Test and Reliability Aspects of Computation in Memory

Mehdi Tahoori
Outline

- The need for Computing in Memory
- Realization of Computing in Memory
- Testing and Reliability Challenges and Solutions
- Concluding Remarks
Observation and Opportunity

- High latency and high energy caused by data movement
  - Long, energy-hungry interconnects
  - Energy-hungry electrical interfaces
  - Movement of large amounts of data

- Opportunity: Minimize data movement by performing computation directly (near) where the data resides
  - Processing in memory (PIM)
  - In-memory computation/processing
  - Near-data processing (NDP)
  - General concept applicable to any data storage & movement unit (caches, SSDs, main memory, network, controllers)
Memory and Power Walls

- A memory access consumes \(~100-1000\times\) the energy of a complex addition
- 62.7% of the total system energy is spent on data movement
Perils of Processor-Centric Design

- Grossly-imbalanced systems
  - Processing done only in **one place**
  - Everything else just stores and moves data: **data moves a lot**
    - Energy inefficient
    - Low performance
    - Complex

- Overly complex and bloated processor (and accelerators)
  - To tolerate data access from memory
  - Complex hierarchies and mechanisms
    - Energy inefficient
    - Low performance
    - Complex
We Need A Paradigm Shift To …

- Enable computation with *minimal data movement*

- Compute where it makes sense *(where data resides)*

- Make computing architectures more *data-centric*
In-memory computing

- 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
Basics of Computation-in-Memory: Classification

- **Computation-Out-Memory**
  - 1. Far (COM-F)
  - 2. Near (COM-N)

- **Computation-In-Memory**
  - Modification of normal memory
  - Any memory technology
  - 3. Periphery (CIM-P)
    - Basic v Hybrid
  - 4. Array (CIM-A)
    - Basic v Hybrid

- **Computation-in-Memory Configurations**
  - Memory configuration: read, write
  - Computation configuration: compute

[Source: H. A. D. Nguyen, JETC, 2020]
Outline

- The need for Computing in Memory
- Realization of Computing in Memory
- Testing and Reliability Challenges and Solutions
- Concluding Remarks
Resistance-based memory devices

- **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 range**
- ReRAM: $10^3$-$10^7$
- PCM: $10^4$-$10^7$
- STT-MRAM: $10^3$-$10^4$

**Access time**
- ReRAM: 10ns - 100ns
- PCM: ~ 100ns
- STT-MRAM: < 10ns

**Endurance**
- ReRAM: $10^6$-$10^9$
- PCM: $10^6$-$10^9$
- STT-MRAM: $> 10^{14}$
Logic design using resistance-based memory devices

- 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

Stateful logic

‘Memristive’ switches enable ‘stateful’ logic operations via material implication

- 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

**Borghetti et al., Nature (2010)**

**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

Read a memory cell (Memory)  Read & operate on two cells (Computation)

+ Less requirements on endurance
+ One access per operation
+ No major changes in array
- Modification of Sense Amplifiers and Address Decoders
- Data alignment

[SOURCE: L. Xie, et.al, A novel memristor-based logic design for resistive computing, ISVLSI 2017]
MVM using resistive memory

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

- 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

Outline

- The need for Computing in Memory
- Realization of Computing in Memory
- Testing and Reliability Challenges and Solutions
- Concluding Remarks
CiM Testing and Reliability Challenges (I)

- Need to cross the boundaries of
  - structural and functional testing
  - digital and analog testing
  - memory and logic testing

- Classification of faulty and fault-free at various abstraction levels
  - Functional level considering CiM operation
  - Structural level of resistive crossbar arrays
  - Individual resistive memory cells

- The non-determinism in correct operation require major changes
  - Fault modeling, test generation, and fault tolerance of CiM circuits
CiM Testing and Reliability Challenges (II)

- Traditional fault models of digital logic are inapplicable
  - building blocks are mostly analog
  - New fault models should be developed for CiM circuits

- Structural vs Functional Testing
  - Cannot ignore functionality during (structural) testing
  - Inherent approximation imperfection in output quality
  - Some structural faults may have benign effect on the output quality

- Need for yield improvement and defect tolerance
  - Use of emerging non-volatile resistive memory technologies
  - Immaturity of the fabrication processes
Fault Modeling and Test Generation for RRAM-based Scouting Logic
Test Approach for Scouting Logic-Based CIM

1. Defect Modeling
   - Model real physics of defects

2. Fault Modeling
   - Verify fault space to identify realistic faults

3. Test Development
   - Test for realistic faults

- Needs to be done for every configuration!
RRAM – Defects

- **Forming**
  - Final production step: create CF in the oxide
  - Large impact on RRAM operations
  - Lower forming current: Higher resistance & more variability
  - Higher forming current: Lower resistance & less variability

- **Forming Defect**
  - Due to variabilities in oxide, electrodes, forming current, etc.
  - Affect shape of filament
    - $l_{CF}, \varphi_T$
    - $\varphi_T < \varphi_B$

- **Transistor and interconnect defects**
  - Impurities, broken wires, dielectric variations, line roughness

**Physical parameters:** $t_{ox}, l_{CF}, l_{gap}, \varphi_T, \varphi_B$
Defect Modeling

- Scouting Logic production process
  - FEOL -> BEOL -> RRAM -> Forming
  - Same defects in both configurations

- Transistor and interconnect defects
  - Traditional linear resistors
  - Shorts (sh), bridges (br), opens (op)

- RRAM device forming defects
  - Forming current has strong impact on device performance
  - Non-linear device
  - Linear resistors not sufficient [Fieback ITC 2019]
  - Device-aware defect modeling
  - Relate I_{form} to device parameters l_{CF}, \phi_T

---

Physical parameters:
- \( t_{ox} \)
- \( l_{CF} \)
- \( l_{gap} \)
- \( \phi_T \)
- \( \phi_B \)
Fault Modeling

Fault space definition

• Analytical definition of all possible faults
Fault Modeling: Fault Space Validation

Fauls

Easy to detect (ETD)
- Always cause errors.
- Deterministic behaviour
- Detection guaranteed with e.g., March test
- E.g., <0r0:0r0 / 0:0 / 1> AND

March test, BIST,...

Hard to detect (HTD)
- May cause errors
- undefined state/read value
- Weak faults
- March test does NOT guarantee the detection
- E.g., <1r1 / U / ?>

Special DFT, Stress tests, ...

Defect $R_{br}$ and forming defect
Fault Modeling and Test Generation for STT-based CiM
Manufacturing defects (in MTJs)

- **Short** between the free layer and the reference layer
  - E.g. due to sputtering effect in the ion beam etching

- **Open** via or metal contacts
  - Internal damage of the MTJ

An open contact defect between the BEC and the underlying Cu layer

Cross-section TEM of a MTJ with a pinhole defect
MTJ defect implications

- Defects change the behavior of an MTJ
  - High resistance defects
  - Low resistance defects

- Short defect
- Stuck-at-P defect
- Stuck-at-AP defect
- Open defect

- Low Resistance defect region
- High Resistance defect region
Retention Faults

- Possibility to flip state due to thermal noise
  - Uni-directional (HRS → LRS far more likely)
- Depending on the NVM device manufacturing parameters
- Expected retention times of several years
  - Significant reductions (< ms) due to process variation and runtime temperature
Why CiM Testing is Challenging?

- Memory Read vs. CiM-Operation
- Reference resistance needed
  - Between distinguishable resistance states
- Large sense margin for memory read operation
- Multiple MTJs with different state combinations narrow down sense margin
- Additional faults possible
  - Which are not covered by conventional read tests
Defect Modeling Methodology

- 12 intra cell defects
  - Opens on all connections inclusive internal node
  - Shorts between all nodes
  - Shorts towards Vdd/Gnd

- Coupling defects
  - With an unrelated cell
  - With the second operand
Results I

- Extensive evaluation of read and bitwise AND/OR
- Typical faults from memory read
  - Incorrect Read Fault (IRF0/IRF1)
- Newly observed faults for CiM operations
  - Incorrect AND Fault (IANDF0/IANDF1)
  - Incorrect OR Fault (IORF0/IORF1)

- Cases with **only** IRF
- Cases with **only** IANDF/IORF  Not detectable by regular memory tests!
- Cases with **both**
Results II – Fault Groups

- Regular memory tests cover IRF
  - Also cover IRF + IANDF/IORF combined fault conditions

- IANDF-only / IORF-only faults need dedicated testing

- Optimization: possible to merge regular memory tests with CiM tests
Test Pattern Generation

- Multiple operations sensitize the same defect condition
  - E.g. a short between the internal node and Vdd can be found with AND(0,1) as well as AND(1,0)

- Find a minimal test coverage for all CiM specific faults
  - Possible solution: AND(0,1), OR(0,0) and OR(1,0)

- Calculate all operations on the memory as a test suite
Fault Tolerant Solutions for CiM
Retention Failures: Methodology

- Trained two-layer binary neural network on MNIST

- Mapped the result on two MTJ crossbars
  - W1 → A1
  - W2 → A2
  - (-1) → P-State and (+1) AP-state

- Asymmetric retention fault behavior
  - AP (HRS) → P (LRS) far more likely

- Evaluate and mitigate asymmetric retention faults
Baseline Training

- Inference accuracy drops over time
  - Due to retention failures
- Inference accuracy reduction rate
  - Depends on thermal stability factor
  - Thermal stability \( \geq 40 \) reasonable
- Conventional training with around half of the cells in HRS
Modified Training

- Reduce the percentage of cells in HRS
  - Reduce HRS cells to ~3%
  - HRS cells scattered over the array

- The inference accuracy decay not improved!
  - Each individual retention failure is more severe
  - Even worse results!
Hybrid Array and modified Training

- Use 10% high retention cells ($\Delta=60$) in array
  - Map the most important weights to them
  - Column-wise
- Inference accuracy decay greatly improved!
  - Only due to HRS weights not mapped to
    - High retention cells
Hybrid array + modified training

- Lower retention MTJ allow for smaller access transistors
  - 1T1MTJ cell $\Delta=60$ to $\Delta=40$
    - Reduce cell area by 37% and leakage by 43%
  - Use 10%/90% $\Delta=60/\Delta=40$ mix
    - Reduce array area by 33% and leakage by 39%
Reliable NVM-CiM with Voltage Tuning
Reliable CiM with Voltage Tuning

- Voltage-dependency of NVMs
  - Leveraged for improving Read Decision Failures (RDF)
- The variation optimized voltage tuning
  - Vdd is as small as possible and VwL is as large as possible
- ReRAM technology:
  - Relatively high distance between LRS and HRS $\rightarrow$ appropriate for CiM-P
  - Relatively large writing energy (1.1 nj) $\rightarrow$ Not appropriate for CiM-A
- STT-MRAM technology:
  - Relatively low distance between LRS and HRS $\rightarrow$ Not appropriate for CiM-P
  - Relatively small writing energy (0.43 pj) $\rightarrow$ appropriate for CiM-A
Voltage Bias Impact on Variability
Temperature Variability

- Temperature Increase

- Temperature Increase

- Temperature Increase

- Temperature Increase
Voltage Tuning and RDF: Logic OR (NOR)
Voltage Tuning and RDF: Logic AND (NAND)
Pareto Analysis

CiM-2, 4, 8 operations for 256kB STT-CiM, CiM-8 operations for 256kB ReRAM-CiM
Outline

- The need for Computing in Memory
- Realization of Computing in Memory
- Testing and Reliability Challenges and Solutions
- Concluding Remarks
Conclusions

- Computation-in-Memory based on non-volatile memory crossbars
  - Significantly reduces costs of data movement
  - However introduces new defects and fault behaviors

- Test generation for CiM needs to deal with new challenges
  - Memory test not sufficient
  - Computation configuration tests are required

- Emerging devices create test complication
  - Compute operations can be used to detect these faults

- Reliable CiM
  - Dealing with NVM non-idealities
Thanks for your attention!

For questions and comments please contact:

mehdi.tahoori@kit.edu