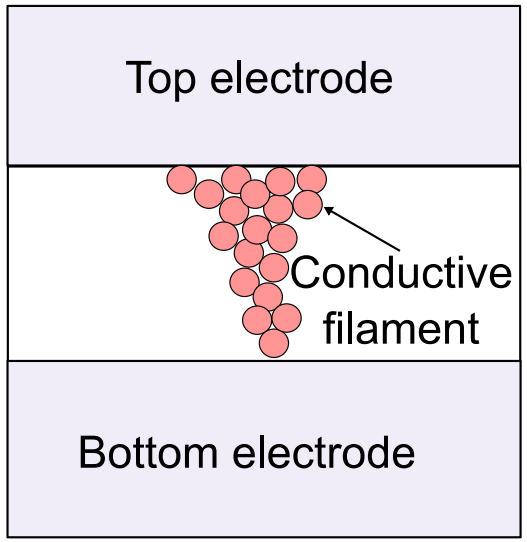

- The current $I_{DS}$ is a function of $V_t$, $V_{DS}$ and $V_{GS}$
- By fixing $V_{DS}$, Kirchhoff's current law can be employed to perform MVM
- **Matrix elements are stored in terms of $V_t$** and the **binary input vector is used to modulate $V_{GS}$**

*Diorio et al., IEEE TED, 43, 1972 (1996)*

*Merrikh-Bayat et al., IEEE Trans. Neural Networks and Learning Systems, 29, 4782 (2018)*

# Resistance-based memory devices

## ReRAM

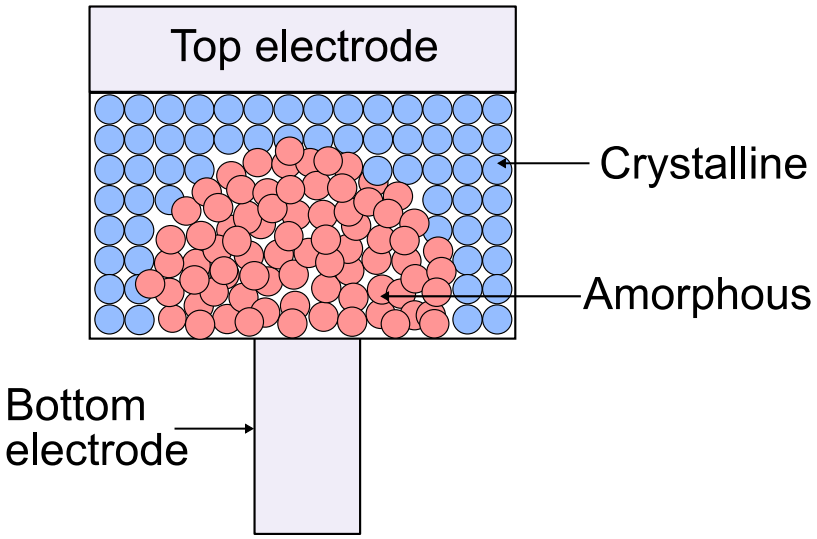



Resistance range = $10^3$-$10^7$

Access time (write) = 10ns - 100ns

Endurance = $10^6$-$10^9$

## PCM

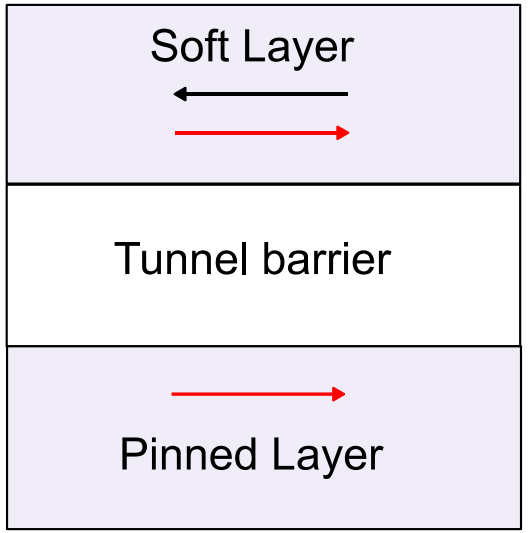



Resistance range = $10^4$-$10^7$

Access time (write) ~ 100ns

Endurance = $10^6$-$10^9$

## STT-MRAM




Resistance range = $10^3$-$10^4$

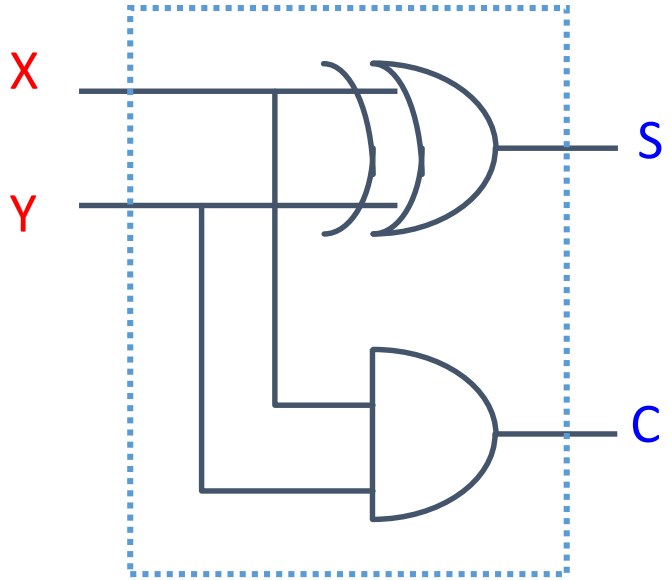
Access time (write) < 10ns

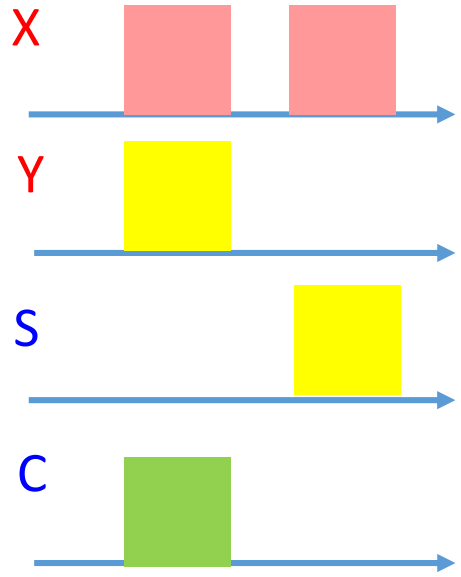
Endurance > $10^{14}$

- **ReRAM:** Migration of defects such as oxygen vacancies or metallic ions
- **PCM:** Joule-heating induced reversible phase transition
- **STT-MRAM:** Magnetic polarization of a free layer with respect to a pinned layer
- Resistance-based memory devices also referred to as **memristive devices**

***Wong and Salahuddin, Nature Nanotechnology (2015)***

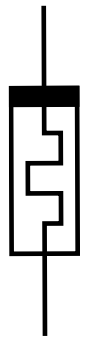
# Logic design using resistance-based memory devices




**Low conductance (Logic "0")**

**High conductance (Logic "1")**



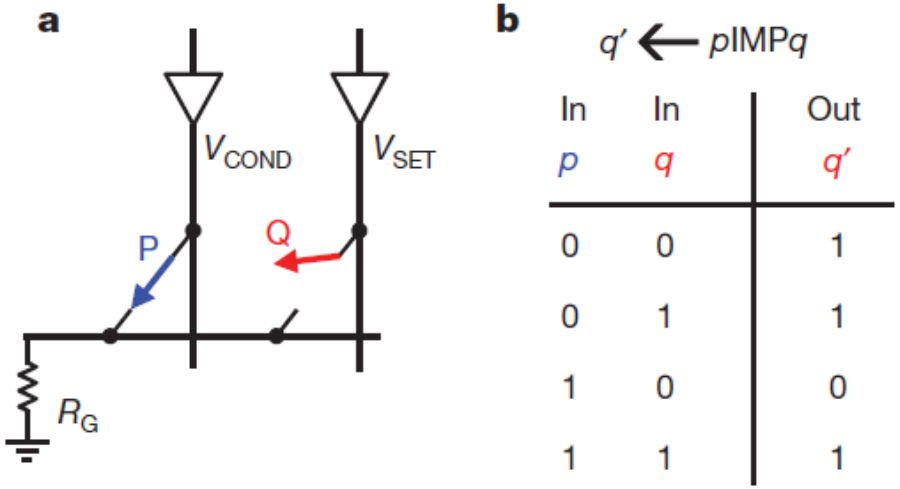
- Voltage serves as the sole logic state variable in conventional CMOS
- CMOS gates regenerate this state variable during computation
- How about using the resistance state of memristive devices as a logic state variable?
- Can toggle the states by applying voltage signals; only binary storage required
- **Logical operations enabled by the interaction between voltage and resistance state variables**

*Borghetti et al., Nature (2010)* *Vourkas, Sirakoulis, IEEE CAS Magazine (2017)*

# Stateful logic

**'Memristive' switches enable 'stateful' logic operations via material implication**

Julien Borghetti[1], Gregory S. Snider[1], Philip J. Kuekes[1], J. Joshua Yang[1], Duncan R. Stewart[1]† & R. Stanley Williams[1]

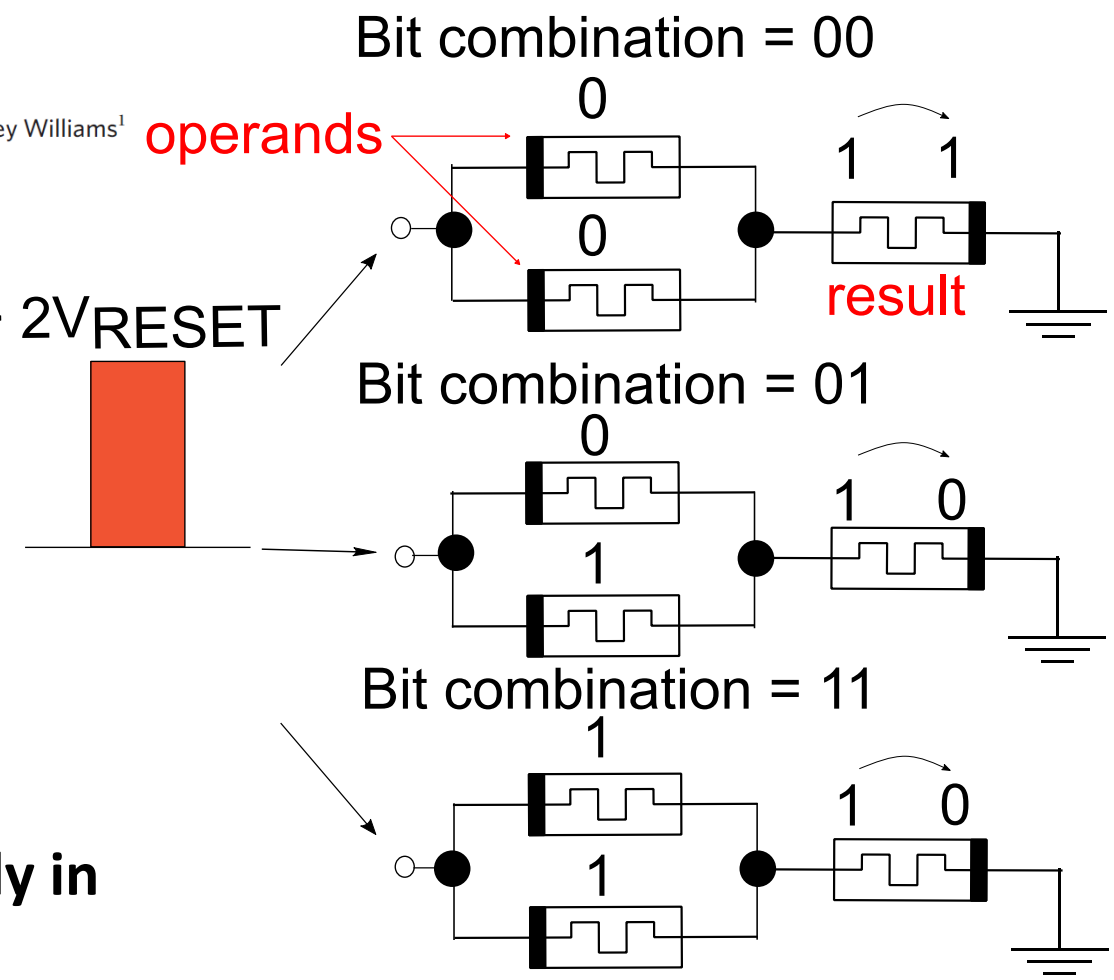



| In $p$ | In $q$ | Out $q'$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

*Borghetti et al., Nature (2010)*

- The Boolean variable is represented **only in terms of the resistance state**
- Both the operands and result are stored in terms of the resistance state variable

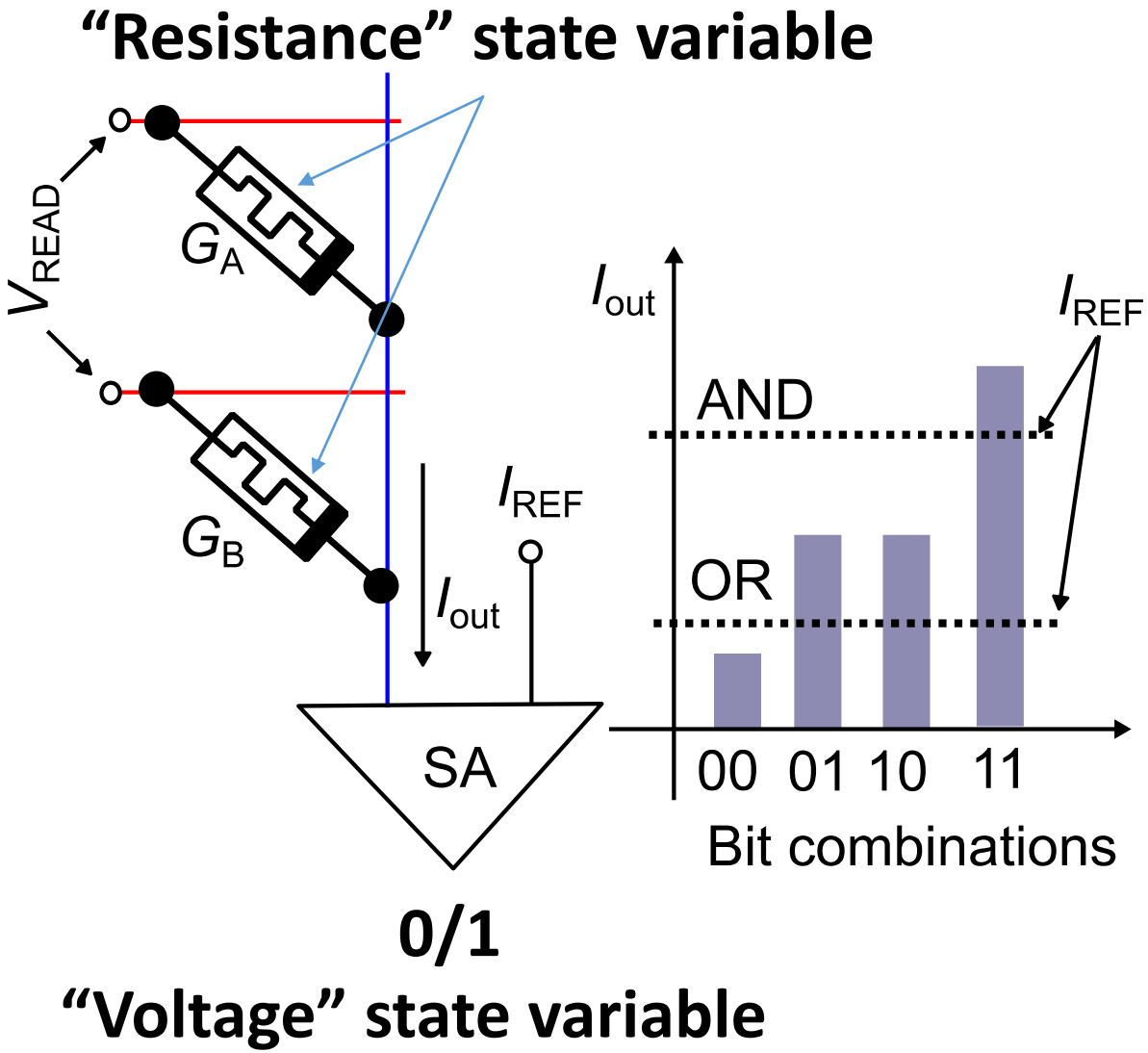
**MAGIC: NOR Logic**




*Kvatinsky et al., IEEE TCAS (2014)*

# Non-stateful logic

- Both resistance and voltage state-variables co-exist
- Data is stored in terms of **resistance logic state-variables**; However, the logical operations are implemented in the periphery
- Eg. by **simultaneously sensing multiple memristive devices connected to the same sense amplifier**
- **Key advantage:** Memristive devices are programmed rather infrequently → limited cycling endurance is not a challenge




*Li et al., Proc. DAC (2016), Xie et al., Proc. ISVLSI (2017), Hamdioui et al., DATE (2019)*

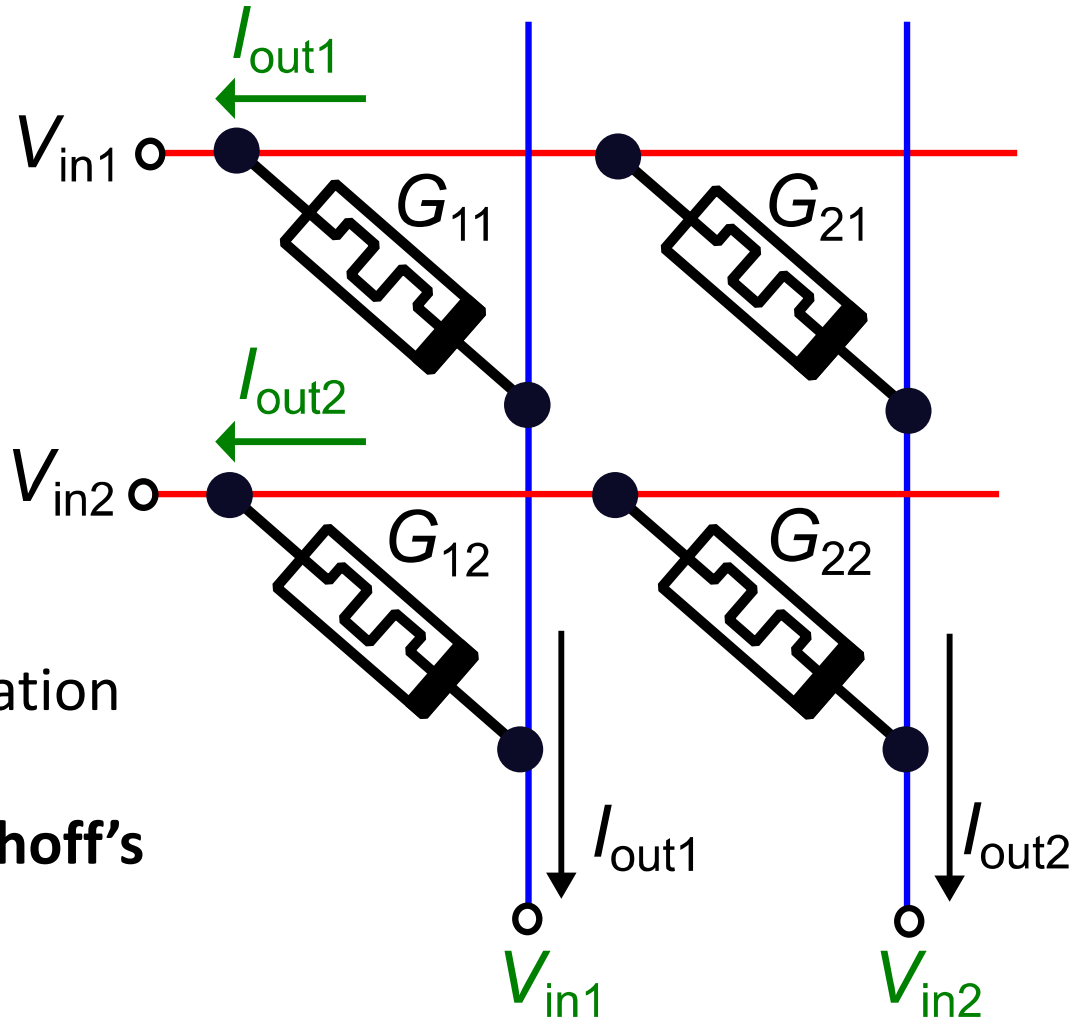
# MVM using resistive memory

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

MAP to conductance values

MAP to read voltage

DECIPHER from the current

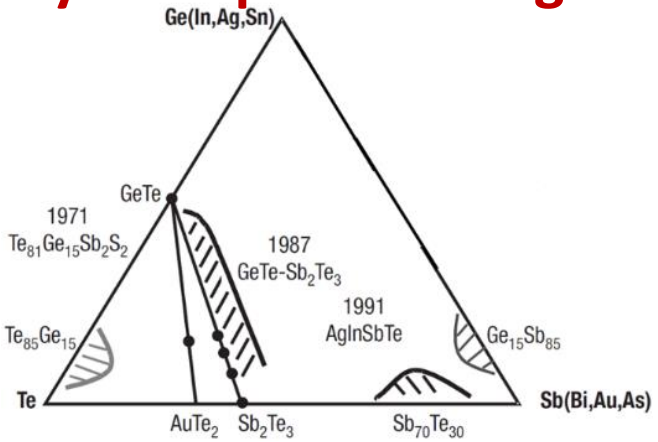



- In-place matrix-vector multiply (MVM) operation with O(1) time complexity
- Exploits **analog storage capability and Kirchhoff's circuits laws**
- Can also implement **MVM with the matrix transpose**

*Burr et al., Adv. Phys. X (2017), Xia and Yang, Nature Materials (2019)*

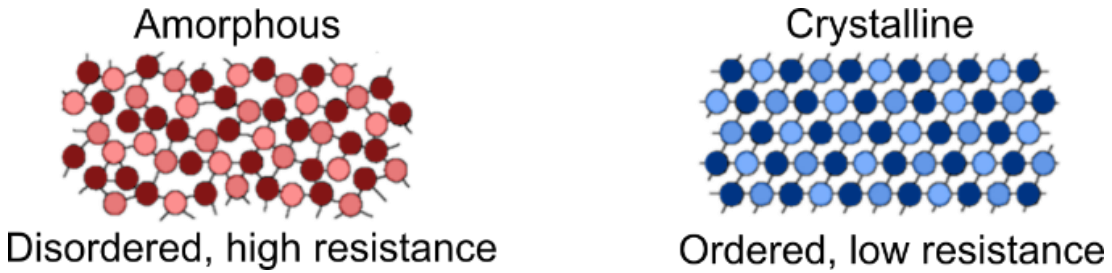
# PCM: A prototypical resistance-based memory

## Commonly used phase change materials

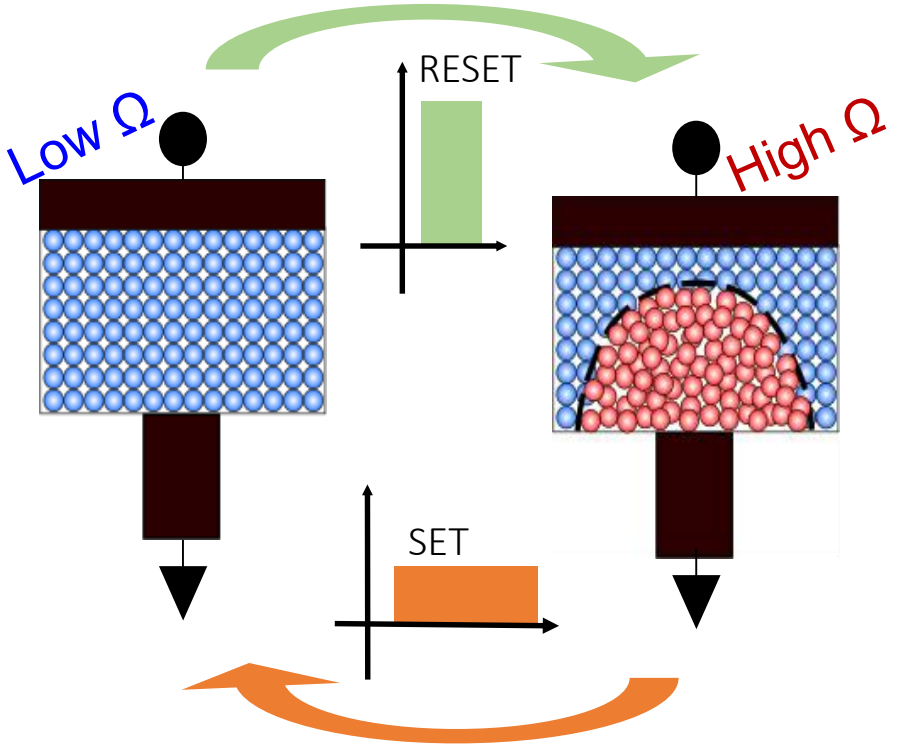



*Wuttig & Yamada, Nature Materials (2007)*

*Burr et al., JETCAS (2016)*

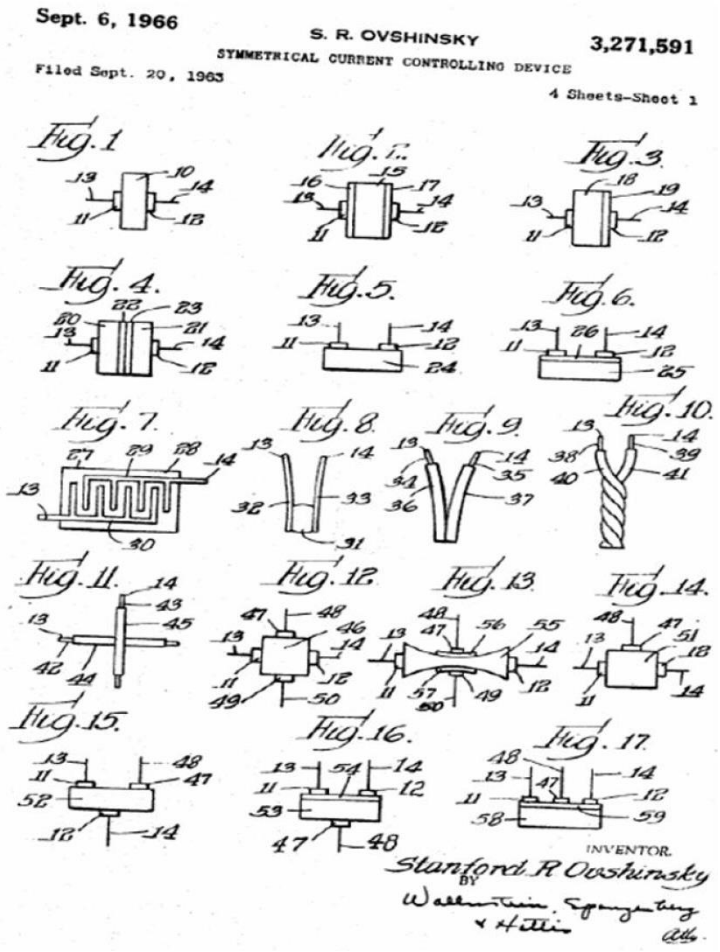


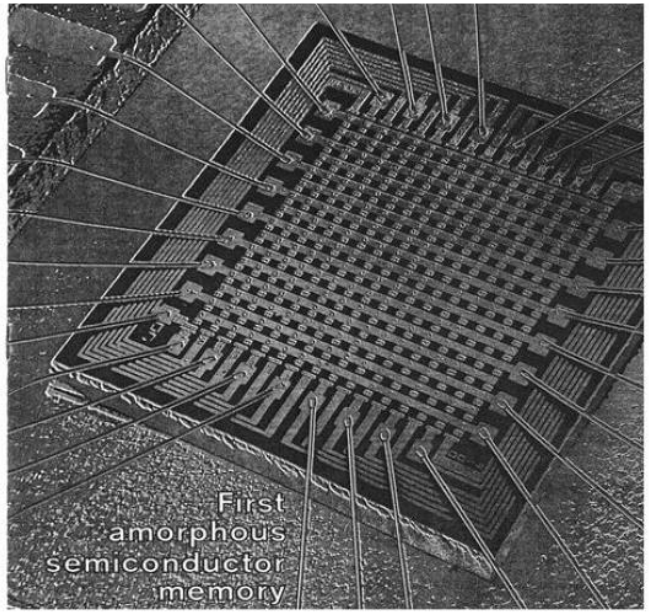

- A nanometric volume of phase change material between two electrodes
- "WRITE" Process
  - ✓ By applying a voltage pulse the material can be changed from crystalline phase (SET) to amorphous phase (RESET)
- "READ" process
  - ✓ Low-field electrical resistance



# A brief history of phase change memory

*Stan Ovshinsky (1960s)*



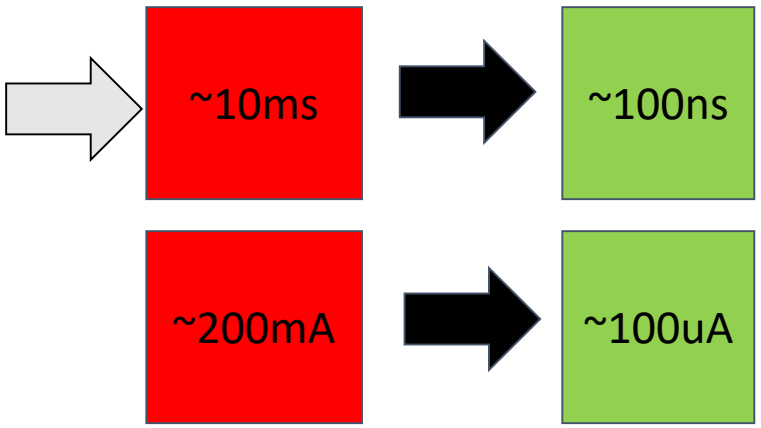

*R. G. Neale, D. L. Nelson and G. E. Moore., Electronics (1970)*

Capacity: 256 bits
RESET: ~200mA, <25V, 5 us
SET: 5mA, ~25V, 10ms
Read: 2.5mA, <5V

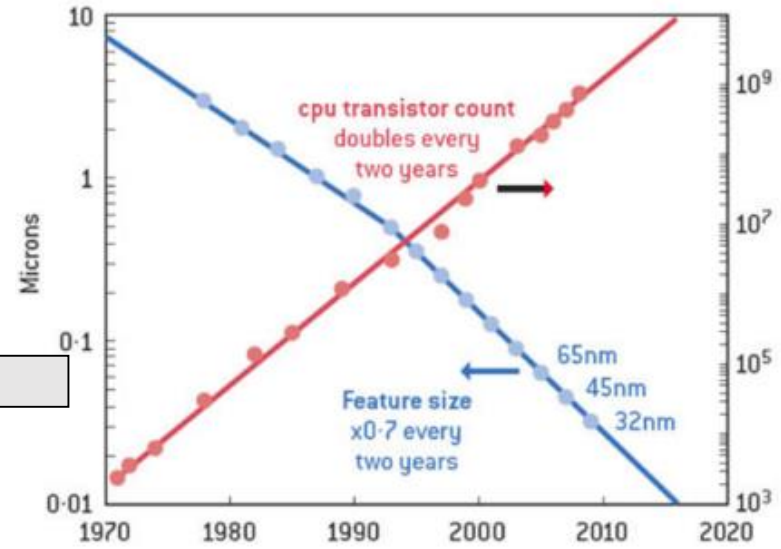
# A brief history of phase change memory

**Commercial success of optical recording**

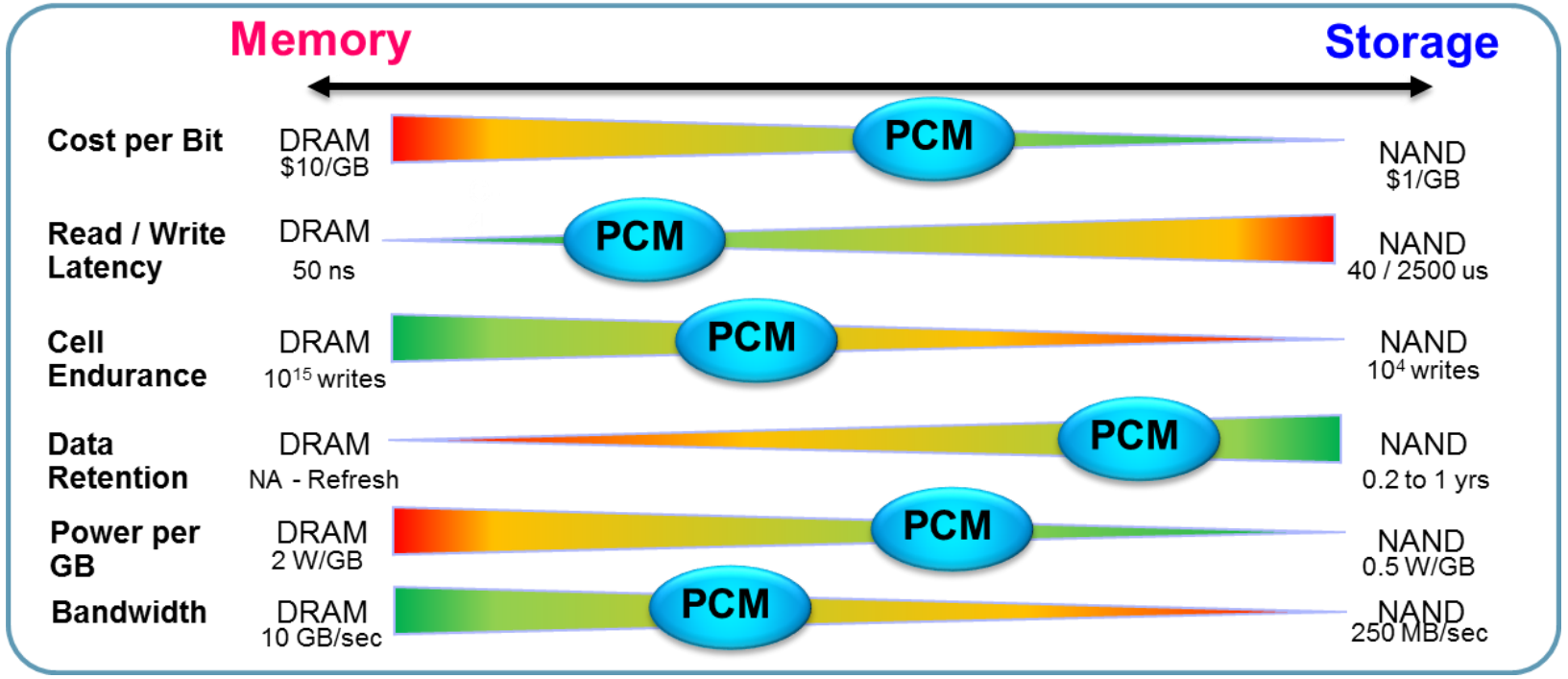
SET Speed

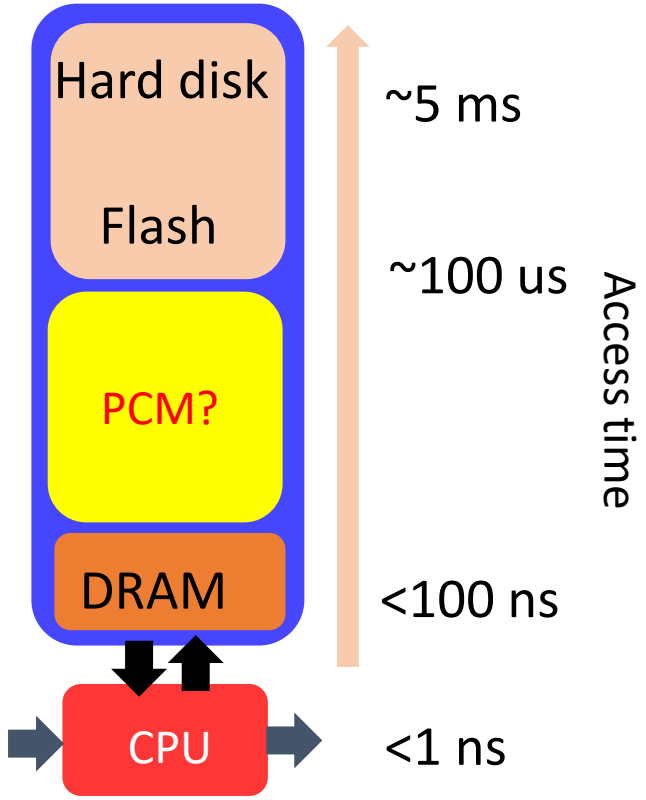



**Advances in semiconductor manufacturing**



# PCM as storage class memory




- Latency: much faster than FLASH (100's of ns vs. 100's of us)
- Write endurance: 1,000 x FLASH
- Nonvolatile, true random access capability, write in-place
- Very good scaling potential demonstrated (beyond 10nm node)
- Cost: between FLASH and DRAM (as technology matures)

**Commercialized as SCM by Intel/Micron (3D Xpoint)**

*Burr et al., IBM JRD (2008), Lee et al. ISCA (2009), Cappelletti, IEDM (2015)*

# Why PCM for in-memory computing?

**Strong field and temperature dependence**

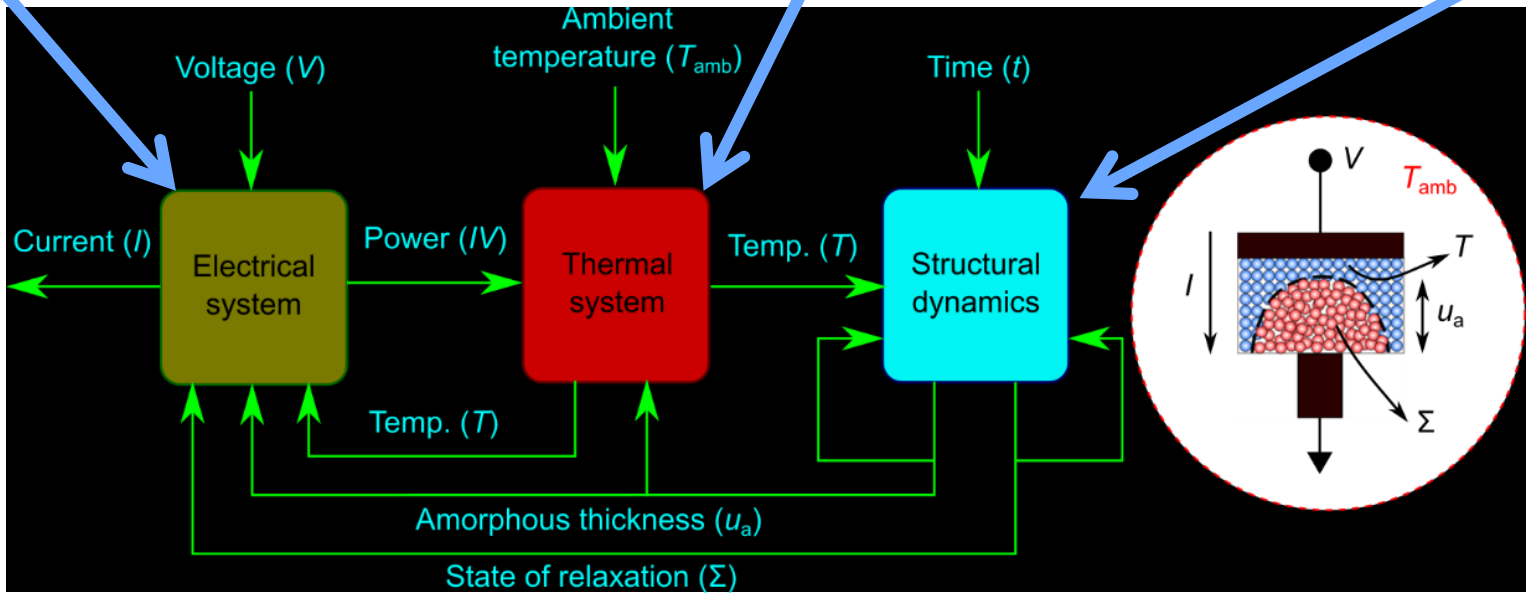
*Ielmini, Zhang, JAP (2007)*
*Le Gallo et al., New J. Phys. (2015)*
*Le Gallo et al., J. Appl. Phys. (2016)*

**Nanoscale thermal transport, thermo-electric effects**

*Lee et al., Nanotechnology (2012)*
*Athmanathan et al., SISPAD (2015)*

**Phase transitions, structural relaxation**

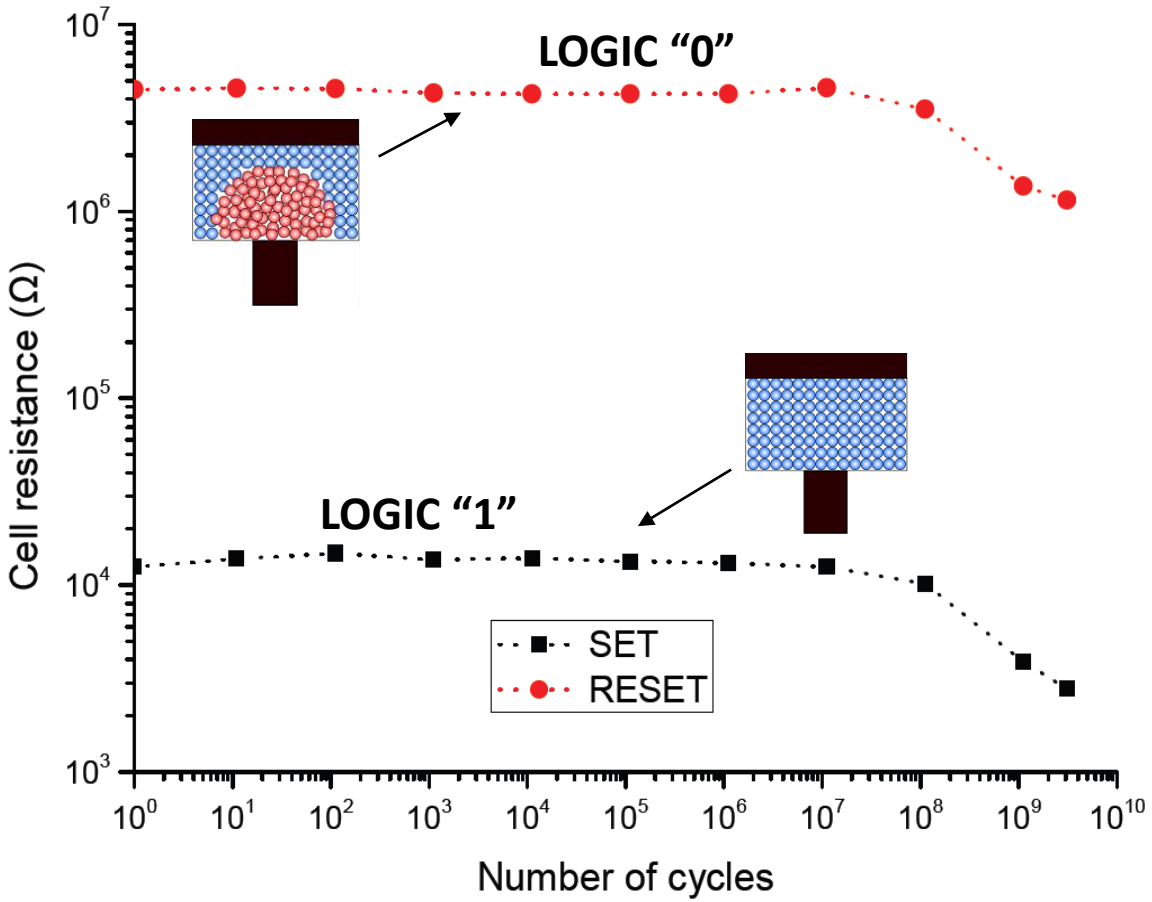
*Sebastian et al., Nature Comm. (2014)*
*Boniardi, Ielmini, APL (2011)*
*Le Gallo et al., Adv. Electr., Mat. (2018)*
*Salinga et al., Nature Materials (2018)*



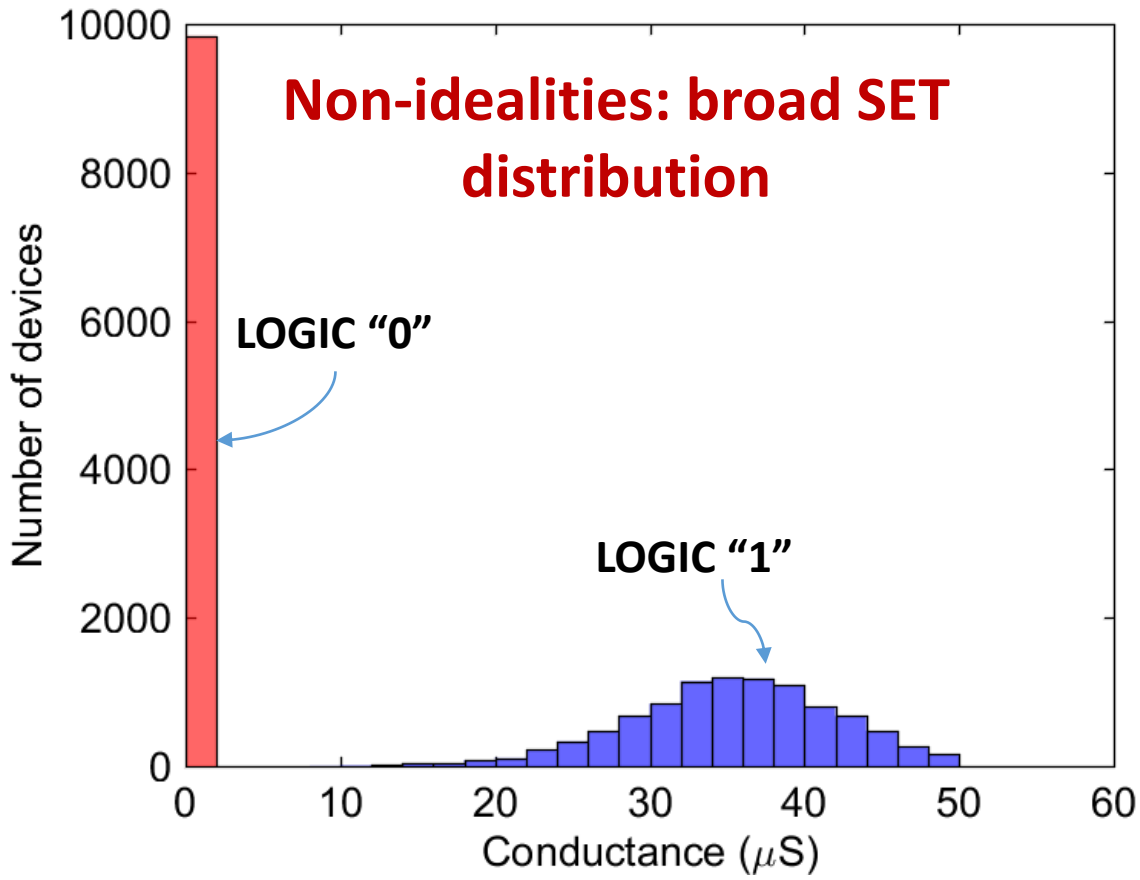

- Successfully commercialized
- **Well understood device physics**

***Le Gallo and Sebastian, J. Phys. D: Appl. Phys. (2020)***

# Key physical attribute I: Non-volatile binary storage

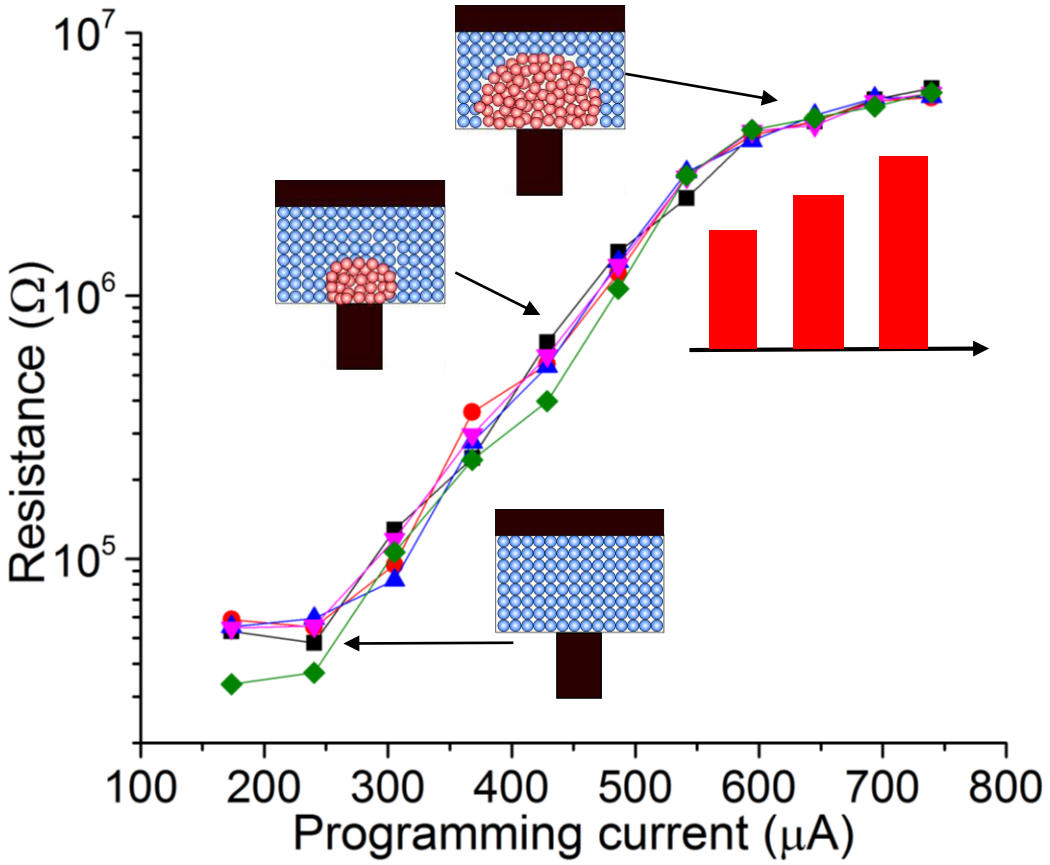



*Tuma et al., Nature Nanotechnology (2016)*



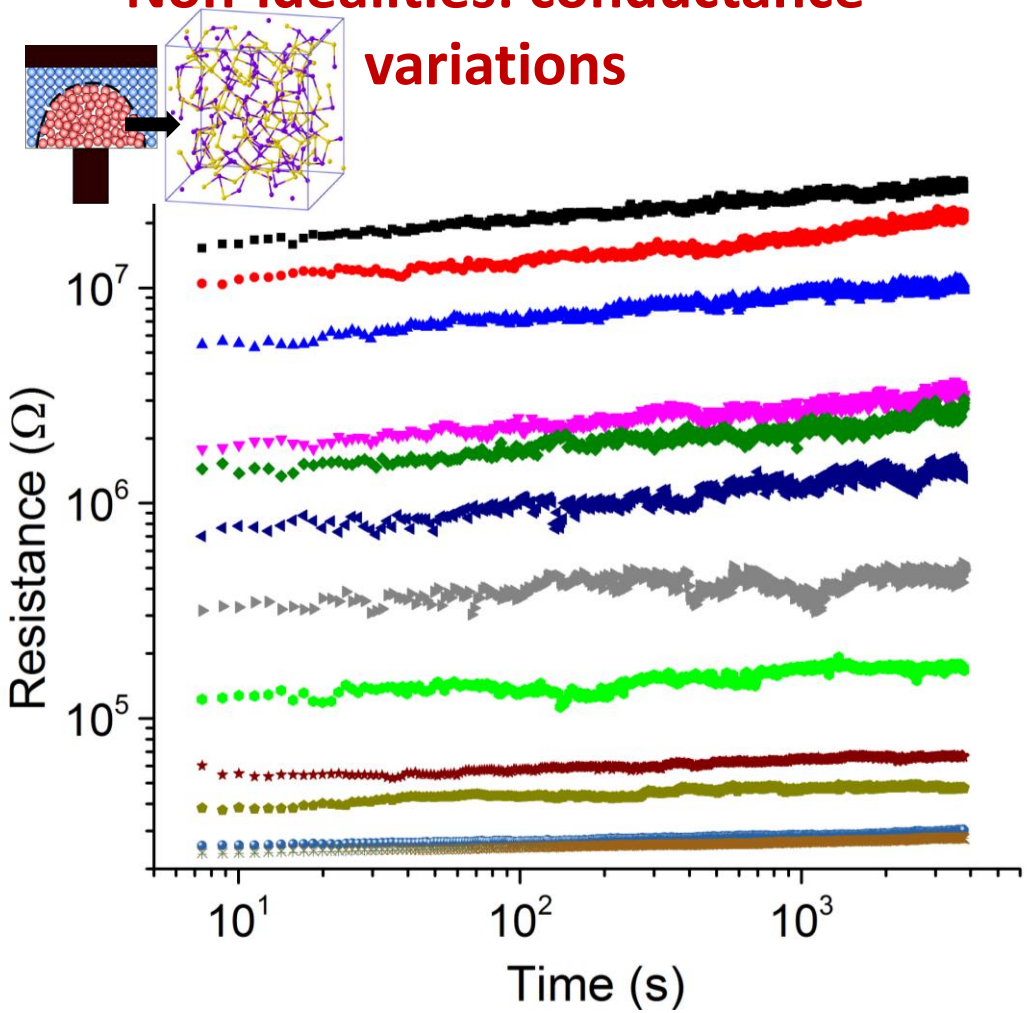

- A **binary storage device**, **with a distribution of SET and RESET conductance values**

# Key physical attribute II: Analog storage capability




*Sebastian et al., J. Appl. Phys. (2018)*

**Non-idealities: conductance variations**




*Le Gallo et al., Adv. Electr. Mat (2018)*

- An **analog storage device**, **but with noise and drift**

# MVM using PCM

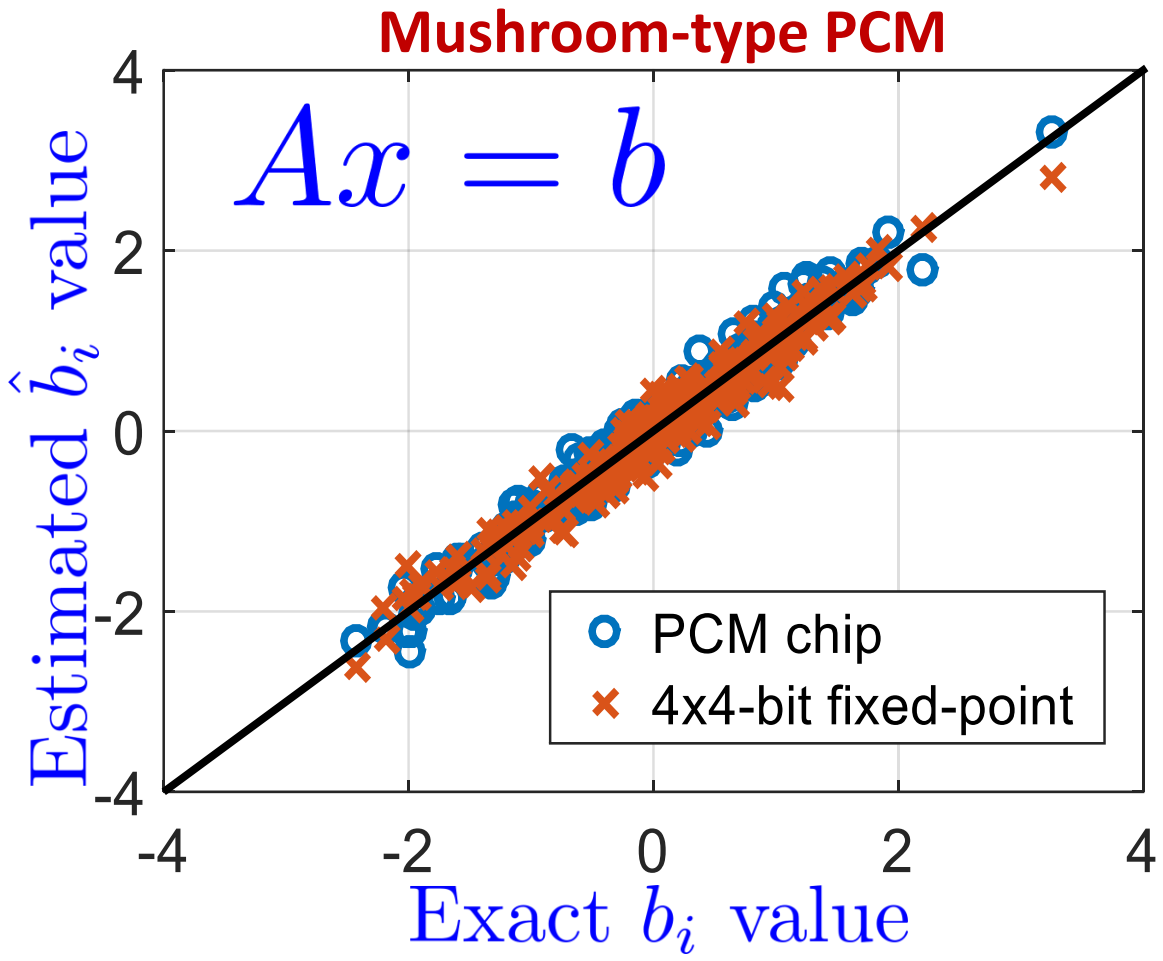
$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

MAP to conductance values

MAP to read voltage

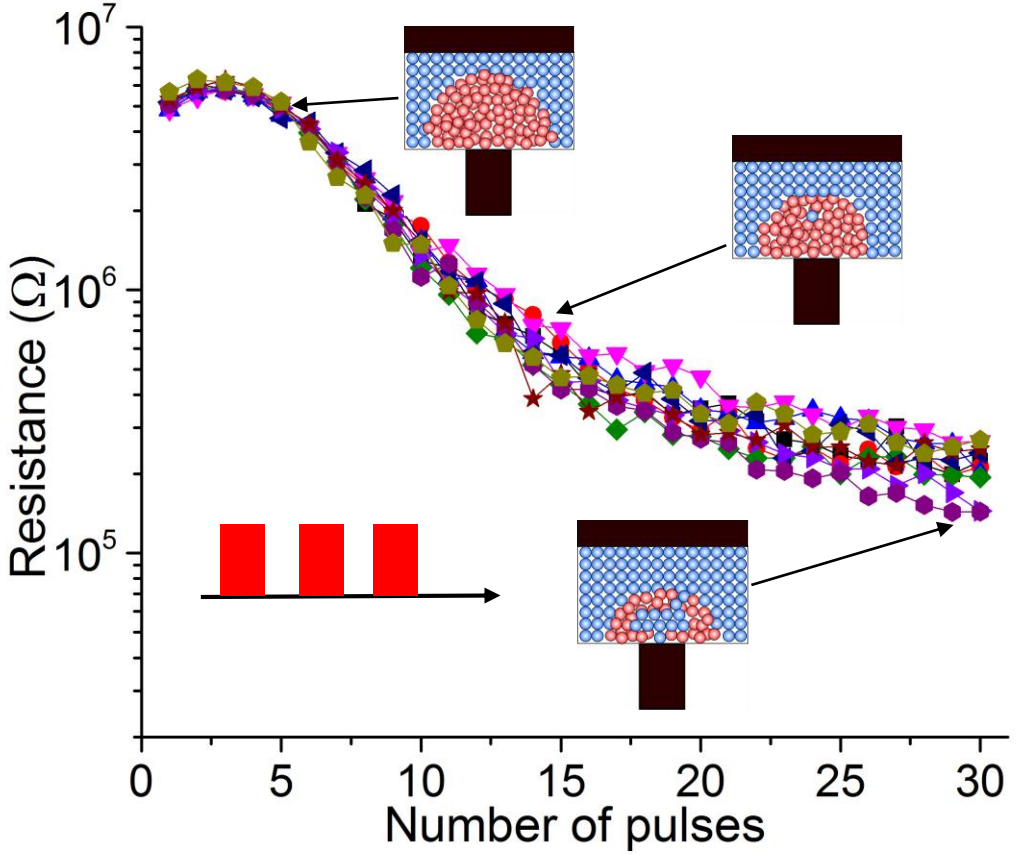
DECIPHER from the current

- *A* is a 256X256 Gaussian matrix coded in a PCM chip
- *x* is a 256-long Gaussian vector applied as voltage
- Precision equivalent to **4-bit fixed point arithmetic**



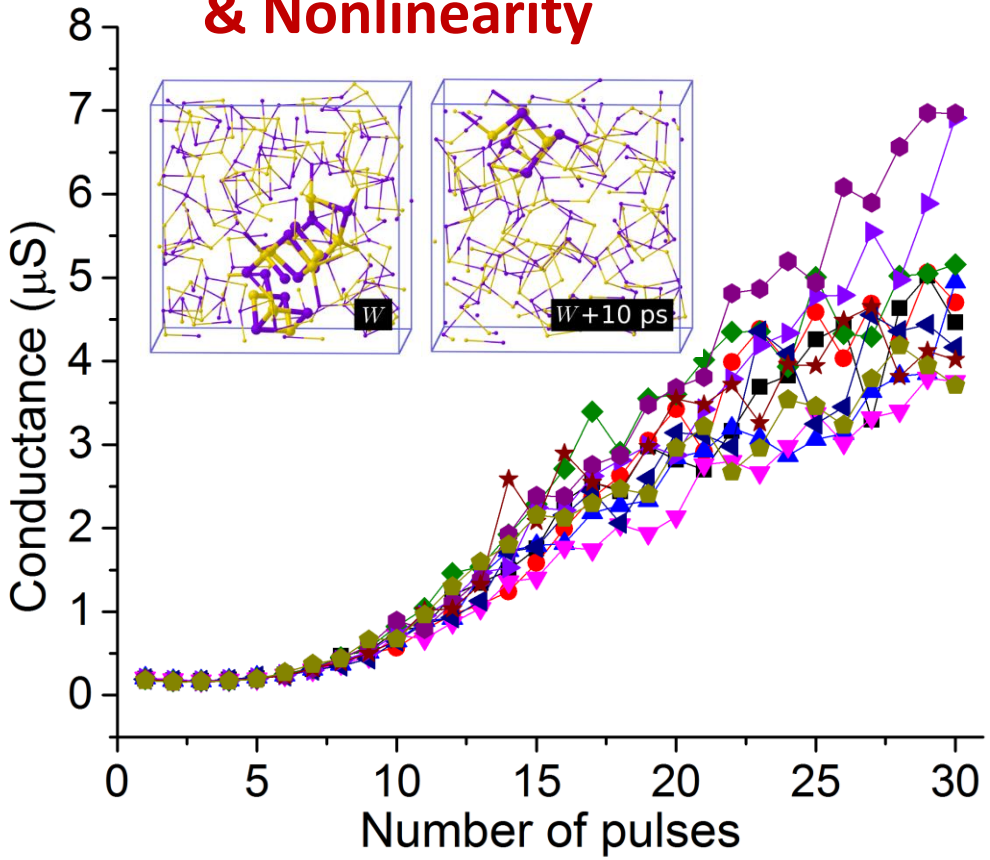

*Le Gallo et al., Nature Electronics (2018)*

# Key physical attribute III: Accumulative behavior




*Sebastian et al., J. Appl. Phys. (2018)*

**Non-idealities: Stochasticity & Nonlinearity**



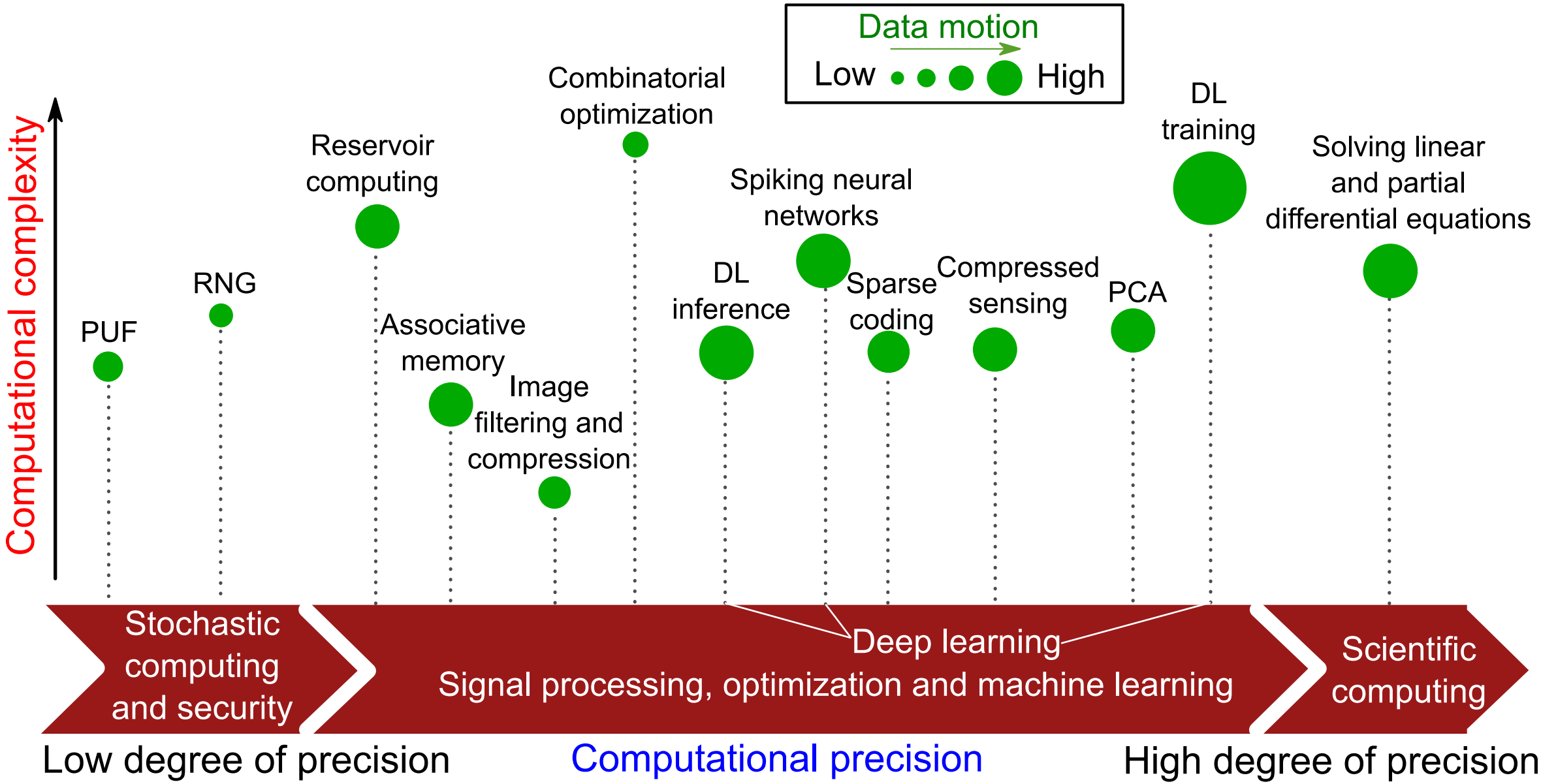

*Le Gallo et al., ESSDERC (2016)*

- A **non-volatile integrator**, but **non-linear and stochastic**

# Outline

- Introduction
- Memory devices and computational primitives
  - ✓Charge-based memory devices & Computational primitives
  - ✓Resistance-based memory devices & Computational primitives
  - ✓Phase change memory: A prototypical resistance-based memory
- Applications
  - ✓Exploiting non-volatile binary storage
  - ✓Scientific computing
  - ✓Signal processing & Machine learning
  - ✓Deep learning
  - ✓Stochastic computing and security
- Discussion
  - ✓Increasing the precision of in-memory computing
  - ✓Photonic in-memory computing
  - ✓Summary

# The application landscape

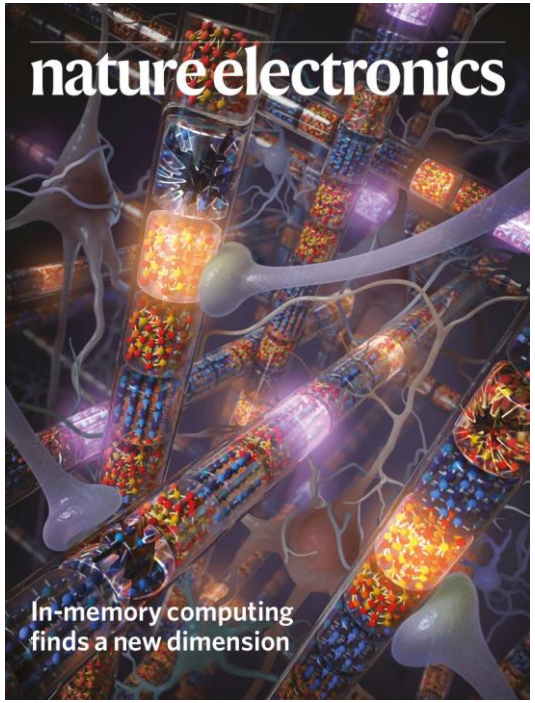


# Applications exploiting non-volatile binary storage

## Database query

*Giannopoulos et al., Adv. Int. Sys. (2020)*

## Hyperdimensional computing




*Karunaratne et al., Nature Electr. (2020)*

# Database query

**Information on newly discovered stars**

| | Dist. | Size | Year |
|---|---|---|---|
| A | 55 | Large | 2016 |
| B | 23 | Medium | 2014 |
| C | 43 | Small | 2015 |
| D | 60 | Medium | 2016 |
| E | 25 | Medium | 2000 |
| F | 34 | Medium | 2001 |
| G | 18 | Small | 2012 |
| H | 30 | Small | 2011 |

**Bitmap representation**

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| Far | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| Near | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| Large | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Medium | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| Small | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| New | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Old | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| OR | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| AND | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Which star is far or large?

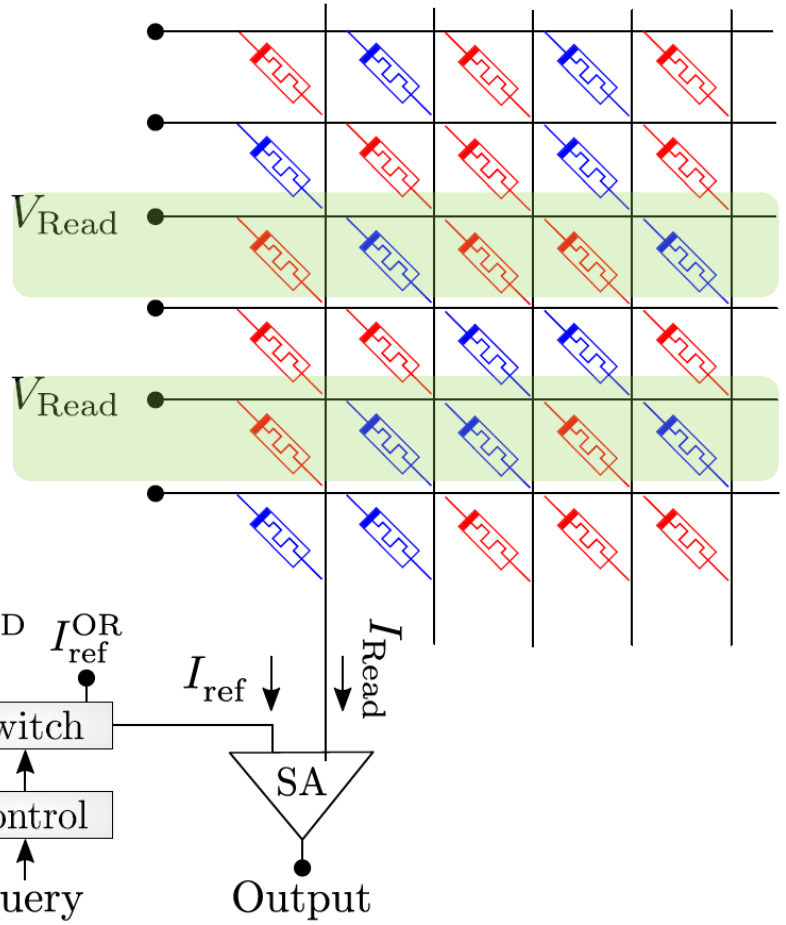
Which star is new and medium?

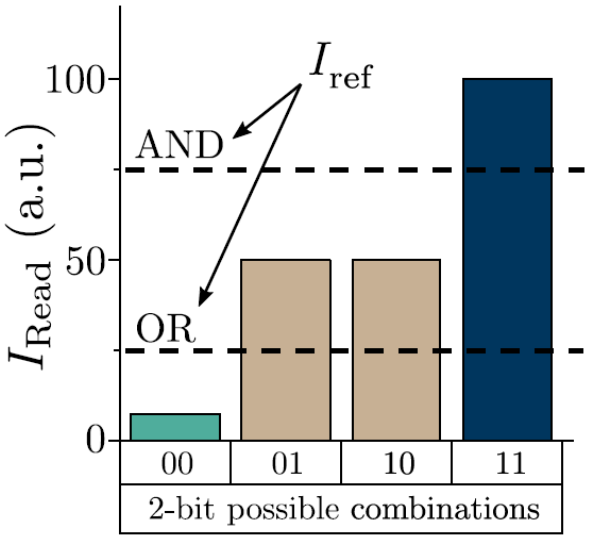
- Database query involves a high percentage of logical operations
- Key challenge: Retrieving the stored data and bringing it to the processor that will execute the query

*Hamdoui et al., DATE (2019)*

# In-memory database query

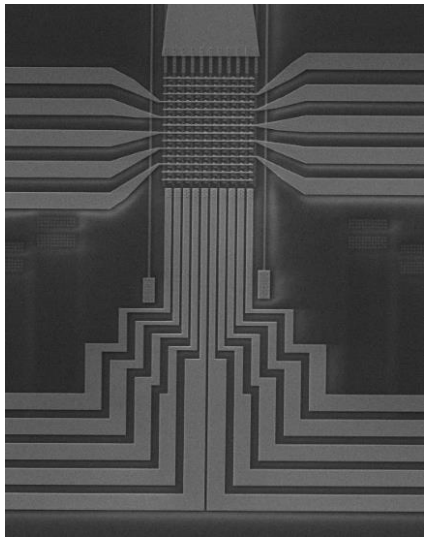
| Attribute \ Entry | #1 | #2 | #3 | #4 | #5 |
|---|---|---|---|---|---|
| A | 0 | 1 | 0 | 1 | 0 |
| B | 1 | 0 | 0 | 1 | 0 |
| C | 0 | 1 | 0 | 0 | 1 |
| D | 0 | 0 | 1 | 1 | 0 |
| E | 0 | 1 | 1 | 0 | 1 |
| F | 1 | 1 | 0 | 0 | 0 |
| C OR E | 0 | 1 | 1 | 0 | 1 |
| A AND D | 0 | 0 | 0 | 1 | 0 |

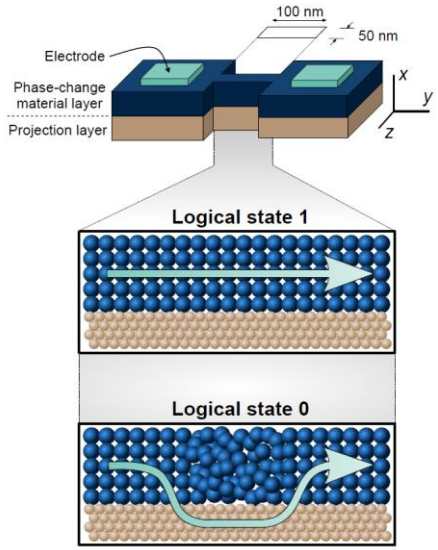



- Database stored in terms of the conductance states of memristive devices
- To perform logical operations, multiple rows are biased simultaneously, and the resulting current is sensed per column using variable reference SAs.

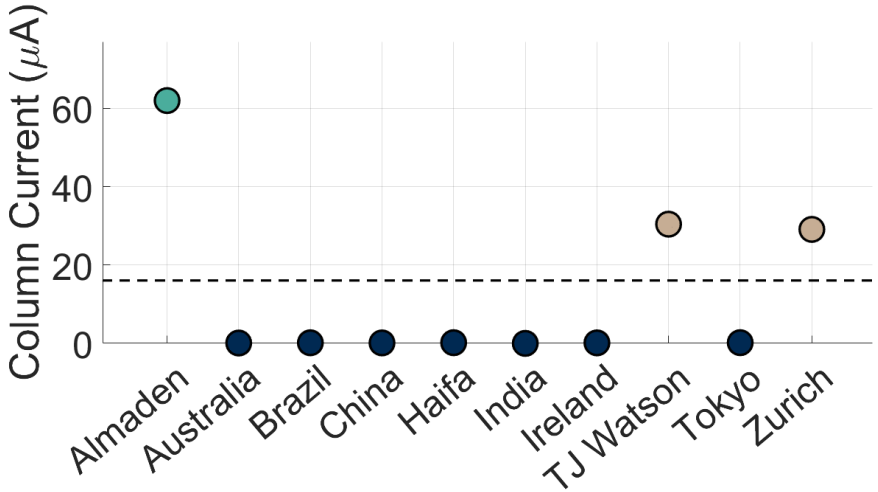
# Experimental demonstration on a PCM array






| IBM Research Labs | 1 Almaden | 2 Australia | 3 Brazil | 4 China | 5 Haifa | 6 India | 7 Ireland | 8 TJ Watson | 9 Tokyo | 10 Zurich |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 – Age 45-75 y | | | | | ● | | | ● | | ● |
| 2 – Age 20-45 y | ● | | | ● | | ● | | | ● | |
| 3 – Age < 20 y | | ● | ● | | | | ● | | | |
| 4 – Size (S) | | ● | ● | ● | | ● | ● | | ● | |
| 5 – Size (L) | ● | | | | ● | | | ● | | ● |
| 6 – Nobel prize | ● | | | | | | | ● | | ● |
| 7 – Turing award | ● | | | | | | | ● | | |
| 8 – Kavli prize | ● | | | | | | | | | ● |
| 9 – N. Hem. | ● | | | ● | ● | ● | ● | ● | ● | ● |
| 10 – S. Hem. | | ● | ● | | | | | | | |

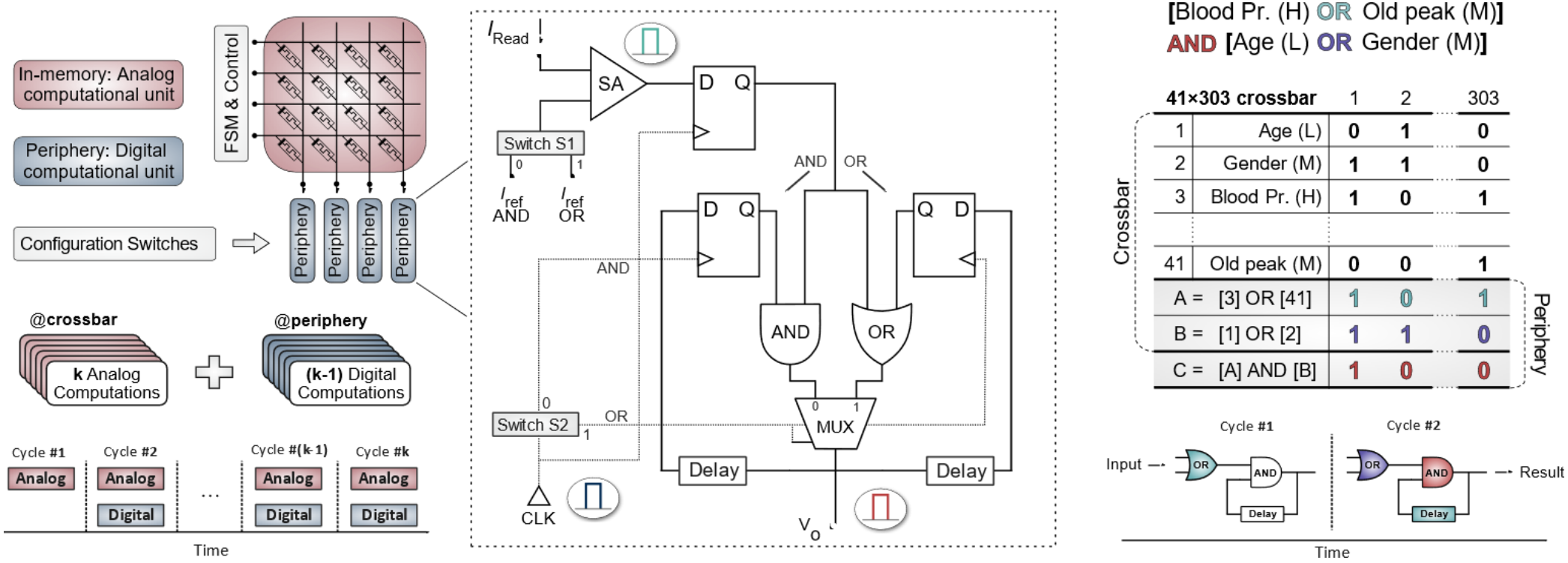
Query: “Turing Award” OR “Kavli Prize”




- A PCM array holds the database in a non-volatile fashion
- By employing non-stateful AND and OR in-memory logic operations, it is possible to query this database

*Giannopoulos et al., Adv. Int. Sys. (2020)*

# Cascaded database query



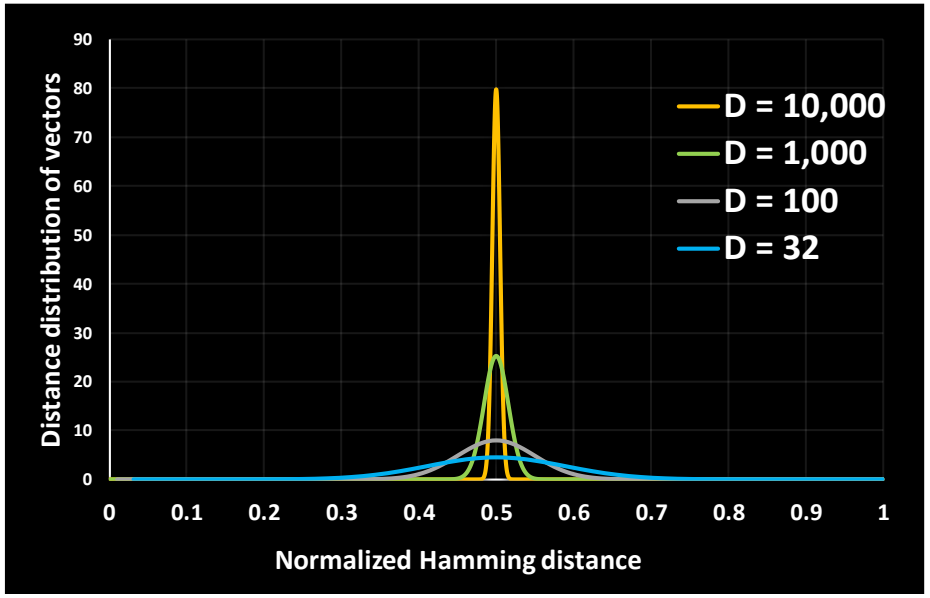

- Real-world database queries consist of a **multitude of subqueries**
- Any query can be expressed as the sum of products (SOP) or the product of sums (POS) where sum and product operators correspond to OR and AND, respectively
- Possible to perform such as **cascaded query both in-memory and near-memory**

# Hyperdimensional computing

Pentti Kanerva, Redwood Center for Theoretical Neuroscience, UC Berkeley

*Kanerva, "An introduction to computing in distributed representation with high-dimensional random vectors", Cogn. Comp., 2009*

**Orthogonality of randomly drawn HD vectors**




- The **brain's circuits are massive** in terms of numbers of neurons and synapses
- **Remarkably robust** to failures and imperfections
- How about computing with **holographic hyperdimensional (HD) binary vectors (~10,000)?**
- The vectors in an HD space are **nearly orthogonal to each other**
- By manipulating such vectors one can efficiently realize certain machine learning tasks

# Example: European language classification


- **Find a language prototype vector using trigrams**

How to encode “ICH BIN”?

ρρI = [1 0 0 1 0 0 … 1 1 0 0]

ρC = [1 0 1 1 1 1 … 1 0 1 0]

H = [1 1 1 0 0 1 … 1 0 1 1]

**ITEM MEMORY**

A = [1 0 0 1 0 … 0 1 0]
B = [0 1 0 1 1 … 1 0 1]
⋮
Z = [0 1 1 1 0 … 0 0 1]
# = [0 1 1 1 0 … 0 0 1]

“ICH” = ρρI*ρC*H = [1 1 0 0 1 0 … 1 1 0 1]

“CH ” = ρρC*ρH*# = [0 0 1 1 0 1 … 1 0 1 0]

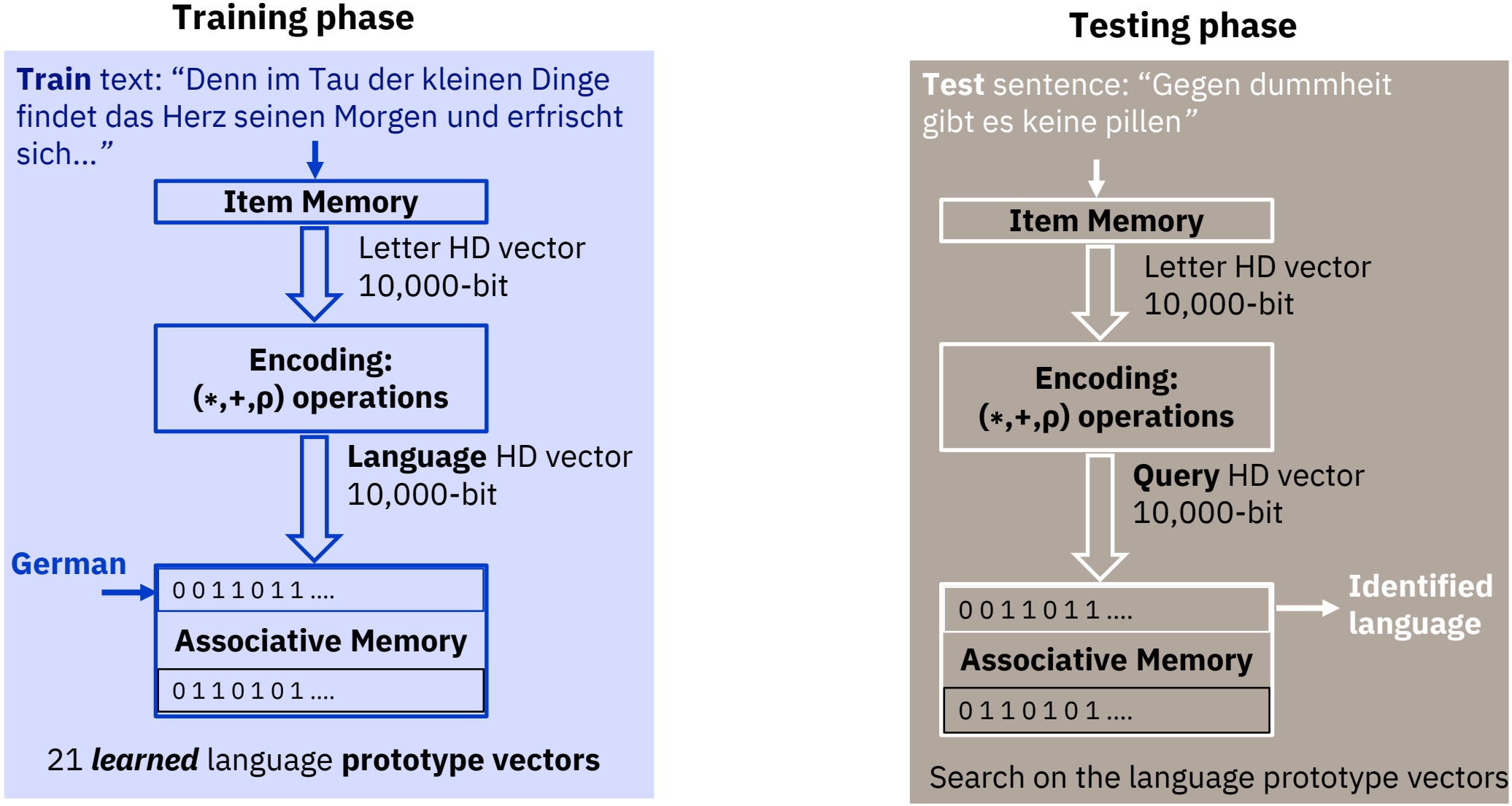
“H B” = ρρH*ρ#*B = [0 0 1 1 1 0 … 0 1 0 1]

“ BI” = ρρ#*ρB*I = [1 0 1 0 0 0 … 1 0 0 0]

“BIN” = ρρB*ρI*N = [0 1 1 0 0 1 … 0 1 1 0]

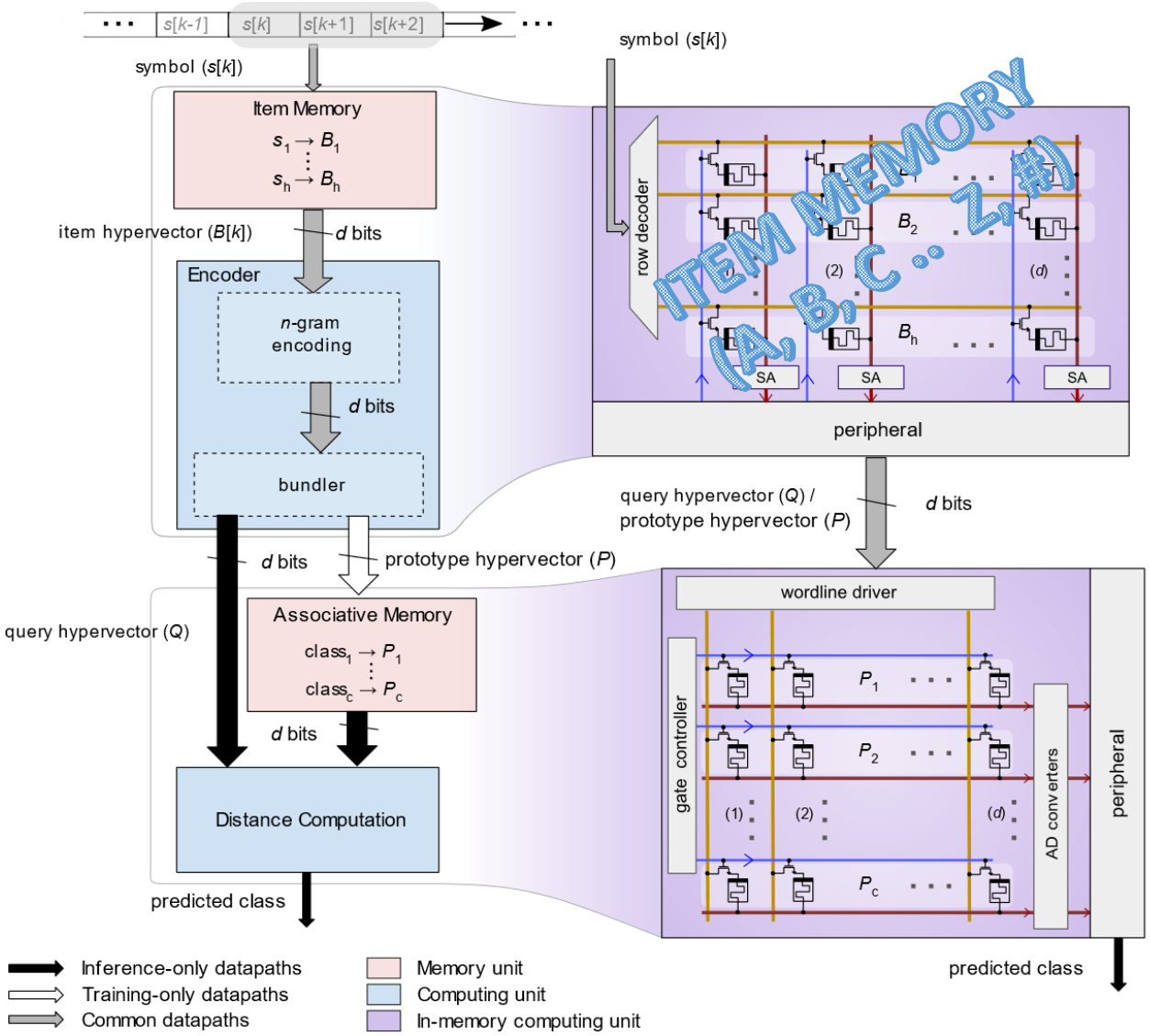
**“ICH BIN” = “ICH” + … = [0 0 1 1 0 1 … 1 1 0 0]**

# Example: European language classification

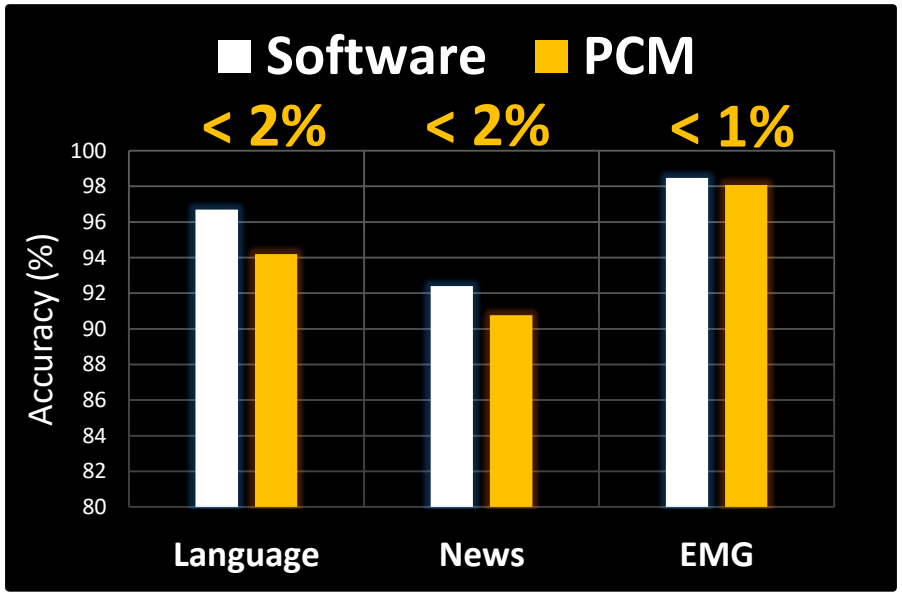



- Find the **closest prototype HD vector** to the **query HD vector**

# In-memory hyperdimensional computing



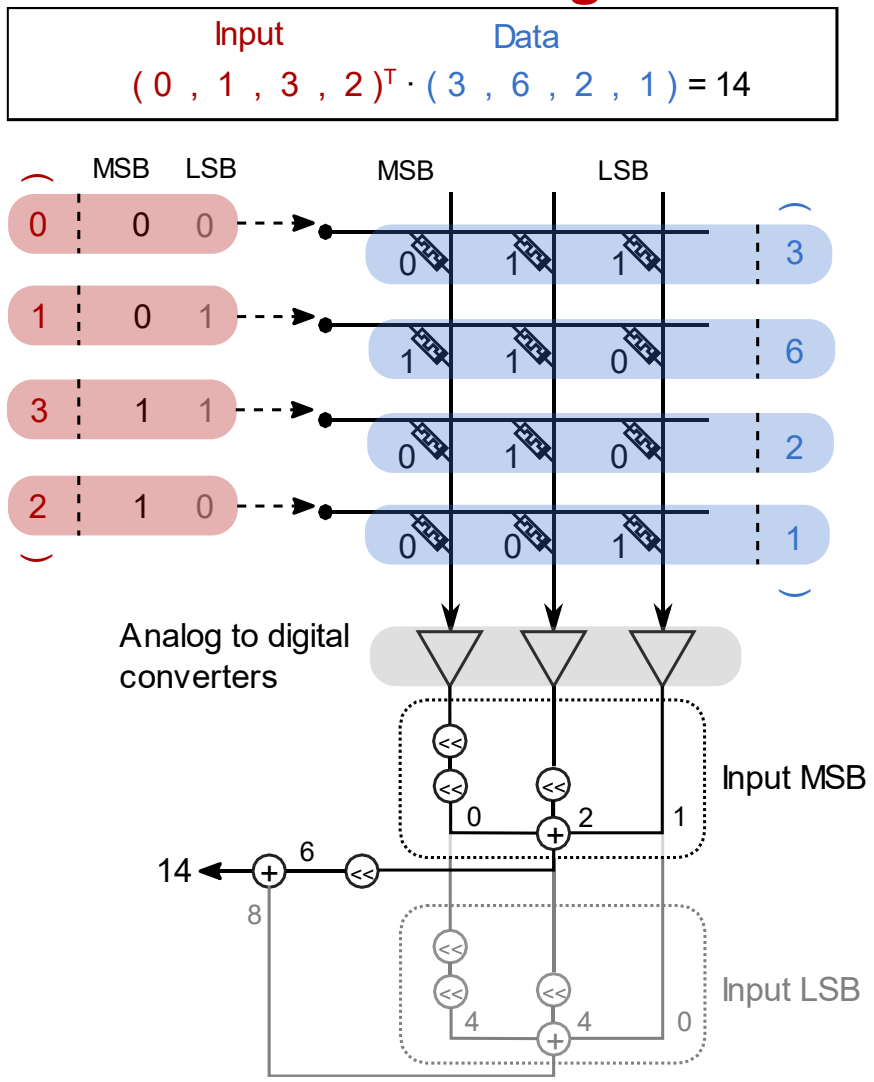

- **HD computing involves manipulation and comparison of vectors**
- **Highly robust to computational errors**
- **Encoding**
  - ✓ In-memory read logic
- **Associative memory search**
  - ✓ In-memory dot-product
- **Exploits non-volatile binary storage**




*Karunaratne et al., Nature Electronics (2020)*

# Scientific computing

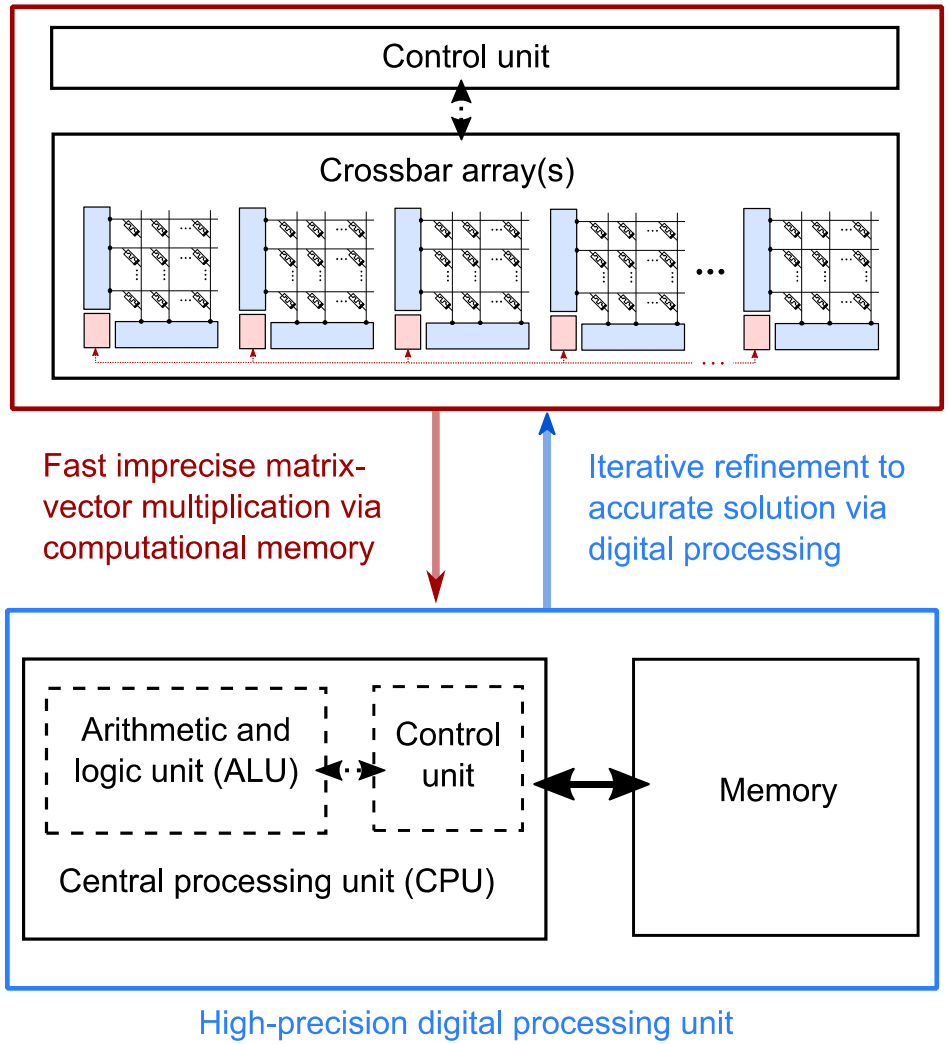
## Bit slicing




*Bojnordi et al., HPCA (2016)*

## Mixed-precision

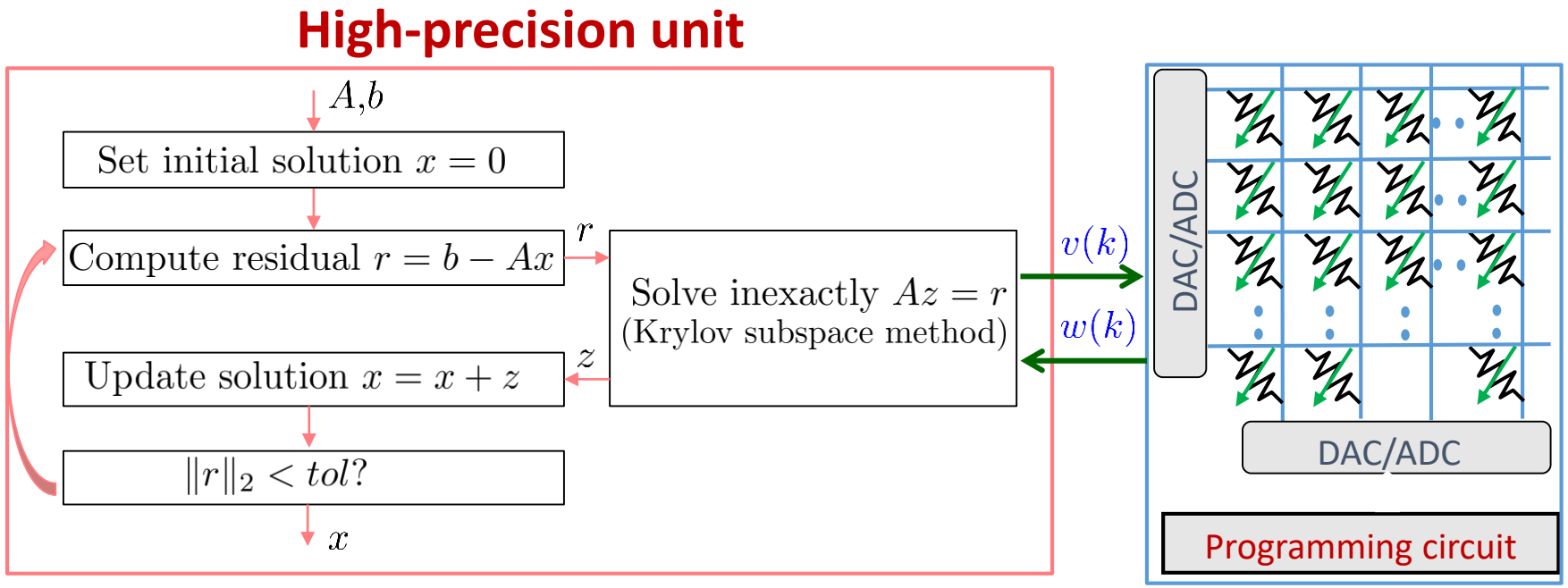
Low-precision computational memory unit




*Le Gallo et al., Nature Electronics (2018)*

# Mixed-precision linear solver

$$\text{if } Ax = b, \text{ find } x$$

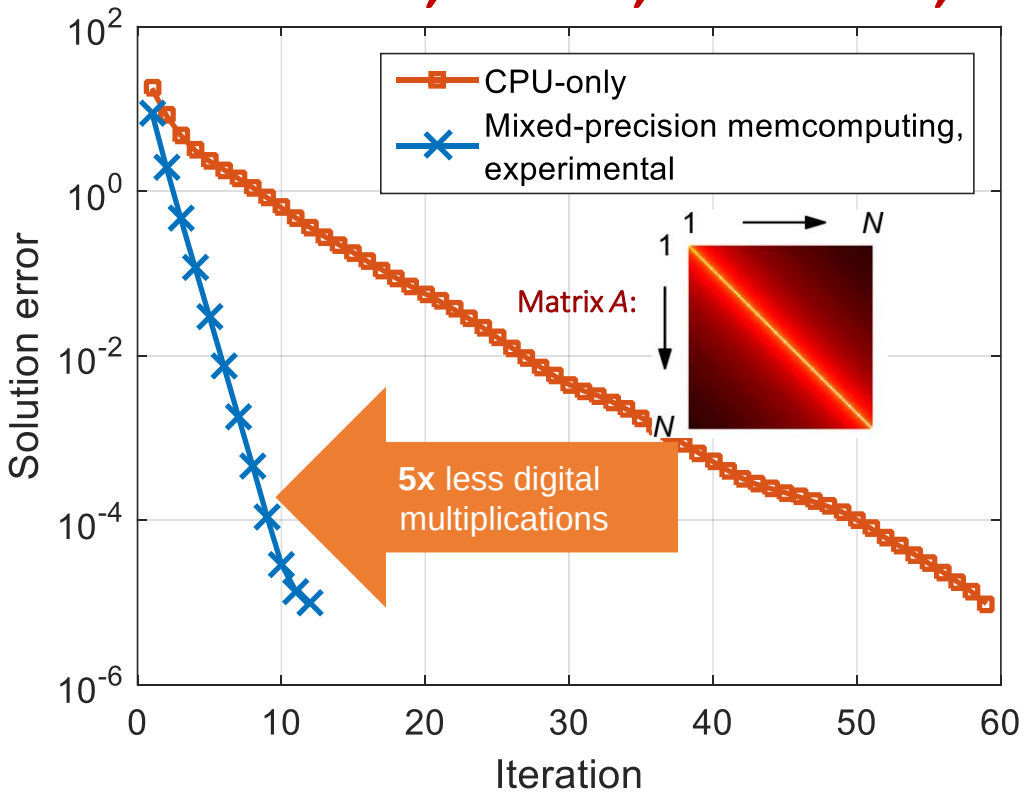



- Solution iteratively updated with **low-precision error-correction terms (iterative refinement)**
- Correction terms are **obtained using an inexact inner solver**
- The matrix multiplications in the inner solver are performed using in-memory computing

***Le Gallo et al., Nature Electronics (2018)***

# Mixed-precision linear solver: Experimental results

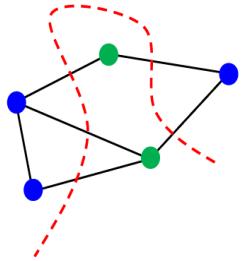
**Experimental result: 10,000x10,000 matrix, 959,376 PCM devices**




- Measured energy savings on end-to-end system w/o computational memory:
  - ✓ Speed-up of 7x and energy reduction of 7x over CPU-only (POWER8 with 8 threads)
  - ✓ Speed-up of 3.6x and energy reduction of 7x over GPU-only (Nvidia P100)
  - ✓ Expected energy savings of 20x with improved PCM devices
- More accurate in-memory computing → Higher gain in performance

# Signal processing and optimization

## Combinatorial optimization

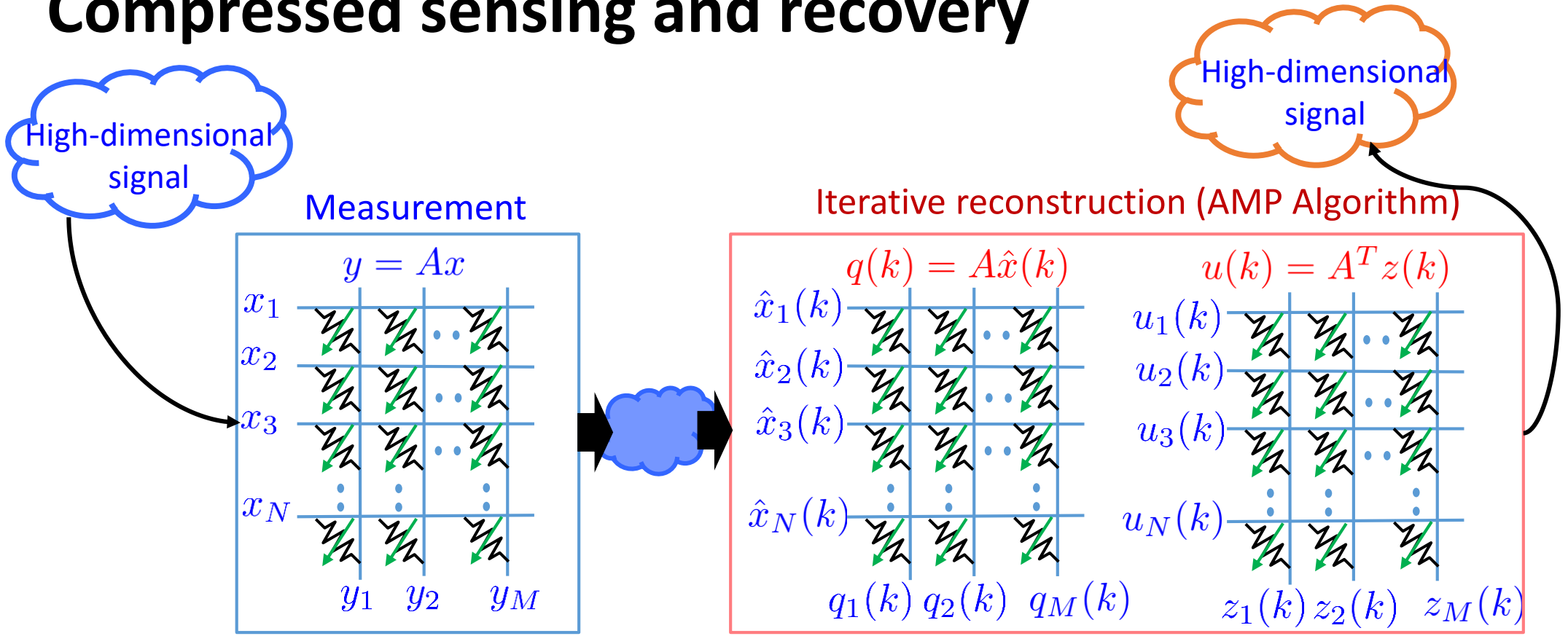


*Bojnordi et al., HPCA (2016)*
*Cai et al., Nature Electr. (2020)*

## Compressed sensing


*Le Gallo et al., Proc. IEDM (2017)*
*Li et al., Nature Electr. (2018)*

# Compressed sensing and recovery

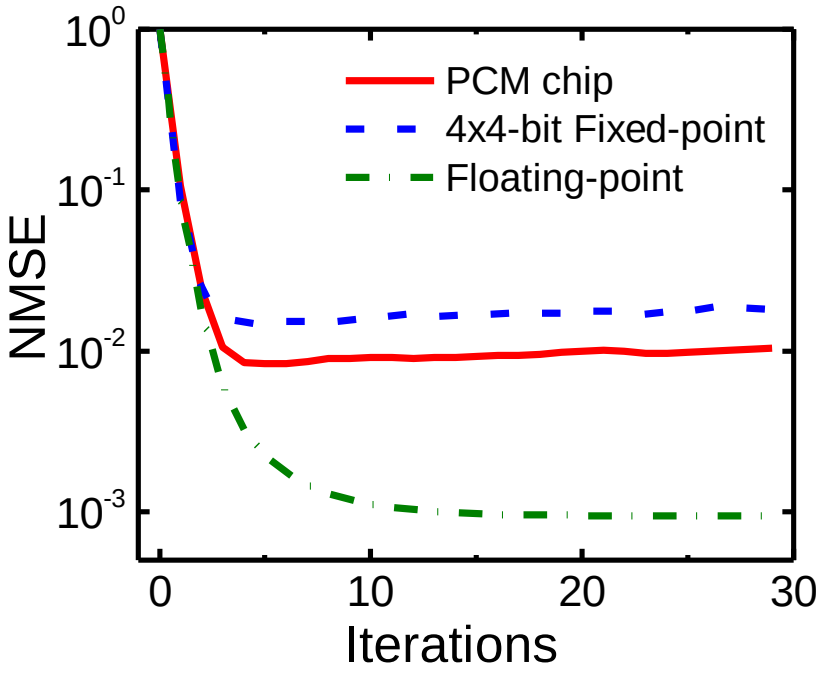


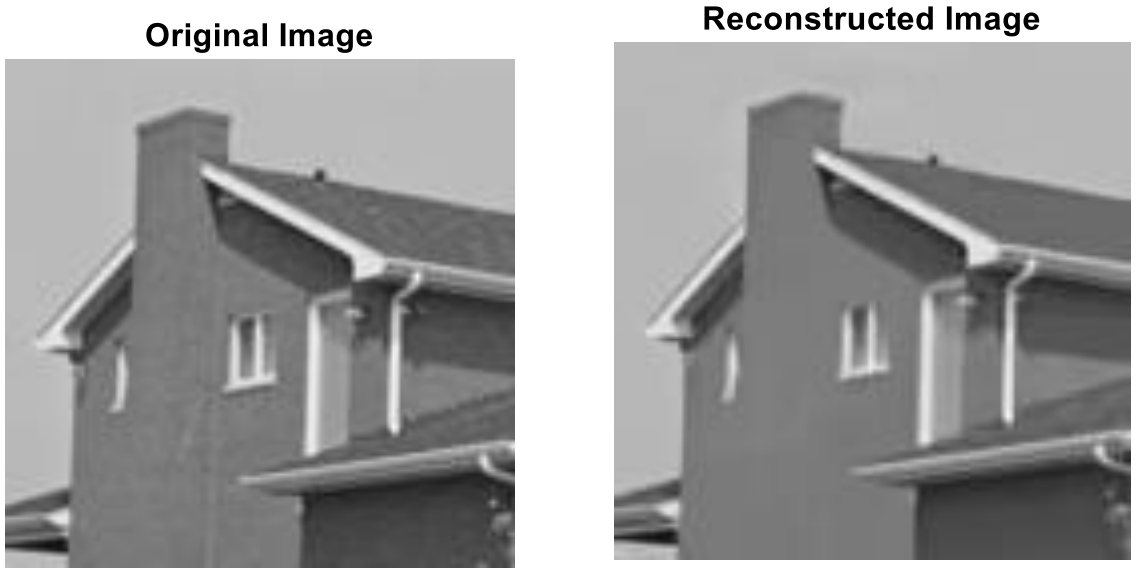

- Store the **measurement matrix** in a cross-bar array of resistive memory devices
- The same array used for both compression and reconstruction
- Reconstruction complexity reduction: **O(NM) → O(N)**

***Le Gallo et al., Proc. IEDM (2017)  Le Gallo et al., IEEE Trans. Electr. Dev. (2018)***

# Compressed sensing and recovery: Experiments

**Experimental result: 128X128 image, 50% sampling rate, Computation memory unit with 131,072 PCM devices**

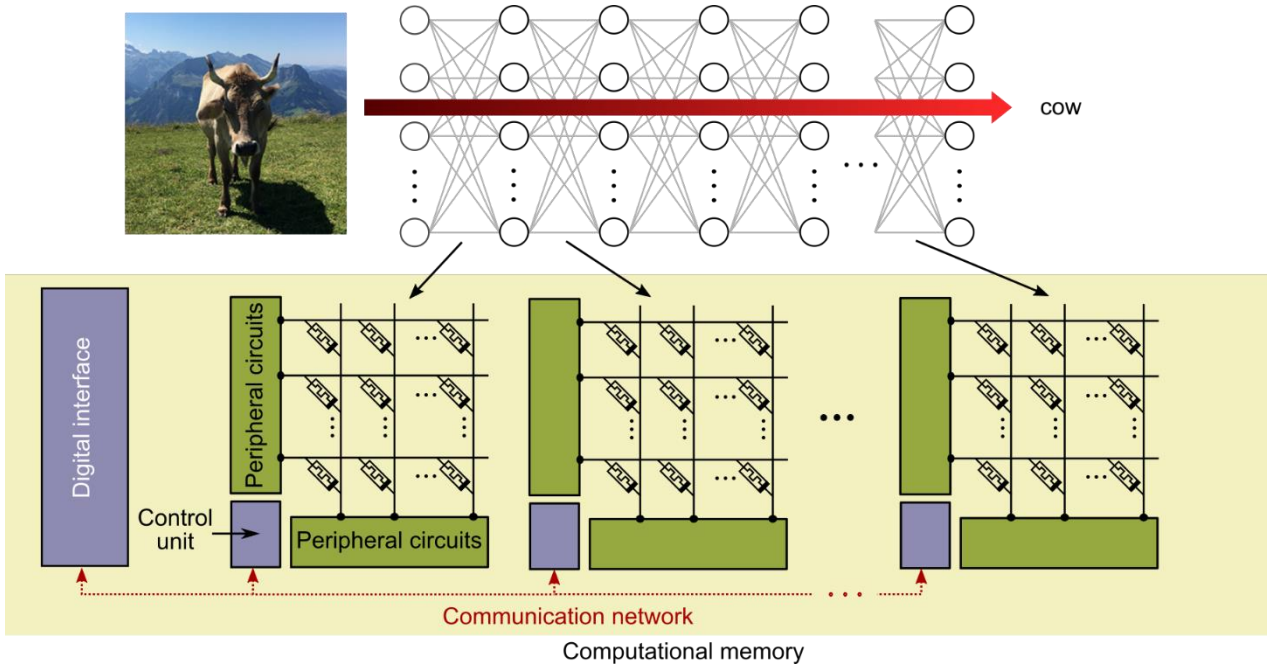



- Estimated **power reduction of 50x** compared to using an optimized 4-bit FPGA matrix-vector multiplier that delivers same reconstruction accuracy at same speed

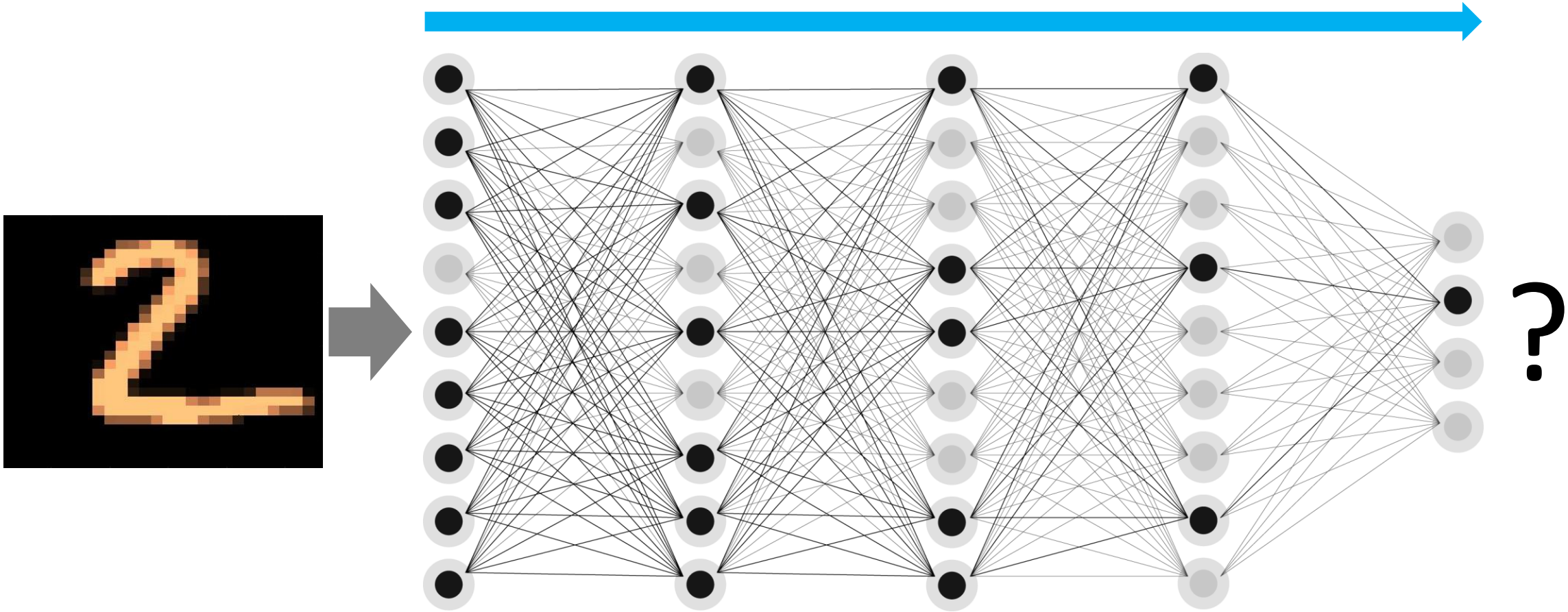
*Le Gallo et al., Proc. IEDM (2017)  Le Gallo et al., IEEE Trans. Electr. Dev. (2018)*

# Deep Learning



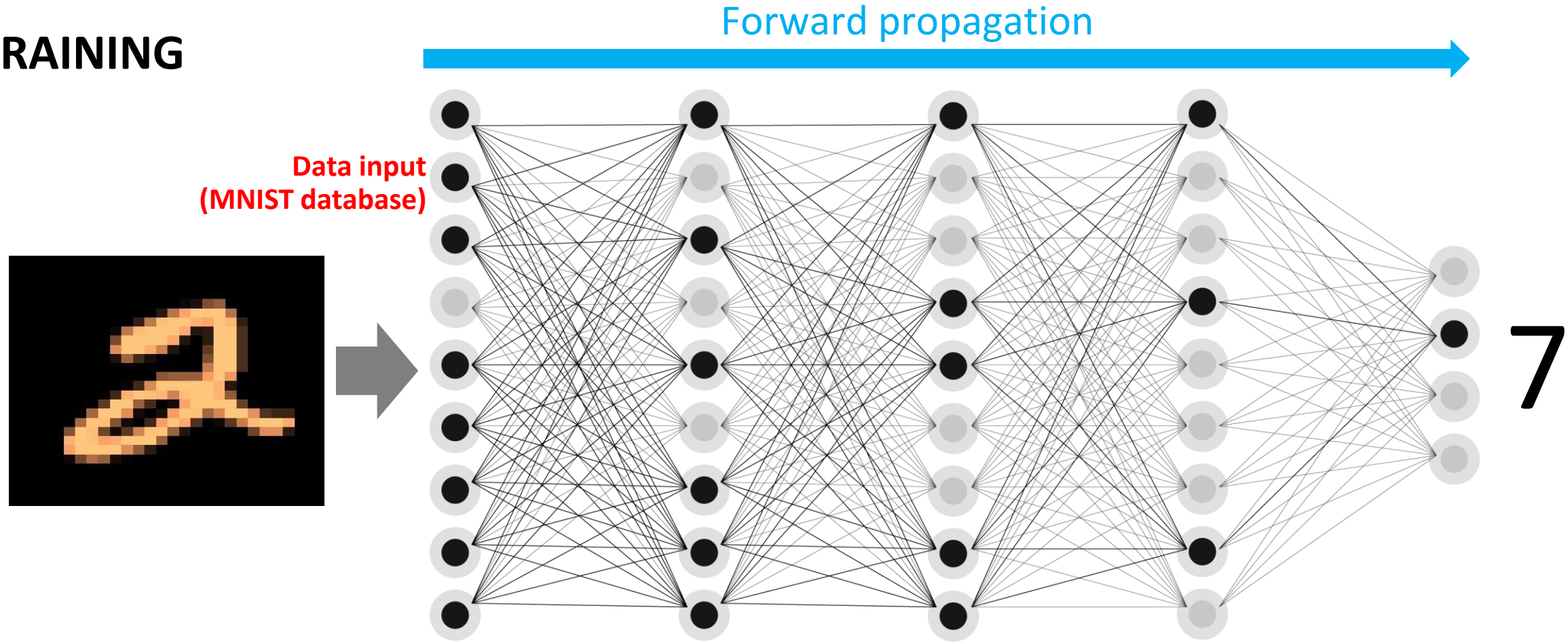

*Sebastian et al., VLSI (2019)*
*Eleftheriou et al., IBM JRD (2019)*
*Joshi et al., Nature Comm. (2020)*
*Nandakumar et al., Front. Neuroscience (2020)*

# Deep Learning: Training and Inference



# Deep Learning: Training and Inference

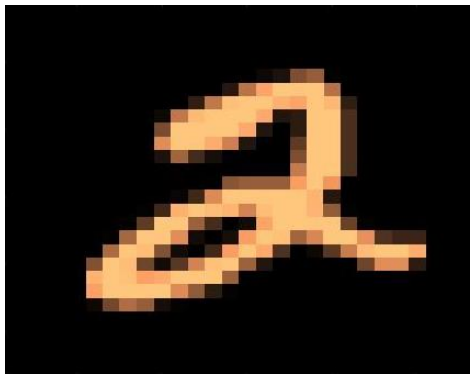
## TRAINING

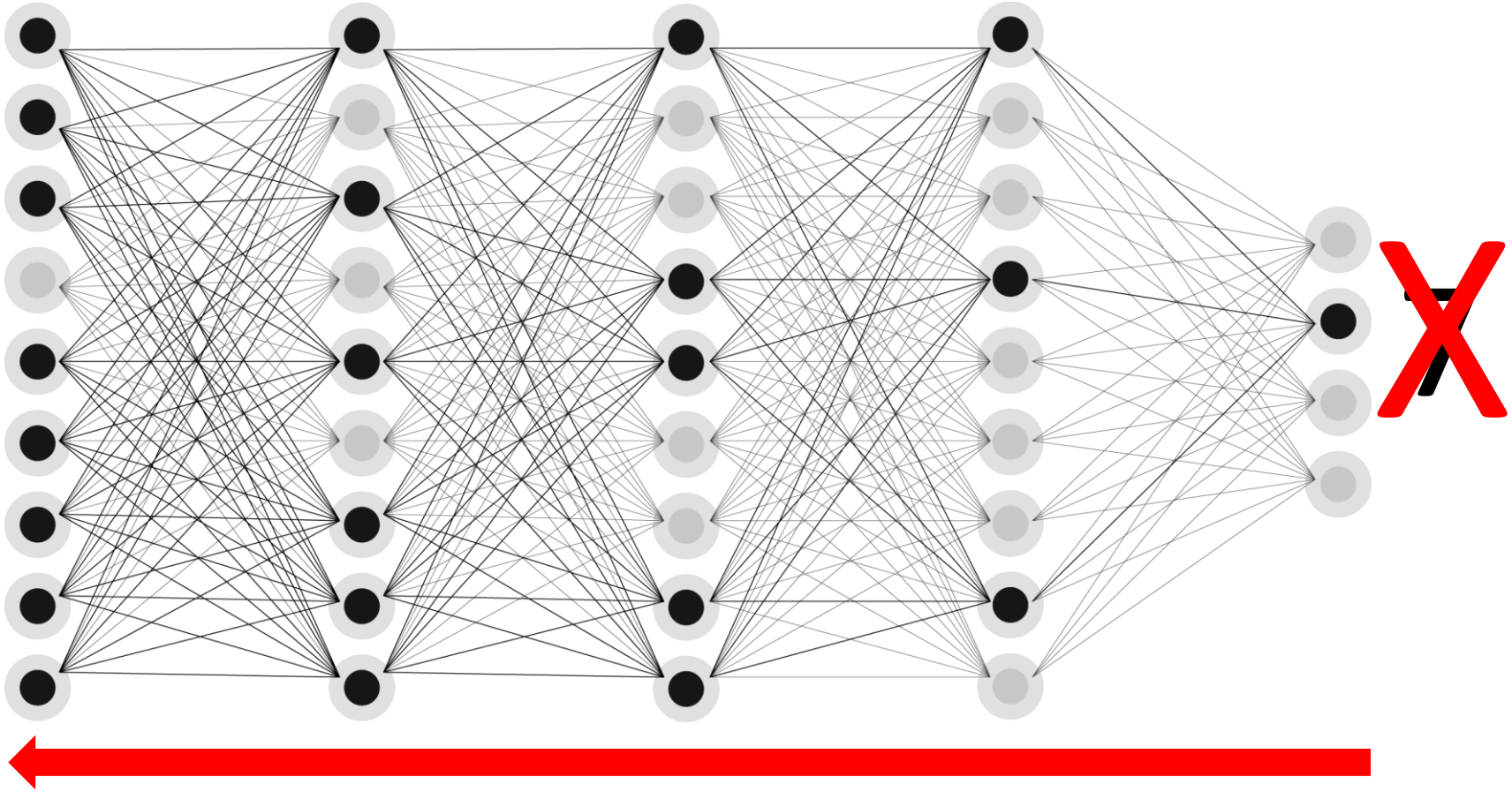


# Deep Learning: Training and Inference

## TRAINING

Data input
(MNIST database)



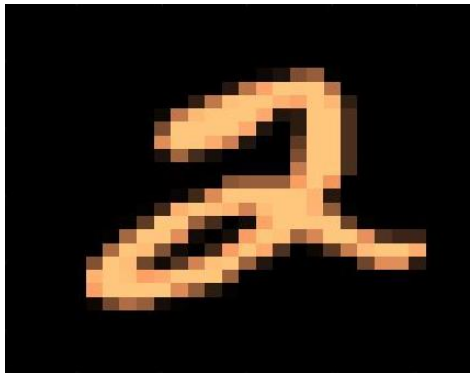


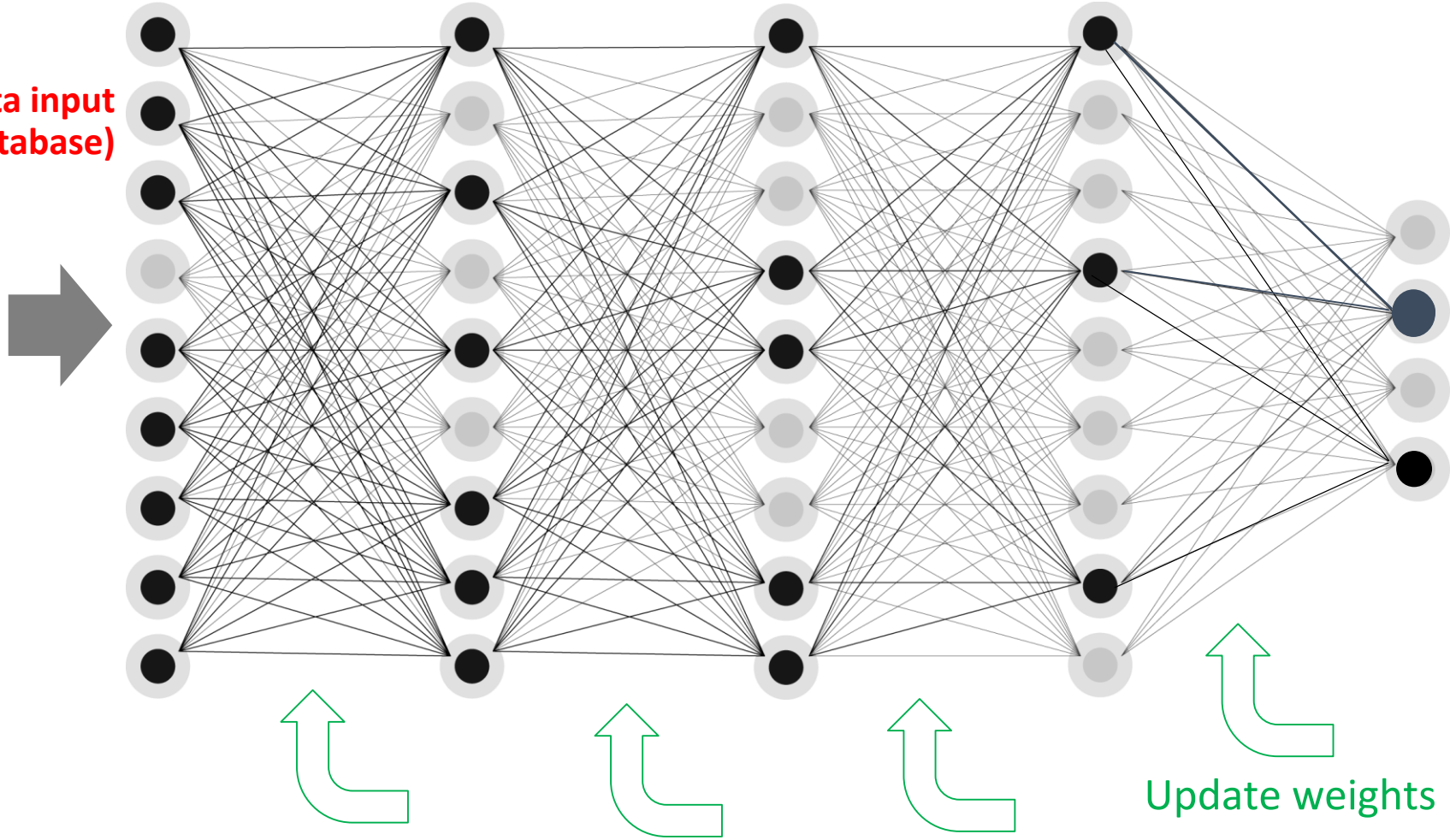
Backward propagation:  this is a two!

# Deep Learning: Training and Inference

## TRAINING

Data input
(MNIST database)

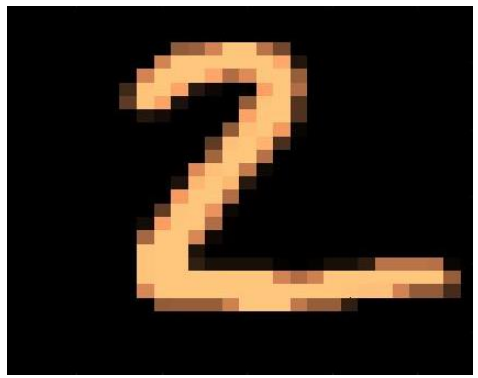


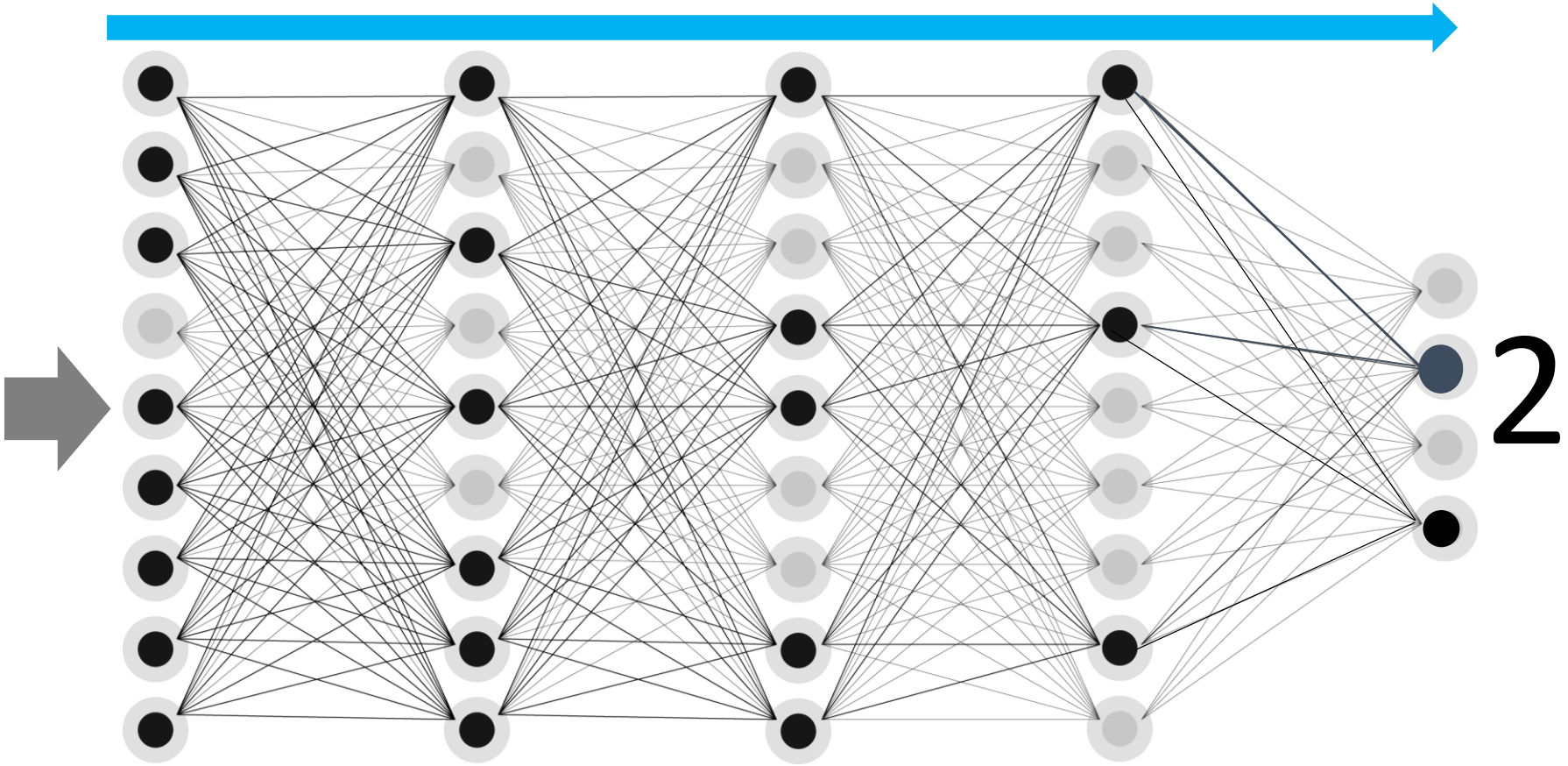


# Deep Learning: Training and Inference

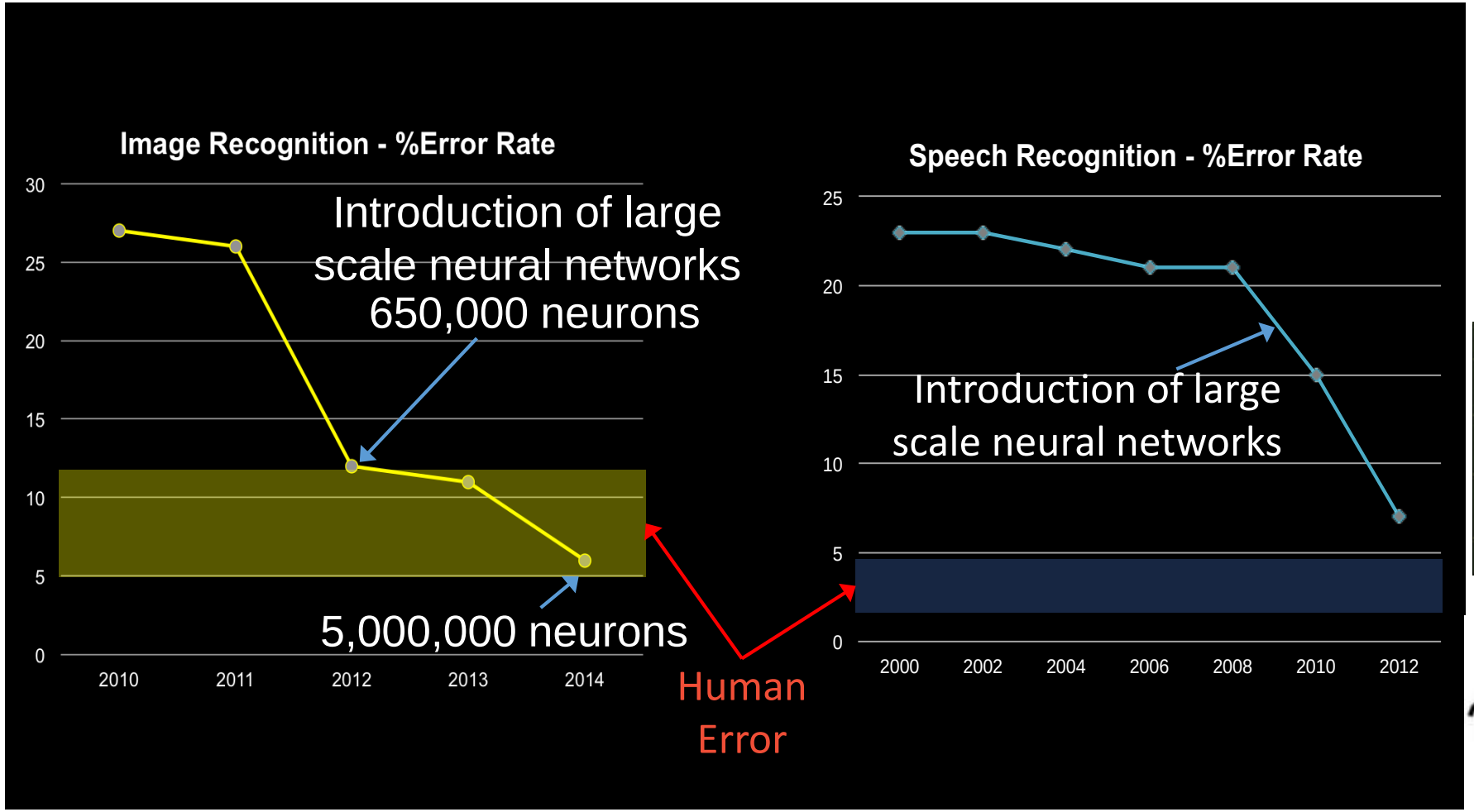
## INFERENCING

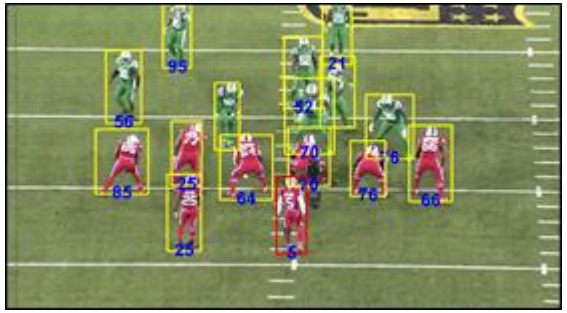
Fully Trained Network





# Deep Learning: Approaching human accuracy



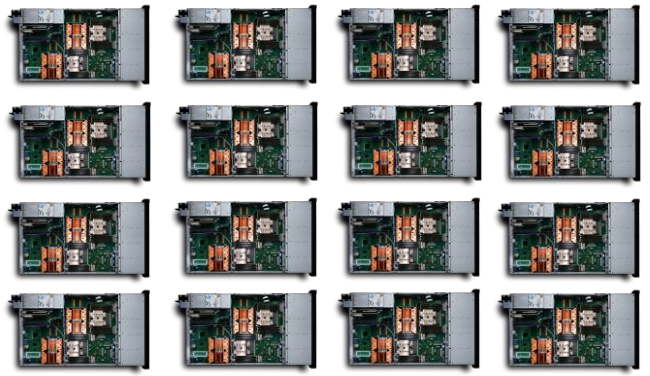

- Key reasons: **Large amounts of data and immense computing power**
- Significant role played by the semiconductor industry and computer architects!

# The computational efficiency problem of DL

**Training Image recognition model**
**Dataset: ImageNet-22K**
**Network: ResNet-101**

4 GPUs
16 days
~385 kWh

256 GPUs
7 hours
~450kWh



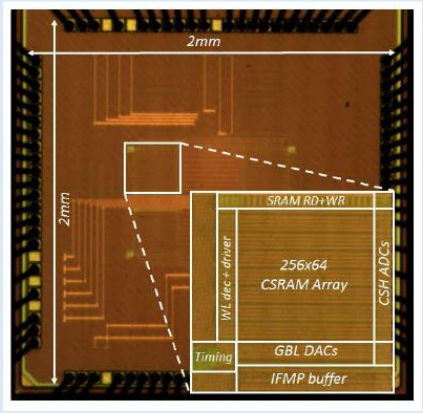
For reference: 1 model training run is ~2 weeks of home energy consumption

*https://arxiv.org/abs/1708.02188*

- Deep learning is **computationally intensive**
- Time consuming even with high-performance computing resources
- Power consumption **prohibitive for applicability in domains such as internet of things**

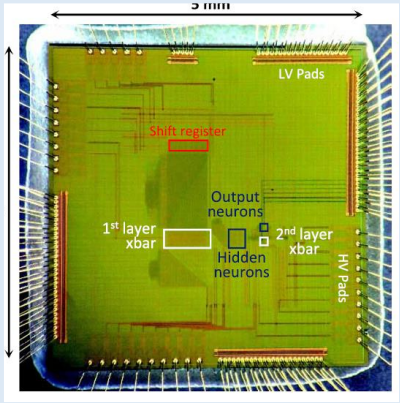
# Deep learning based on in-memory computing

## SRAM




Biswas et al., IEEE JSSC 2019
Valavi et al., IEEE JSSC, 2019

## Flash




Merrick-Bayat et al., IEEE TNNLS, 2017
Wang et al., IEEE TVLSI, 2019

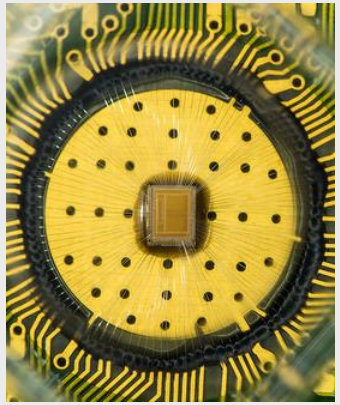
CHARGE-BASED MEMORY

RESISTANCE-BASED MEMORY

## RRAM

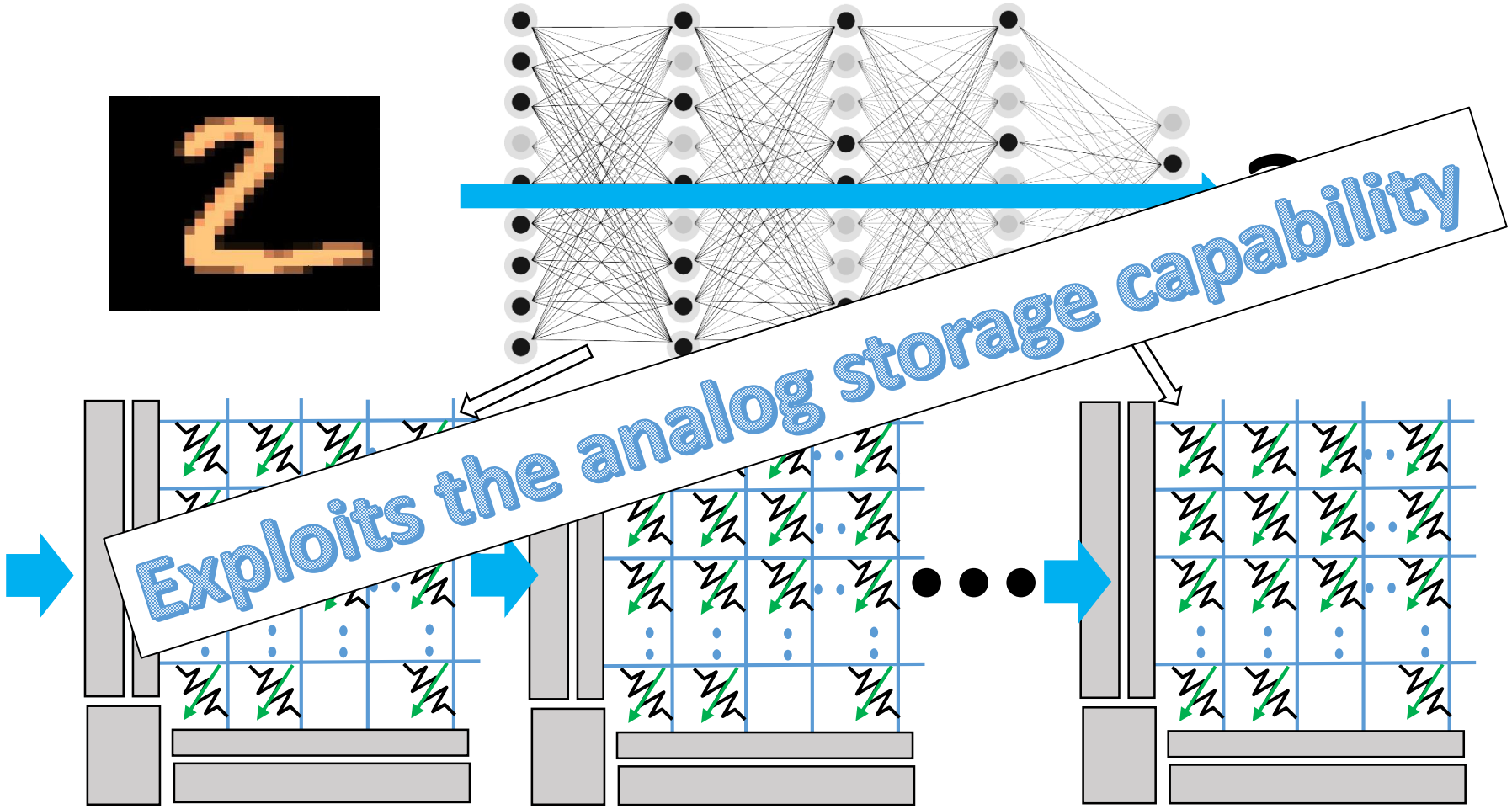

Hu et al., Adv. Mat., 2018
Xue et al., ISSCC, 2019

## PCM



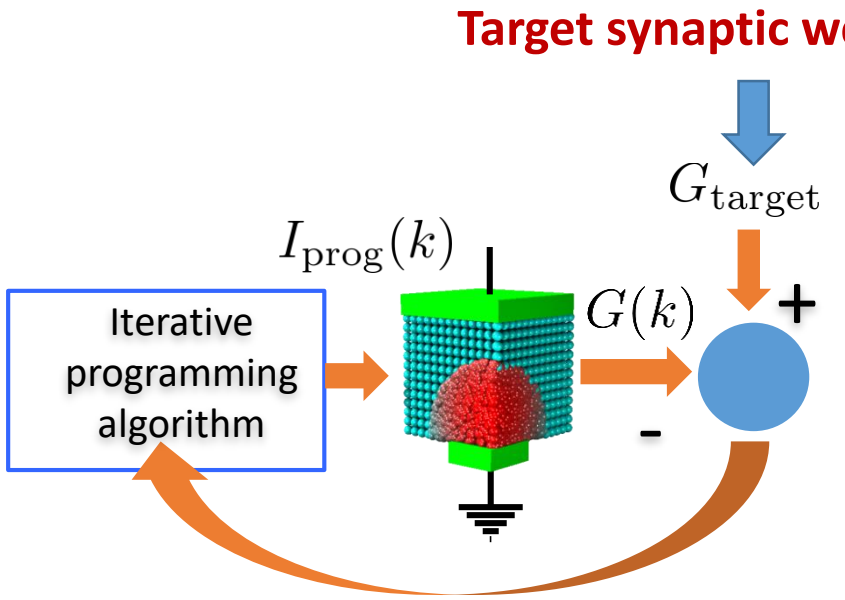
Ambrogio et al., Nature, 2018
Sebastian et al., VLSI, 2019

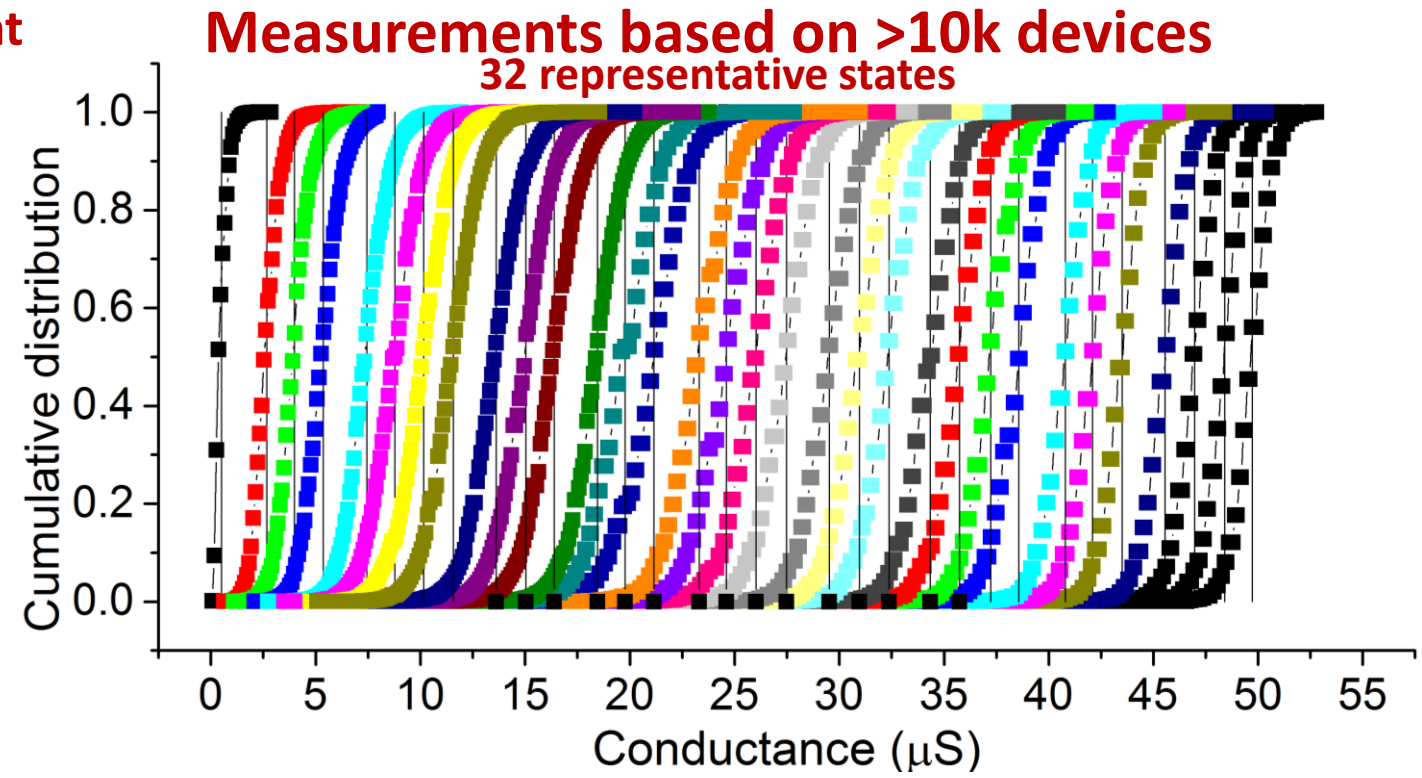
# Deep learning: Inference




The trained synaptic weights are mapped to an array of computational memory cores performing matrix vector multiply operations corresponding to each layer

# Mapping synaptic weights to PCM devices



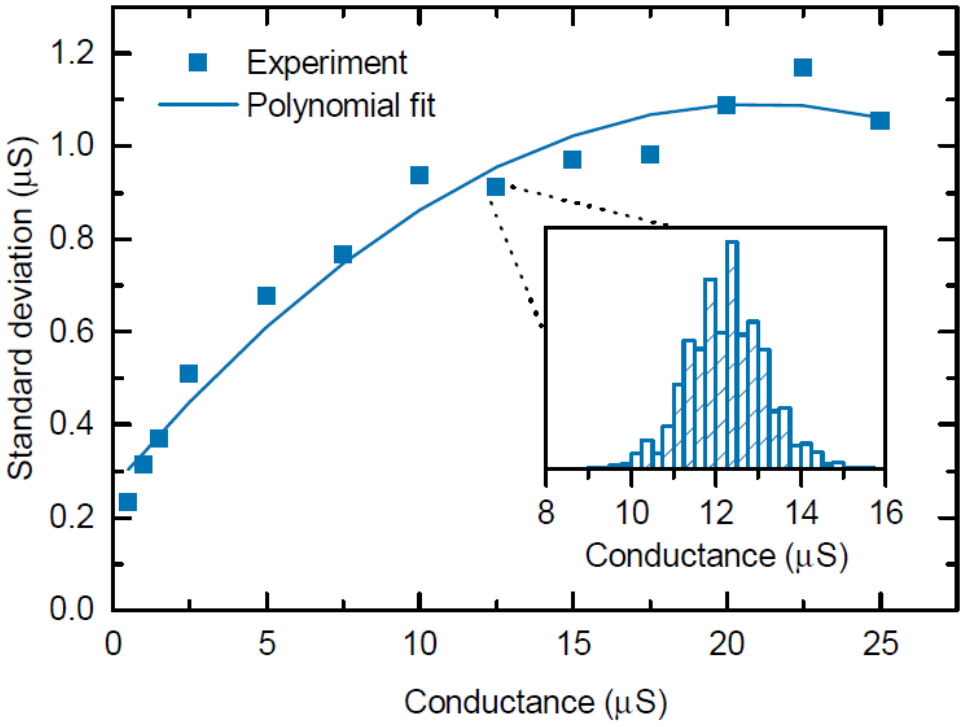



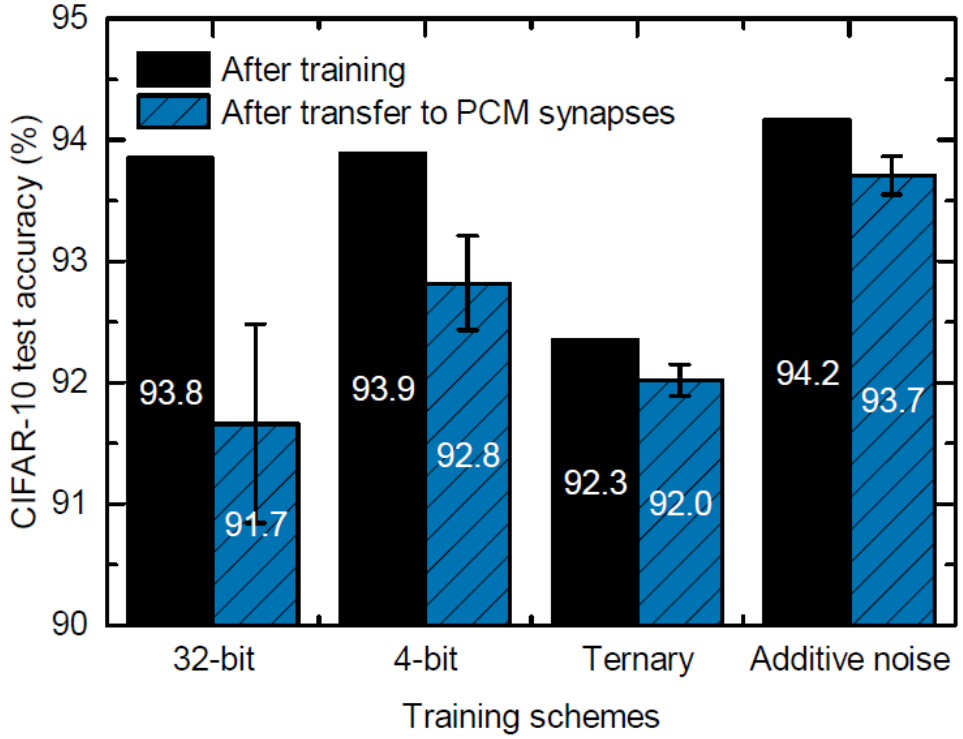

- **Iterative programming algorithms** used to achieve a target conductance value
- Non-ideal analog storage → Distribution of conductance values

*Papandreou et al., ISCAS (2011)*
*Sebastian et al., E/PCOS (2016)*

# Mapping synaptic weights to PCM devices

**ResNet-32 on CIFAR-10**

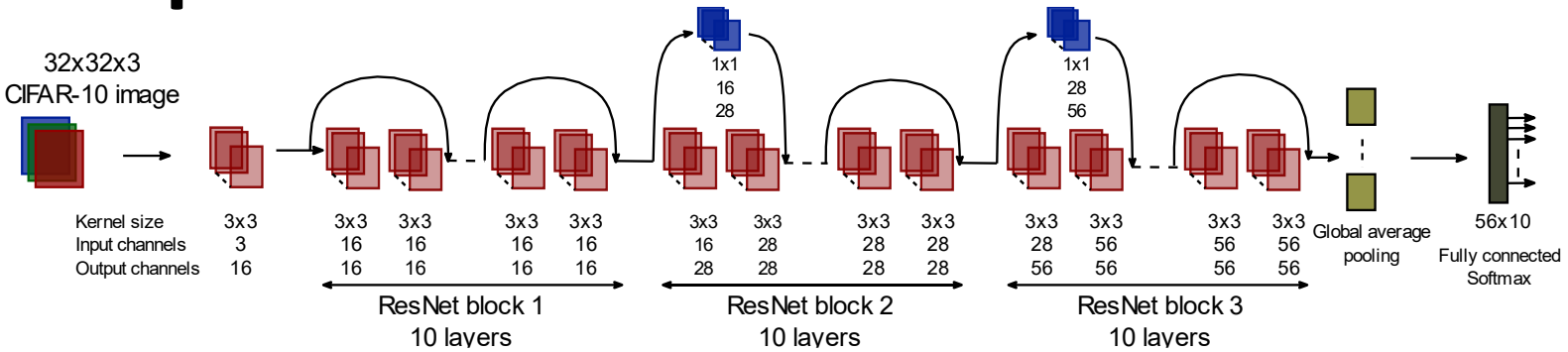



- A **custom training approach** needed to account for the conductance distributions
- Possible to achieve software-equivalent classification accuracies

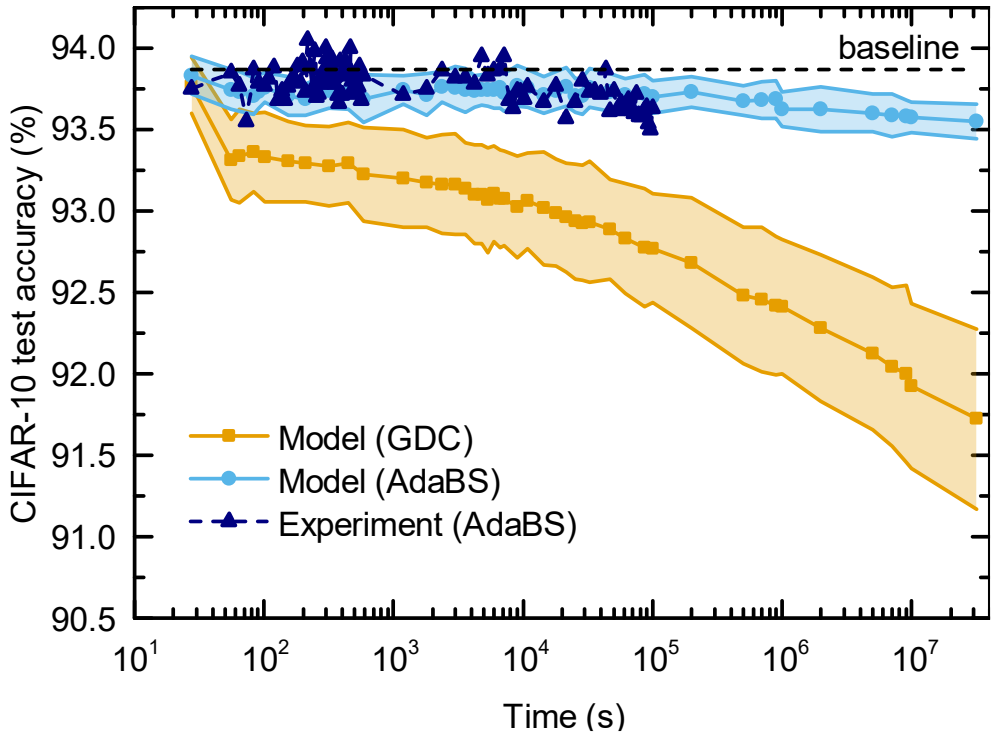
*Joshi et al., Nature Comm. (2020)*

# Inference experiment: ResNet-32 on CIFAR-10




**723,444 PCM devices (1T1R)**

*Joshi et al., Nature Comm. (2020)*

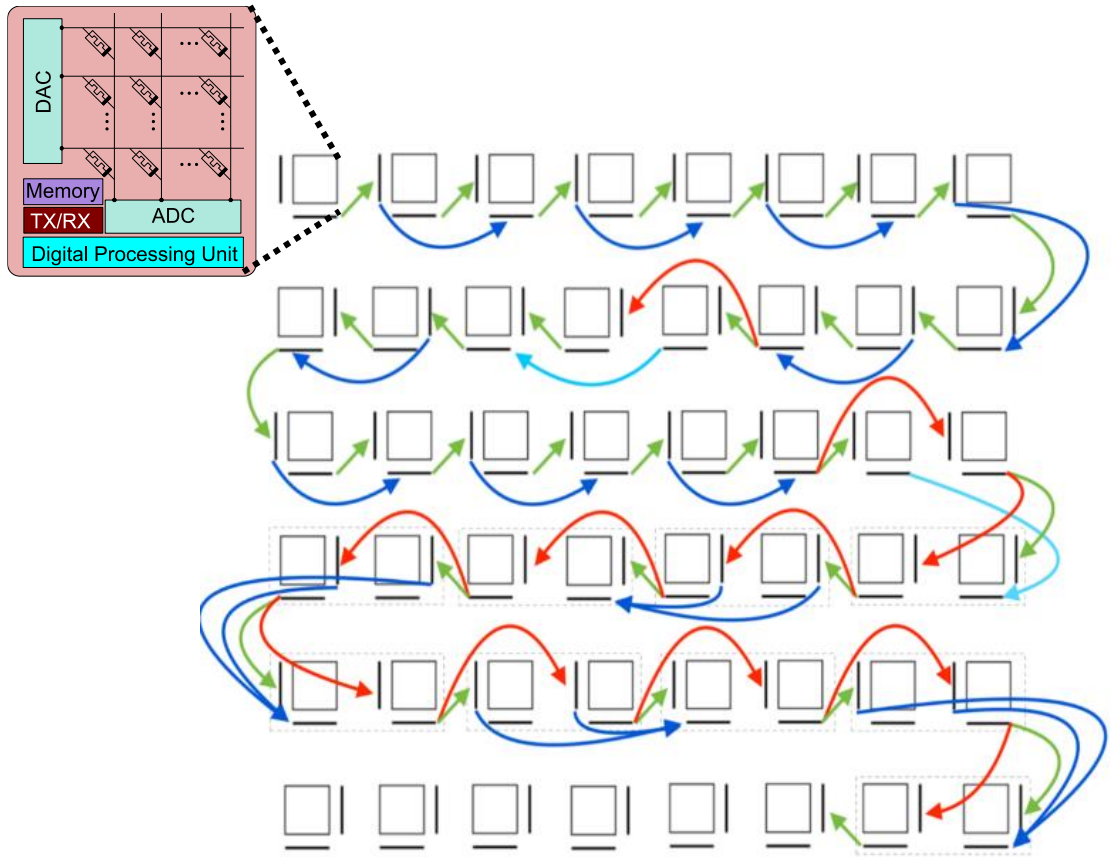



- With a **custom noise-injective training**, software equivalent accuracies can be achieved

# System integration: Communication fabric

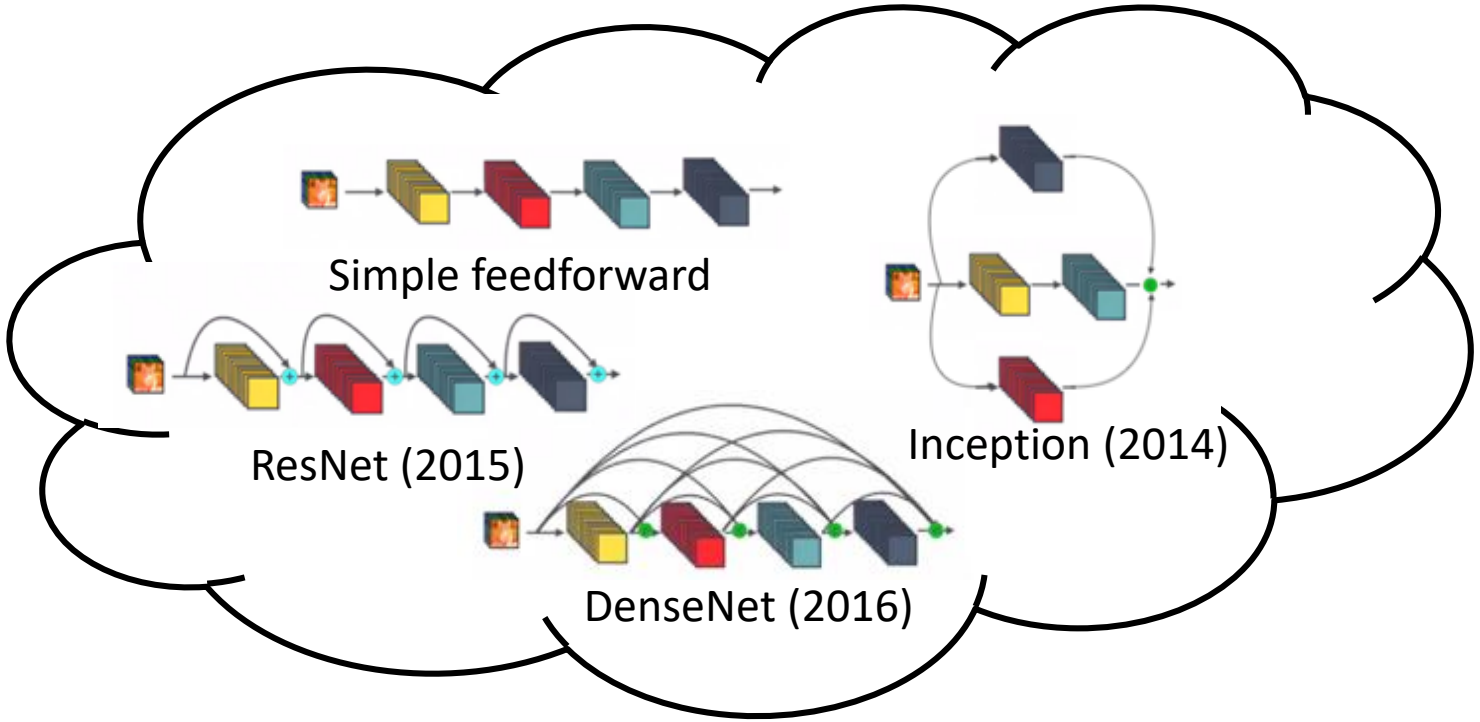
- Compared to all-digital implementations, in-memory computing is more amenable to highly pipelined dataflows

- Communication fabric should facilitate efficiently movement of activations from one computational memory unit to another

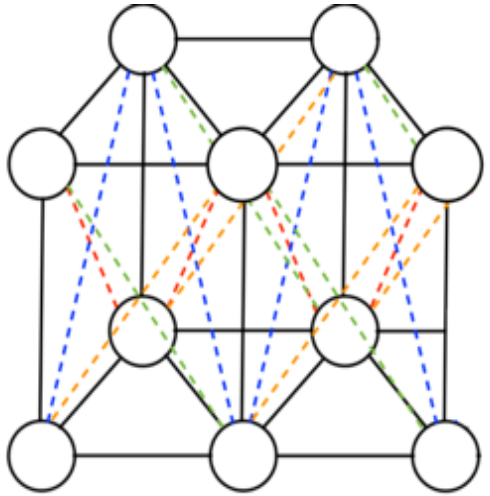


# Communication fabric for CNNs

**Consolidated graph representation of CNNs**



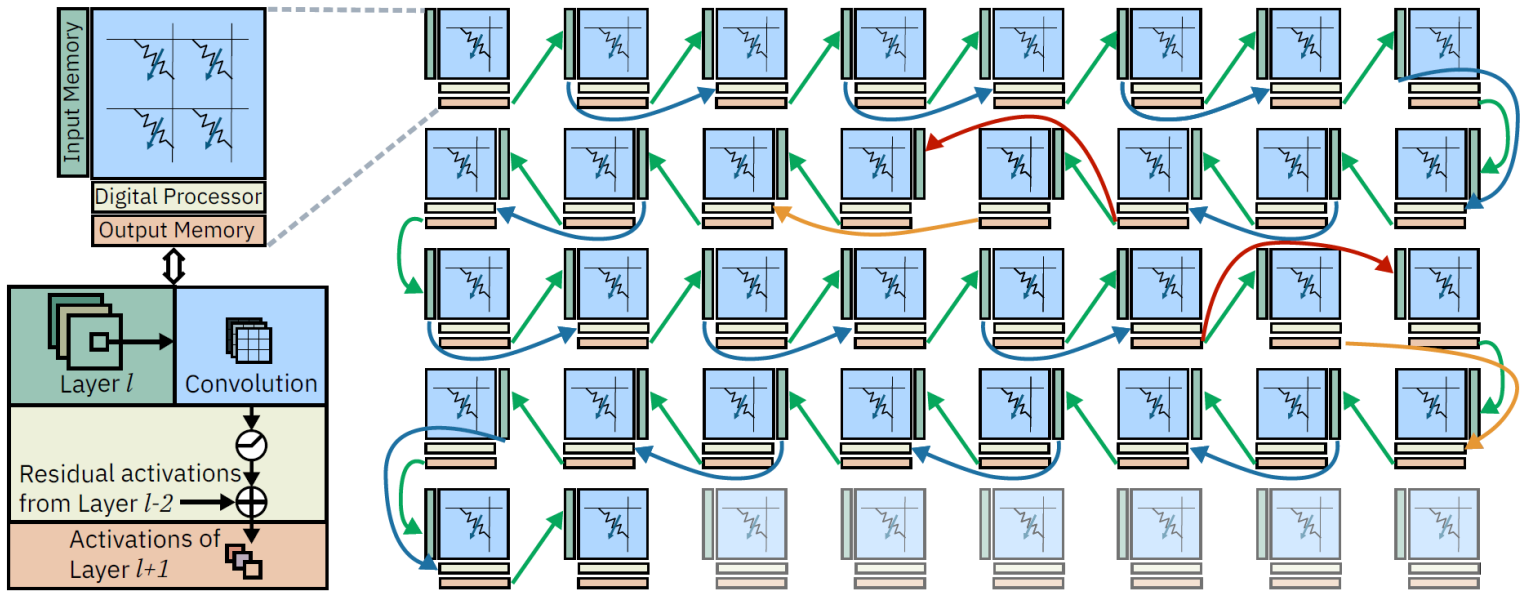

**5 Parallel Prism topology**



***Dazzi et al., MLSys Workshop @NeurIPS, 2019***

- The key distinguishing feature of the various CNN architectures is their connectivity
- Obtain a consolidated graph representation for all state-of-the-art CNNs (***C***)
  - Vertices represent convolution layers
  - Edges represent activations
- Communication fabric with 5-parallel prism topology (***F***)
- CNN executable in a pipelined fashion on F if there exists a homomorphism ***h: C → F***

# Mapping of ResNet-32 on an array of CM cores



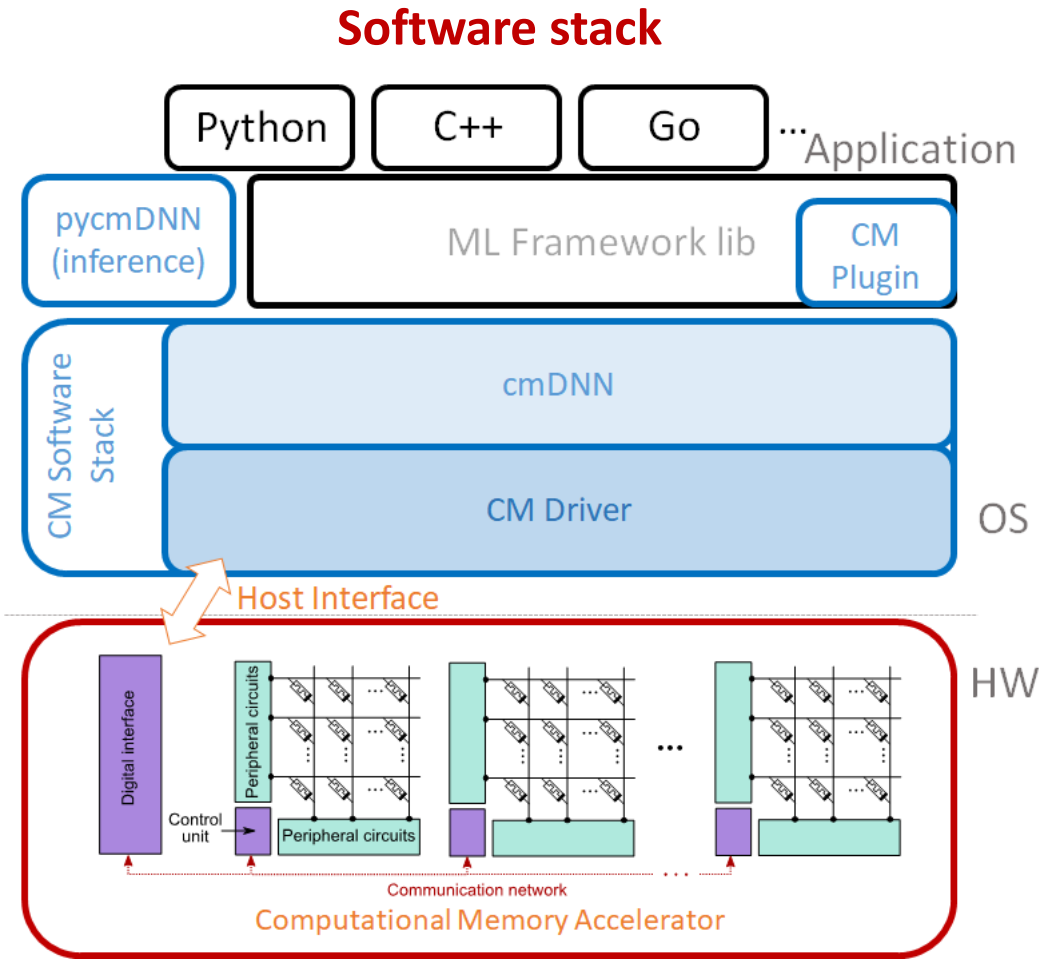

- The communication fabric is 5PP
- The links active in this implementation verify the homomorphism
- Each CM core has modest digital processing capability and its own input and output memory
- Each core stores the weights corresponding to each convolution layer
- The input memory stores the pixel neighborhood required for the convolution and the result of the dot product computation is stored in the output memory
- The estimated throughput is 38600 Images/s!

# System integration: Software stack

- Essential to develop a software stack that can compile a NN model into operators suitable for the accelerator
  - ✓ Compile the model into optimized operations and routing
  - ✓ Orchestrates the data movement to and from the accelerator

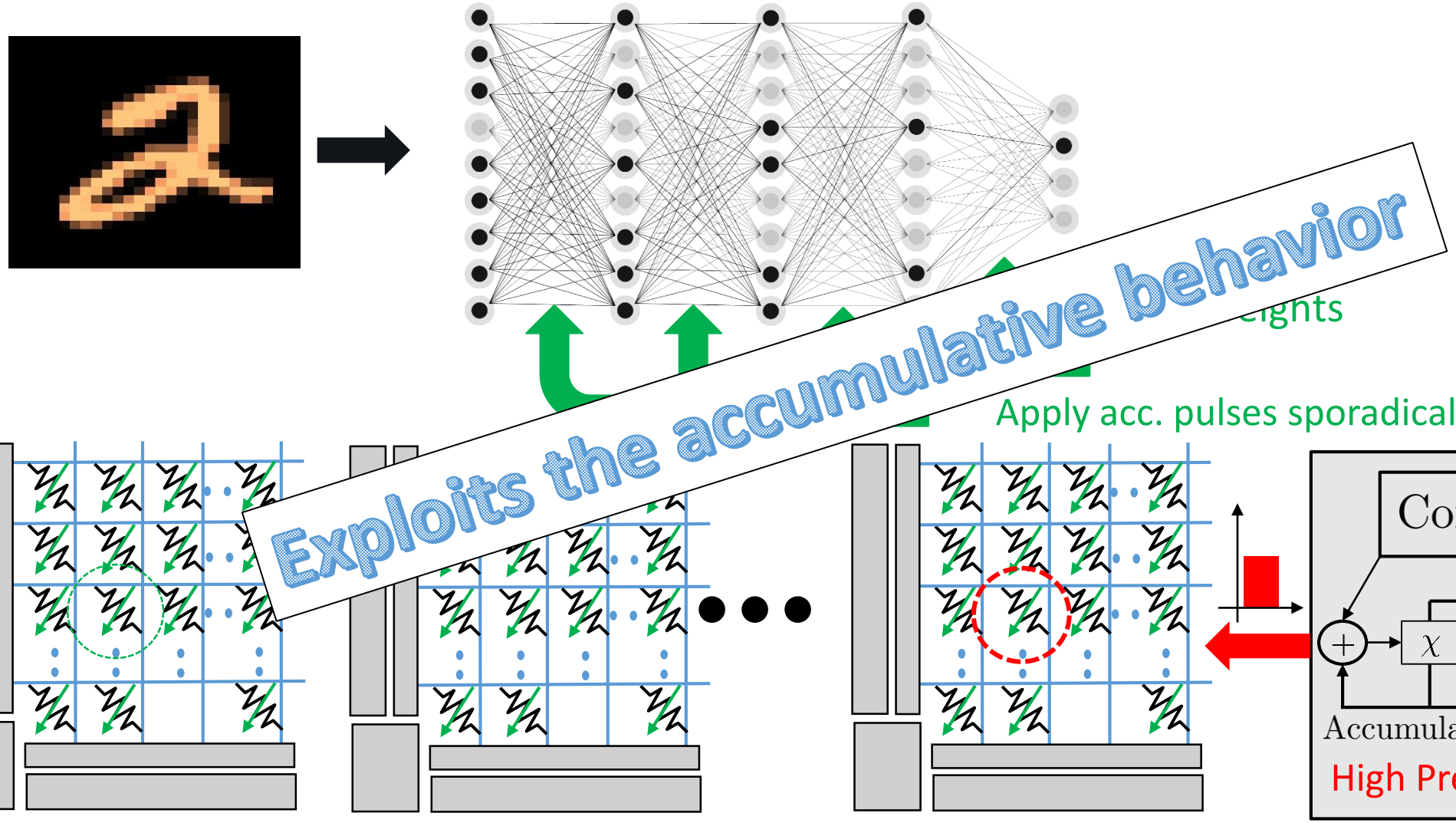
- Three essential software components
  - ✓ The computational memory OS driver
  - ✓ The computational memory compiler
  - ✓ A library that allows inference hiding low-level details

- A non-trivial task and fertile area of research




*Eleftheriou et al., "Deep learning acceleration based on in-memory computing", IBM J. Res. Dev., 2019*
*Kourtis et al., "Compiling neural networks for a computational memory accelerator", Proc. SPMA (EuroSys), 2020*

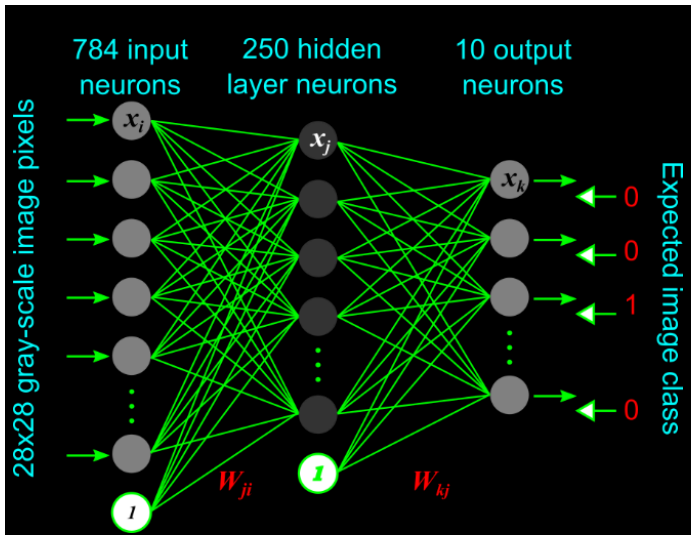
# Deep learning training

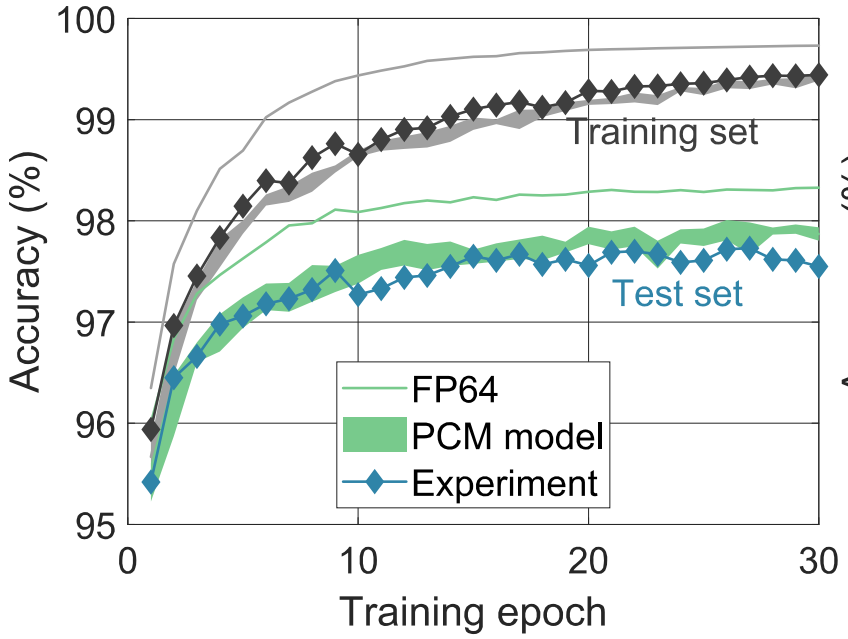


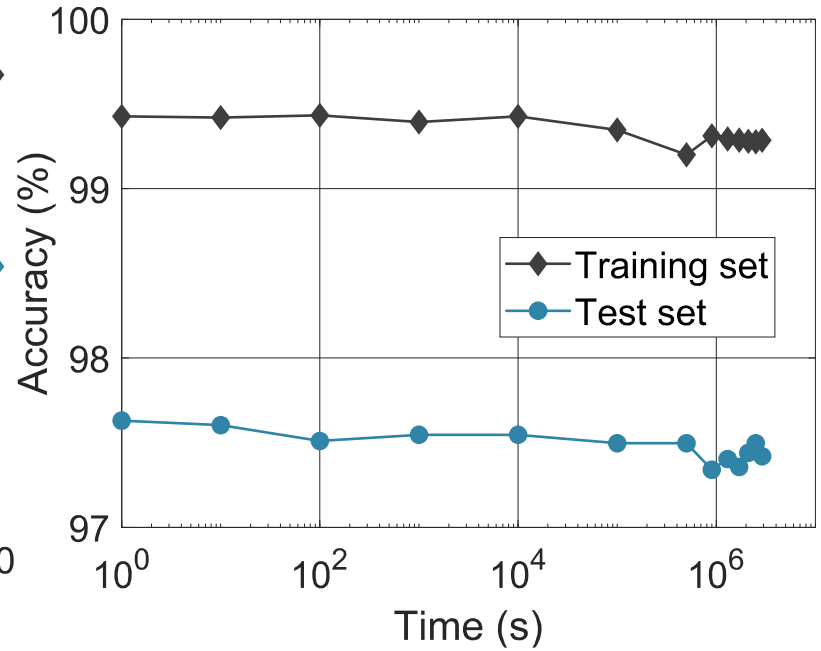

*Nandakumar et al., ArXiv, 2017*

*Sebastian et al., VLSI, 2019*

# Mixed-precision training: Experimental results



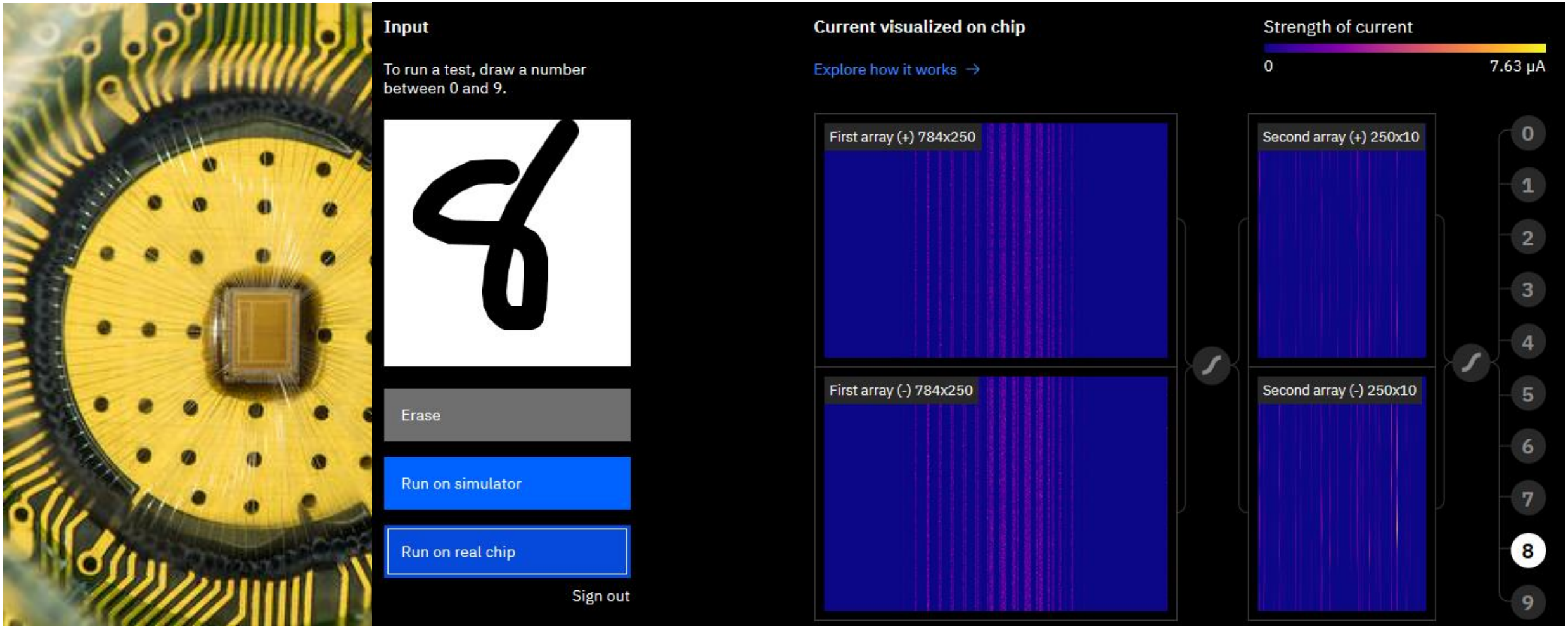




- Each synaptic weight mapped to two PCM devices (~400,000 PCM devices)
- Comparable test accuracy as FP32 training
- Negligible accuracy drop during inference after training

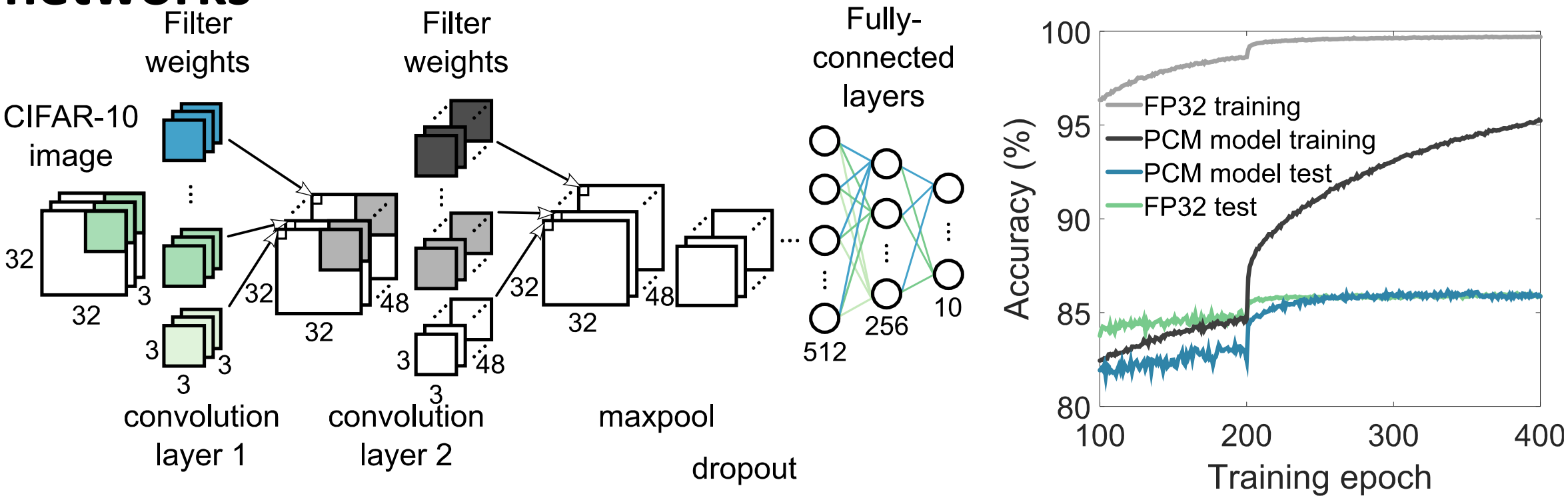
https://analog-ai-demo.mybluemix.net

# Cloud Demo

https://analog-ai-demo.mybluemix.net



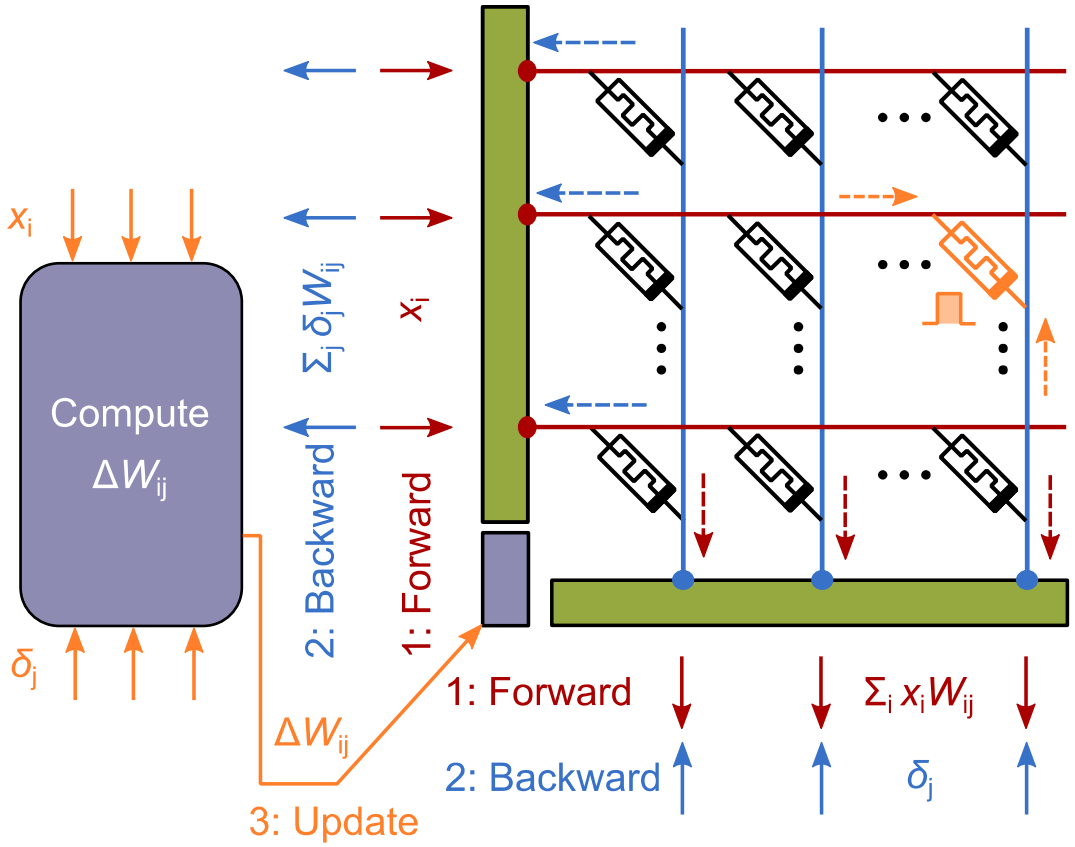
# Mixed-precision training: Extension to larger networks




- Convolutional neural network with approx. 1.5 Million parameters
- Better generalization that FP32 based training due to the use of stochastic devices
- Also applicable to long-short term memory (LSTM) networks and generative adversarial networks (GANs)

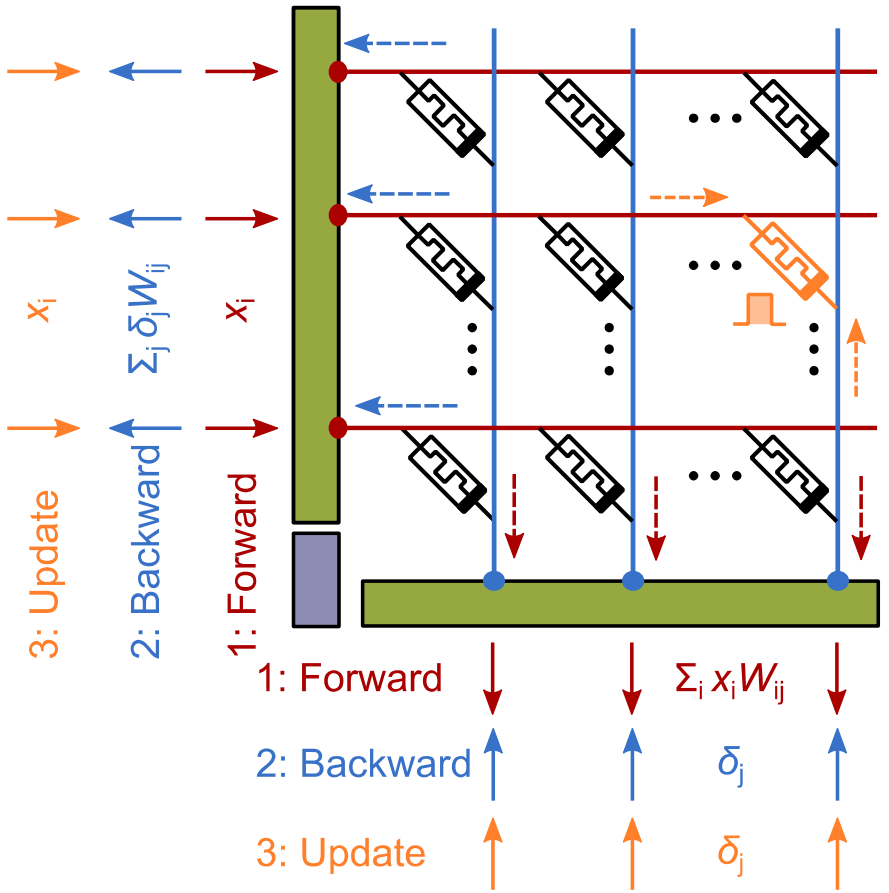
# Deep learning training: Other approaches

## Approach I (Mixed-precision)




*Prezioso, Nature (2015)*
*Nandakumar et al., Proc. ISCAS (2018)*
*Yu, Proc. IEEE (2018)*
*Eleftheriou et al., IBM JRD (2019)*

## Approach II (In place weight update)




*Alibart et al., Nature Comm. (2013)*
*Gokmen and Vlasov, Front. Neuroscience (2016)*
*Ambrogio et al., Nature (2018)*
*Gokmen and Haensch, Front. Neuroscience (2020)*

# Spiking Neural Networks (SNNs)

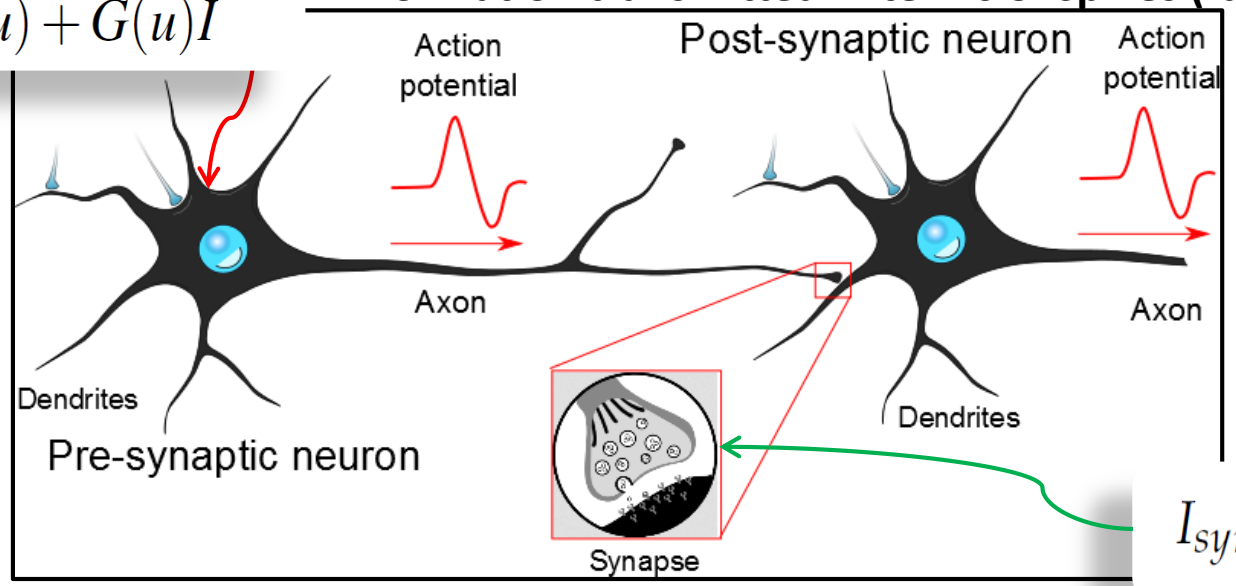
**Information transmitted as floating point numbers!**

**Conventional Neural Networks**


**Neuronal dynamics**

$$du/dt = F(u) + G(u)I$$

**Information transmitted in terms of spikes (rate, timing etc.)**




- **Asynchronous**
- **Local, event-based learning**
- **Employed by the brain**

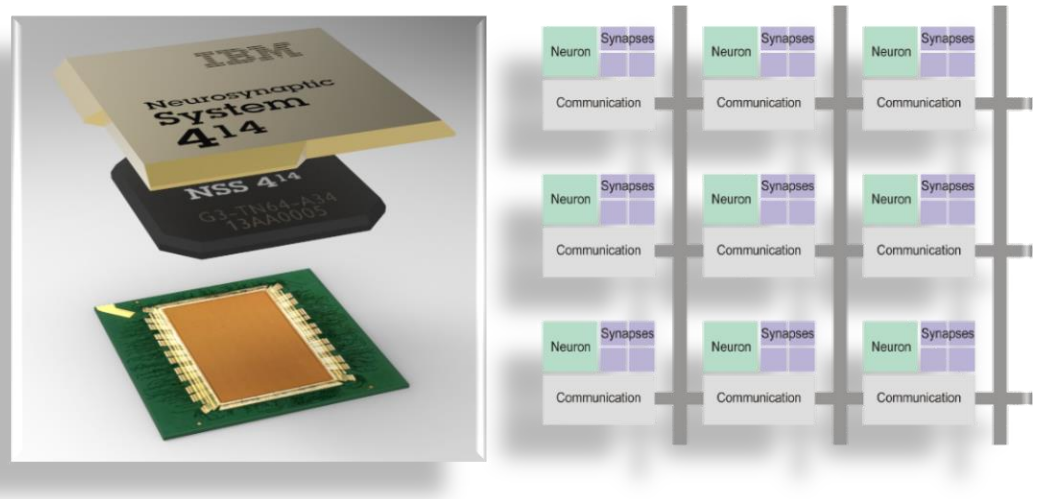
**Synaptic dynamics**

$$I_{syn} = g_{syn} S(V - E_{syn})$$

- Objective 1: **Exploit the temporal coding and synaptic/neuronal dynamics** to transcend deep learning
- Objective 2: **Develop computing substrates** for efficient realization of neuronal and synaptic dynamics

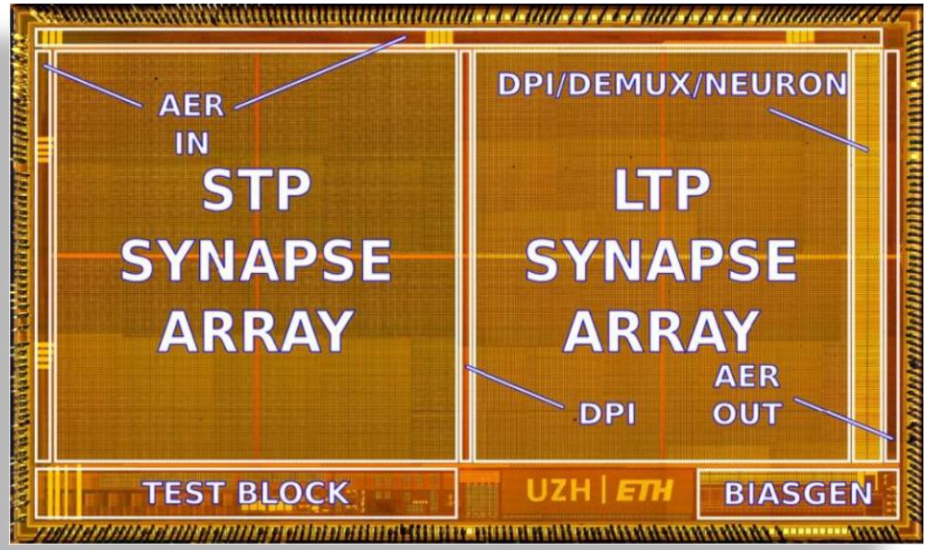
***Pfeiffer and Pfeil, Front. Neuroscience (2018), Rajendran et al., IEEE SP Magazine (2019)***

# SNN co-processors (Digital and Analog CMOS-based)




- Computation of neuronal and synaptic dynamics in **digital CMOS circuitry**

*Merolla et al., Science (2014)*
*Davies et al., IEEE MICRO (2018)*

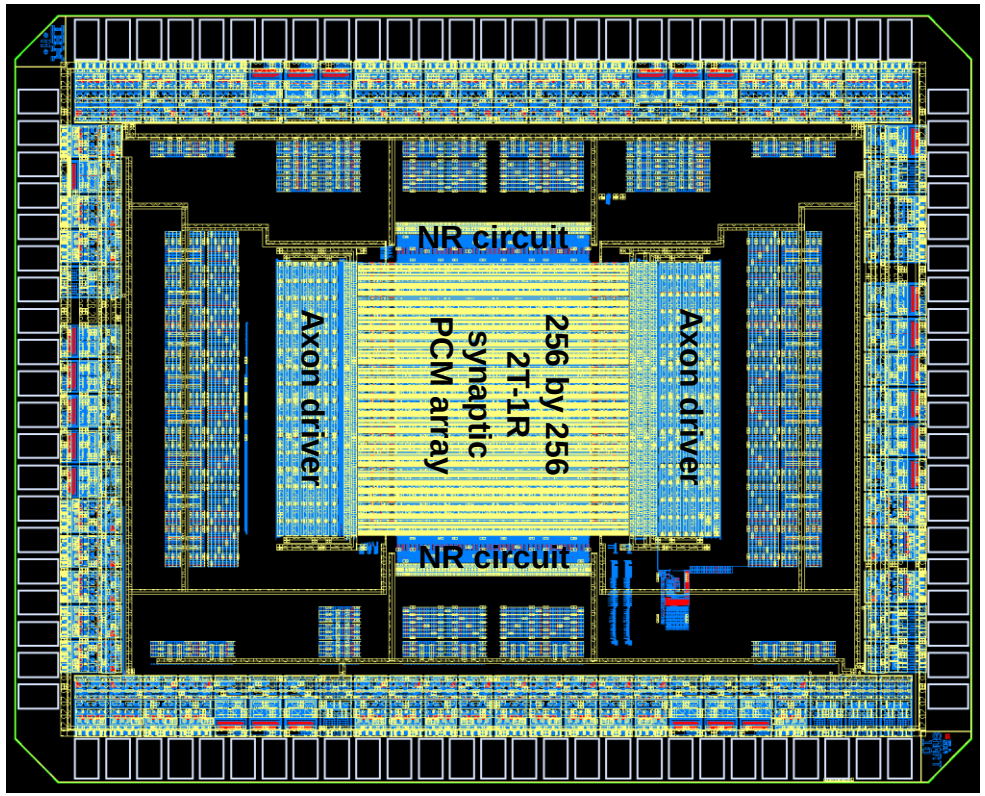



- Exploit **subthreshold MOSFET characteristics** to directly emulate neuronal and synaptic dynamics

*Benjamin et al., Proc. IEEE (2014)*
*Qiao et al., Front. Neuroscience (2015)*
*Moradi et al., IEEE Trans. Biomed. Circuits Syst. (2018)*

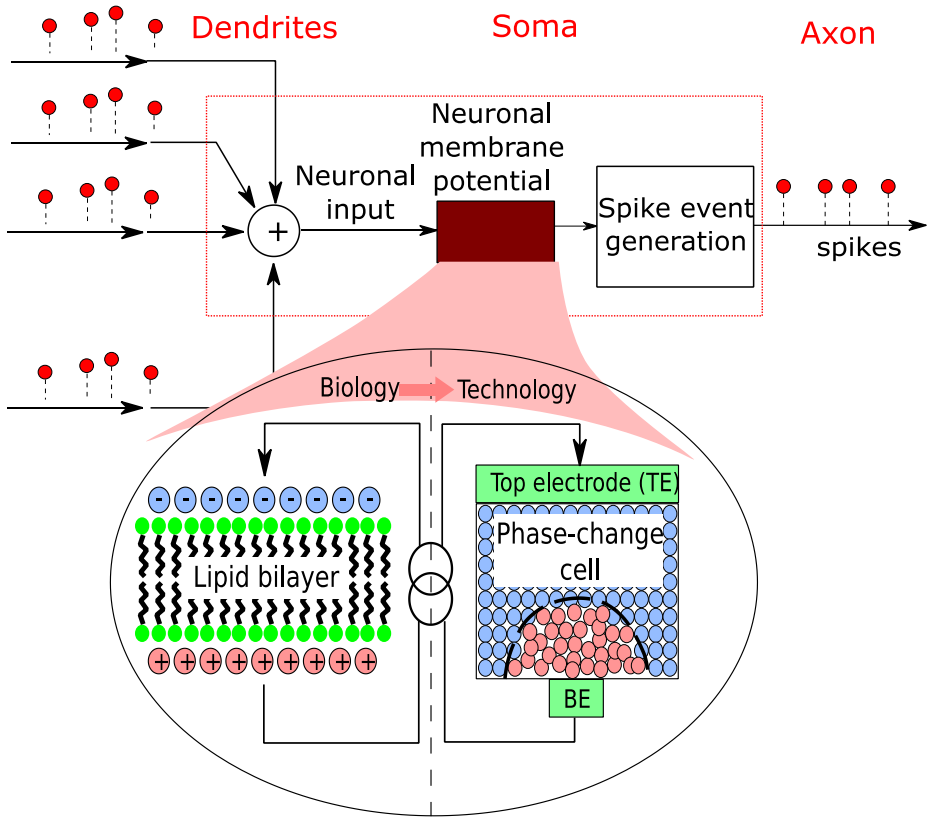
# SNNs using in-memory computing

## Synapse




*Yu et al., IEEE TED (2011)*
*Kuzum et al., NanoLetters (2012)*
*Kim et al., Proc. IEDM (2015)*
*Wang et al., Nature Mat. (2017)*

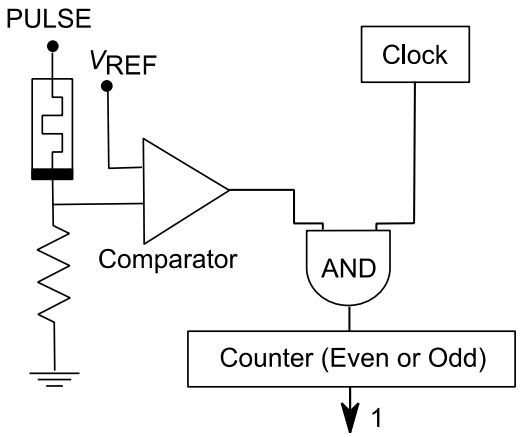
## Neuron




*Al-Shedivat et al., IEEE Trans. Emerg. Sel. Topics Circuits Syst. (2015)*
*Tuma et al., Nature Nanotech. (2016)*
*Mehonic and Kenyon, Front. Neurosci. (2016)*

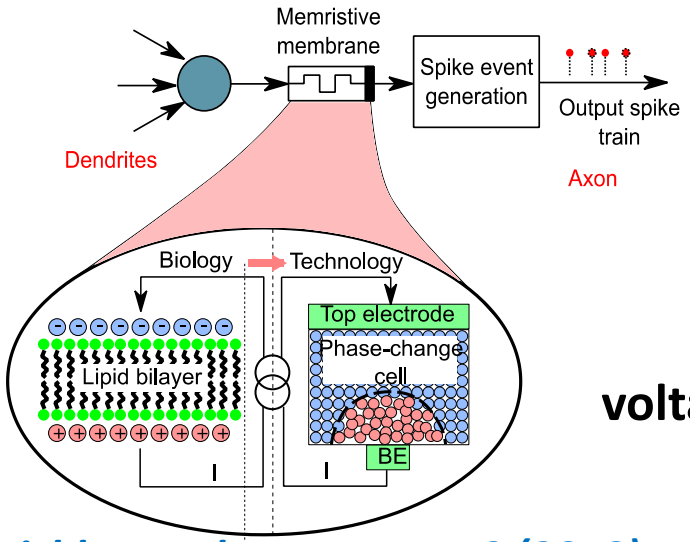
# Stochastic computing and security

## True random number generator



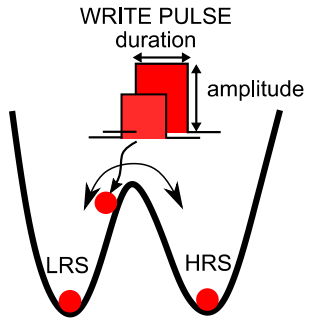

*Jiang et al., Nature Comm., 8 (2017)*

## Stochastic neurons/synapses




*Bichler et al., IEEE TED, 59 (2012)*
*Tuma et al., Nature Nano., 11 (2016)*

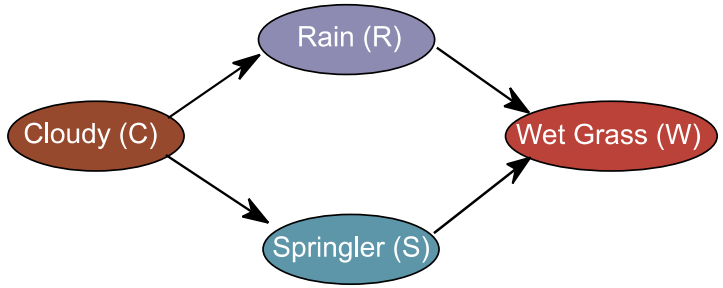
**Inherent stochasticity associated with resistance switching**




*Carboni & Ielmini, Adv. Electr. Mat. (2019)*

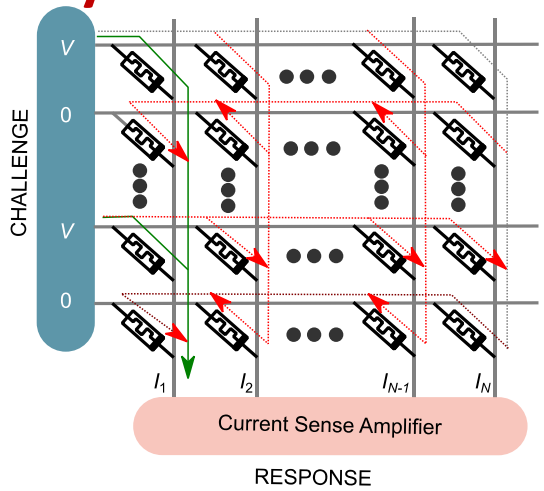
**Ability to control the stochasticity via the voltage/duration of write pulses!**

## Probabilistic inference




*Shim et al., Sci. Reports, 7 (2017)*
*Mizrahi et al., Nature Comm 9 (2018)*

## Physical unclonable function



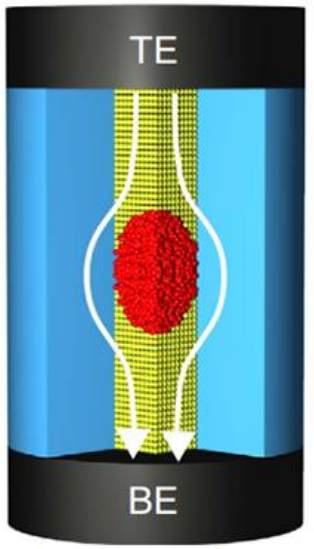

*Nili et al., Nature Electr., 1 (2018)*

# Outline

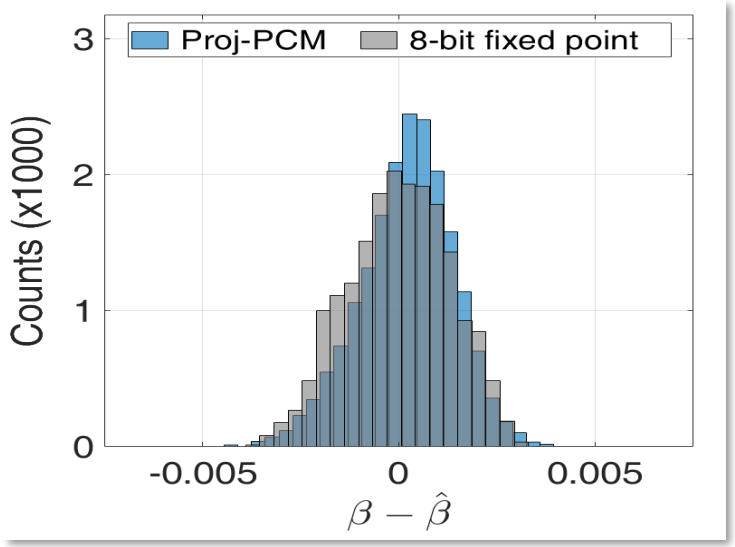
- Introduction
- Memory devices and computational primitives
  - ✓Charge-based memory devices & Computational primitives
  - ✓Resistance-based memory devices & Computational primitives
  - ✓Phase change memory: A prototypical resistance-based memory
- Applications
  - ✓Exploiting non-volatile binary storage
  - ✓Scientific computing
  - ✓Signal processing & Optimization
  - ✓Deep learning
  - ✓Stochastic computing and security
- Discussion
  - ✓Increasing the precision of in-memory computing
  - ✓Photonic in-memory computing
  - ✓Summary

# Increasing the precision of in-memory computing

**Non-ideal analog storage:** Projected phase-change memory



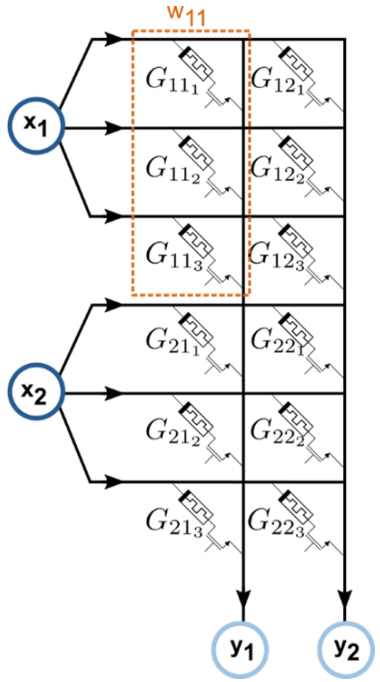

$R_{CRYST} << R_{PROJ}$
$R_{AMOR} >> R_{PROJ}$

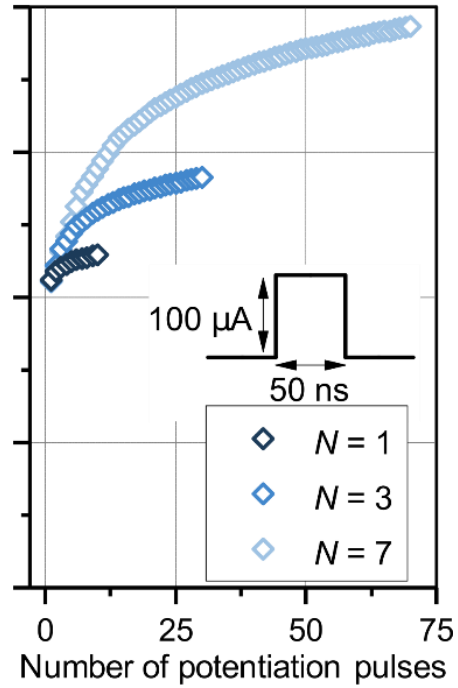



Sub 10fJ, 8b scalar multiplication without moving data!

*S. Kim et al., IEDM (2013)*
*Koelmans et al., Nature Comm. (2015)*
*Giannopoulos et al., IEDM (2018)*

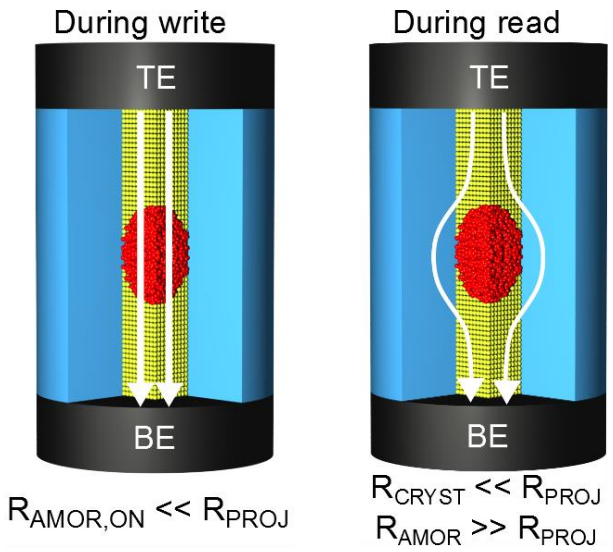
**Non-ideal accumulative behavior:** Multi-device synaptic architectures

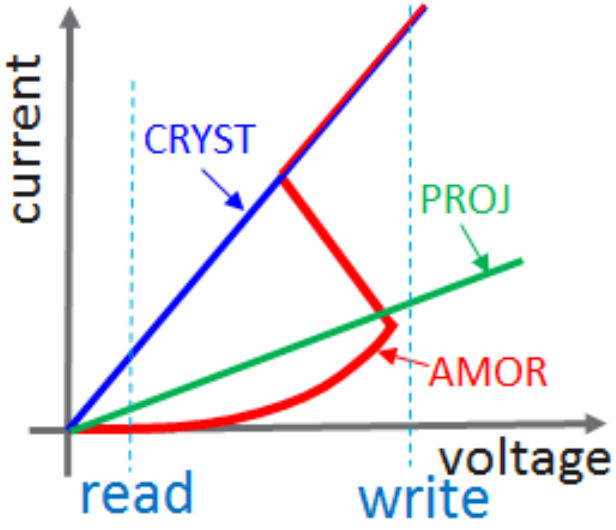


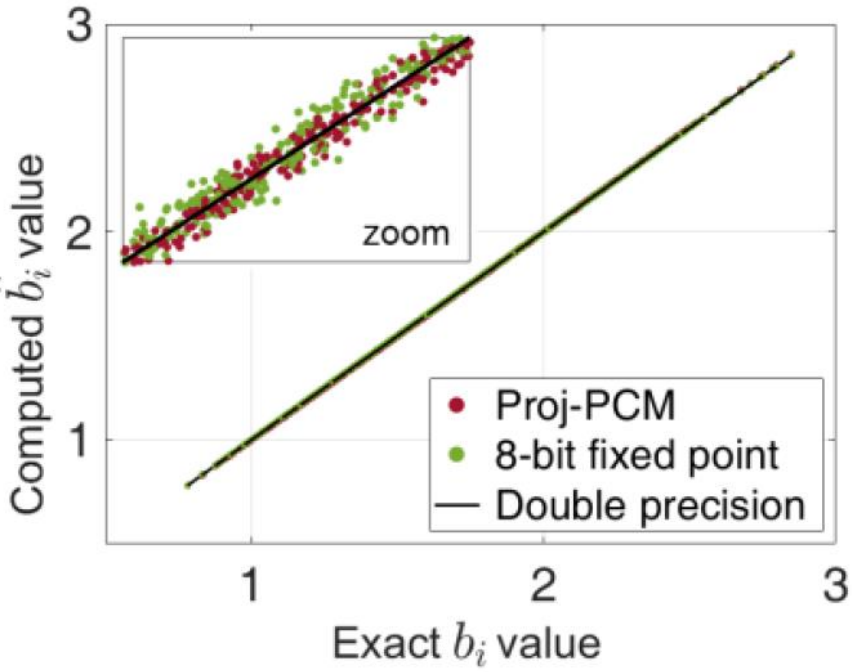




*Boybat et al., Nature Comm. (2018)*
*Boybat et al., Proc. ISCAS (2019)*

# Projected phase-change memory



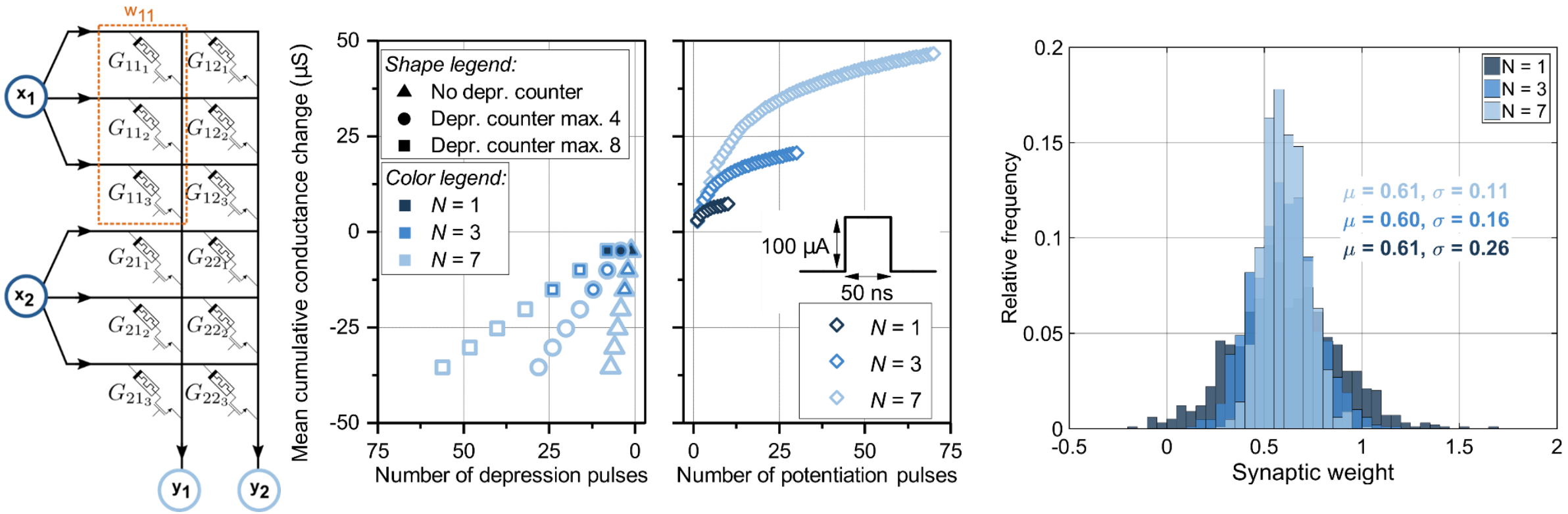




- Modified PCM device concept
- Exploits the I-V characteristic of phase change materials
- Substantially lower drift and conductance fluctuations arising from 1/f noise
- **Precision equal to 8-bit fixed-point arithmetic**

***Kim et al., Proc. IEDM (2013), Koelmans et al., Nature Comm. (2015), Giannopoulos et al., IEDM (2018)***

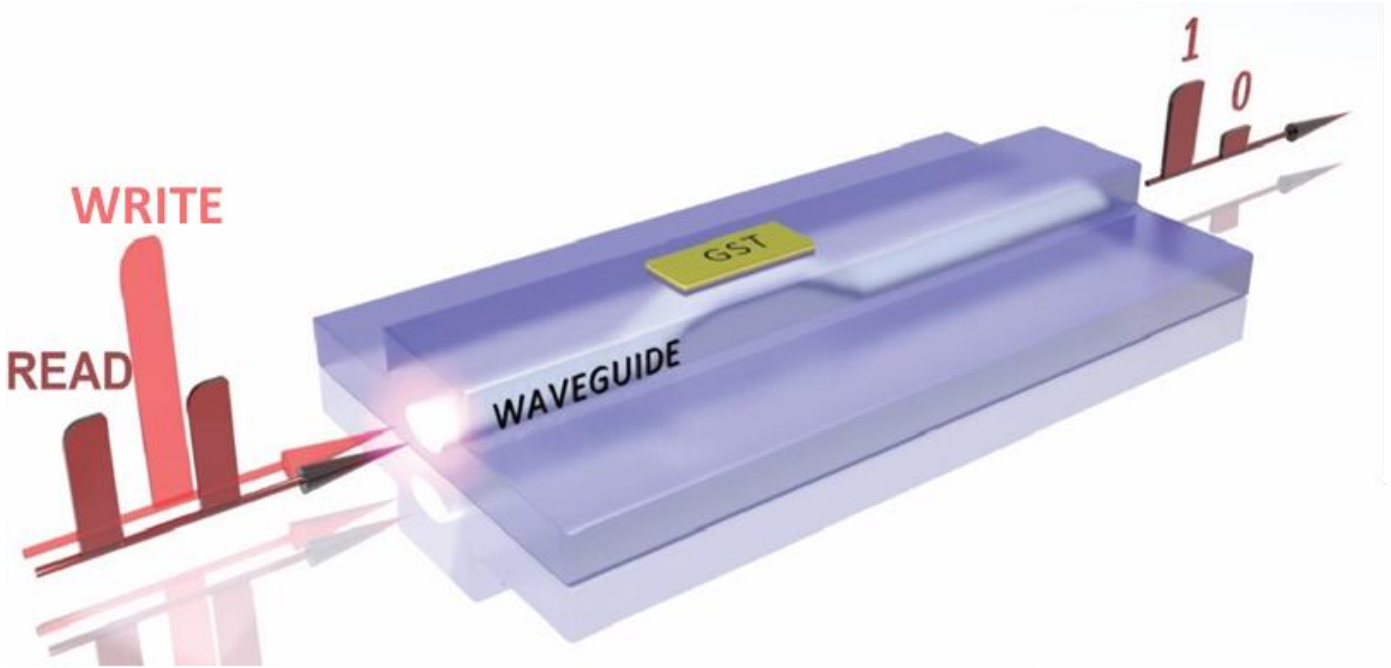
# Multi-device synaptic architectures

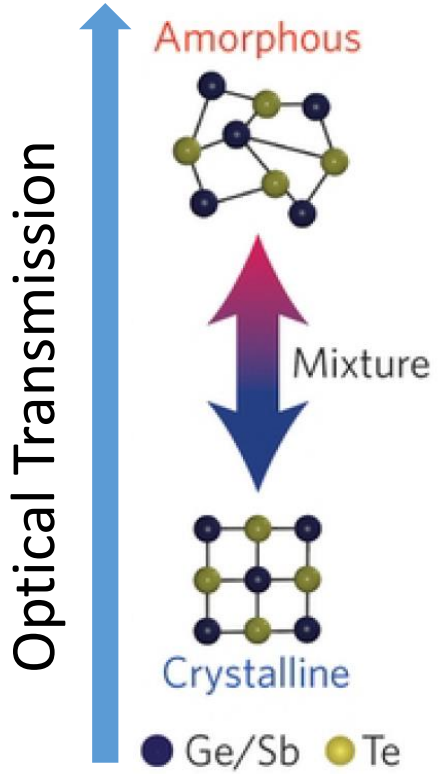



- Using multi-device architectures, an **increased dynamic range/conductance change granularity**, **extended linear behavior** and improvements in **conductance change stochasticity** can be achieved
- It is possible to device very innovative **arbitration schemes for device selection**

***Boybat et al., Nature Communications (2018)***

# Photonic memory devices

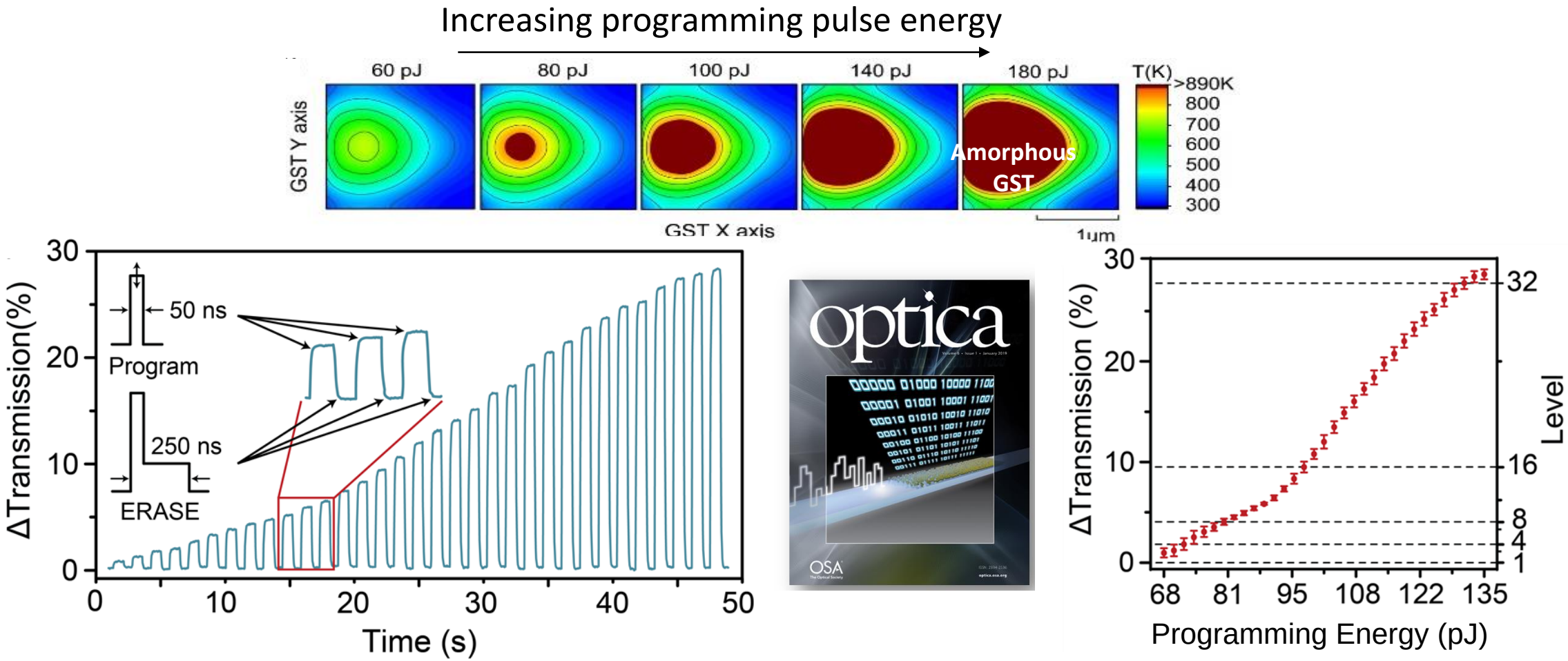



- Information stored in the phase configuration of a PCM segment on top of a nanophotonic waveguide
- **Write:** Evanescent coupling to the PCM
- **Read:** Monitor changes in the optical transmission

***Rios et al., Nature Photonics, 9, pp. 725 (2015)***

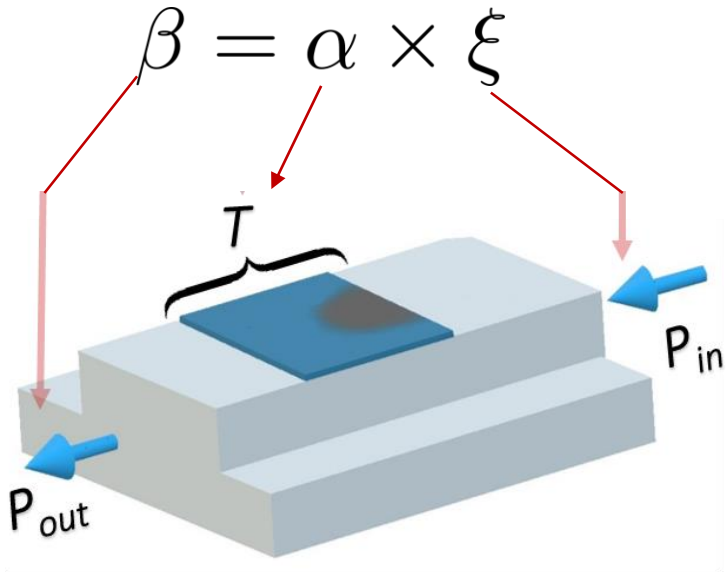
# Analog storage capability



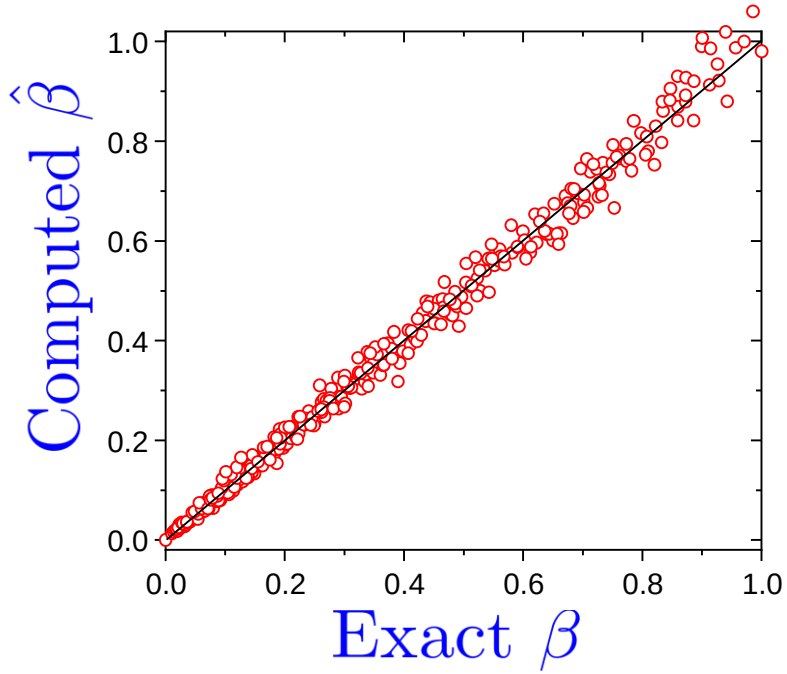

- Possible to achieve a continuum of transmission levels

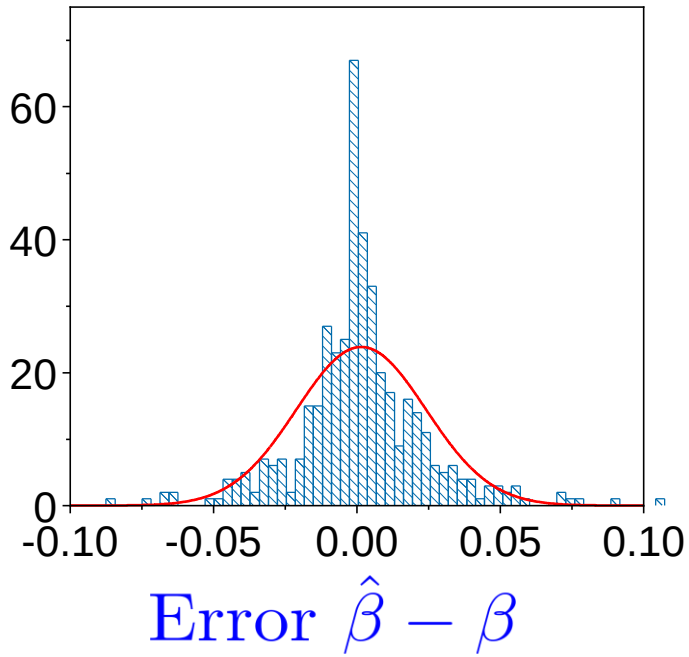
*Li et al., Optica 6(1), 1-6 (2019)*

# Photonic in-memory computing




**scalar multiplication**

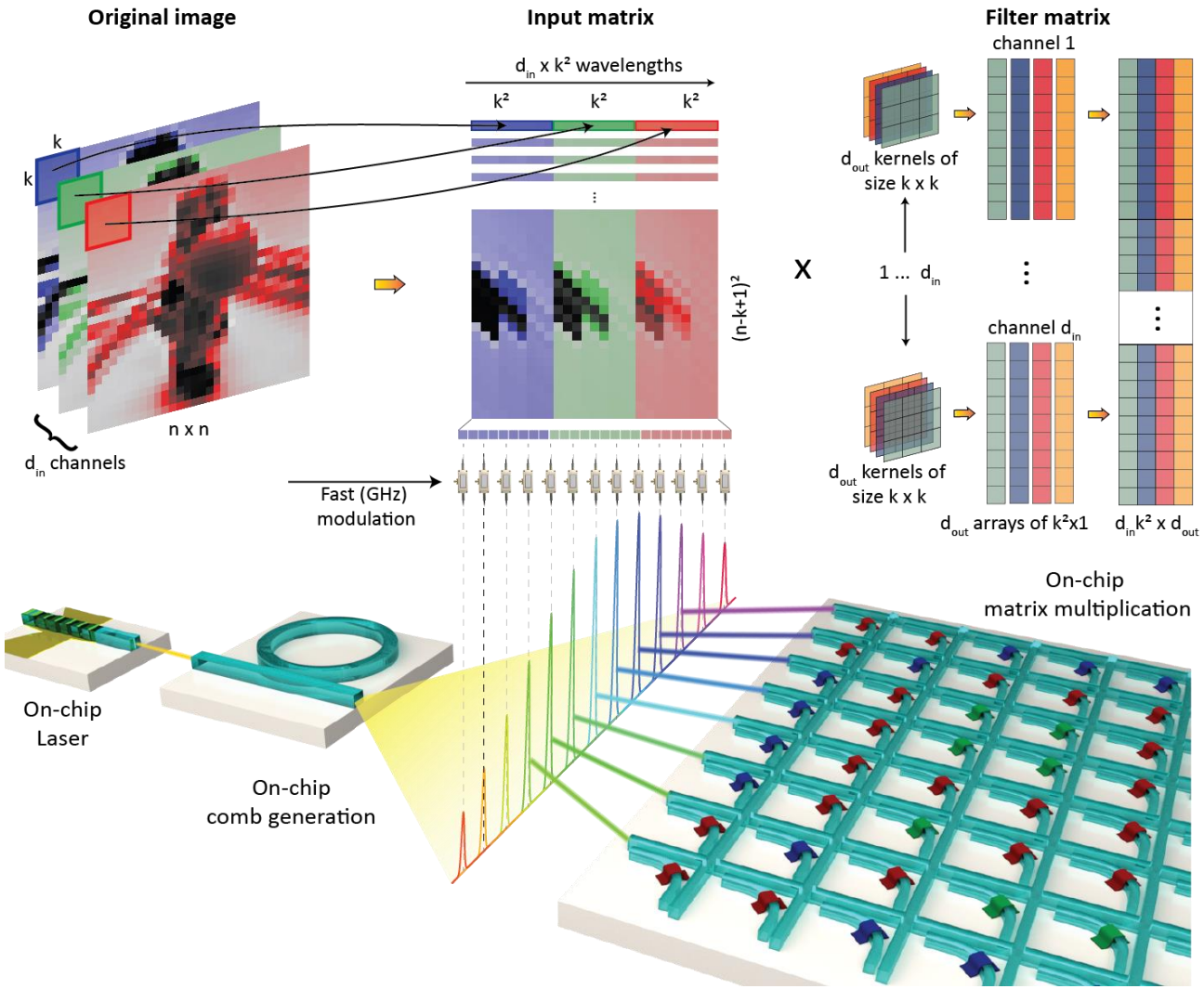


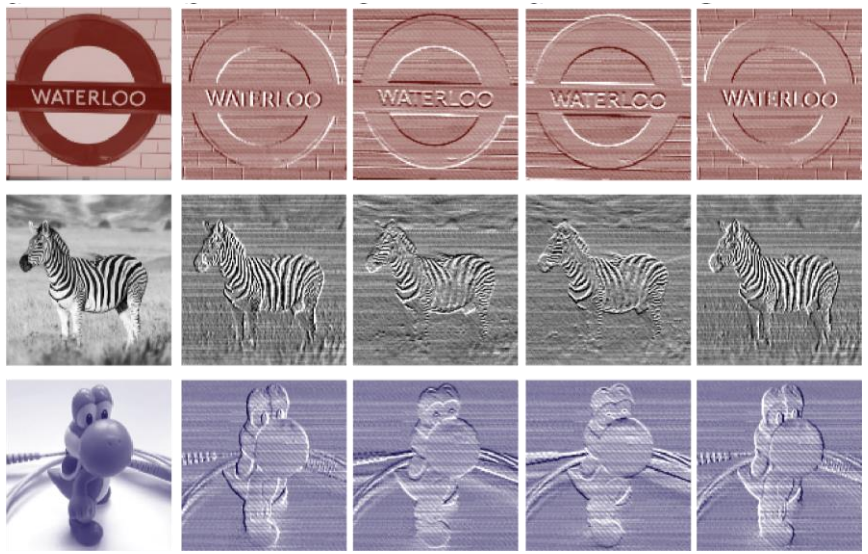




- Can perform in-memory scalar multiplication analogous to the electrical counterpart
- Larger areal footprint
- Exhibits higher linearity and improved accuracy
- Potential gain in speed
- **Inherent wavelength division multiplexing capability**

***Rios et al., Science Advances, 5(2), (2019)***

# Integrated photonic tensor core




- By exploiting the Inherent wavelength division multiplexing capability, it is possible to perform a complete convolution operation in a single time step!

***Feldmann et al., ArXiv (2020)***

# Summary

- In-memory computing is an emerging von Neumann computing paradigm where the **physical attributes of memory devices are exploited to compute in place**
- Can realize several **logical and arithmetic primitives** using both **charge-based** as well as **resistance-based** memory devices
- **Non-volatile binary storage**, **analog storage** and **accumulative behavior** typically exploited when computing with phase-change memory
- The applications span **high precision scientific computing** to **stochastic computing** that relies on imprecision
  - ✓ **Data-base query** and **hyperdimensional computing** facilitated by non-volatile binary storage
  - ✓ Approaches such as **bit slicing and mixed-precision computing** required to meet the precision requirements of scientific computing
  - ✓ **Signal processing and optimization** are particularly attractive application domains for in-memory computing
  - ✓ Using **custom noise-injective training approaches**, it is possible to achieve almost software-equivalent classification accuracies in **deep learning inference**
  - ✓ Using **mixed-precision training**, it is possible to achieve almost software-equivalent classification accuracies in **deep learning training**
- Emerging device-level concepts include **projected memory** and **photonic in-memory computing**

# Review article on in-memory computing


nature nanotechnology

FOCUS | REVIEW ARTICLE

https://doi.org/10.1038/s41565-020-0655-z

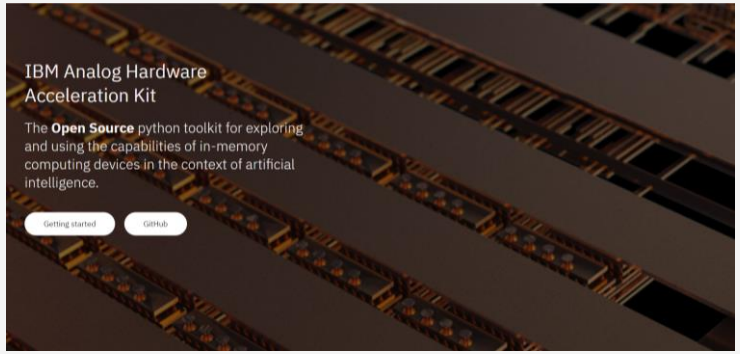
Check for updates

## Memory devices and applications for in-memory computing

Abu Sebastian ✉, Manuel Le Gallo, Riduan Khaddam-Aljameh and Evangelos Eleftheriou

# Analog AI Hardware Acceleration Toolkit

https://analog-ai.mybluemix.net/




## Current Capabilities Include:

- Simulate analog MVM operation including analog backward/update pass
- Simulate a wide range of analog AI devices and crossbar configurations by using abstract functional models of material characteristics with adjustable parameters
- Abstract device (update) models
- Analog friendly learning rule
- Hardware-aware training for inference capability
- Inference capability with drift and statistical (programming) noise models

## Roadmap:

- Integration of more simulator features in the PyTorch interface
- Tools to improve inference accuracy by converting pre-trained models with hardware-aware training
- Algorithmic tools to improve training accuracy
- Additional analog neural network layers
- Additional analog optimizers
- Custom network architectures and dataset/model zoos
- Integration with the cloud
- Hardware demonstrators

# Acknowledgements


- In-memory computing group, IBM Research – Europe
- Several other groups @ IBM Research – Europe
- IBM AI Hardware Center (https://www.research.ibm.com/artificial-intelligence/ai-hardware-center/)
- IBM TJ Watson Research Center
- IBM Research - Almaden
- New Jersey Institute of Technology, University of Patras, ETH Zürich, École polytechnique fédérale de Lausanne, RWTH Aachen, Oxford University, University of Münster