# Future computer Architectures: Computing in Memory

**Said Hamdioui**  
Delft University of Technology  
The Netherlands

**ASCI Spring School**  
on Heterogeneous Computing Systems  
May 29 - June 1, 2017

## Outline

- Motivation
  - The need of new technology and architectures
- Memristor (memristive devices)
  - Promising device, principal of working, potential
- Memrisor for memories
  - Straightforward application
- Memristor for logic
  - Different styles
- **Computation-in-memory architecture**
  - Combining all together
- Some results/ potential of CIM
  - Does it make sense?
- Conclusion
Motivation: Computing walls

1. Power Wall
   - Dominated by com & memory
   - 70 to 90% for data-ints. Appl

2. Memory Wall
   - Slow Limited bandwidth
   - Communication bottleneck
   - Stored program principle

3. ILP Wall
   - Insufficient parallelism at instr. level
   - Programmability Complexity & overhead

=> Reduced / Saturated performance
   - Enhancement based on expensive on chip memory (~70% of area)
   - Requires LD & ST: killers of overall perf

Need of new architectures

Motivation: Technology walls

1. Leakage Wall
   - High static leakage (volatile)

2. Reliability Wall
   - Unreliable components
   - Expensive solutions
   - Economically not affordable

3. Cost Wall
   - Complex manufacturing
   - Low yield, High cost
   - Limited scalability
   - Comes at additional cost

=> Less/no economical benefit

Need for new device technologies
Motivation: tech & comp architectures

Conventional Technology + Architectures

Today & near future
- Heterogeneous Architectures
  - Accelerators specialised logic, DSP, FPGA's

Near to far future
- Near memory computing
- Near Threshold computing

In Memory computing Architectures
- Quantum, Neuromorphic/ bio-inspired
  - Resilient

Unconventional Technologies + Architectures

[Source: Jan van Lunteren, IBM, Zurich]
Motivation: computing for the future

- **Can storage and computation integrated in the same physical location?**
  - Keep data unchanged as much as possible & execute operations
  - Significantly reduces communication/ power bottlenecks
- **Can non-volatile technology used?**
  - Practically zero leakage
- **Can (massive) parallelism supported?**
  - For problems with a lot of data level parallelism

![Computation-in-memory CIM die?](image)

- Can storage and computation integrated in the same physical location?
- Can non-volatile technology used?
- Can (massive) parallelism supported?

![Need device technology enabler?](image)

Memristor: basics

- **1971: Leon Chua**
  - Two terminal non-volatile device
  - Driven by electrical signal (V or I)
  - Resistivity depends on the past state
  - Switchable between 2+ resist. values

- **Physical mechanism**
  - Conductive filament grows by ion migration accelerated by temperature and field
  - Many materials under investigation: TiOx, HfOx, TaOx, ...

- **A lot of industrial interest**
  - First by HP in 2008
  - SK Hynix, HRL Labs, ...

![Memristor: basics](image)
Memristor: Advantages

- Dual functionality
  - Realize both memory and logic functions
  - Enable new computing paradigms
  - Reduce (eliminate) memory wall
- Low energy consumption
  - Low/zero leakage: Non-volatility
  - Reduce the overall power consumption
- Scalability/ Nanometric dimensions
  - Extreme density at low price and reduce area
  - Sustain the profitability of Moore's law
- CMOS compatibility
  - Enable the heterogeneous integration
  - Enhance manufacturing at low cost
- Two terminal passive device structure
  - Realize dense crossbar architectures
  - Stack on CMOS
- Good endurance & Good Reliability?

Memristor: potential applications

- Non-volatile memory
  - Inc multilevel
- Logic gates
  - Stand alone or hybrid
- Computing: Resistive, Neuromorphic and biological, ... 
  - etc.
Memristive based memories?

• Mainly three popular classes
  • STT-MRAM (Spin-Transfer-Torque Magnetic RAM)
    • Ferromagnetic layers
    • In plane MTJ v Perpendicular MTJ
  • PCM (Phase Change Memories)
    • Chalcogenide materials (crystalline v amorphous)
    • Mushroom Structure v Pillar structure
  • Resistive RAM
    • Oxide-RAM (oxRAM)
    • Conductive bridge RAM (CBRAM)

• Main characteristics:
  • High density
  • Non-volatility; Zero standby power
  • High scalability
  • Two terminal devices
  • Low writing voltage?

Source, S. Yu, et.al, Emerging Memory Technologies, IEEE

Memristive based memories?

• oxRAM seems to be most promising
  • Very high density (cross-point array structure)
  • Smaller and simpler in respect to MRAM
  • Lower consumption in respect to PCM.
  • Lower programming voltage and faster

<table>
<thead>
<tr>
<th>Features</th>
<th>DRAM</th>
<th>FLASH</th>
<th>MRAM</th>
<th>PCM</th>
<th>ReRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration</td>
<td>FE</td>
<td>FE</td>
<td>BE</td>
<td>BE</td>
<td>BE</td>
</tr>
<tr>
<td>Scalability</td>
<td>32 nm</td>
<td>15 nm</td>
<td>20-30 nm</td>
<td>10-20 nm</td>
<td>10 nm</td>
</tr>
<tr>
<td>Density</td>
<td>$4\cdot10^2$</td>
<td>$4f^2$</td>
<td>$35-40f^3$</td>
<td>$6-8f^3$</td>
<td>$4-6f^3$</td>
</tr>
<tr>
<td>Write voltage</td>
<td>$V_{NOM}$</td>
<td>$&gt;10V$</td>
<td>1V</td>
<td>3.5V</td>
<td>1.25V</td>
</tr>
<tr>
<td>Write time</td>
<td>50ns</td>
<td>0.1ns</td>
<td>20ns</td>
<td>10ns</td>
<td>10-50ns</td>
</tr>
<tr>
<td>Write energy</td>
<td>90fJ/bit</td>
<td>2.5pJ/bit</td>
<td>20pJ/bit</td>
<td>10-100fJ/bit</td>
<td>10-100fJ/bit</td>
</tr>
<tr>
<td>Endurance</td>
<td>$10^{15}$</td>
<td>$10^{6-5}$</td>
<td>$10^{12-15}$</td>
<td>$10^3$</td>
<td>$10^{6-9}$</td>
</tr>
</tbody>
</table>

Source, S. Yu, et.al, Emerging Memory Technologies, IEEE

[ref: Clermidy-2014]
Memristor for memories: RRAM are becoming reality

- Many prototypes
- 64 Mb Multi-Layered (2010) - Unity Semiconductor
  - Cross-Point Array Architecture, Meal Oxide
- 8 Mb Multi-Layered - Panasonic (2012)
  - Cross-Point Array Architecture, Meal Oxide
- 32 Gb 2-Layer - Sandisk/Toshiba (2013)
  - Cross-Point Array with Selective device Architecture
  - Metal Oxide Technology (24nm)
- 16 Gb – Micron/Sony (2014)
  - 1T1R Array Architecture
  - Metal Oxide Technology (27nm)
- 3D Xpoint (2015)-Intel/Micron
  - PCM technology
  - March 2017: SDD for data centers? 375GB?

Memristor for logic

- Boolean logic
  - CMOS like design [Voukas]
  - Rationed Logic [Kvatinsky]
  - MAGIC logic [Kvatinsky]
  - Other [Snijder, Xie]

- Implication logic
  - Logic operations on two propositions: p & q
  - Out= p → q (if p, then q)
  - Different implementations [Snider, Linn, Kvatinsky]

- Threshold/majority logic
  - Sum of weighted inputs compared with a certain threshold
  - Different implementations [Rose, Gao]
Memristor for logic

- **Boolean logic** ([G. Snider, APhy’05, Lei, et.al, ICCD’15])
  - Any logic function can be implemented
  - E.g., 2NAND
  - 2 Control voltages: \( V_w, V_h, \text{GND} \)
    - \( V_w > V_th > V_h \)
    - \( V_h = V_w/2 \)

- **Working Principle**
  1. Program all devices to \( R_{off} \)
  2. Program \( p \) to \( R_{on} \)
  3. Process NAND
    - Apply \( V_p = V_h, V_q = V_h, V_f = V_w \)
    - \( V_w - V_x = V_w - V_h = V_h \)
    => \( f = 1 \) (\( R_{off} \))

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Requires a sequence of multiple accesses!

Memristor for logic

- **Implication logic** ([Borghetti et.al, Nature, 2010])
  - Logic operations on two **propositions**: \( p \) & \( q \)
  - \( \text{Out} = p \rightarrow q \) (if \( p \), then \( q \))
  - 2 Control voltages: \( V_w, V_h, \text{GND} \)
    - \( V_w > V_th > V_h \)
    - \( V_h = V_w/2 \)

- **Working principal**
  1. Program \( p \) to \( R_{on} \)
  2. Program \( q \) to \( R_{off} \)
  3. Process imply
    - Apply \( V_p = V_h \) & \( V_q = V_w \)
    => \( V_x \approx V_h \)
    => \( q = R_{off} \)
    \( V_w - V_x = V_w - V_h < V_th \)

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( p \rightarrow q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Requires a sequence of multiple accesses!
Memristor for logic

- **Threshold logic**
  - \( f(x_1, x_2, \ldots, x_n) = \begin{cases} 1 & \text{if } \sum^n x_i T \\ 0, \text{ otherwise} \end{cases} \)
  - Two control voltages: Vdd & GND
  - Two logic states: 0 & 1

- **Example**
  
  Assume \( n=3, T=V_{th}=V_{dd}/2 \)
  1. Program all devices to Ron
  2. Provide the input voltages
    - Vdd, Vdd, 0
  3. \( V_f=1 \) (Roff)
    - \( V_x = (4/7) \) Vdd > Vth

Computation-in-memory: Is there any benefits?

- \( n_p \) processors
- Latency \( \propto (t_L + t_S + t_{ALU}) \times (n/n_p) \)

- **Better overall performance**
  - \( t_{L} + t_{ALU} \approx t_{S} + t_{ALU} \)
  - \( t_{S} + t_{ALU} \) is \( c \) constant
  - \( t_L \) depends on miss rate
  - E.g. Large data sizes \( \Rightarrow \) higher miss rate

- **Reduced energy**
  - Significant communication reduction
  - Reduce memory & power wall

- **Parallellism is program dependent**
  - CIM consumes much less than cores
  - Higher \( n_p \) \( \Rightarrow \) higher power \( \Rightarrow \) dark silicon

- **Potential applications**
  - Loops on the same data sets
  - Bit-wise operation
  - High data volume and reuse
  - E.g., bio-sequencing, graph processing.
Computation-in-memory: ideal example

\[
\begin{pmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{pmatrix}
\begin{pmatrix}
h_{11} & h_{12} \\
h_{21} & h_{22}
\end{pmatrix}
= \begin{pmatrix}
a_{11}h_{11} + a_{12}h_{21} & a_{11}h_{12} + a_{12}h_{22} \\
a_{21}h_{11} + a_{22}h_{21} & a_{21}h_{12} + a_{22}h_{22}
\end{pmatrix}
\]

Source: H. A. Du Nguyen, et. al. NANOACRH 2015

Computation-in-memory: requirements & challenges

- Some requirements
  - Crossbar based: dense
  - Heterogeneous integration
  - Good/Enough endurance
  - Specific applications
  - New program models
  - Etc

- Some challenges
  - Logic and arithmetic operations within the crossbar (memory)
  - Sneak path currents
  - High voltage drivers
  - Existing logic design requires multiple accesses to perform a single operation
    - Reduce endurance & increase latency
  - Need better schemes for logic and arithmetic operations

Scouting Logic?

- Perform operations while reading the operands
- No write of the devices during EX
- Use lower-voltage & simple control

Ref. L. Xie, et. al., ISVLSI 2017
**Computation-in-memory: Scouting Logic**

**Read a memory cell**

Input:
- \( I_{in} \)
- \( V_r / R_H \)
- \( V_r / R_L \)

Output:
- \( 0 \)
- \( 1 \)

**Read & operate on two cells**

- **OR operation**
  - Input: \( 00 \), \( 10/01 \), \( 11 \)
  - Output: \( 00 \), \( 10/01 \), \( 11 \)

- **AND operation**
  - Input: \( 00 \), \( 10/01 \), \( 11 \)
  - Output: \( 00 \), \( 10/01 \), \( 11 \)

- **XOR operation**
  - Input: \( 00 \), \( 10/01 \), \( 11 \)
  - Output: \( 00 \), \( 10/01 \), \( 11 \)

*Ref. L. Xie, et al., ISVLSI 2017*
Computation-in-memory: Scouting Logic

Voltage based design VSA

Current based design CSA

Switch Configurations

<table>
<thead>
<tr>
<th>Operation</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR/Read</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>AND</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>XOR</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
</tbody>
</table>

Delay (ns) | Power (uW) | Area (mm²)

- **VCA**
  - Delay: 9.31
  - Power: 8.65
  - Area: 6.00
  - Page fault latency: 165 cycles
  - Page fault rate: 0.0001
  - Instructions
  - m: % of memory accesses

- **VSA**
  - Delay: 2.73
  - Power: 15.99
  - Area: 17.19
  - Page fault rate: 0.0001
  - Instructions
  - m: % of memory accesses

- **CSA**
  - Delay: 2.73
  - Power: 15.99
  - Area: 17.19
  - Page fault rate: 0.0001
  - Instructions
  - m: % of memory accesses

June 2, 2017
Ref. L. Xie, et.al, ISVLSI 2017

Computation-in-memory: potential

- **Multicore** [Intel Xeon E5-2680]
  - nC = 4 cores, each 2.5 GHz
  - L1=32KB, 1CC access latency
  - L2=256KB, 2CC access latency
  - mrL1/mrL2: miss rate L1 / L2
  - 8GB DRAM, latency: 165 cycles
  - Page fault latency: 800 cycles
  - Page fault rate: 0.0001
  - Instructions
  - m: % of memory accesses
  - nL: % of memory access due to L
  - nR: % of memory access due to R
  - CIM area = ~ area of on-chip multicore caches

- **Application example**: Bit map index

<table>
<thead>
<tr>
<th>Dist.</th>
<th>Size</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>35</td>
<td>2016</td>
</tr>
<tr>
<td>B</td>
<td>23</td>
<td>2014</td>
</tr>
<tr>
<td>C</td>
<td>40</td>
<td>2015</td>
</tr>
<tr>
<td>D</td>
<td>60</td>
<td>2016</td>
</tr>
<tr>
<td>E</td>
<td>25</td>
<td>2010</td>
</tr>
<tr>
<td>F</td>
<td>34</td>
<td>2005</td>
</tr>
<tr>
<td>G</td>
<td>28</td>
<td>2012</td>
</tr>
<tr>
<td>H</td>
<td>30</td>
<td>2011</td>
</tr>
</tbody>
</table>

(a) Original Dataset (b) Bitmap Operations

June 2, 2017
Computation-in-memory: Potential

• Examples
  • Healthcare: DNA sequencing
    • we assume we have 200 GB of DNA data to be compared to
    • A healthy reference of 3GB for 50% coverage
    **E. A. Worthey, Current Protocols in Human Genetics, 2001
  • Mathematic: $10^6$ parallel additions

• Assumptions
  • Conventional architecture
    • FinFET 22nm multi-core implementation, with scalable number of clusters, each with 32 ALU (e.g., comparator)
    • 64 clusters; each cluster share a 8KB L1 cache
  • CIM architecture
    • Memristor 10nm crossbar implementation
    • The crossbar size equals to total cache size of CMOS computer

[Source: S. Hamdioui, et.al, DATE 2015]

Computation-in-memory: Potential

• Metrics
  • Energy-delay/operation
  • Computing efficiency: number of operations per required energy
  • Performance area: number of operations per required area

• Results

<table>
<thead>
<tr>
<th>Metric</th>
<th>Archit.</th>
<th>DNA sequencing</th>
<th>$10^6$ additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy –Delay/operations</td>
<td>Conv.</td>
<td>2.02e-03</td>
<td>1.5043e-18</td>
</tr>
<tr>
<td></td>
<td>CIM</td>
<td>2.34e-06</td>
<td>9.25702-21</td>
</tr>
<tr>
<td>Computing Efficiency</td>
<td>Conv.</td>
<td>4.11e01</td>
<td>6.5226e+9</td>
</tr>
<tr>
<td></td>
<td>CIM</td>
<td>3.70e04</td>
<td>3.9063e+12</td>
</tr>
<tr>
<td>Performance Area</td>
<td>Conv.</td>
<td>5.73e06</td>
<td>5.1118e+09</td>
</tr>
<tr>
<td></td>
<td>CIM</td>
<td>8.28e09</td>
<td>4.9164e+12</td>
</tr>
</tbody>
</table>

> x100
> x100
> x100

Key drives: Reduced memory bottleneck, non-volatile technology & parallelism
Conclusion

- Von-Neumann based computers
  - Memory & communication bottleneck
  - Complex programmability of multi-cores
  - Higher power consumption
  - \( \Rightarrow \) Unable to solve (today) and future application at affordable cost

- Short term
  - Specialization: application-specific accelerators (reduced prog)
  - Near memory computing, accelerator around memories (data-centric model)

- Long term
  - Alternative architecture, beyond Von Neumann & using new device tech
  - Resistive computing has a huge potential (CIM architecture)
  - But many open questions: device & materials, HW& SW, algorithms, etc

Future computer Architectures: Computing in Memory

Said Hamdioui
Delft University of Technology
The Netherlands

ASCI Spring School
on Heterogeneous Computing Systems
May 29 to June 1, 2017