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

TU Delft
Delft University of Technology

Outline

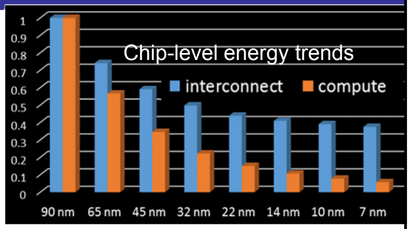
- Motivation
  - *The need of new technology and architectures*
- Memristor (memristive devices)
  - *Promising device, principal of working, potential*
- Memrisor for memories
  - *Straightforward application*
- Memristor for logic
  - Different styles
- **Computation-in-memory architecture**
  - *Combining 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

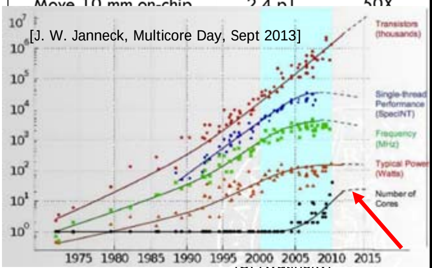
=> Reduced / Saturated performance
- Enhancement based on expensive on chip memory (~70% of area)
- Requires LD & ST: killers of overall perf

Need of new architectures



[S. Borkar, "Exascale Computing: a fact or a fiction?," IPDPS'13

Operation (8-bit operand)	Energy/Op (45 nm)	Cost (vs. ALU)
ALU operation	0.05 pJ	1X
Move 10 mm on-chip	2.4 nJ	50X

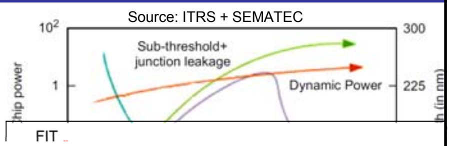


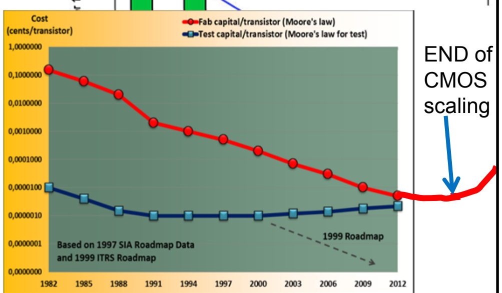
[Ref, D. Patterson, future of computer Architecture, 2006]

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

[Ref, S. Hamdioui, et. al, DATE 2017]

Motivation: tech & comp architectures

Conventional Technology + Architectures

Today & near future

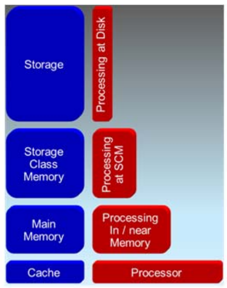
Heterogeneous Architectures

Accelerators specialised logic, DSP, FPGA's

Near to far future

Near memory computing

Near Threshold computing



[source: Jan van Lunteren, IBM, Zurich]

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

Far future

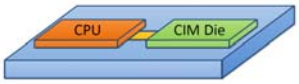
In Memory computing Architectures

Quantum, Neuromorphic/ bio-inspired Resistive

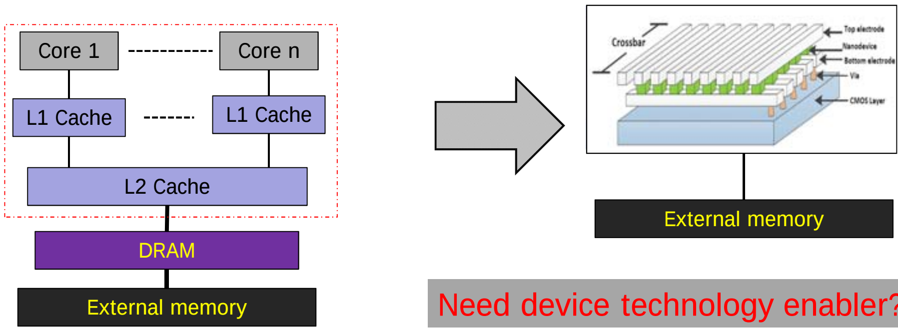
Unconventional Technologies + Architectures

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



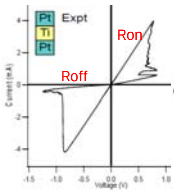
Computation-in-memory CIM die?

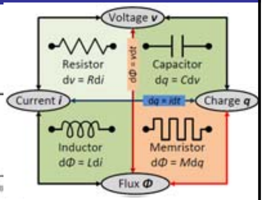


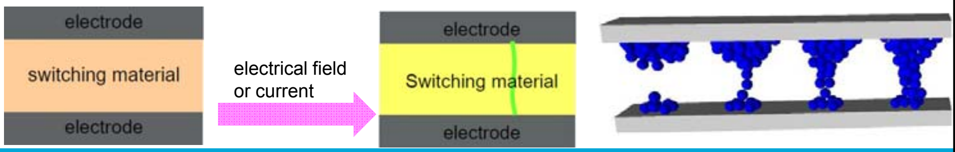
Need device technology enabler?

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, …

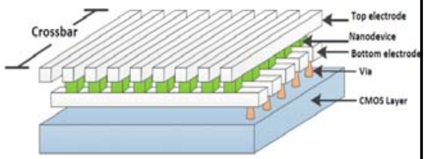


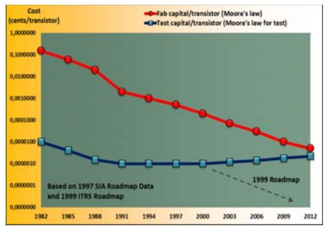


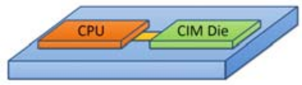


Memristor: Advantages

- 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?

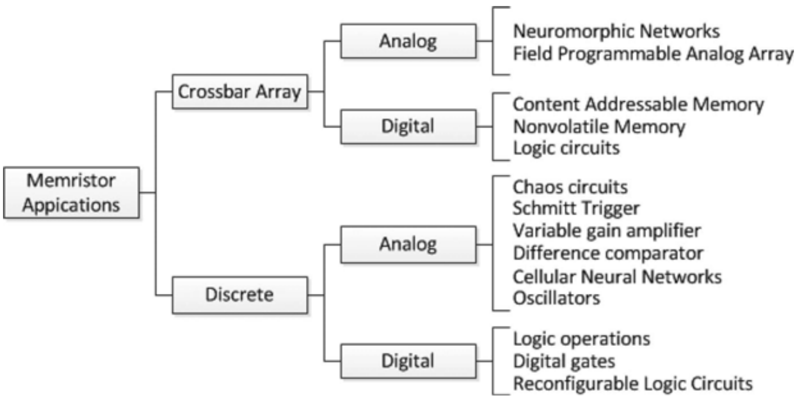






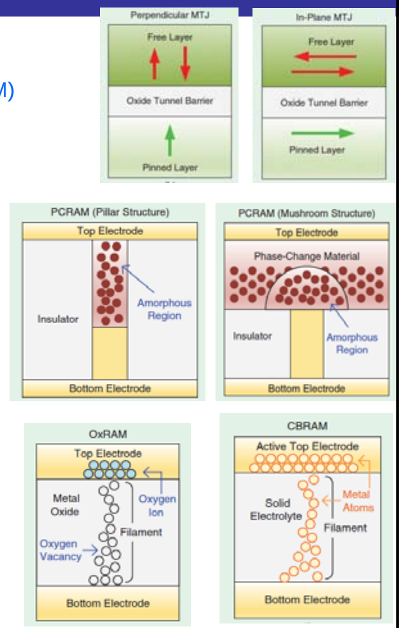
Memristor: potential applications

- 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 MJT 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 SOLID-STATE CIRCUITS MAGAZINE, spring 2016*

Memristive based memories?

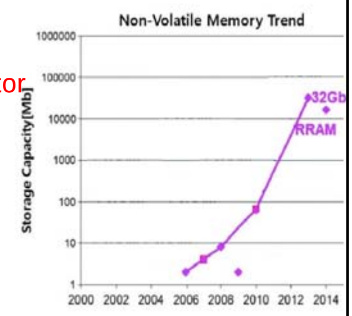
- oxRAM seems to be most promising
  - Very high density (cross-point array structure)
  - Smaller and simpler in respect to MRAM
  - Lower consumption in respect to PCM.
  - Lower programming voltage and faster

Features	DRAM	FLASH	MRAM	PCM	ReRAM	
		Nand	STT		OxRAM	CBRAM
Integration	FE	FE	BE	BE	BE	BE
Scalability	32 nm	15 nm	20-30 nm	10-20 nm	10 nm	10-20 nm
Density	4-6f^2	4f^2	35-40f^2	6-8f^2	4-6f^2	4-6f^2
Write voltage	V_{NOM}	>10V	1V	3-5V	1-2.5V	1-2.5V
Write time	50ns	0.1ms	20ns	10ns	10-50ns	100-1000ns
Write energy	90fJ/bit		2.5pJ/bit	20pJ/bit	10-100fJ/bit	10-100fJ/bit
Endurance	1E^{+15}	1E^{+4-5}	1E^{+12-15}	1E^{+9}	1E^{+6-10}	1E^{+5-6}

[ref: Clermidy-2014]

Memrisor 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?



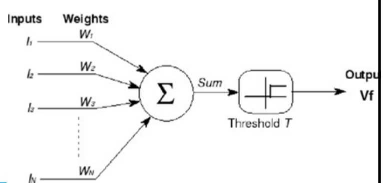
Memristor for logic

- 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]

p	q	$p \Rightarrow q$
true	true	true
true	false	false
false	false	true
false	true	true

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

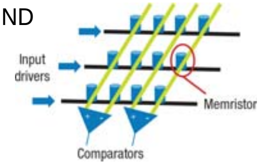


Memristor for logic

- Boolean logic [G. Snider, APhy'05, Lei, et.al, ICCD'15]
  - Any logic function can be implemented
  - E.g., 2NAND
  - 2 Control voltages: Vw, Vh, GND
    - Vw> Vth > Vh
    - Vh=Vw/2

Logic states:
Ron=1
Roff=0

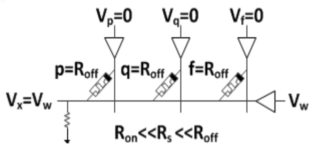
p	q	f
0	0	1
0	1	1
1	0	1
1	1	0



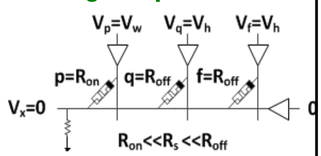
- Working Principle
  1. Program all devices to Roff
  2. Program p to Ron
  3. Process NAND
     - Apply Vp=Vh, Vq=Vh, Vf=Vw
     - Vw-Vx=Vw-Vh=Vh

  => f= 1 (Roff)

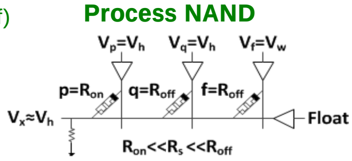
Program to Roff



Program p to Ron



Process NAND



Requires a sequence of multiple accesses!

Memristor for logic

- Implication logic [Borghetti et.al, Nature, 2010]
  - Logic operations on two *propositions*: p & q
  - Out= p → q (if p, then q)
  - 2 Control voltages: Vw, Vh, GND
    - Vw> Vth > Vh
    - Vh=Vw/2

p	q	P→q
0	0	1
0	1	1
1	0	0
1	1	1

Logic states:
Ron=1
Roff=0

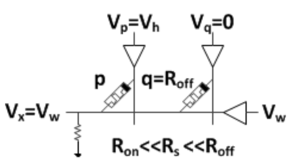
- Working principal
  1. Program p to Ron
  2. Program q to Roff
  3. Process imply
     - Apply Vp=Vh & Vq=Vw

       => Vx≈ Vh

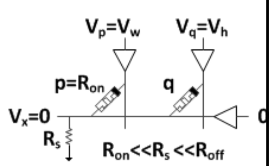
  => q= Roff

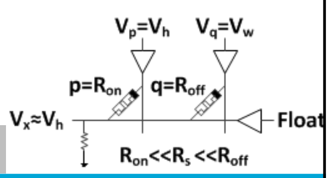
  Vw-Vx= Vw-Vh<Vth

Program q to Roff



Program p to Ron

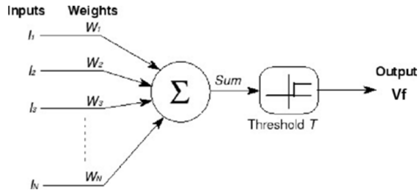




Requires a sequence of multiple accesses!

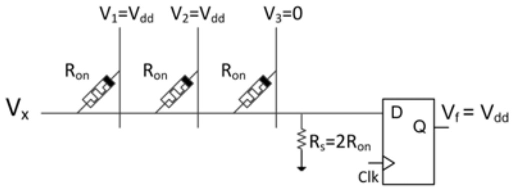
Memristor for logic

- Threshold logic
  - $f(x_1, x_2, .. x_n) = \begin{cases} 1 & if\ \sum_1^n x_i \geq T \\ 0, & otherwise \end{cases}$
  - Two control voltages: Vdd & GND
  - Two logic states: 0 & 1

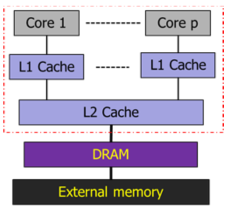


- Example

  Assume n=3, T=Vth=Vdd/2
  1. Program all devices to Ron
  2. Provide the input voltages
     - Vdd, Vdd, 0
  3. Vf=1 (Roff)
     - Vx =(4/7) Vdd >Vth

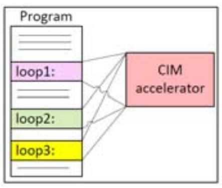


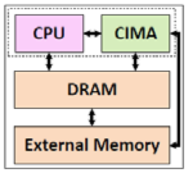
Computation-in-memory: Is there any benefits?



- n_p processors
- Latency ∝ $(t_L+t_S+t_{ALU})*(n_i/n_p)$

Assume a program with n_i instructions





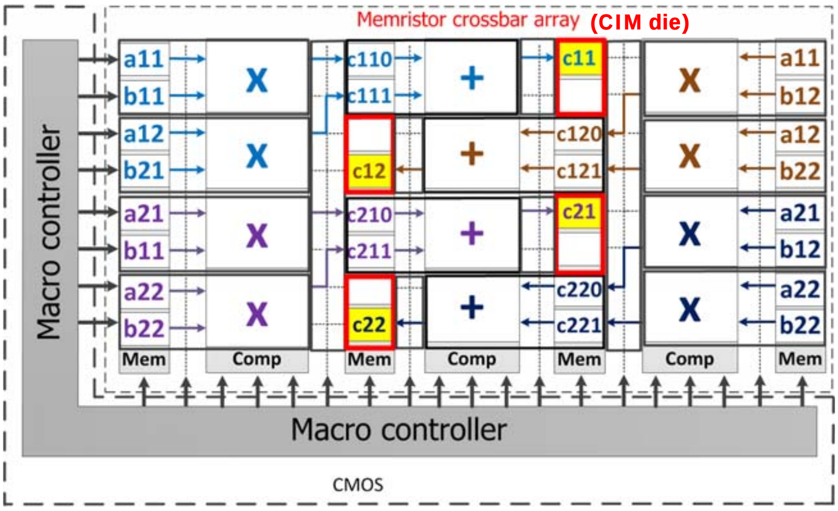
- n_a parallel crossbar arrays
- Latency ∝ $(t'_S+t'_{ALU})*(n_i/n_a)$
- Data already loaded in CIM

$$\text{Speed-up} \propto \frac{(t_L+t_S+t_{ALU})}{(t'_S+t'_{ALU})} * \frac{n_a}{n_p}$$

- **Better overall performance**
  - $t'_S+t'_{ALU} << t_L+t_S+t_{ALU}$
  - $t'_S+t'_{ALU}$ is ~ constant
  - t_L depends on miss rate
  - E.g. Large data sizes => higher miss rate
- **Reduced energy**
  - Significant communication reduction
  - Reduce memory & power wall
- **Parallelism is program dependent**
  - CIM consumes much less than cores
  - Higher n_p => higher power => 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

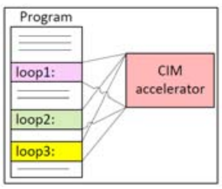
$$\begin{pmatrix} a11 & a12 \\ a21 & a22 \end{pmatrix} * \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix} = \begin{pmatrix} a11*b11+a12*b21 & a11*b12+a12*b22 \\ a21*b11+a22*b22 & a21*b12+a22*b22 \end{pmatrix}$$

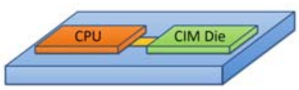


Source: H. A. Du Nguyen, et .al . NANOACRH 2015

Computation-in-memory: requirements & challenges

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

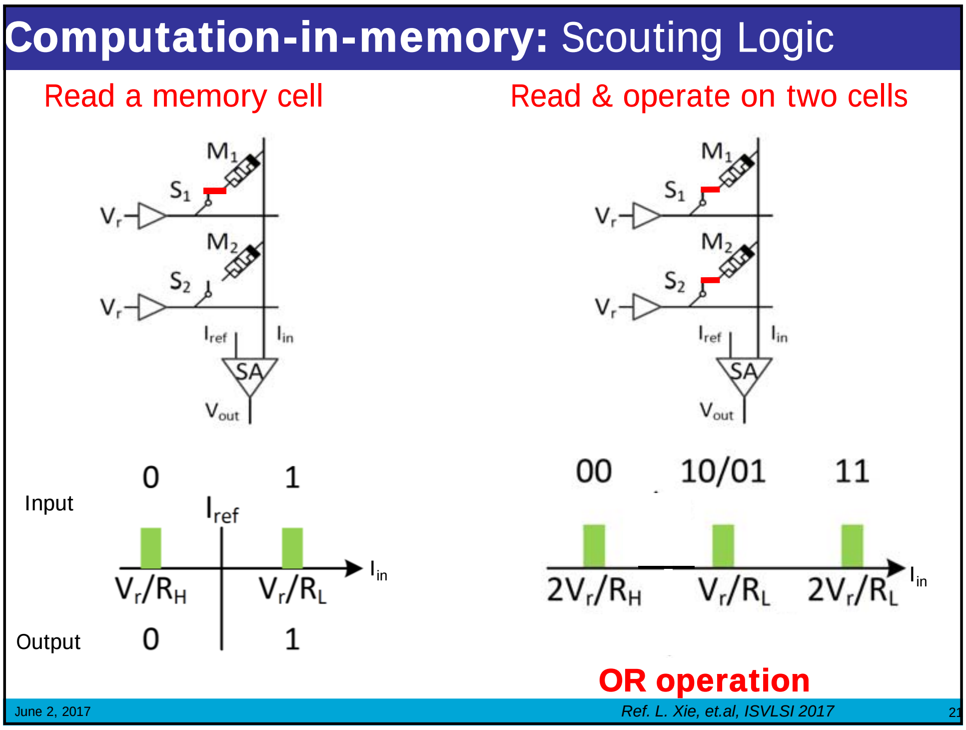


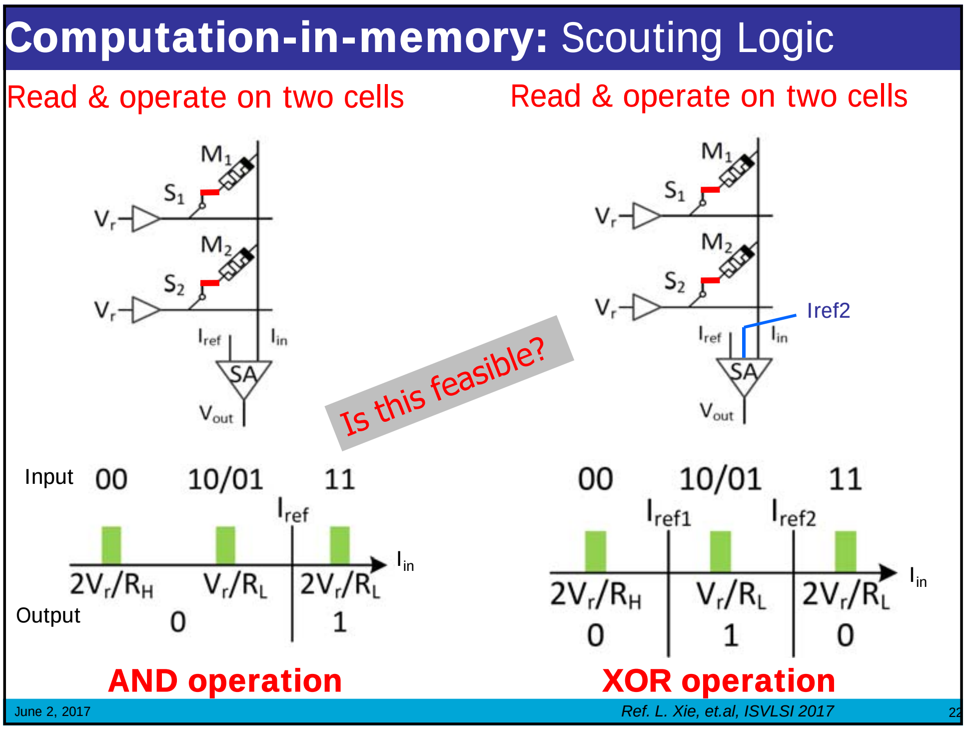


- Some challenges
  - Logic and arithmetic operations within the crossbar (memory)
  - Sneak path currents
  - *High voltage* drivers
  - Existing logic design requires *multiple accesses* to preform a single operation
    - *Reduce endurance & increase latency*

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

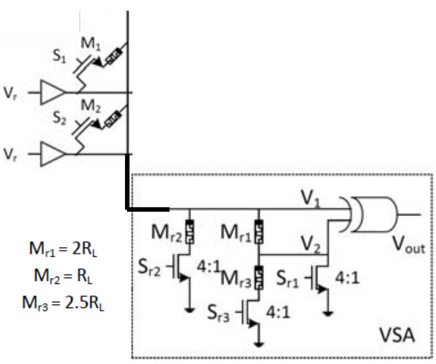

Computation-in-memory: Scouting Logic
Read a memory cell
Read & operate on two cells
M1
S1
Vr
M2
S2
Iref
Iin
SA
Vout
Input
0
1
Iref
Vr/RH
Vr/RL
Iin
Output
0
1
00
10/01
11
2Vr/RH
Vr/RL
2Vr/RL
OR operation
June 2, 2017
Ref. L. Xie, et.al, ISVLSI 2017
21

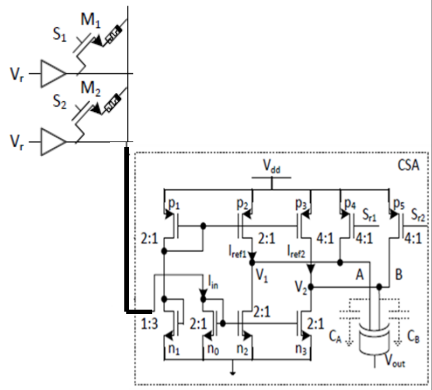

Computation-in-memory: Scouting Logic
Read & operate on two cells
Read & operate on two cells
M1
S1
Vr
M2
S2
Iref
Iin
SA
Vout
Iref2
Is this feasible?
Input
00
10/01
11
Iref
2Vr/RH
Vr/RL
2Vr/RL
Iin
Output
0
1
AND operation
00
10/01
11
Iref1
Iref2
2Vr/RH
Vr/RL
2Vr/RL
0
1
0
XOR operation
June 2, 2017
Ref. L. Xie, et.al, ISVLSI 2017
22

Computation-in-memory: Scouting Logic

Voltage based design VSA

Current based design CSA





Switch Configurations

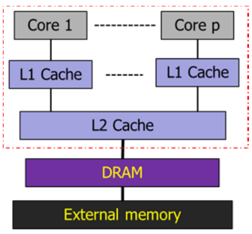
Operation	S_{r1}	S_{r2}	S_{r3}
OR/Read	ON	OFF	OFF
AND	ON	ON	OFF
XOR	OFF	OFF	ON

	Delay (ns)	Power (uW) OR	AND	XOR	Area (um^2)
CSA	2.73	15.99	17.19	17.59	29.7216
VCA	9.31	8.65	6.00	11.01	22.324

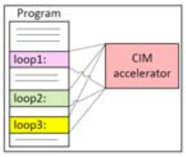
Switch Configurations

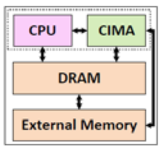
Operation	S_{r1}	S_{r2}
OR/Read	OFF	ON
AND	ON	OFF
XOR	OFF	OFF

Computation-in-memory: potential



- Multicore [Intel Xeon E5-2680]
  - n_p=4 cores, each 2.5 GHz
  - L1=32KB, 1CC access latency
  - L2=256KB, 2CC access latency
  - mr_{L1}/ mr_{L2}: miss rate L1 / L2
  - 8GB DRAM, latency: 165 cycles
  - Page fault latency: 800 cycles
  - Page fault rate: 0.0001* mr_{L2}
  - Instructions n_i
  - m : % of memory accesses





- CIM
  - Access time CIM die: 2 or 4 cycles
  - CPU – CIM communication: **82 cycles** (165/2)
  - Instructions n_i = n_l **(logic)** + n_r (rest)
  - m_l : % of memory access due to n_l
  - m_r : % of memory access due to n_r
  - CIM area= ~ area of on-chip multicore caches
- Application example : Bit map index

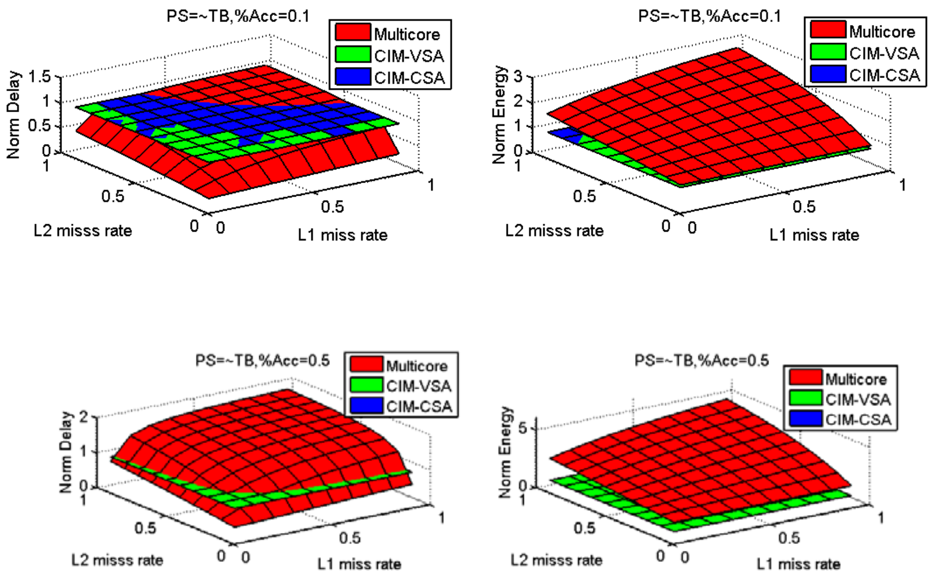
	Dist.	Size	Year
A	55	Large	2016
B	23	Medium	2014
C	43	Small	2015
D	60	Medium	2016
E	25	Medium	2000
F	34	Medium	2001
G	18	Small	2012
H	30	Small	2011

(a) Original Dataset

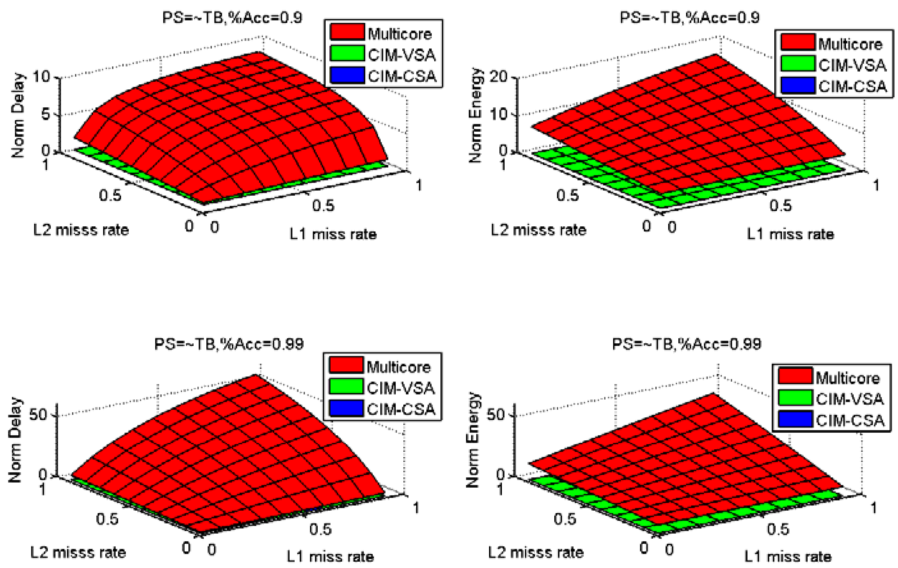
	A	B	C	D	E	F	G	H
Far	1	0	1	1	0	0	0	0
Near	0	1	0	0	1	1	1	1
Large	1	0	0	0	0	0	0	0
Medium	0	1	0	1	1	1	0	0
Small	0	0	1	0	0	0	1	1
New	1	0	0	1	0	0	0	0
Old	0	1	1	0	1	1	1	1
OR	1	0	1	1	0	0	0	0
AND	0	0	0	1	0	0	0	0

(b) Bitmap Operations

Computation-in-memory: potential



Computation-in-memory: potential



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

Metric	Archit.	DNA sequencing	10^6 additions	
Energy –Delay/ operations	Conv.	2.02e-03	1.5043e-18	> x100
	CIM	2.34e-06	9.25702-21	
Computing Efficiency	Conv.	4.11e01	6.5226e+9	> x100
	CIM	3.70e04	3.9063e+12	
Performance Area	Conv.	5.73e06	5.1118e+09	> x100
	CIM	8.28e09	4.9164e+12	

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

Conclusion

- Von-Neumann based computers
  - Memory & communication bottleneck
  - Complex progammability of multi-cores
  - Higher power consumption
  - => 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

Thanks

TUDelft
Delft University of Technology