2024 IEEE International Symposium on Integrated Circuits and Systems (ISICAS 2024)

18-19 October

New Delhi India

PRESENTATION TITLE

Accelerated Image Processing Through IMPLY-Based NoCarry Approximated Adders

Presenter Name

Nima TaheriNejad

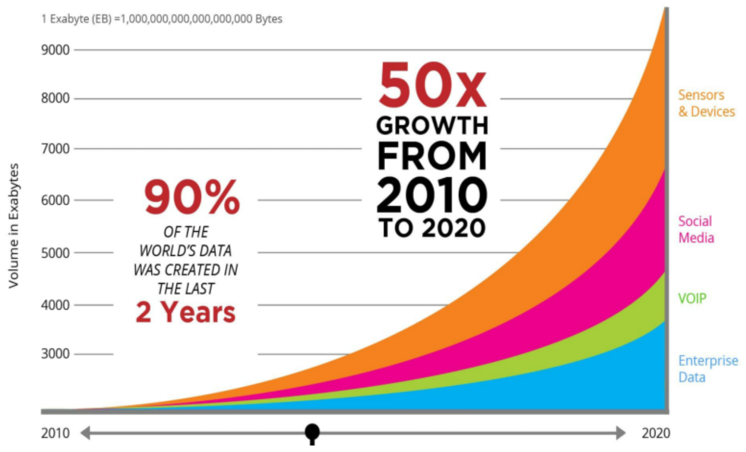
ISICAS 2024 DELHI, INDIA

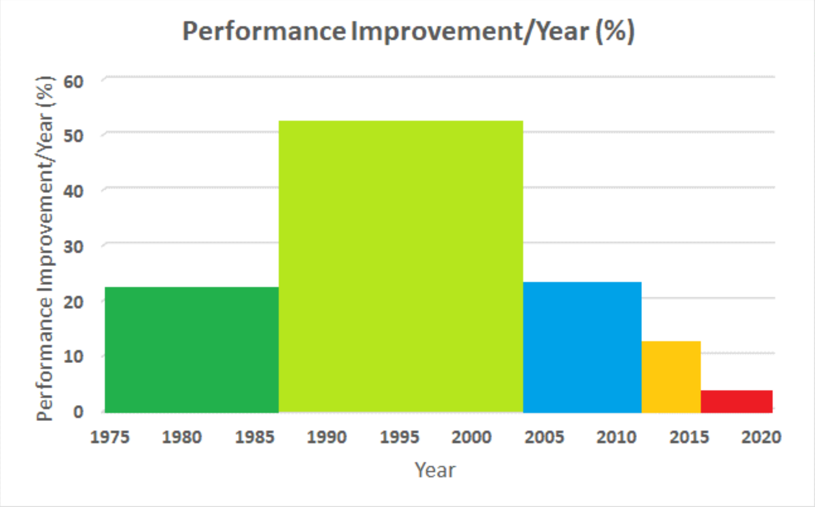
SYSTEMS & CIRCUITS FOR A SUSTAINABLE SOCIETY

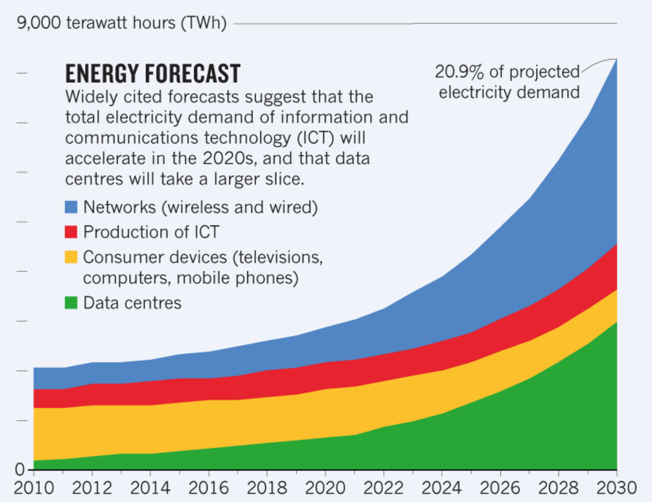
OUTLINE

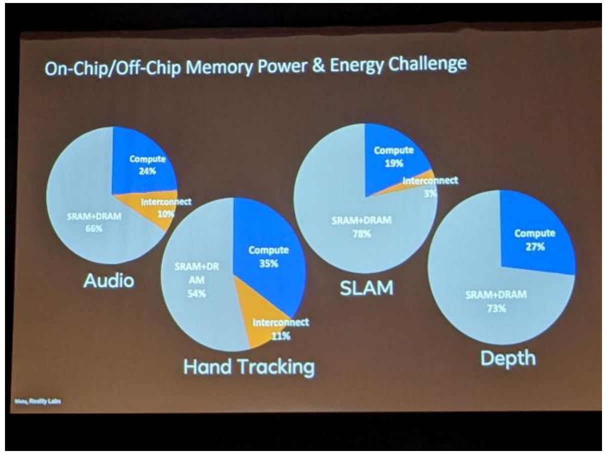
- Introduction & Motivation
- Problem Statement
- Methodology
- Results
- Image Processing Applications
- Conclusion
- References

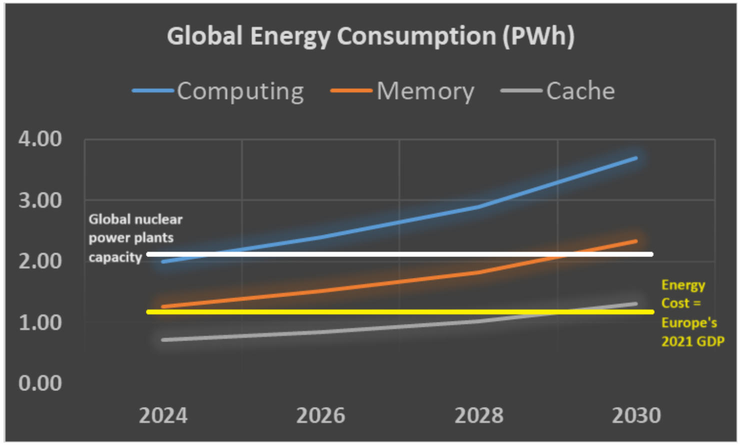
MOTIVATION


1 Exabyte (EB) =1,000,000,000,000,000,000 Bytes
Volume in Exabytes
9000
8000
7000
6000
5000
4000
3000
90%
OF THE WORLD'S DATA WAS CREATED IN THE LAST
2 Years
50x
GROWTH FROM 2010 TO 2020
Sensors & Devices
Social Media
VOIP
Enterprise Data
2010
2020


Performance Improvement/Year (%)
Performance Improvement/Year (%)
60
50
40
30
20
10
0
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
Year

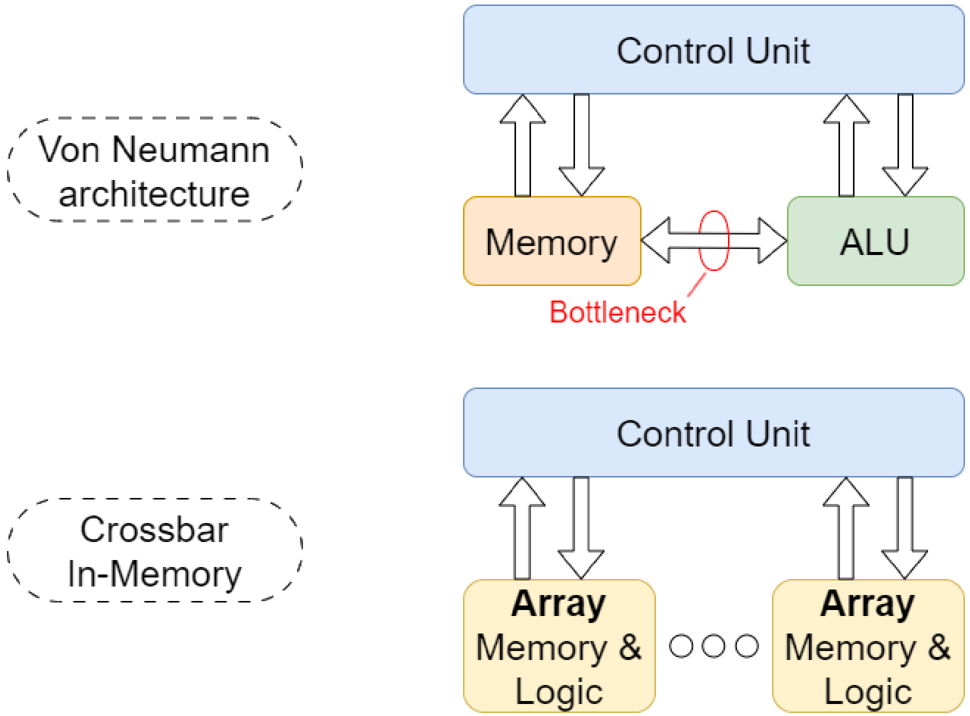

9,000 terawatt hours (TWh)
ENERGY FORECAST
Widely cited forecasts suggest that the total electricity demand of information and communications technology (ICT) will accelerate in the 2020s, and that data centres will take a larger slice.
20.9% of projected electricity demand
Networks (wireless and wired)
Production of ICT
Consumer devices (televisions, computers, mobile phones)
Data centres
0
2010
2012
2014
2016
2018
2020
2022
2024
2026
2028
2030

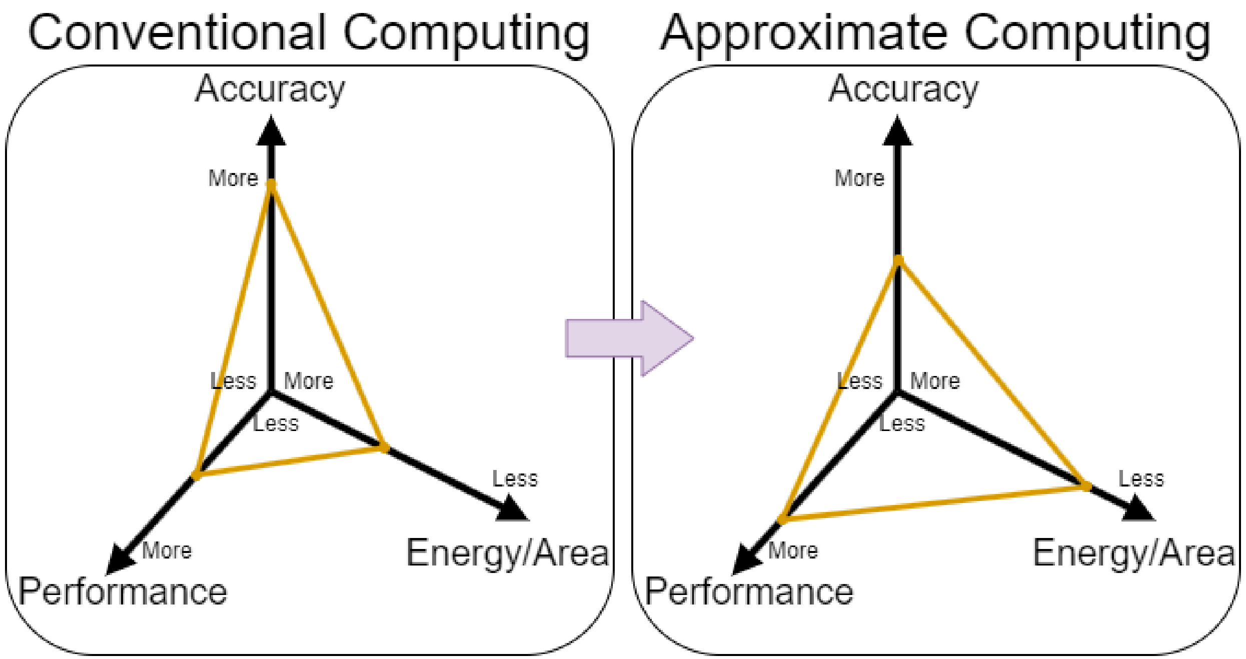

On-Chip/Off-Chip Memory Power & Energy Challenge
Compute 24%
Interconnect 10%
SRAM+DRAM 66%
Audio
Compute 35%
SRAM+DRAM 54%
Interconnect 11%
Hand Tracking
Compute 19%
Interconnect 3%
SRAM+DRAM 78%
SLAM
Compute 27%
SRAM+DRAM 73%
Depth


Global Energy Consumption (PWh)
Computing
Memory
Cache
4.00
3.00
2.00
1.00
0.00
Global nuclear power plants capacity
Energy Cost = Europe's 2021 GDP
2024
2026
2028
2030

INTRODUCTION

- Von Neumann Bottleneck => Memristor & In-Memory Computing [1]
- Power Wall Problem => Approximate Computing





PROBLEM STATEMENT

- Approximate + In-Memory Computing
- Algorithm Design with {→, ⊥}
- Efficient Topology Mapping
- Application-Level Gains

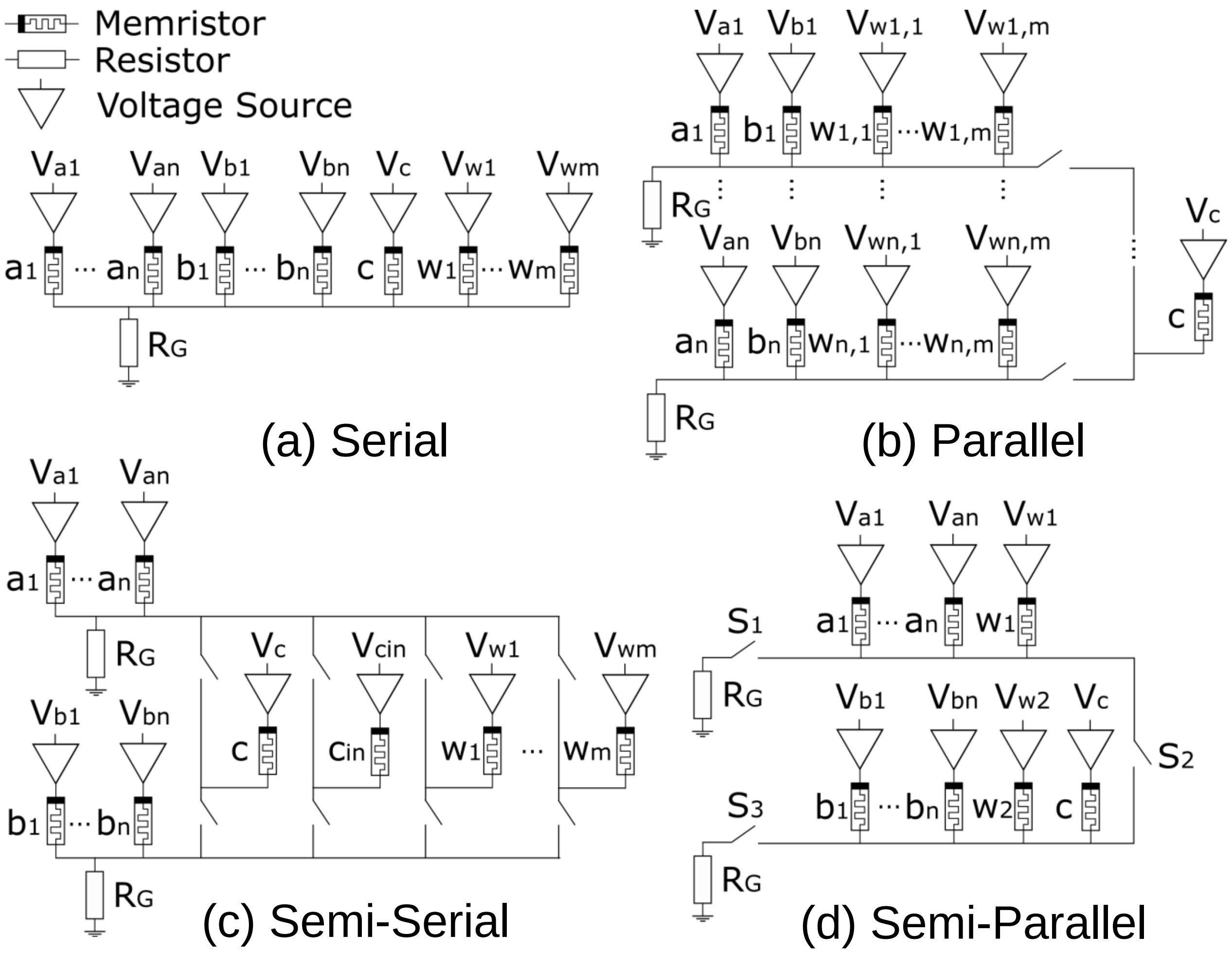
$$Sum = f_s(a, b, cin)$$
$$Cout = f_c(a, b, cin)$$

a	b	$b' = a \rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1

Boolean to IMPLY

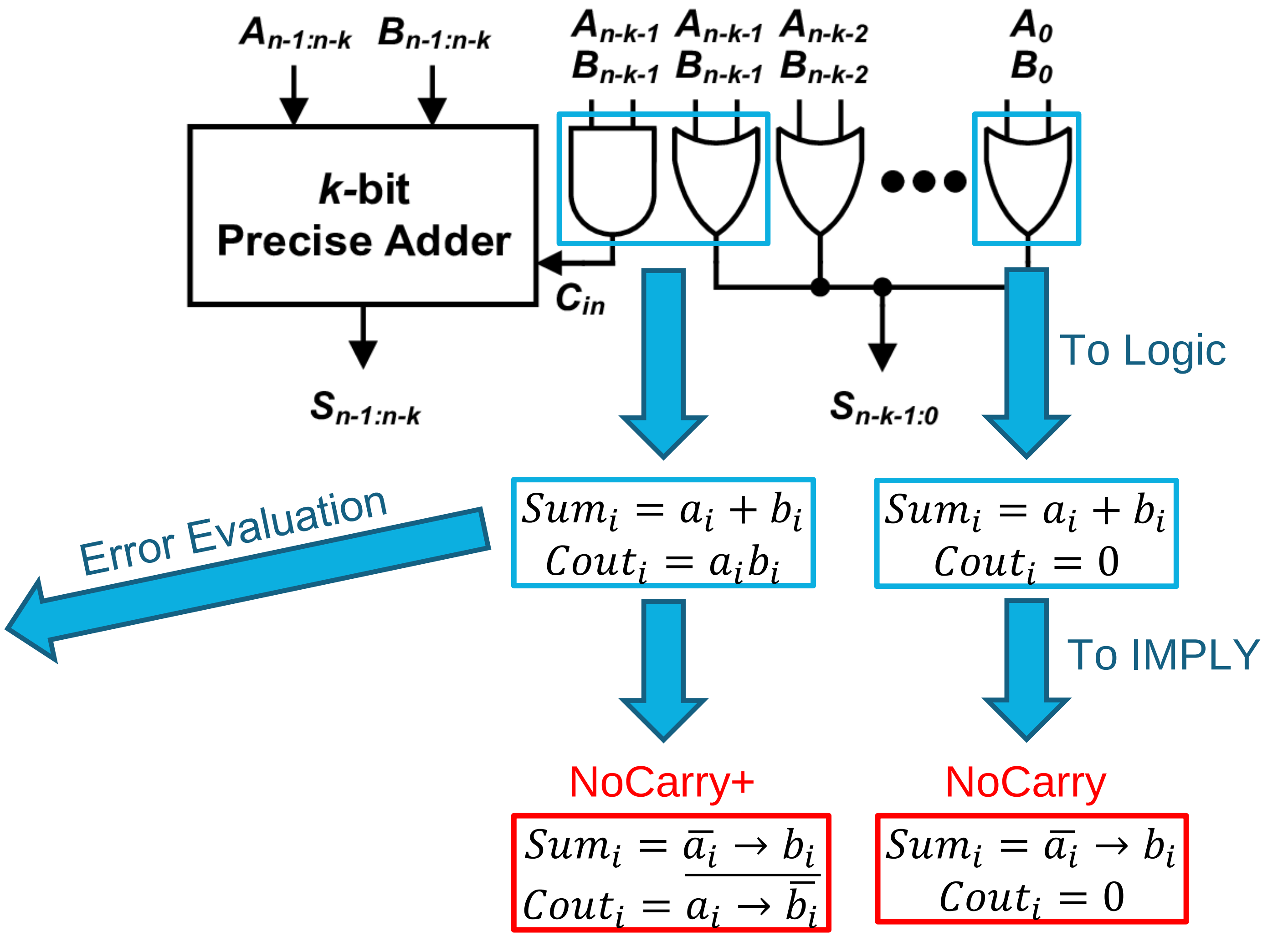
Constraints

$$Sum_{IMPLY} = f_{s,IMPLY}(a, b, cin)$$
$$Cout_{IMPLY} = f_{c,IMPLY}(a, b, cin)$$



METHODOLOGY

- Disregard Carry Propagation [2]
- Functional Representation
- Evaluation
- Equivalent IMPLY Function



Inputs			Exact		No Carry		No Carry +	
a	b	c	Sum	Cout	Sum	Cout	Sum	Cout
0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0
0	1	0	1	0	1	0	1	0
0	1	1	0	1	1	0	1	0
1	0	0	1	0	1	0	1	0
1	0	1	0	1	1	0	1	0
1	1	0	0	1	1	0	1	1
1	1	1	1	1	1	0	1	1

METHODOLOGY – MAPPING

- Implementation of NoCarry/NoCarry+
- Map IMPLY Algorithms
- Topology Advantages
  - => Parallelization
  - => Work Memristor usage

NoCarry+

$$Sum_i = \overline{a_i} \rightarrow b_i$$
$$Cout_i = \overline{a_i \rightarrow \overline{b_i}}$$

Serial

Semi-Serial

Semi-Parallel

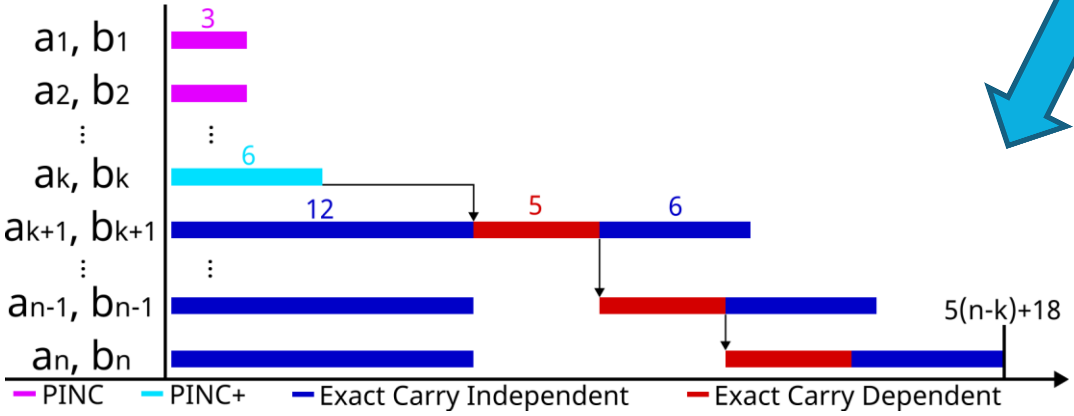
Parallel

Steps	Operation	Equivalent Logic
1	$w_1 = 0,\ w_2 = 0$	$False(w_1, w_2)$
2	$w_1' = a \rightarrow w_1$	$w_1 = \overline{a}$
-	$w_2' = b \rightarrow w_2$	$w_2 = \overline{b}$
3	$b' = w_1' \rightarrow b$	$b = \overline{a} \rightarrow b = a + b = Sum$
-	$w_2'' = a \rightarrow w_2'$	$w_2 = \overline{ab}$
-	$c' = w_2'' \rightarrow c$	$c = \overline{a \rightarrow \overline{b}} = ab = C_{out}$

Serial Implementation (SINC+)

Steps	Section 1	Section 2	Equivalent Logic
-		$w_1 = w_2 = 0$	$False(w_1, w_2)$
switch w_j with w_k $(w_1 \Leftrightarrow w_2)$, $w_j := w_1$, $w_k := w_2$			
1	$w_j' = a \rightarrow w_j'$	$w_k = 0$	$w_j = \overline{a}$ & $False(w_k)$
-		$w_k' = b \rightarrow w_k$	$w_k = \overline{b}$
2	$w_k'' = a \rightarrow w_k'$	$b' = w_j' \rightarrow b$	$b = \overline{a} \rightarrow b = Sum$ & $w_k = a \rightarrow \overline{b}$
-	$c' = w_k'' \rightarrow c$		$c = \overline{a \rightarrow \overline{b}} = ab = C_{out}$

Semi-Serial Implementation (S-SINC+)



Parallel Implementation (PINC+)

Steps	Section 1	Section 2	Between sections	(S1,S2,S3)	Equivalent Logic
1	$w_1 = 0$	$w_2 = 0$	-	(1,0,0) (1,0,1)	$False(w_1)$ & $False(w_2)$
2	$w_1' = a \rightarrow w_1$	$w_2' = b \rightarrow w_2$	-	(1,0,0) (1,0,1)	$w_1 = \overline{a}$ & $w_2 = \overline{b}$
3	-	-	$b' = w_1' \rightarrow b$	(0,1,1)	$b = \overline{a} \rightarrow b = Sum$
-	-	-	$w_2'' = a \rightarrow w_2'$	(1,1,0)	$w_2 = \overline{ab}$
-	-	$c' = w_2'' \rightarrow c$	-	(0,0,1)	$c = \overline{a \rightarrow \overline{b}} = ab = C_{out}$

Semi-Parallel Implementation (S-PINC+)

METHODOLOGY – SIMULATION

- 8 Approximated Algorithms (4 Topologies)
- VTEAM Model [3] in SPICE
- Fitted to measurements
- ATOMIC Pipeline

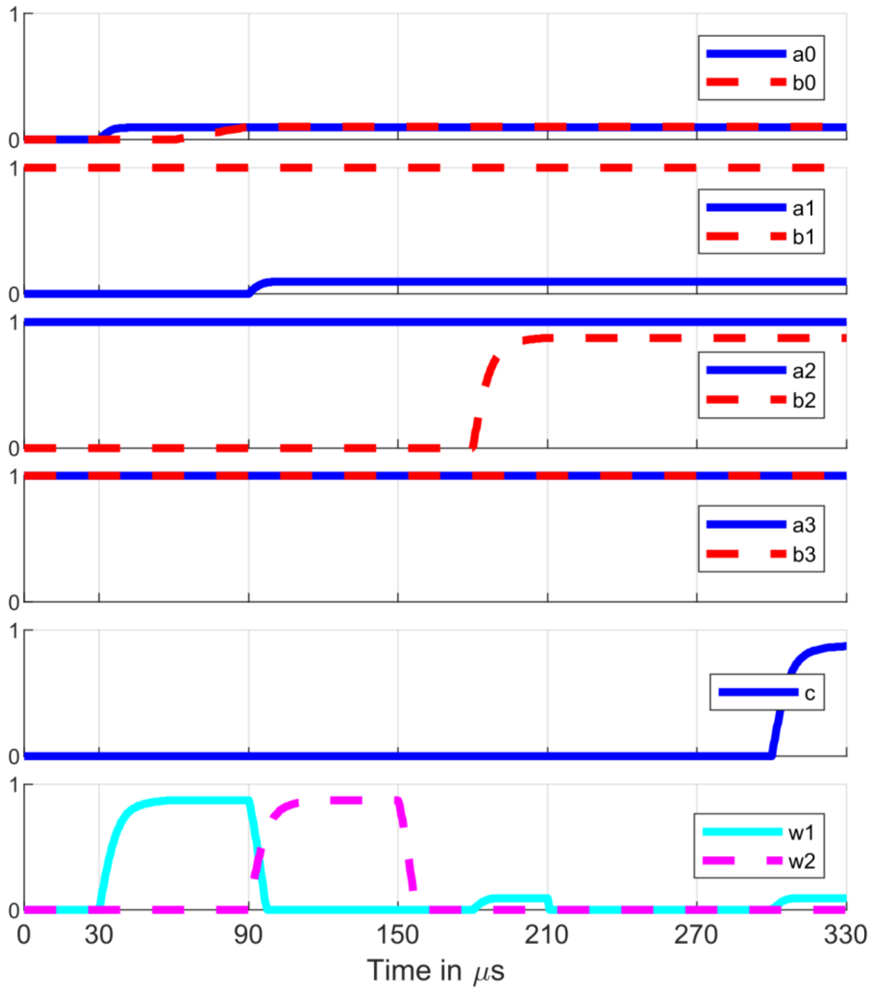
  => Stateful Logic Validation

  => Automated SPICE Simulations

  => Energy Calculations

  => Deviation Experiments
- Correct Behavior with $\pm 30\%$ Deviation

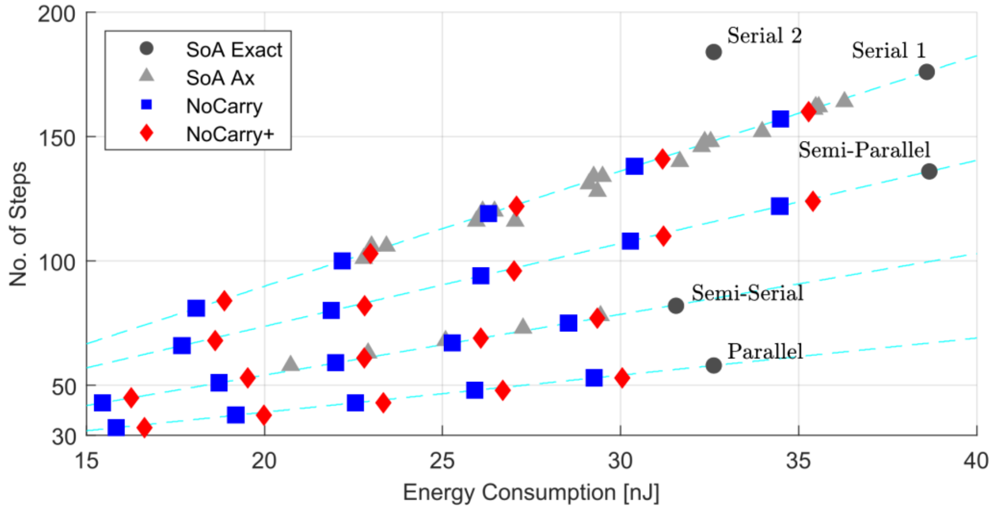
Get ATOMIC access here:
https://github.com/fabianseiler/ATOMIC



S-SINC+ Algorithm with A=0011, B=0101

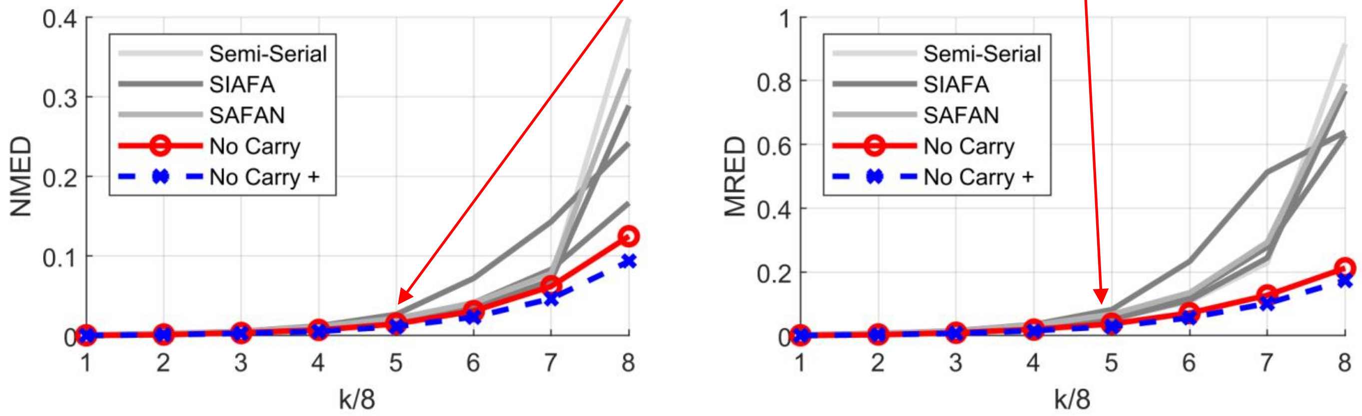
RESULTS

- Comparison to Exact Adder
  - => **6-54**% Faster
  - => **7-54**% less Energy
  - => Up to **12**% fewer Memristors
  - => Up to **63**% fewer Switches



- Comparison to Approximated Adder (5/8)
  - => **17-30**% Faster
  - => **17-33**% less Energy
  - => Improved Error Metrics

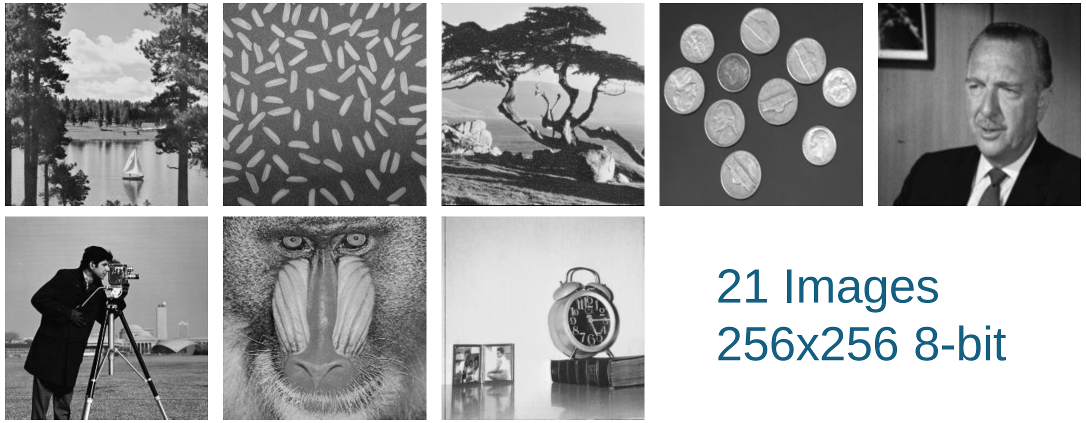
12-56% NMED & 27-64% MRED



Normalized Median Error Distance (NMED)
Mean Relative Error Distance (MRED)

IMAGE PROCESSING – OVERVIEW

- Error-resilient Application
- Quality Metrics
  - => PSNR
  - => SSIM
- Proposed Datasets
  - => Image Addition
  - => Grayscale Filter
- Highway Dataset [4]
  - => Background Subtraction



21 Images
256x256 8-bit

21 RGB-Images
8-bit/Channel

IMAGE PROCESSING – RESULTS

- Image Addition & Grayscale Filter (5/8 Ax Adder)
- Image Subtraction (5/8 Ax Adder) & Gaussian Smoothing (**86**% Ax)

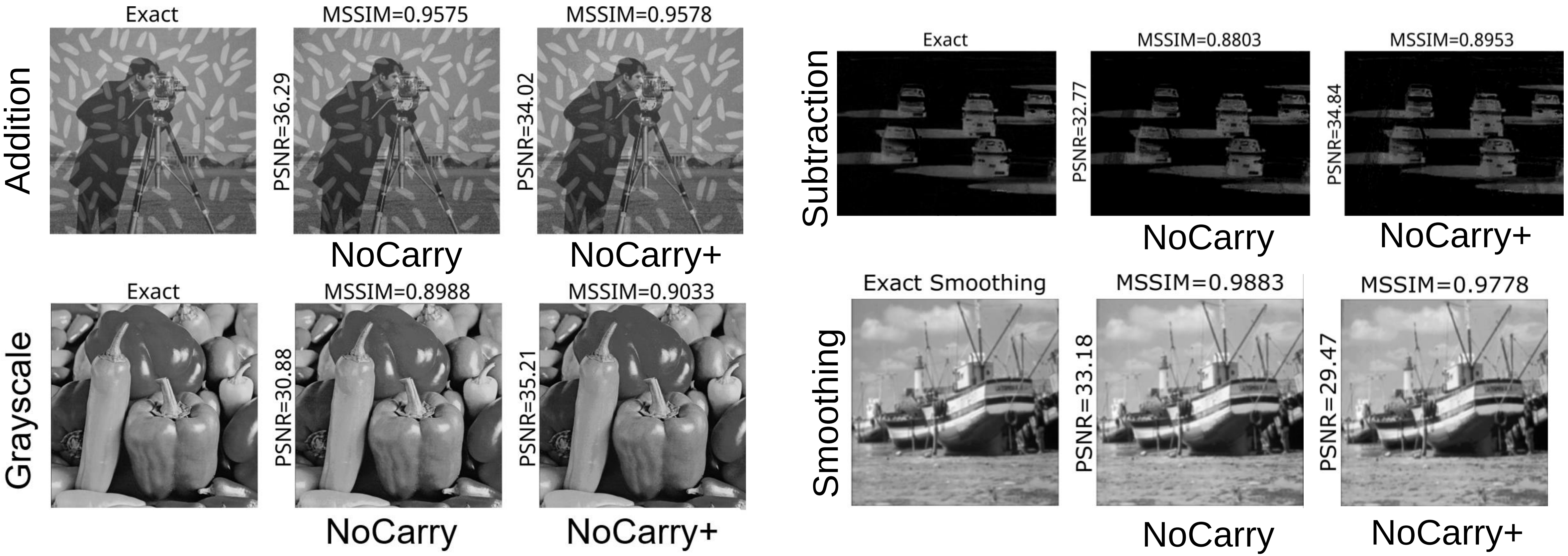
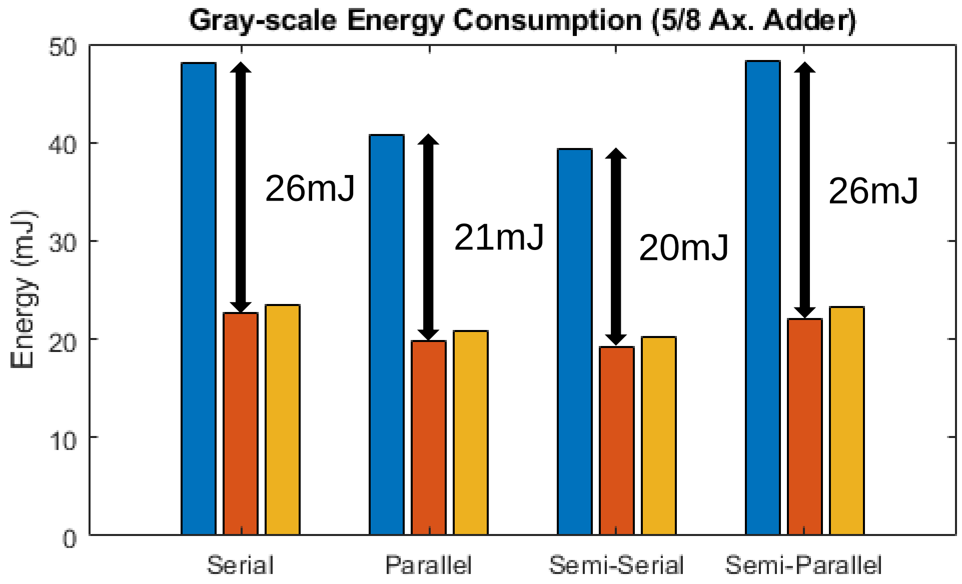
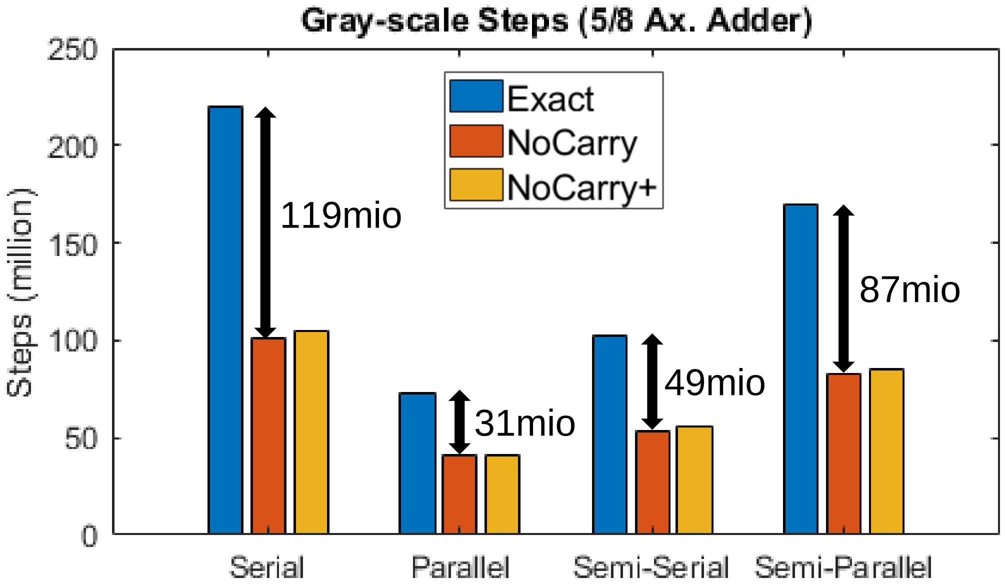


IMAGE PROCESSING – GAINS

- 684x912 RGB-Image
- **20-26** mJ Energy
- **31-119** million Steps





CONCLUSION

- **8** Approximated Implementations
- In **4** Topologies => Design Space Exploration
- Saved **6-54**% Energy & **7-54**% Steps
- Proposed 2 Image Processing Datasets
- Improved Image Quality & More Efficient than SoA

REFERENCES

- [1] N. TaheriNejad. In-memory computing: Global energy consumption,carbon footprint, technology, and products status quo. pp. 1–6, 2024
- [2] H. R. Mahdiani et al. Bio-inspired imprecise computational blocksfor efficient vlsi implementation of soft-computing applications. IEEETransactions on Circuits and Systems I: Regular Papers, 57(4):850–862,2010
- [3] S. Kvatinsky et al. Vteam: A general model for voltage-controlledmemristors. IEEE Transactions on Circuits and Systems II: ExpressBriefs, 62(8):786–790, 2015.
- [4] A. Prati et al. Detecting moving shadows: algorithms and evalua-tion. IEEE Transactions on Pattern Analysis and Machine Intelligence,25(7):918–923, 2003

- **Presented Datasets:** github.com/fabianseiler/Ax-Image-Processing
- **ATOMIC Tool:** github.com/fabianseiler/ATOMIC

Questions & Answers

THANK YOU