Approximate Computing: Ultra Low Power with "Good Enough" Results

Kaushik Roy

A. Raghunathan, George Karakonstantis, Vaibhav Gupta, Deb Mohapatra, George Panagopoulos, Ik-Joon Chang, Jung-Hwan Choi, Nilanjan Banerjee, Prof. Swaroop Ghosh, Prof. Swarup Bhunia, Swagath Venkatramani

Electrical and Computer Engineering,
Purdue University, West Lafayette, USA

January 19, 2014

Approximate Computing: An Analogy

Task: Division

$$is\ \frac{923}{21} > 1.75?$$

$$is\ \frac{923}{21} > 45.27?$$

$$\frac{923}{21} = ?$$

21) 923 (43
-84
83
63

But, I worked harder than needed

Accuracy

Same computation, but application context dictates required accuracy of results

Energy consumed varies based on accuracy required

Glucose consumption of brain increases with task difficulty (Larson *et al.* 1995)

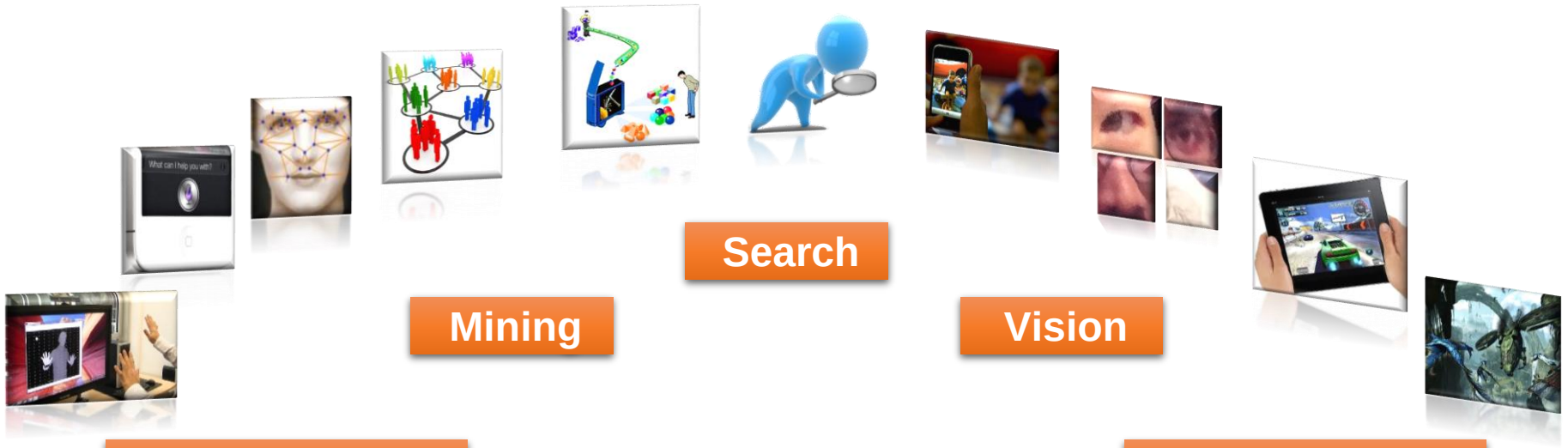
Outline

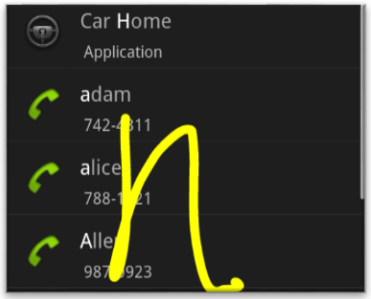
- Why?
  - Motivation for Approximate Computing
- What?
  - Approximate computing: Design philosophy and approach
- How?
  - Technologies for Approximate Computing

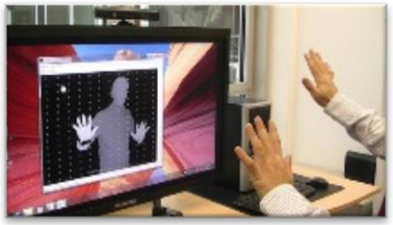
Motivation

- Explosive growth in digital information content and rapid increase in the number of users of applications related to image and video processing, recognition, and mining.
- How to process digital data in an energy-efficient manner while catering to desired user quality requirements?
  - Most of these applications possess an inherent quality of "error"-resilience
  - Considerable room for allowing approximations in intermediate computations, as long as the final output meets the user quality requirements

Evolving Application Landscape



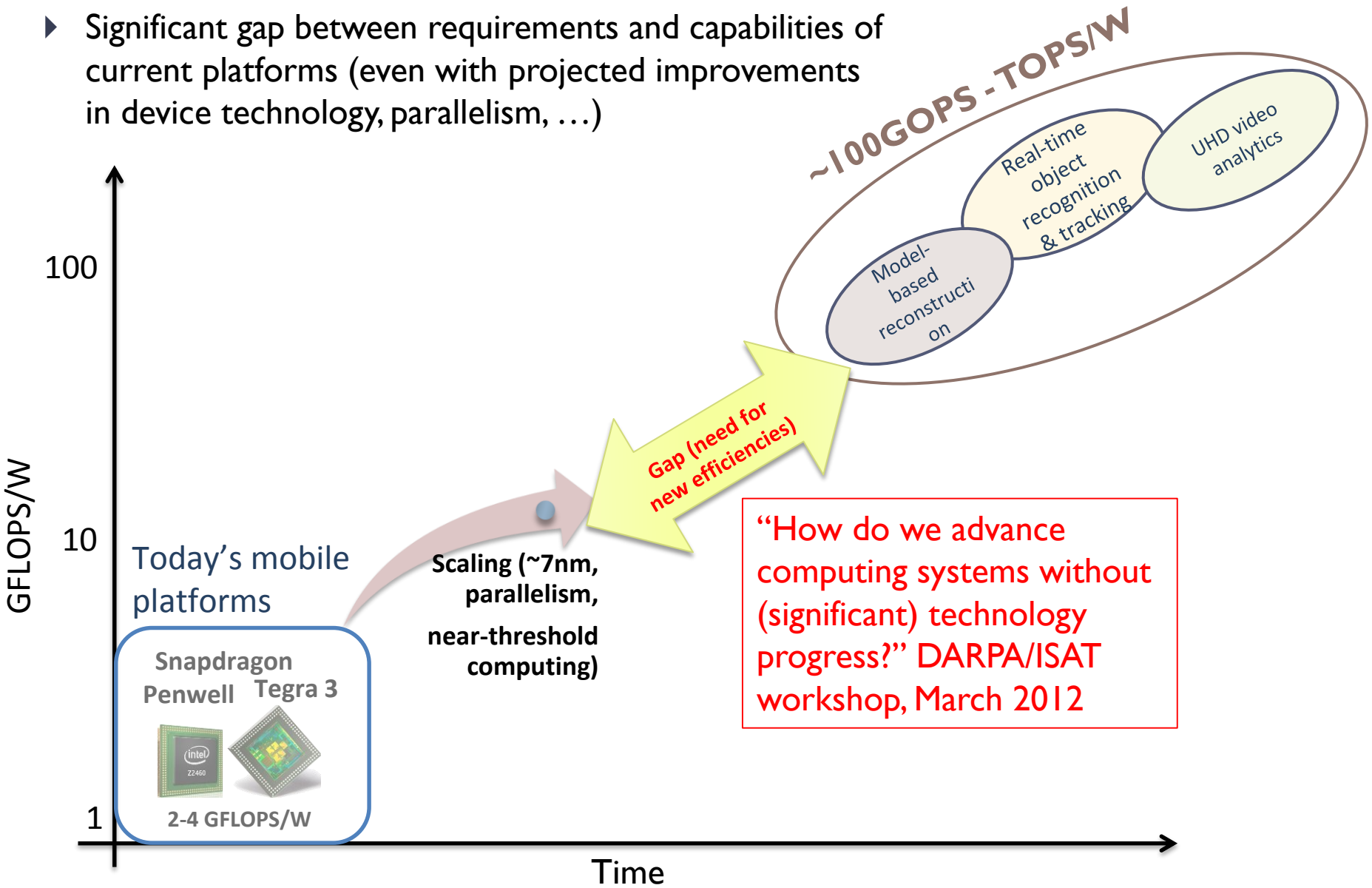




- Cloud: Increasing fraction of compute cycles in the cloud are spent on organizing & making sense of data
- Mobile: Add "intelligence"
  - Natural user interfaces
  - Context awareness

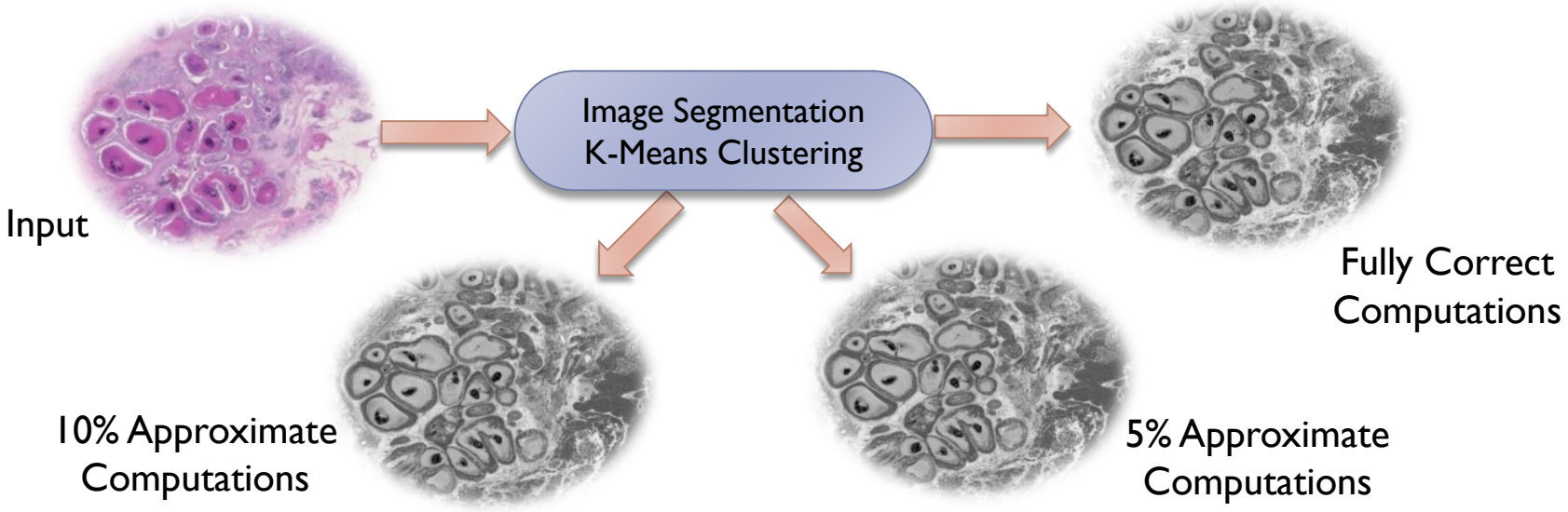
New Workloads Create New Needs

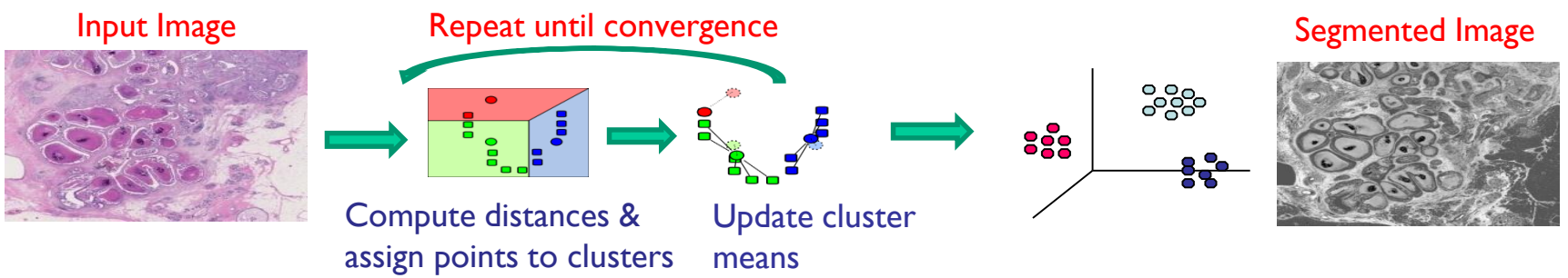
- Significant gap between requirements and capabilities of current platforms (even with projected improvements in device technology, parallelism, …)



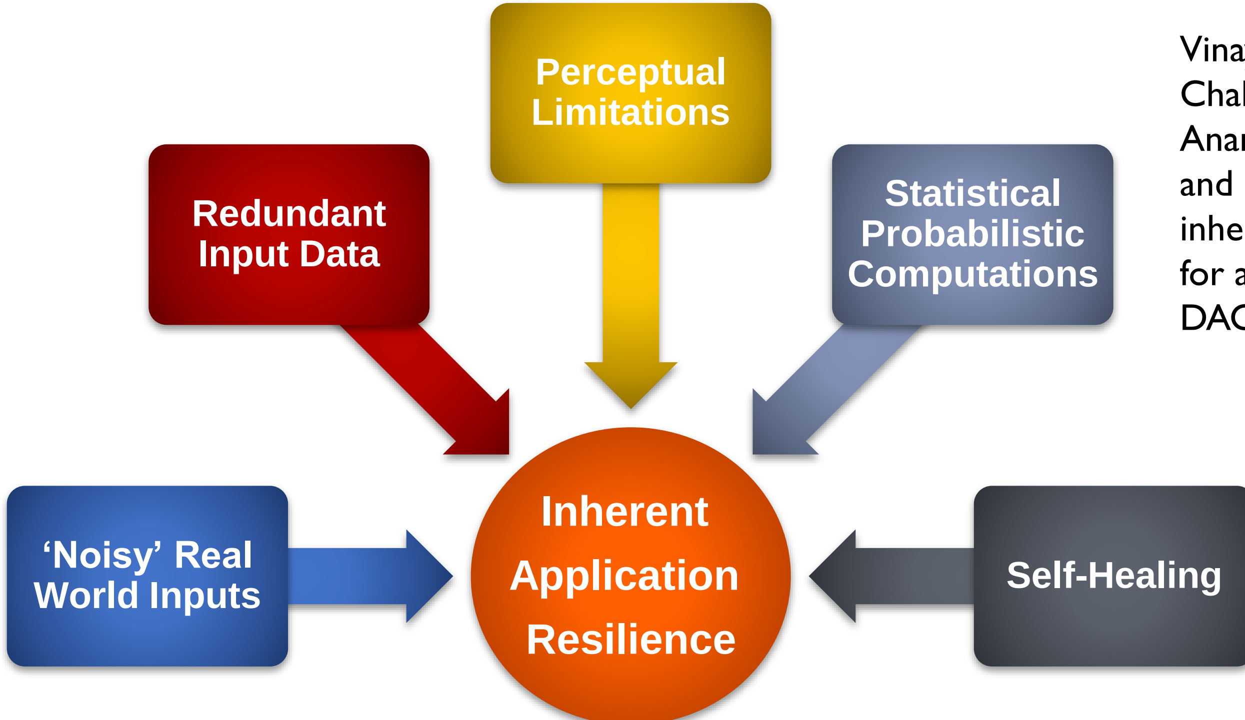
New Workloads Create a New Opportunity!

- Intrinsic application resilience: Ability to produce acceptable outputs despite underlying computations being performed in an approximate manner

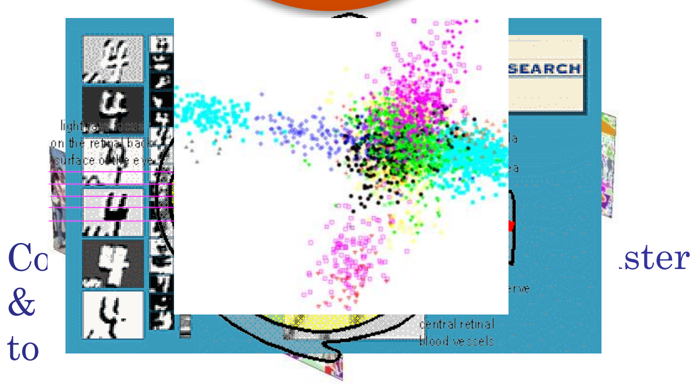




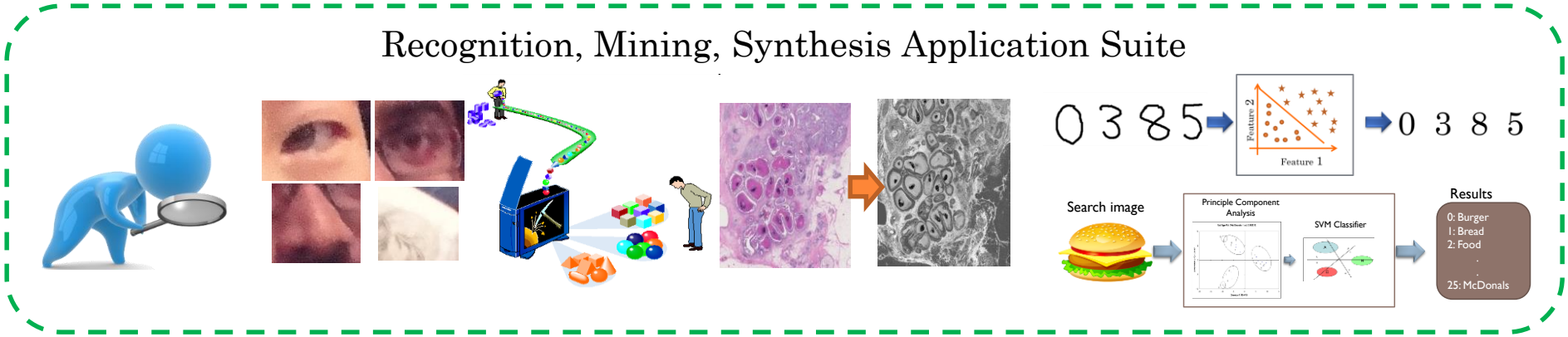
Intrinsic Application Resilience: Sources

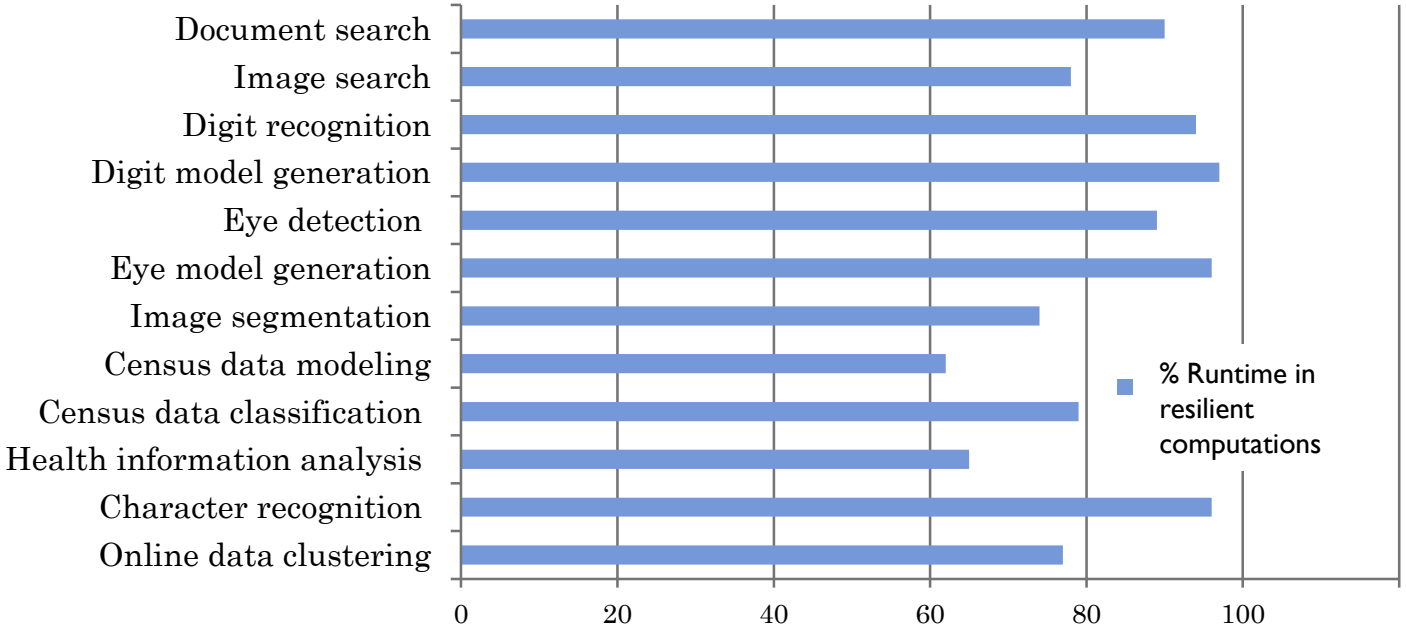


Vinay K. Chippa, Srimat T. Chakradhar, Kaushik Roy, Anand Raghunathan, "Analysis and characterization of inherent application resilience for approximate computing," DAC 2013.



Intrinsic Resilience in RMS Applications





Applications have a mix of resilient and sensitive computations

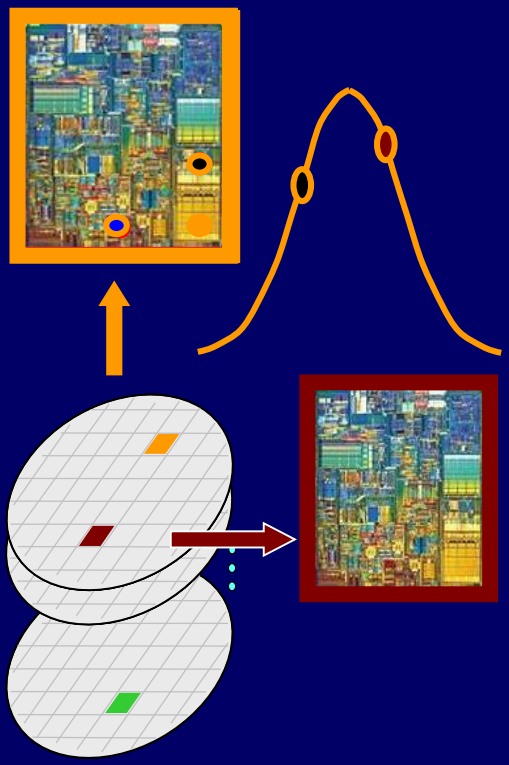
83% of runtime spent in computations that can be approximated

V. K. Chippa, S. T. Chakradhar, K. Roy and A. Raghunathan, "Analysis and characterization of inherent application resilience for approximate computing," DAC 2013.

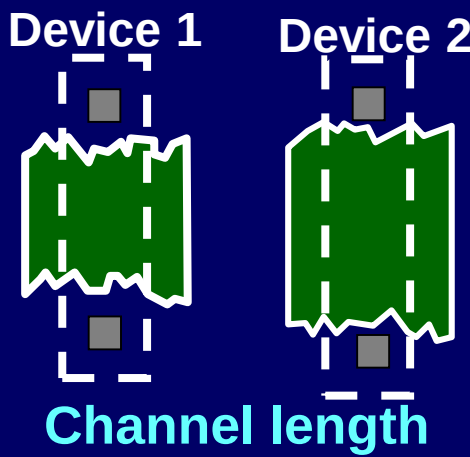
Motivation

- Process parameter variations are large in sub-45nm technologies
- Worst-case design would incur large power consumption
  - Proper approximations can lead to “much better than worst-case design” – low energy consumption with negligible drop in quality
  - Relaxes the design constraints
- Emerging devices like spin-transfer-torque based lateral spin valves are suitable for a class of approximate computing algorithms – brain inspired

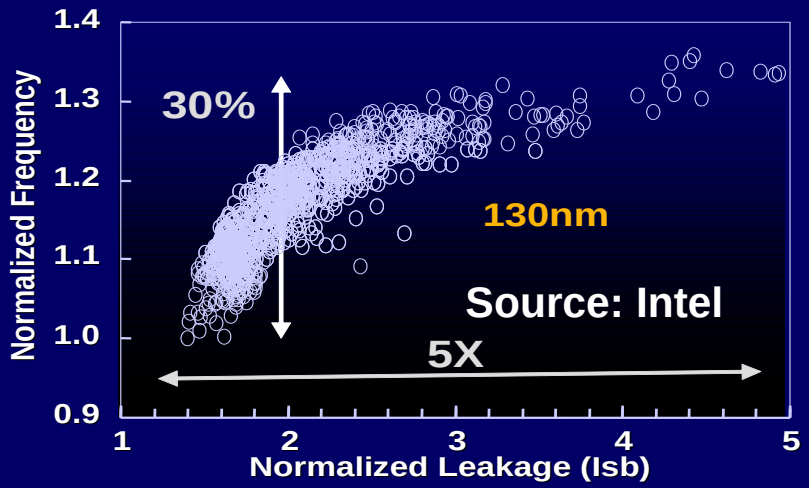
Variation in Process Parameters



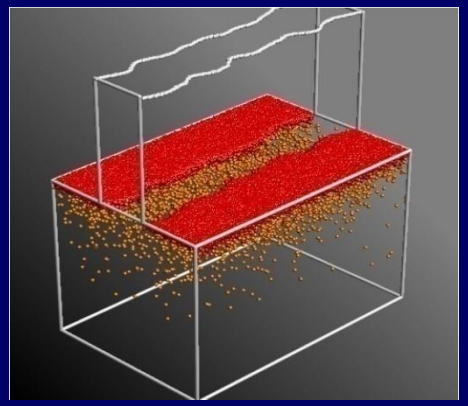
Inter and Intra-die Variations



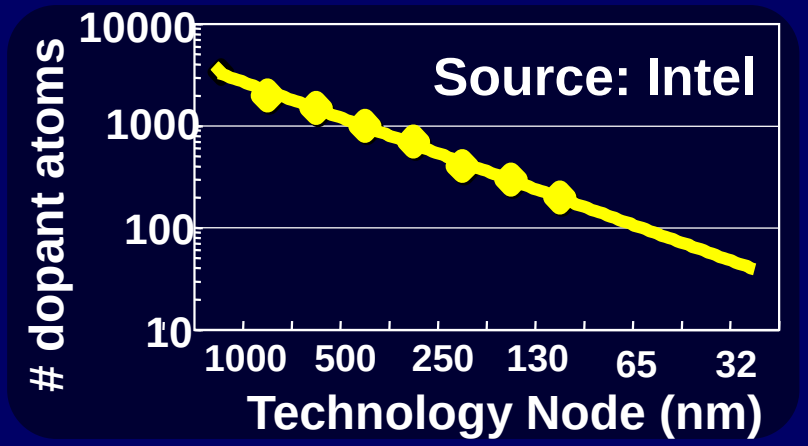
Channel length



Delay and Leakage Spread



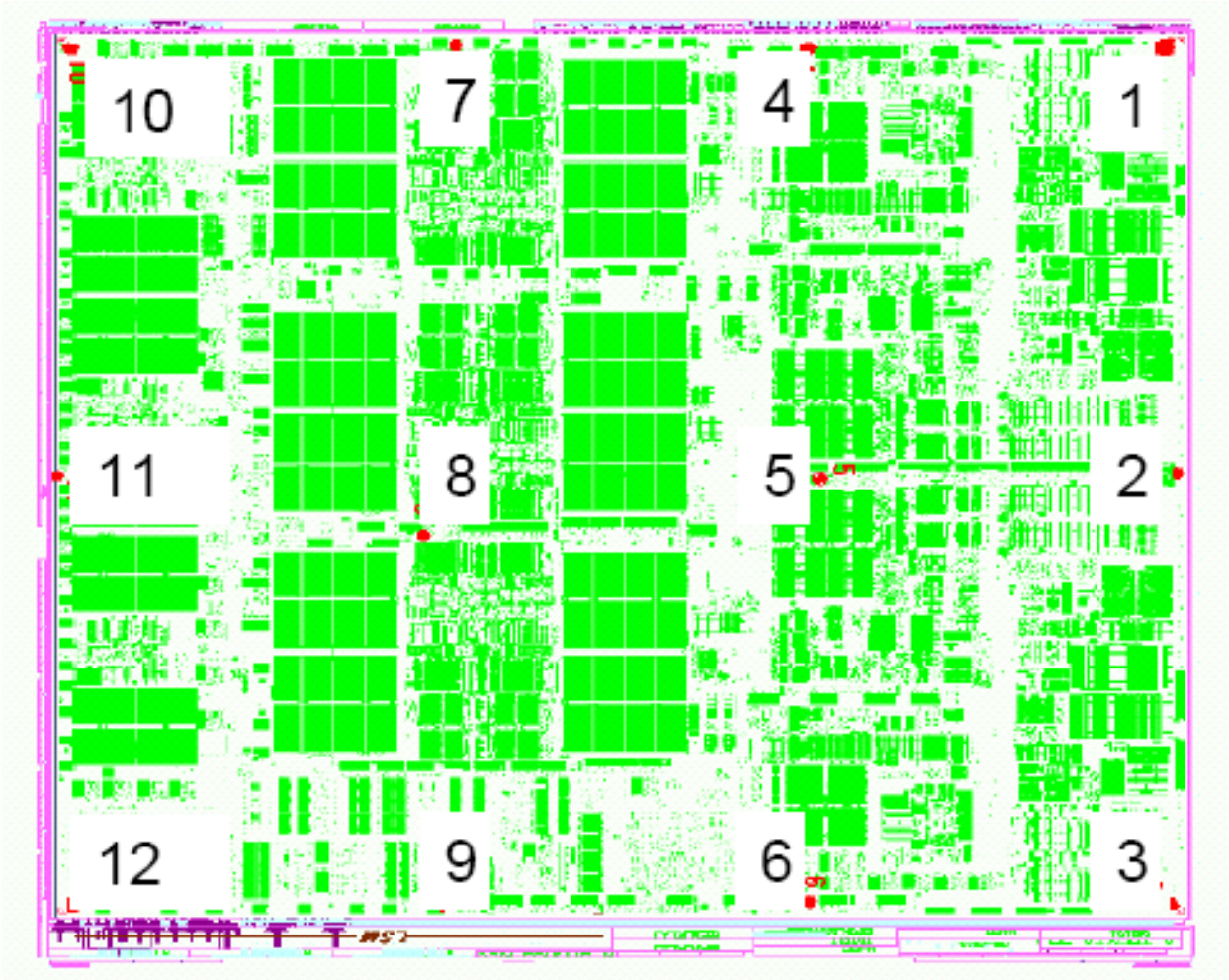
Random dopant fluctuation



Device parameters are no longer deterministic

Significance of Variation

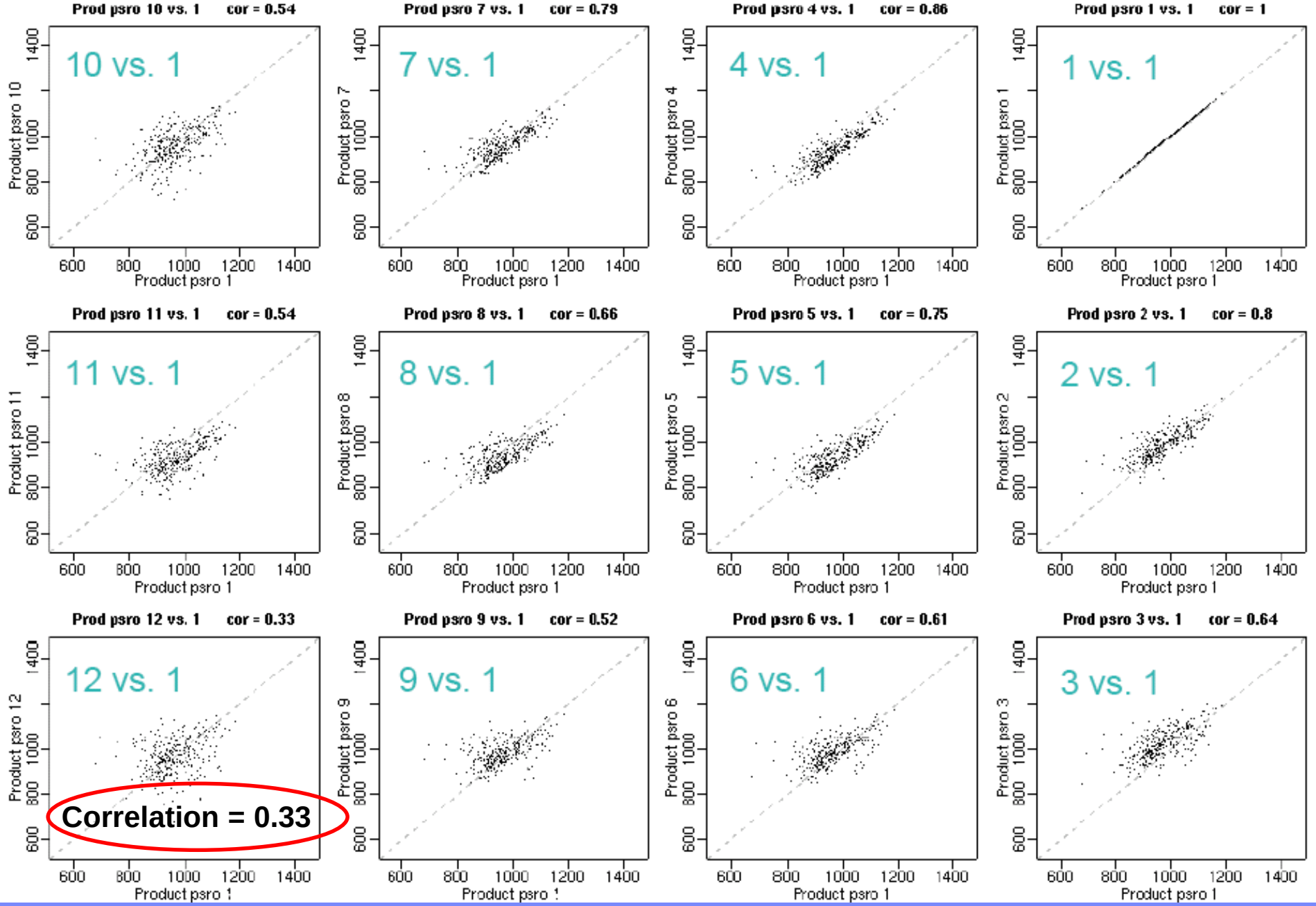
- **12 identical ring oscillators placed across 250 mm^2 chip**



Source: M. Bhushan

Source: M. Bhushan

12 ROs versus RO1

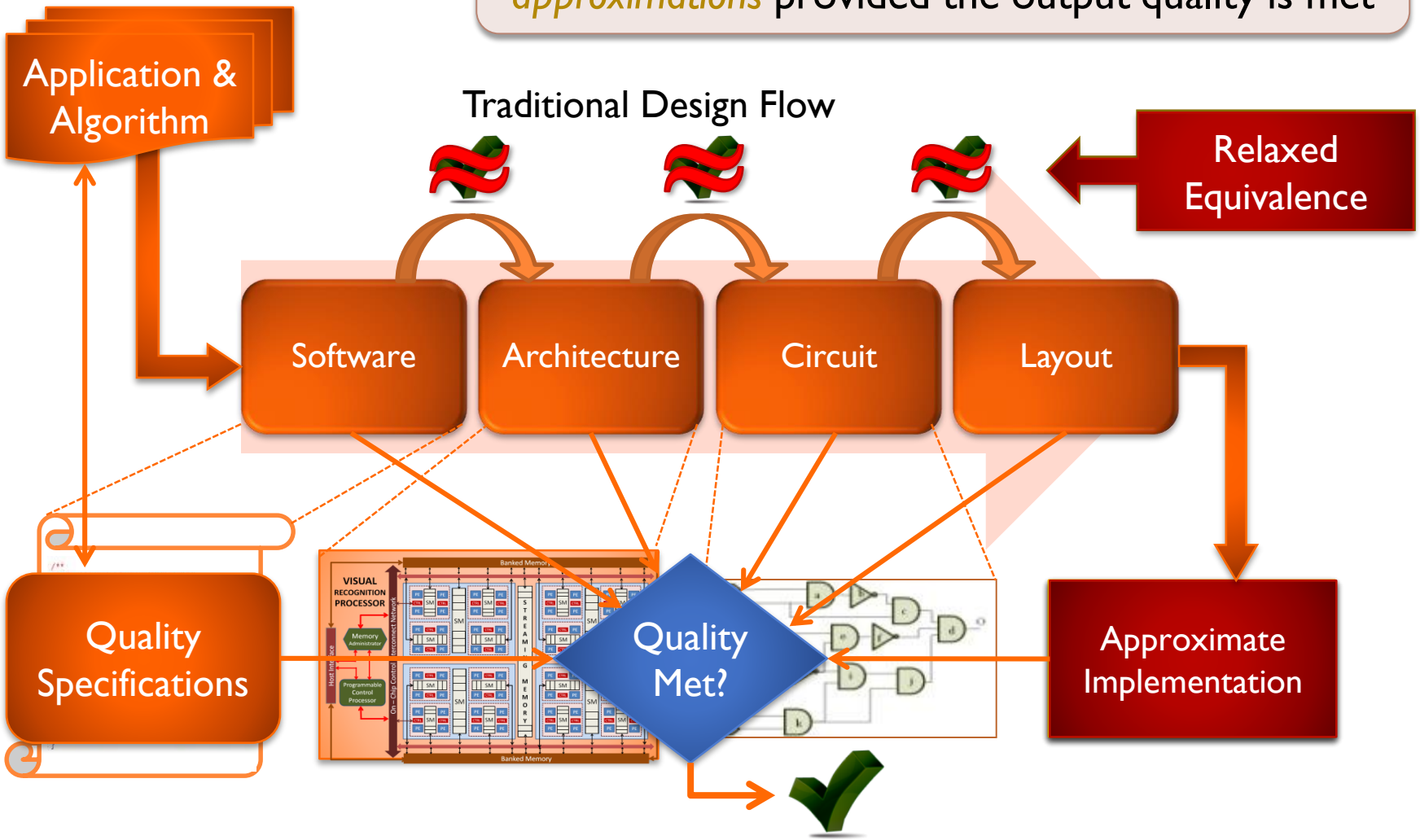


Need A New Design Philosophy

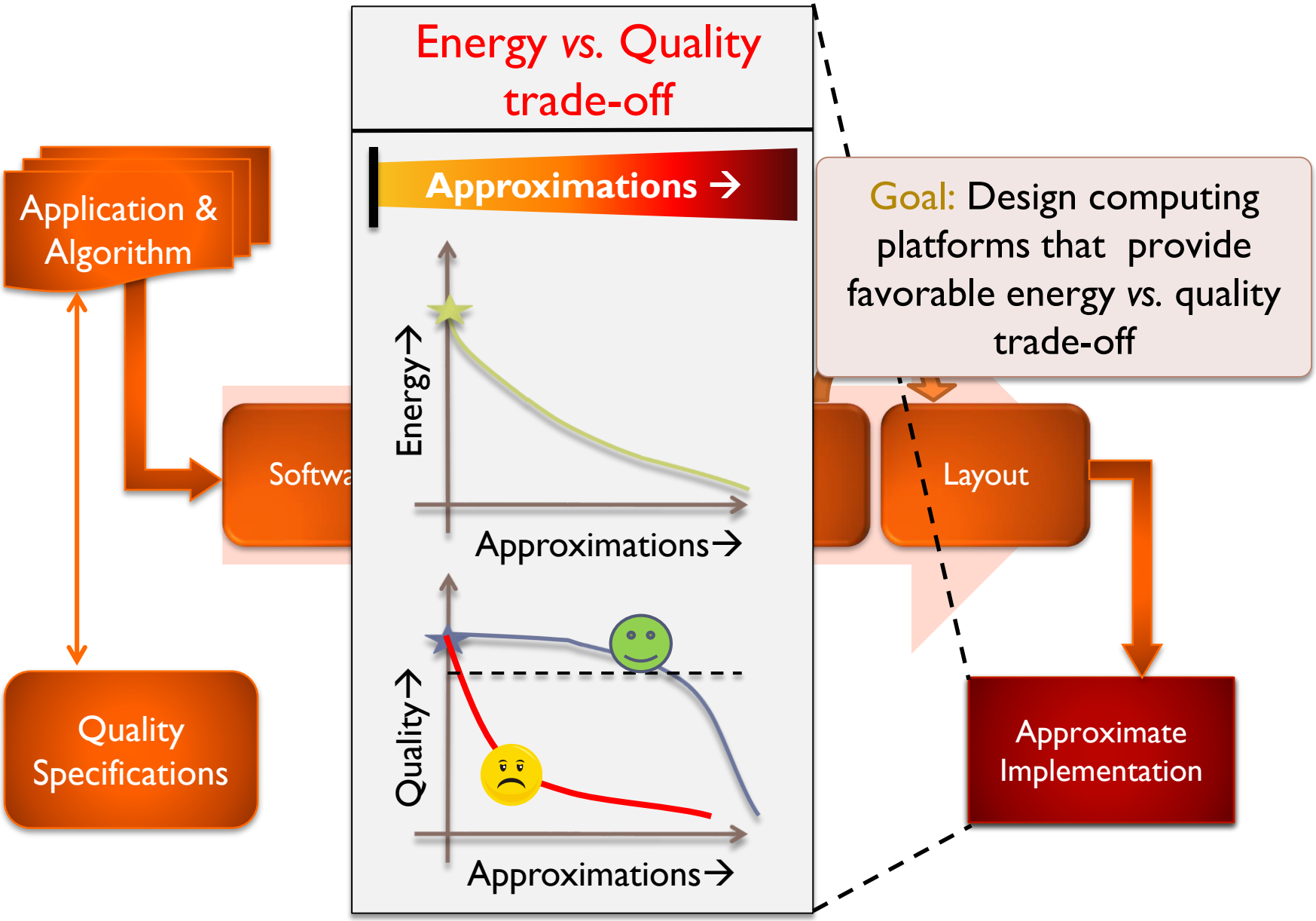
-Much Better than Worst case
-Lower Power consumption
-Resilient to Uncertainties
-Quality as a Metric

Approximate Computing: Design Philosophy

All levels of design abstraction can be subject to *approximations* provided the output quality is met



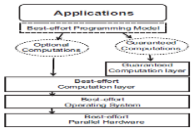
Approximate Computing: Design Philosophy

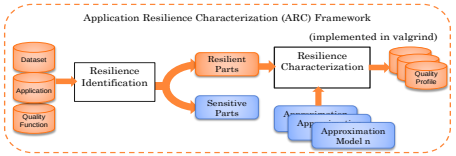


Approximate Computing @ Purdue

Approximate Computing in Software

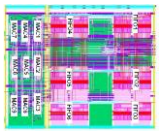
- Best-effort parallel computing (DAC 2010)
- Dependency relaxation (IPDPS 2010)
- Analysis and characterization of inherent application resilience (DAC 2013)
- Approximate Neural Networks (ISLPED 2014)

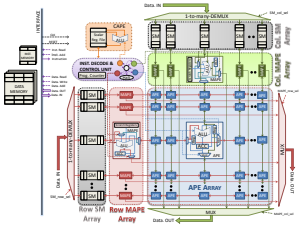




Approximate Architecture & System Design

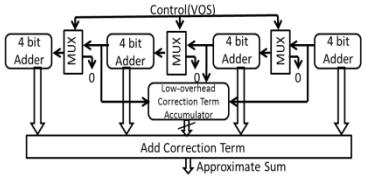
- Scalable Effort Hardware (DAC 2010, DAC 2011, CICC 2013)
- Significance Driven Computation: MPEG, H.264 (DAC2009, ISLPED 2009)
- QUORA: Quality Programmable vector processor (MICRO 2013)

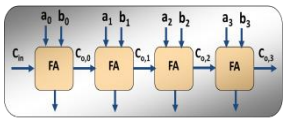


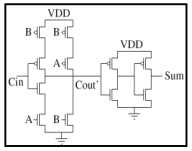


Approximate Circuit Design

- Voltage Scalable meta-functions (DATE 2011)
- Energy-quality tradeoff in DCT (DATE 2006)
- Approximate memory design (DAC 2009)
- IMPACT: Imprecise Adders for low power approximate computing (ISLPED 2011)

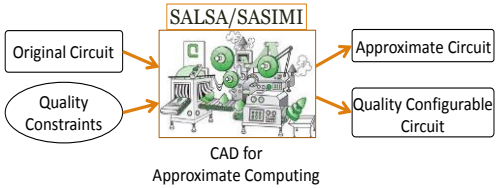






Design Automation for Approximate Computing

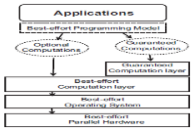
- SALSA: Systematic Logic Synthesis for Approximate Circuits (DAC 2012)
- Substitute-and-Simplify: Design of quality configurable circuits (DATE 2013)
- MACACO: Modeling and Verification of Circuits for Approximate Computing (ICCAD 2011)

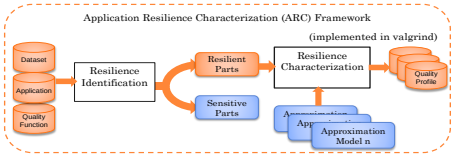


Approximate Computing @ Purdue

Approximate Computing in Software

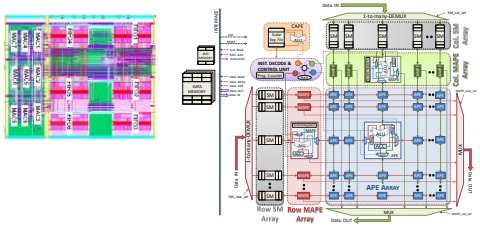
- Best-effort parallel computing (DAC 2010)
- Dependency relaxation (IPDPS 2010)
- Analysis and characterization of inherent application resilience (DAC 2013)
- Approximate Neural Networks (ISLPED 2014)





Approximate Architecture & System Design

- Scalable Effort Hardware (DAC 2010, DAC 2011, CICC 2013)
- Significance Driven Computation: MPEG, H.264 (DAC2009, ISLPED 2009)
- QUORA: Quality Programmable vector processor (MICRO 2013)



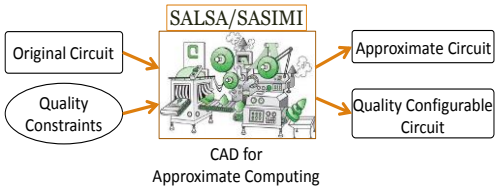
Approximate Circuit Design

- Voltage Scalable meta-functions (DATE 2011)
- Energy-quality tradeoff in DCT (DATE 2006)
- Approximate memory design (DAC 2009)
- IMPACT: Imprecise Adders for low power approximate computing (ISLPED 2011)

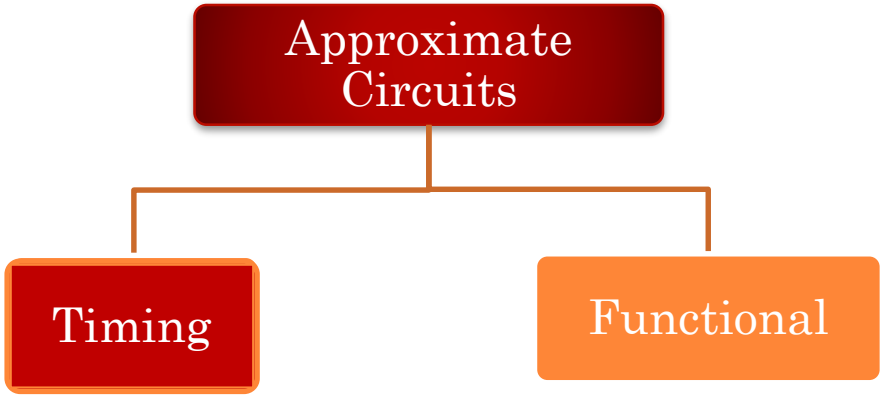
- Overscaled operation
- Functional approximation

Design Automation for Approximate Computing

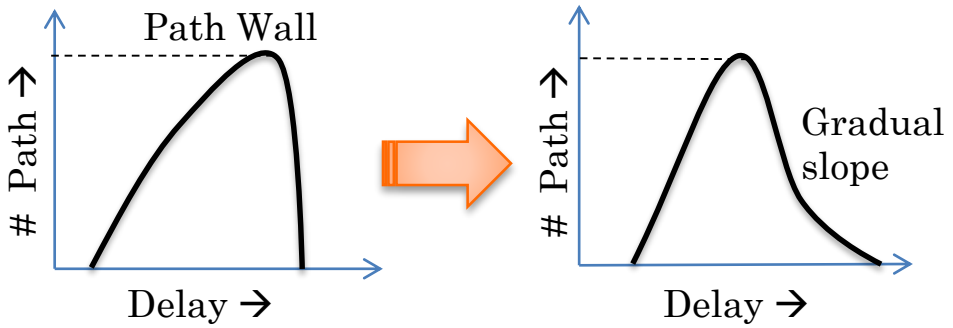
- SALSA: Systematic Logic Synthesis for Approximate Circuits (DAC 2012)
- Substitute-and-Simplify: Design of quality configurable circuits (DATE 2013)
- MACACO: Modeling and Verification of Circuits for Approximate Computing (ICCAD 2011)



Approximate Circuit Design

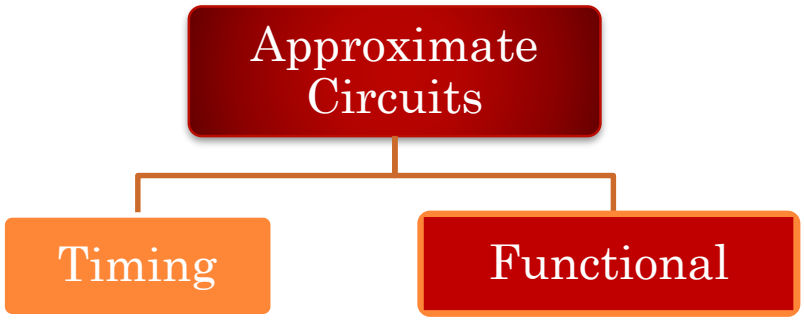


- Circuits subject to voltage over-scaling → timing errors
- Problem: "Wall Effect" Large number of near critical paths

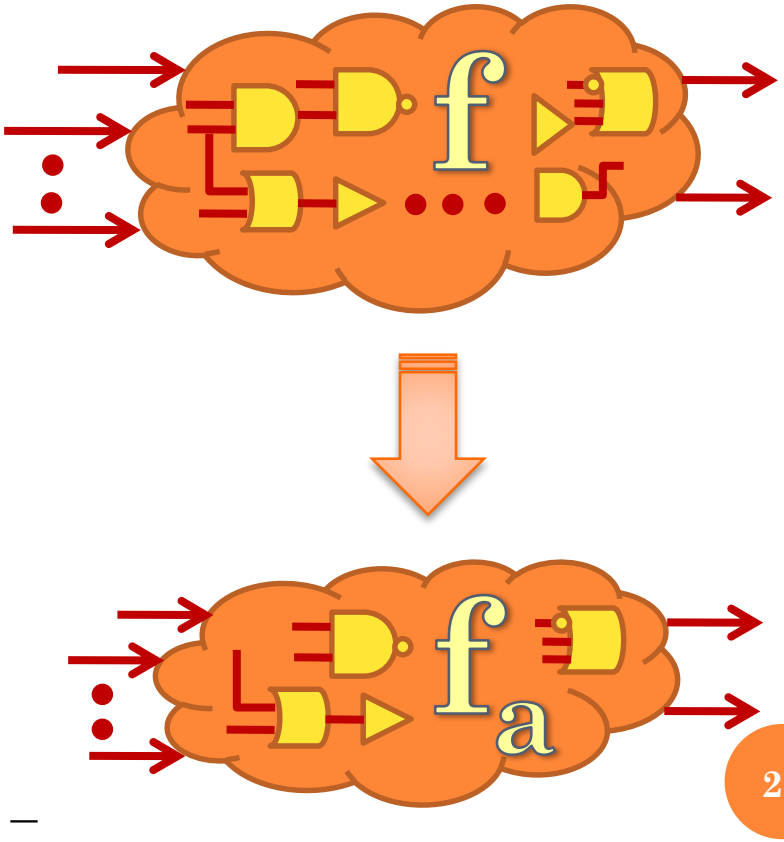


- Slack Redistribution – Kahng *et. al.* – ASPDAC 2010
- Dynamic Segmentation – Mohapatra *et. al.* – DATE 2011
- Adaptive Voltage Over-scaling – Krause *et. al.* – DATE 2011

Approximate Circuit Design



- The functionality is approximated such that logic is simplified



Manual Techniques:

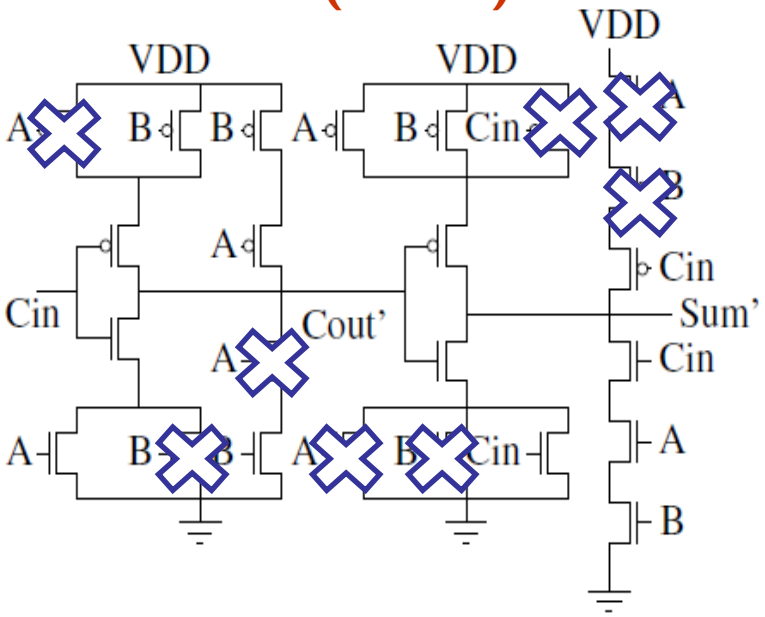
- Specific arithmetic blocks
  - Adders
    - Reverse Carry propagate (RCP) adder – Zhu *et. al.* TVLSI 2010
    - IMPACT – Gupta *et. al.* – ISLPED 2011
  - Multipliers
    - Under designed Multiplier – Kulkarni *et. al.* VLSID 2011

Approximate Circuits

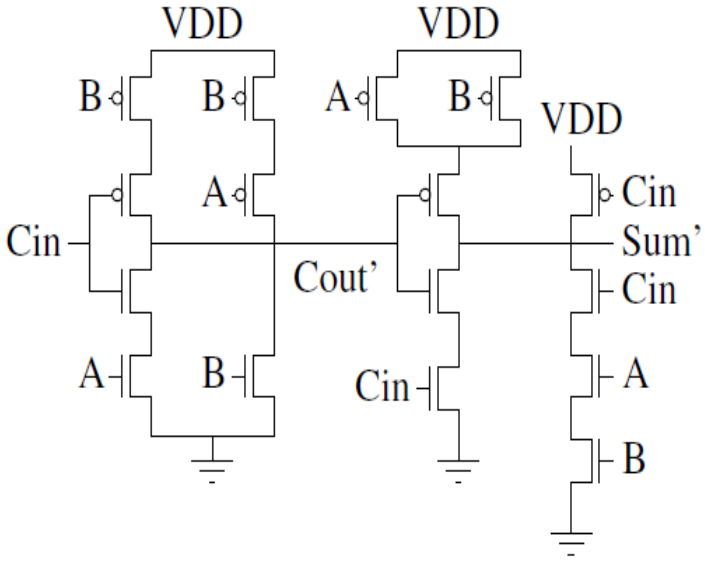
- Functional approximation
  - Modify functionality such that it leads to simplified implementation
- Example: Approximate full adder

Inputs			Approx. 1		Approx. 2		Approx. 3	
A	B	C_{in}	Sum_1	$Cout_1$	Sum_2	$Cout_2$	Sum_3	$Cout_3$
0	0	0	**1**	0	0	0	0	0
0	0	1	1	0	1	0	**0**	0
0	1	0	**0**	**1**	**0**	0	1	0
0	1	1	0	1	**1**	**0**	**1**	**0**
1	0	0	1	0	**0**	**1**	**0**	**1**
1	0	1	0	1	0	1	0	1
1	1	0	0	1	0	1	**1**	1
1	1	1	**0**	1	1	1	1	1

Conventional (mirror) adder



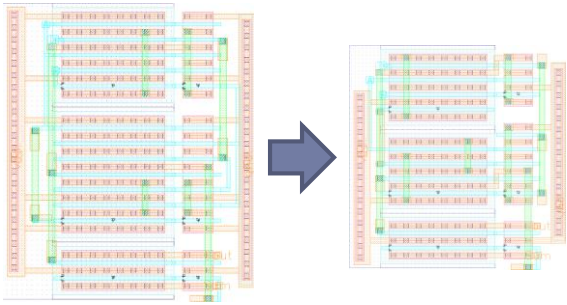
Approximate adder (Approx. 1)



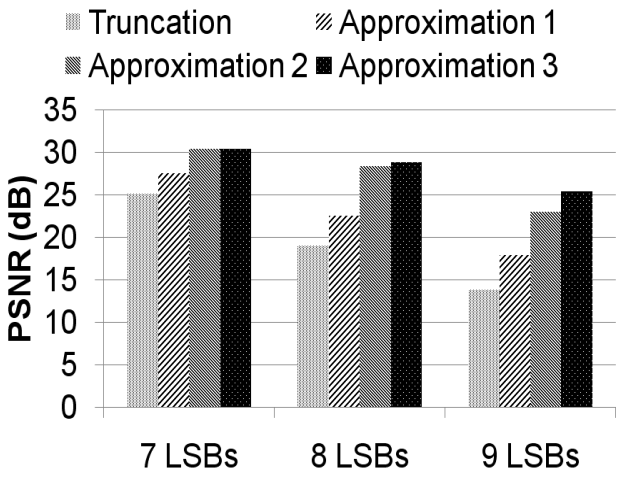
Vaibhav Gupta, Debabrata Mohapatra, Anand Raghunathan, Kaushik Roy: Low-Power Digital Signal Processing Using Approximate Adders. IEEE TCAD, Jan. 2013.

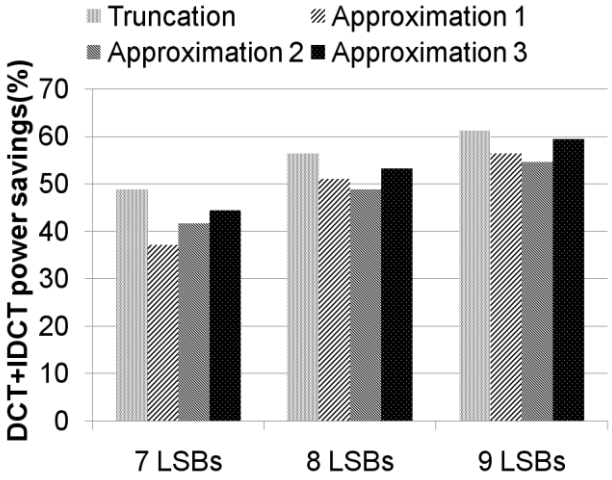
Approximate Circuits

- Benefits: Fewer transistors, lower dynamic & leakage power, shorter critical path, opportunity for down-sizing



Evaluation (JPEG compression)





Accurate

PSNR = 31.16

Truncation

PSNR = 19.04

Approx.

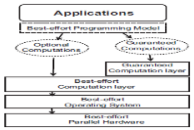
PSNR = 28.9

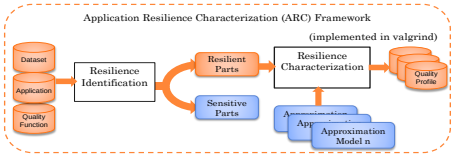
60% power savings and 37% area savings with 5.7 dB loss in output quality (PSNR)

Approximate Computing @ Purdue

Approximate Computing in Software

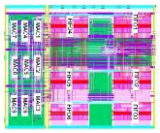
- Best-effort parallel computing (DAC 2010)
- Dependency relaxation (IPDPS 2010)
- Analysis and characterization of inherent application resilience (DAC 2013)
- Approximate Neural Networks (ISLPED 2014)

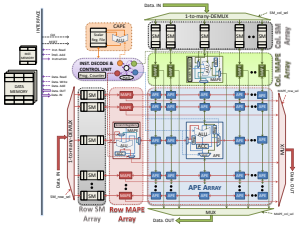




Approximate Architecture & System Design

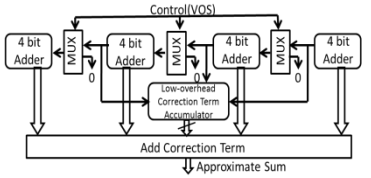
- Scalable Effort Hardware (DAC 2010, DAC 2011, CICC 2013)
- Significance Driven Computation: MPEG, H.264 (DAC2009, ISLPED 2009)
- QUORA: Quality Programmable vector processor (MICRO 2013)

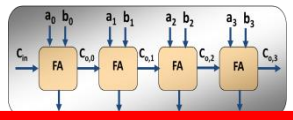


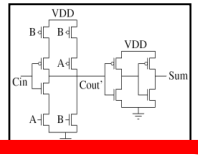


Approximate Circuit Design

- Voltage Scalable meta-functions (DATE 2011)
- Energy-quality tradeoff in DCT (DATE 2006)
- Approximate memory design (DAC 2009)
- IMPACT: Imprecise Adders for low power approximate computing (ISLPED 2011)







Design Automation for Approximate Computing

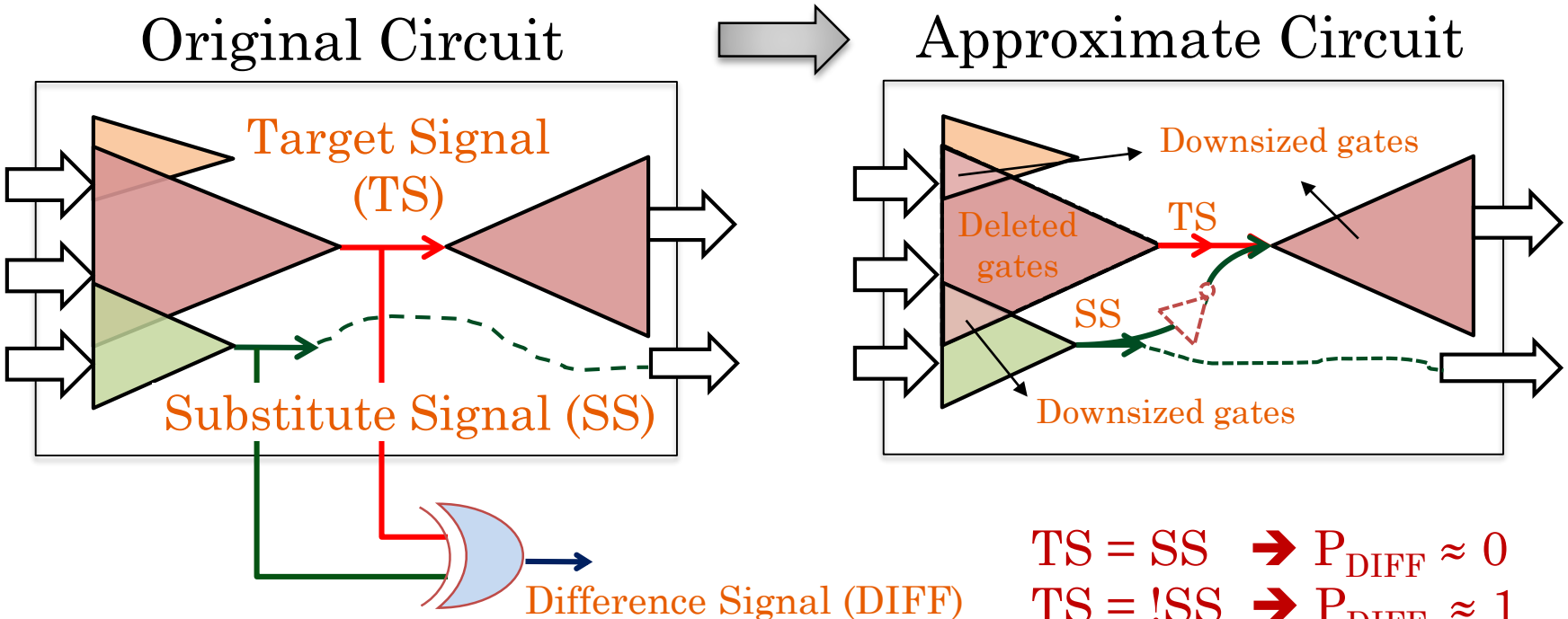
- SALSA: Systematic Logic Synthesis for Approximate Circuits (DAC 2012)
- Substitute-and-Simplify: Design of quality configurable circuits (DATE 2013)
- MACACO: Modeling and Verification of Circuits for Approximate Computing (ICCAD 2011)

- Synthesis of AC
- Verification

DESIGN METHODOLOGY: SUBSTITUTE & SIMPLIFY

Swagath et. Al., DATE 2013

SUBSTITUTE-AND-SIMPLIFY (SASIMI)



TS = SS ➔ $P_{DIFF} \approx 0$

TS = !SS ➔ $P_{DIFF} \approx 1$

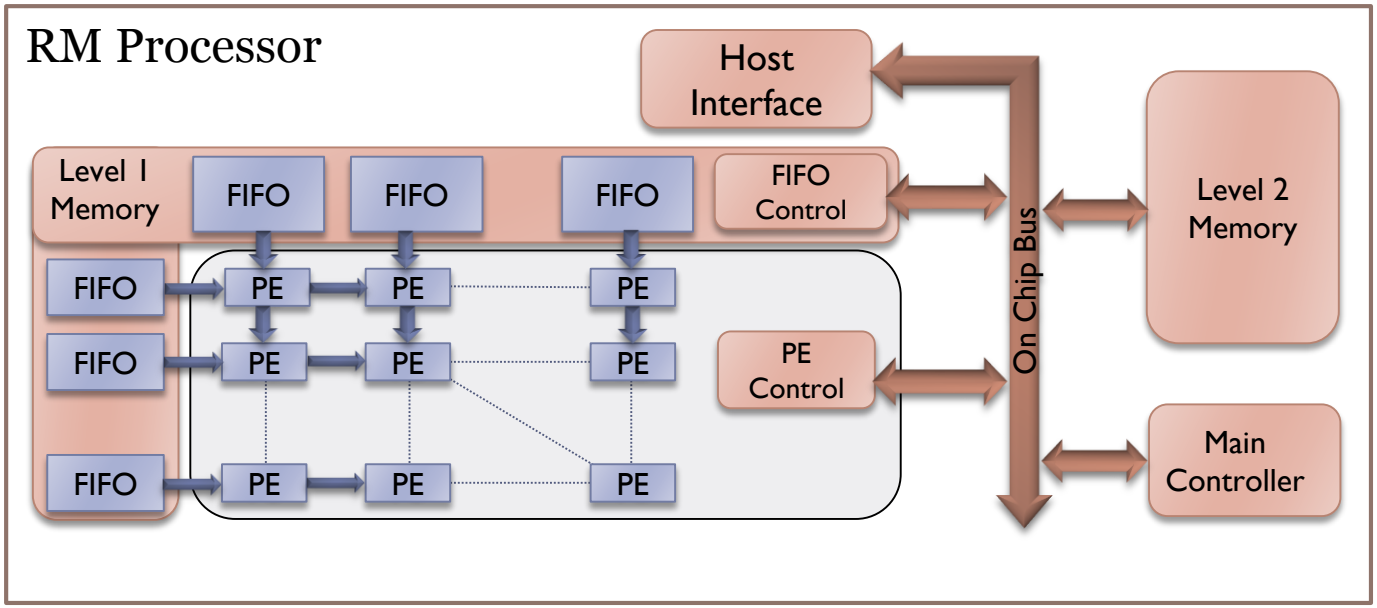
- Key I… funct… …s

Substitution pairs should be judiciously selected!

- Substitute one in place of the other
  - Circuit becomes approximate
- Simplify the circuit: Logic Deletion & Downsizing

Application Level Case Study of SASIMI Generated Circuits

- Processing elements (PEs) replaced with SASMI generated approximate adders and multipliers

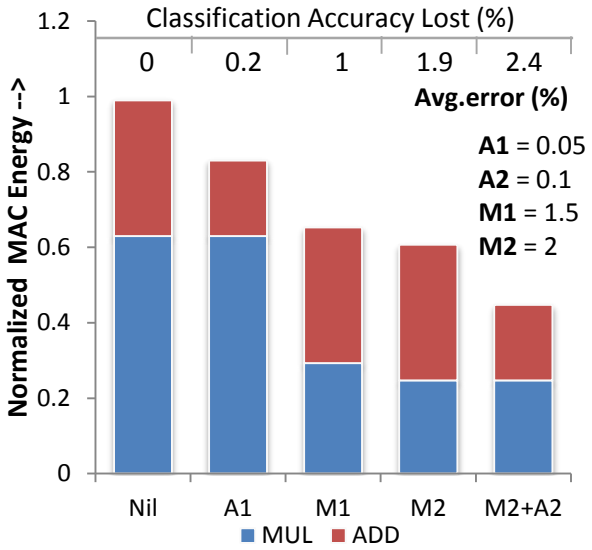


Source: Scalable Effort hardware Chippa *et. al.* – DAC 2010

- Two Recognition applications based on
  - Support Vector Machines (SVMs)
  - K-nearest Neighbors

Results: Application Level Case Study

K-Nearest Neighbors

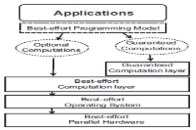


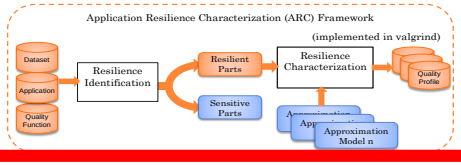
- 30% energy savings in MAC units for 1% loss in classification accuracy
- Savings increase to 55% for <2.5% loss in accuracy
- Quality requirements can be tailored to the needs of the application

Approximate Computing @ Purdue

Approximate Computing in Software

- Best-effort parallel computing (DAC 2010)
- Dependency relaxation (IPDPS 2010)
- Analysis and characterization of inherent application resilience (DAC 2013)
- Approximate Neural Networks (ISLPED 2014)





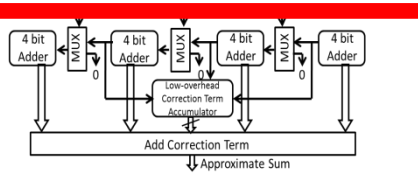
Approximate Architecture & System Design

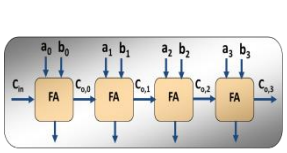
- Scalable Effort Hardware (DAC 2010, DAC 2011, CICC 2013)
- Significance Driven Computation: MPEG, H.264 (DAC2009, ISLPED 2009)
- QUORA: Quality Programmable vector processor (MICRO 2013)

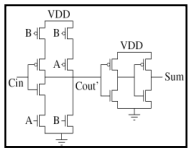
- SDC
- Approximate computing in programmable processors

Approximate Circuit Design

- Voltage Scalable meta-functions (DATE 2011)
- Energy-quality tradeoff in DCT (DATE 2006)
- Approximate memory design (DAC 2009)
- IMPACT: Imprecise Adders for low power approximate computing (ISLPED 2011)

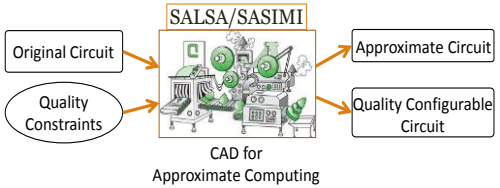






Design Automation for Approximate Computing

- SALSA: Systematic Logic Synthesis for Approximate Circuits (DAC 2012)
- Substitute-and-Simplify: Design of quality configurable circuits (DATE 2013)
- MACACO: Modeling and Verification of Circuits for Approximate Computing (ICCAD 2011)



Significance Driven Computation for Error-resilient Applications

-Algorithm

-Architecture

How do you achieve “graceful degradation”?

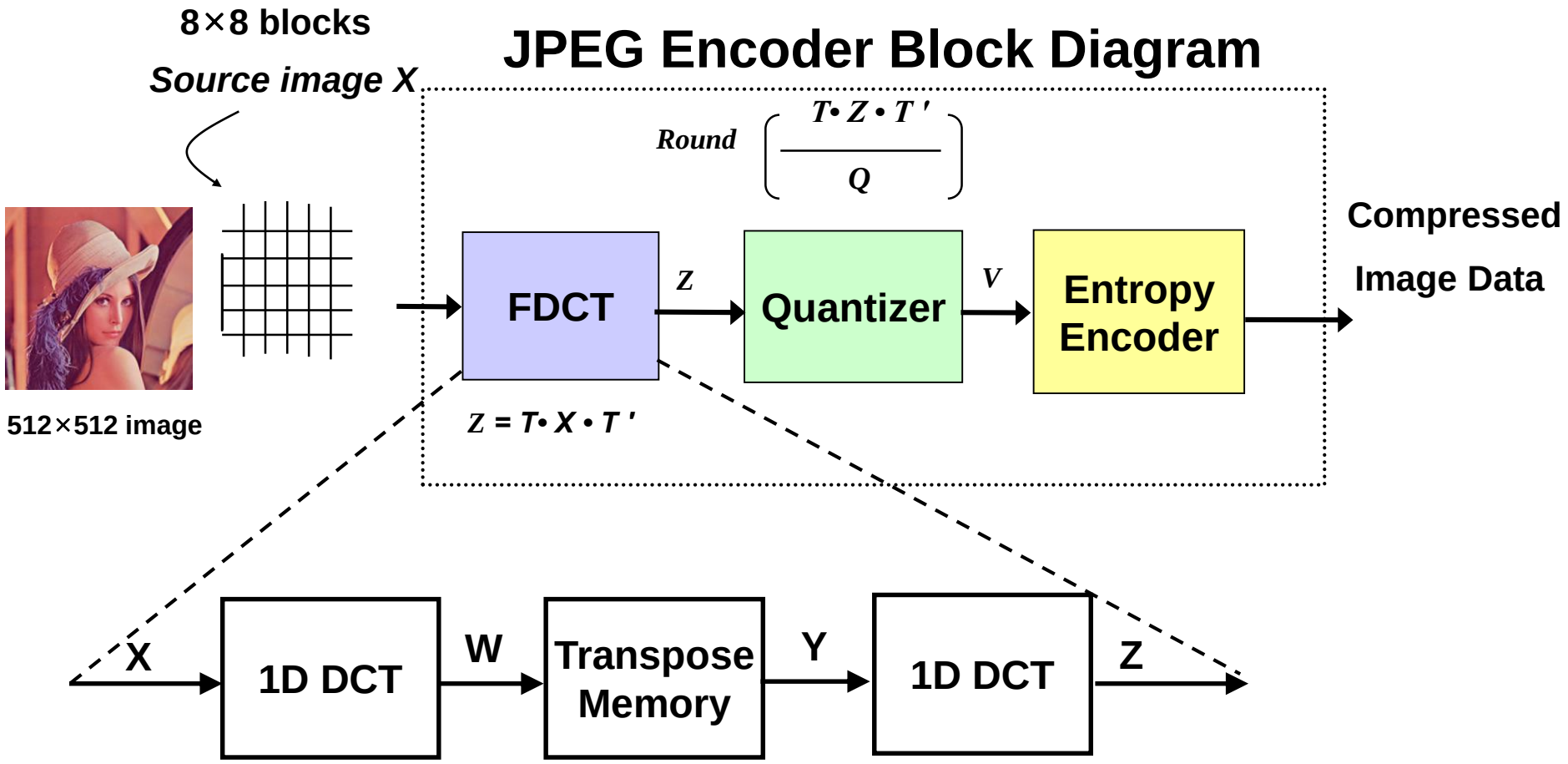
- All computations are “not equally important” for determining outputs
- Identify important and unimportant computations based on output “sensitivity”
- Compute important computations with “higher priority”
- Delay errors due to variations/ Vdd scaling “affect only” non-important computations
- “Gradual degradation” in output with voltage scaling and process variations

APPLICATION TO DSP: LOW-POWER, UNEQUAL ERROR PROTECTION, & ERROR RESILIENCY

Prof. Kaushik Roy

@ Purdue Univ.

Example: Low-Voltage Image Compression



- DCT is used in current international image/video coding standards
  - JPEG, MPEG, H.261, H.263

DCT (Discrete Cosine Transform)

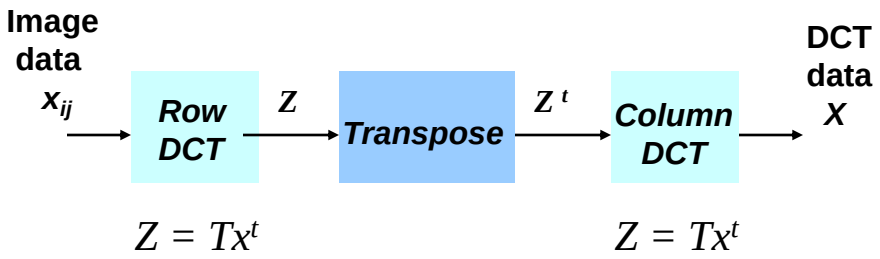
$$X_{kl} = \frac{c(k)c(l)}{4}\sum_{i=0}^{7}\sum_{j=0}^{7} x_{ij}\cos\left(\frac{(2i+1)k\pi}{16}\right)\cos\left(\frac{(2j+1)l\pi}{16}\right)$$

$$k,l = 0,1,....,7 \text{ and } c(k) = \begin{cases} \frac{1}{\sqrt{2}}, k = 0 \\ 1, otherwise \end{cases}$$

$$T = \begin{bmatrix} d & d & d & d & d & d & d & d \\ a & c & e & g & -g & -e & -c & -a \\ b & f & -f & -b & -b & -f & f & b \\ c & -g & -a & -e & e & a & g & -c \\ d & -d & -d & d & d & -d & -d & d \\ e & -a & g & c & -c & -g & a & -e \\ f & -b & b & -f & -f & b & -b & f \\ g & -e & c & -a & a & -c & e & -g \end{bmatrix}$$

Note the symmetry of the DCT coef. matrix

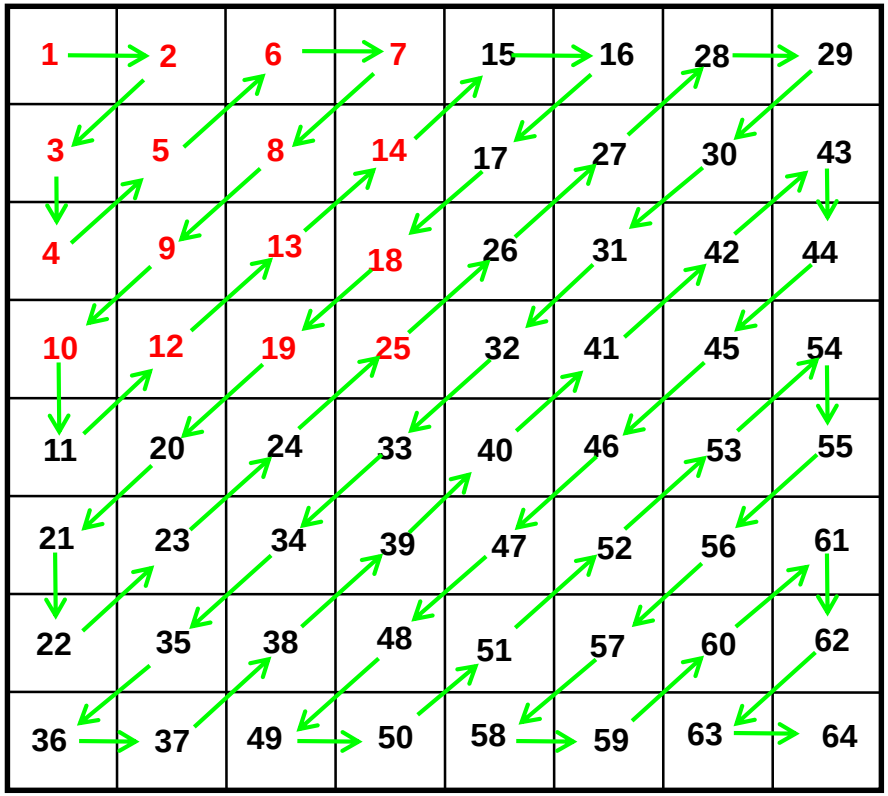
$$Z = Tx^t \ , Z = Tx^t$$



Prof. Kaushik Roy

@ Purdue Univ.

Energy Distribution of a 2D-DCT Output



High energy components (important outputs 75% energy)

Low energy components (less important outputs)

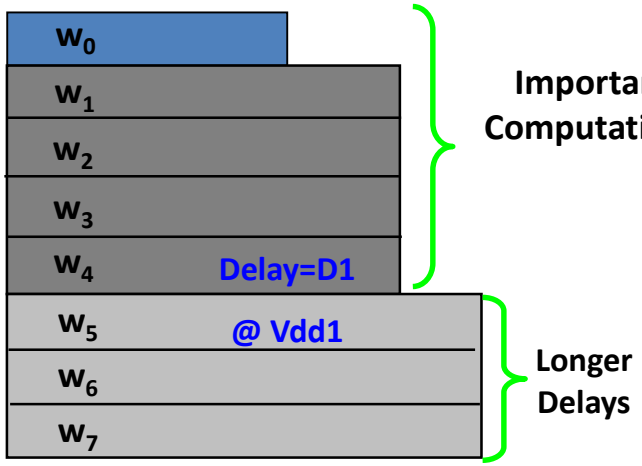
Can important components be computed with higher priority ?

Prof. Kaushik Roy

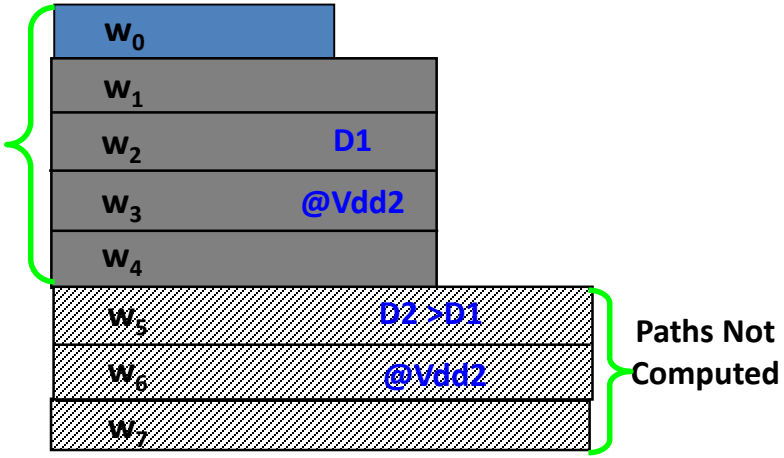
@ Purdue Univ.

Proposed DCT under V_{dd} scaling

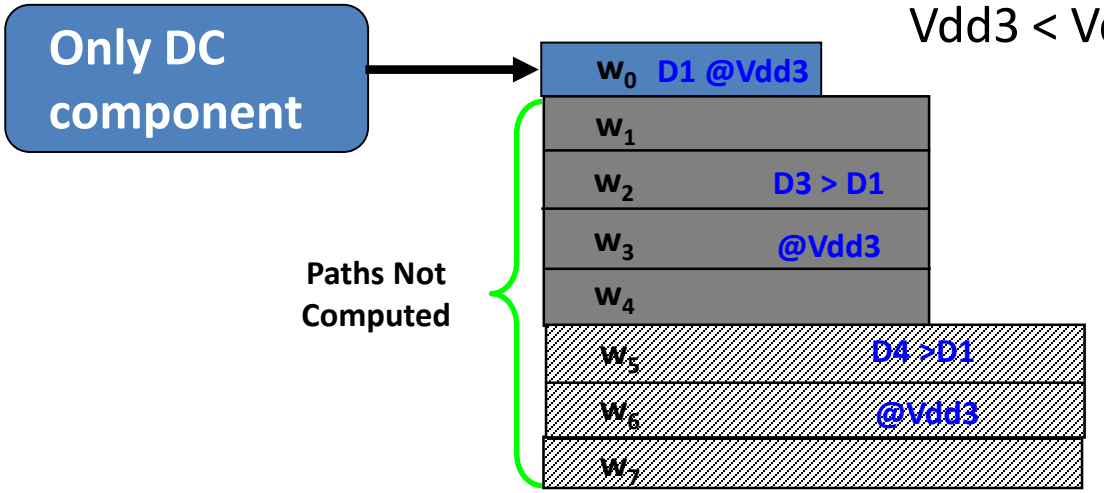
Proposed Design with high/low delay paths



Scaled Vdd: Longer paths under Vdd scaling



Extreme Scaled Vdd: Shorter paths affected

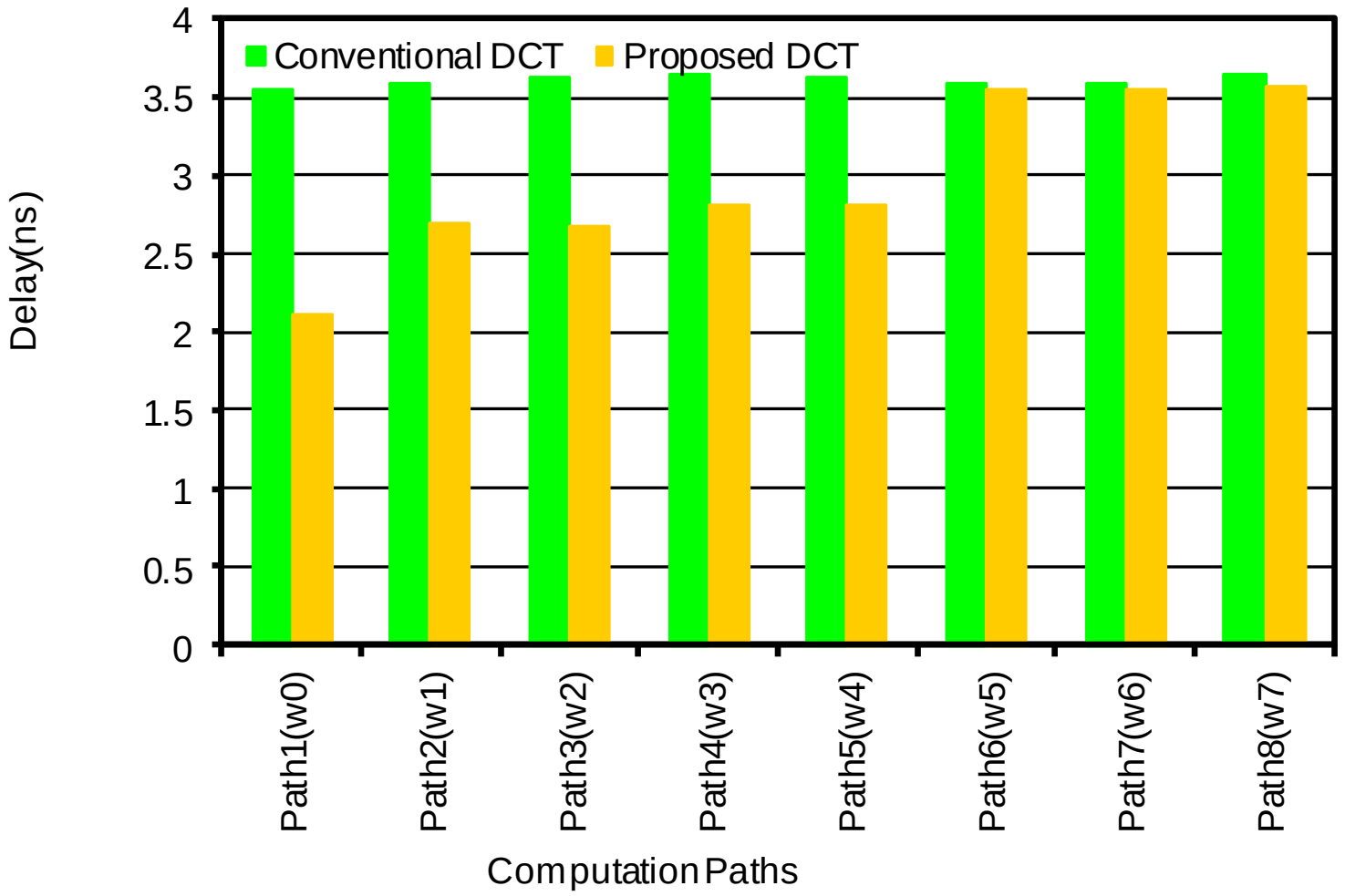


Vdd3 < Vdd2 < Vdd1(nominal)

Prof. Kaushik Roy

@ Purdue Univ.

1D-DCT Path Delay Comparisons



DCT: Approximations with Shared Multiplier

- Specifically targets the reduction of *redundant* computation in the vector scaling operation.

< Coefficient Decomposition >

$c = 111010001100$

$c = 2^9(111) + 2^7(1) + 2^2(11)$

alphabet set = {1, 11, 111}

Alphabets - chosen basic bit sequences

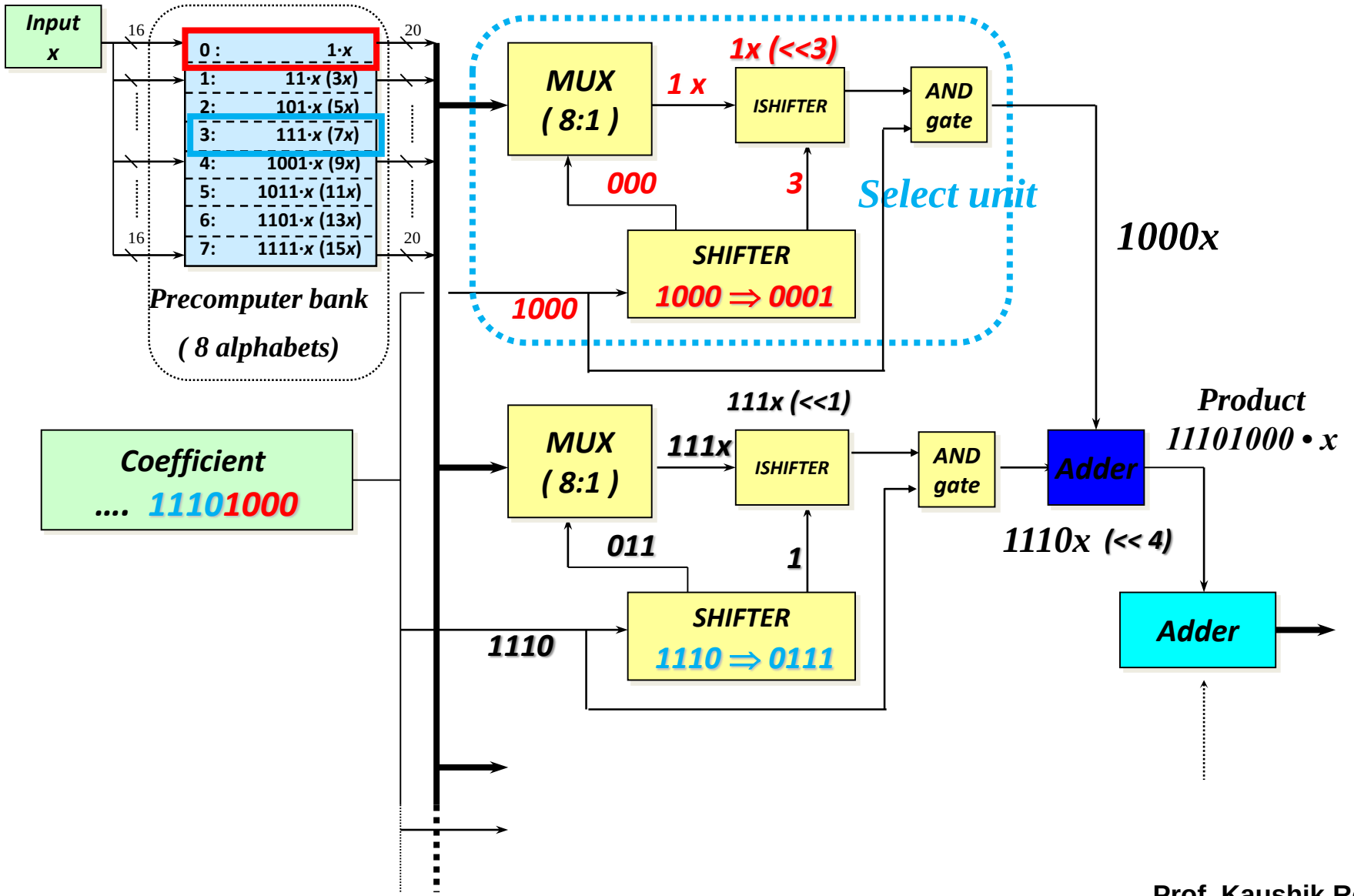
Alphabet set - a set of alphabets that covers all the coefficients in vector C

$c \bullet x = 111010001100 \bullet x$

$c \bullet x = 2^9(0111 \bullet x) + 2^7(0001 \bullet x) + 2^2(0011 \bullet x)$

if 0111 • x , 0001 • x and 0011 • x are available, c • x can be significantly simplified as add and shift operation

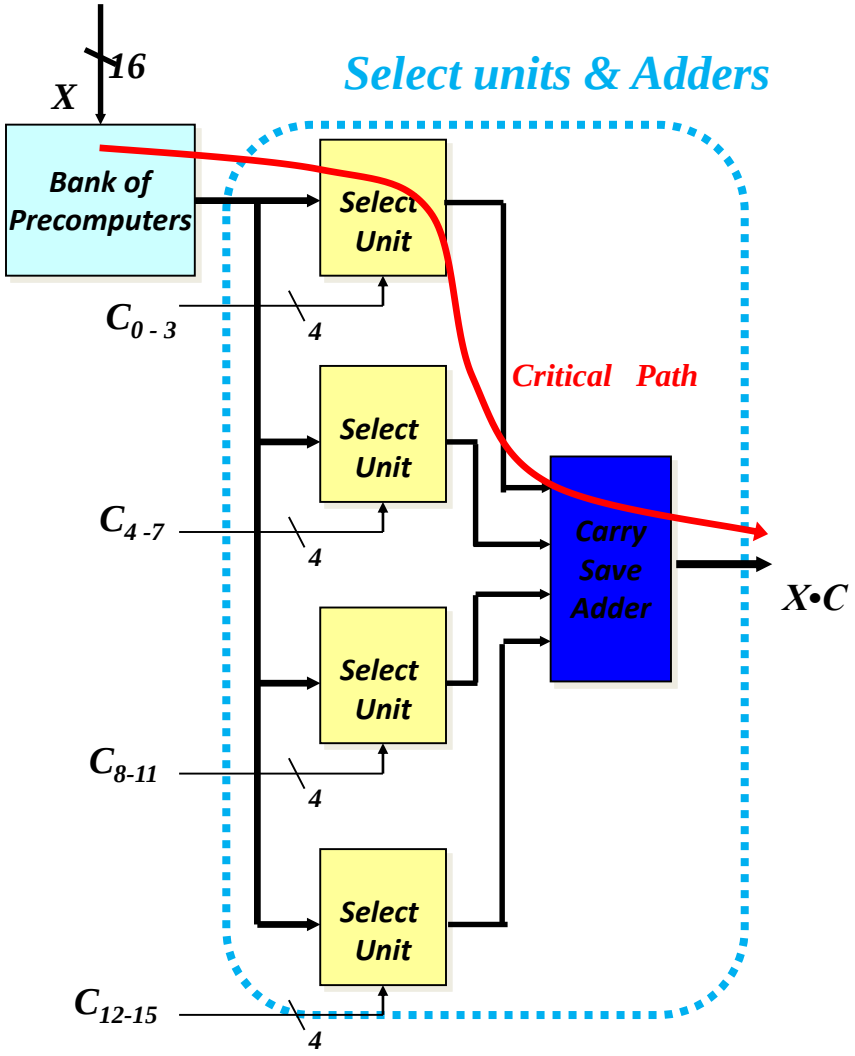
Shared Multiplier Architecture



Prof. Kaushik Roy

@ Purdue Univ.

16×16 Shared Multiplier Implementation

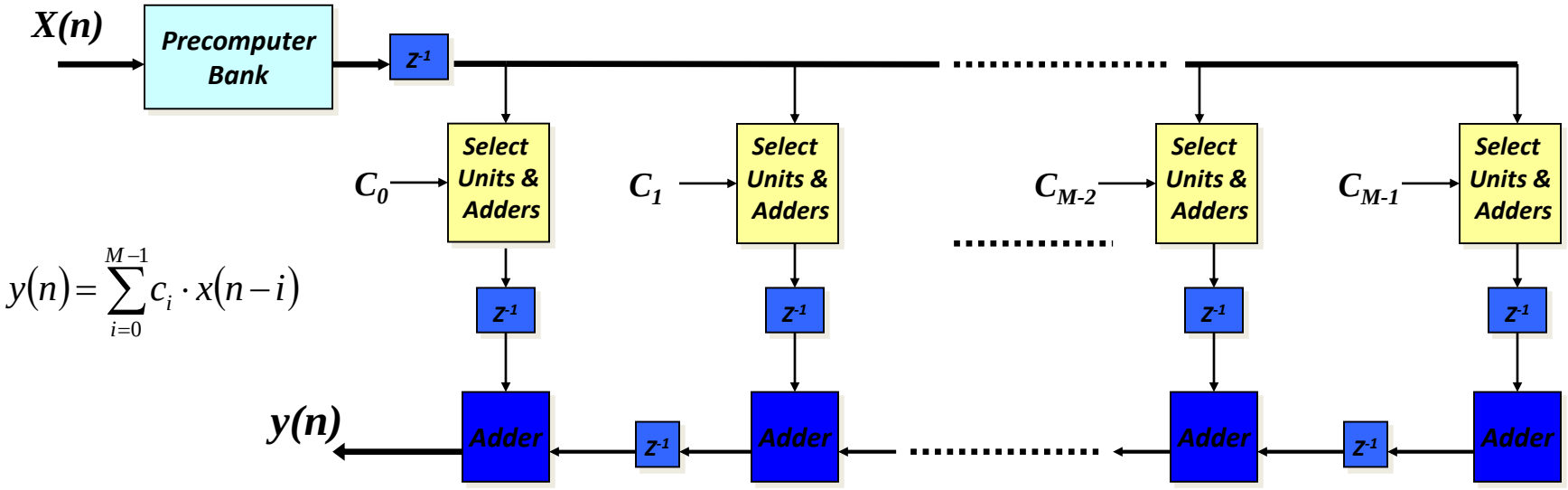


- 16 × 16 Wallace tree multiplier (WTM) and carry save array multiplier (CSAM) are also implemented for comparison.

	Precomputer	*Select units & Adders*	*WTM*	*CSAM*
Delay	*6.923 ns*	*11.231 ns*	*16.638 ns*	*23.398 ns*
Power	*18.06 mW*	*18.91 mW*	*22.80 mW*	*21.78 mW*
Area	*162340 μm²*	*252120 μm²*	*241000 μm²*	*175640 μm²*

- *CMU library (0.35 μm technology)*

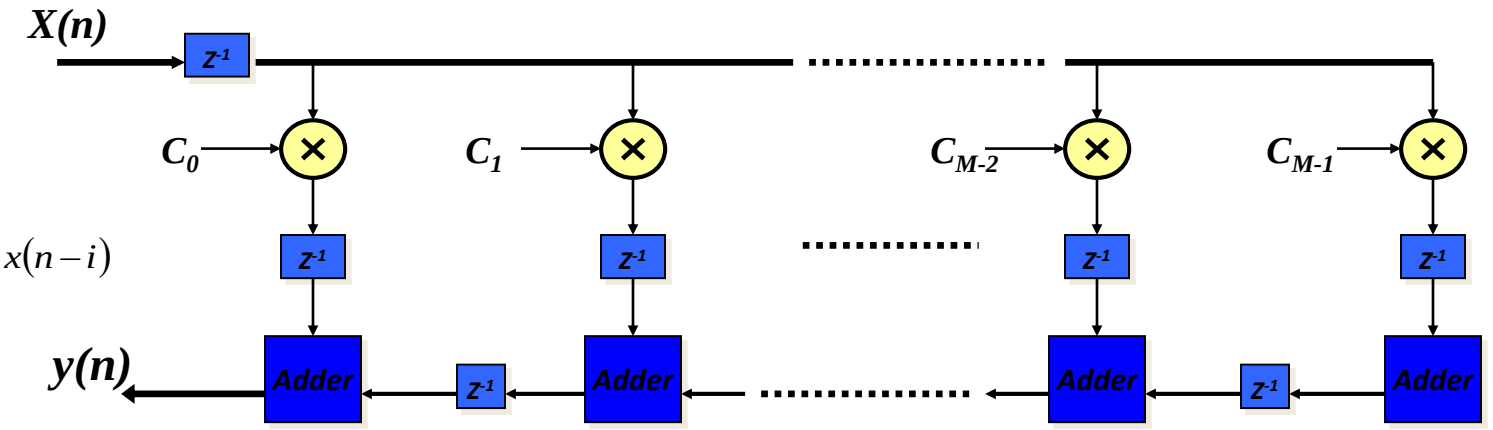
FIR filter using Shared Multiplier



- Computations $a_k \bullet x$ are performed just once for all alphabets and these values are shared by all the select units
- Only *select unit and adders* and lie on the critical path

FIR filter using WT & CSAM

$$y(n) = \sum_{i=0}^{M-1} c_i \cdot x(n-i)$$



Filter	FIR filter using Shared Multiplier	FIR filter using Wallace Tree	FIR filter using Carry Save Array
Clock Cycle	13 ns	18 ns	25 ns
Power	398.4 mW	412.2 mW	401.1 mW
Area	$4.41 \times 10^6\ \mu m^2$	$3.87 \times 10^6\ \mu m^2$	$3.15 \times 10^6\ \mu m^2$

- CMU library (0.35 µm technology)
- Power measured with clock frequency : 25ns

DCT (Background)

Using the Symmetry of the DCT coefficient matrix, the matrix multiplication is simplified.

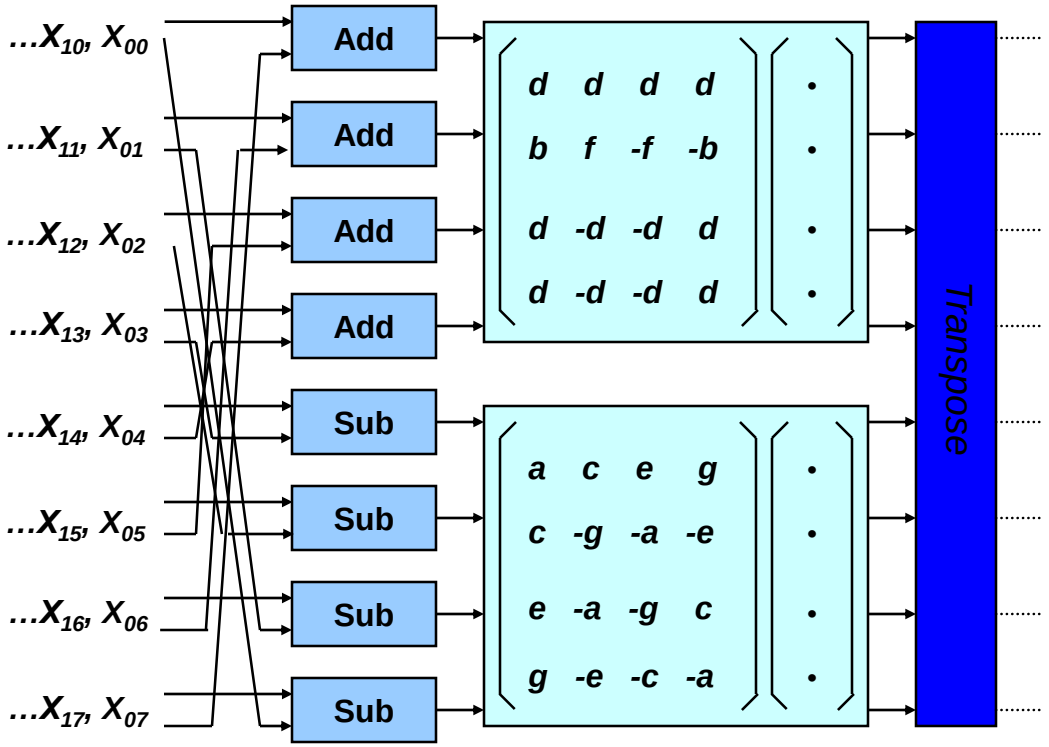
$$Z = Tx^t \ , \ X = TZ^t$$

Even DCT

$$\begin{bmatrix} z_0 \\ z_2 \\ z_4 \\ z_6 \end{bmatrix} = \begin{bmatrix} d & d & d & d \\ b & f & -f & -b \\ d & -d & -d & d \\ f & -b & b & f \end{bmatrix} \begin{bmatrix} x_0 + x_7 \\ x_1 + x_6 \\ x_2 + x_5 \\ x_3 + x_4 \end{bmatrix}$$

Odd DCT

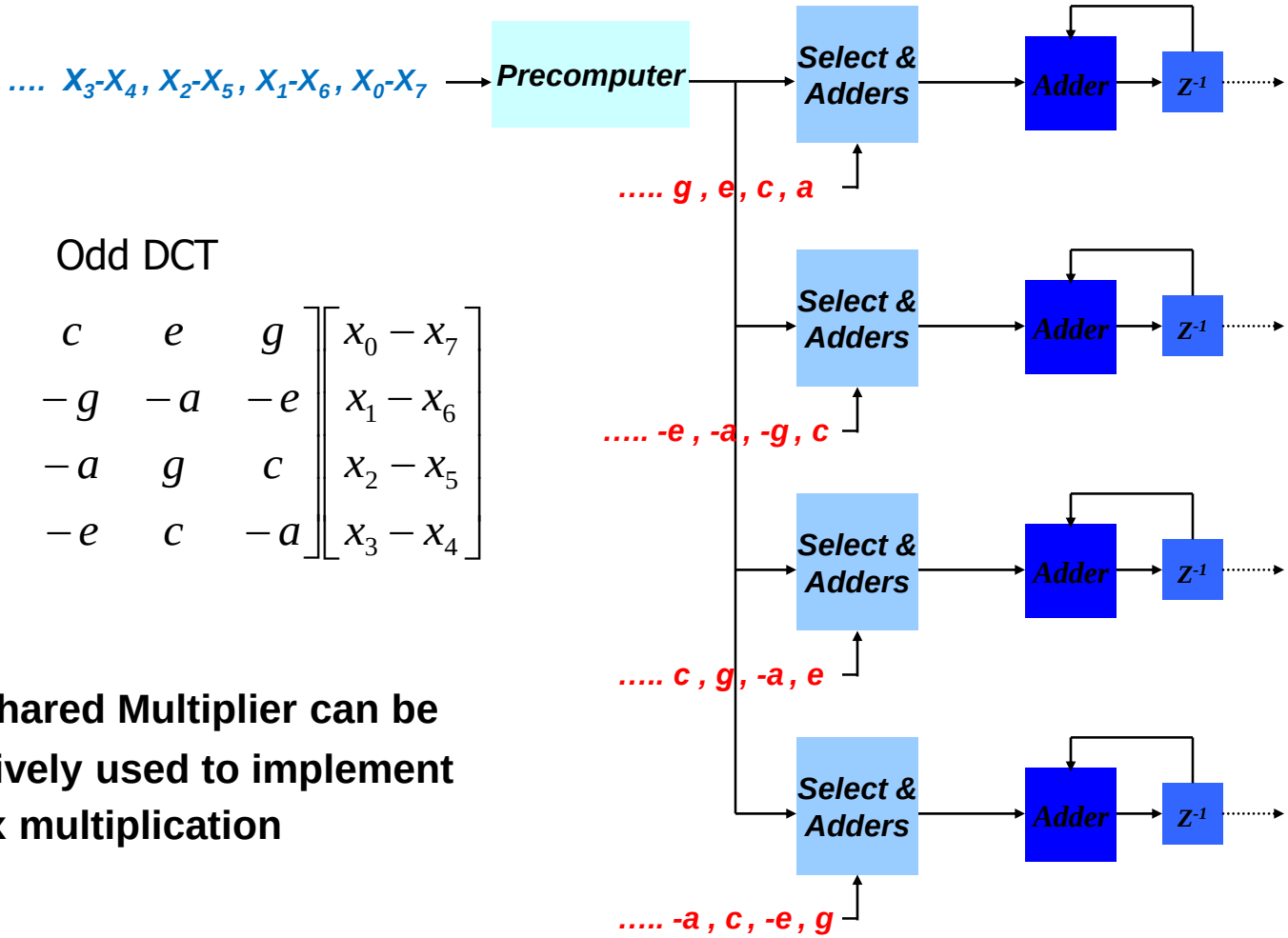
$$\begin{bmatrix} z_1 \\ z_3 \\ z_3 \\ z_7 \end{bmatrix} = \begin{bmatrix} a & c & e & g \\ c & -g & -a & -e \\ e & -a & g & c \\ g & -e & c & -a \end{bmatrix} \begin{bmatrix} x_0 - x_7 \\ x_1 - x_6 \\ x_2 - x_5 \\ x_3 - x_4 \end{bmatrix}$$



Prof. Kaushik Roy

@ Purdue Univ.

DCT using Shared Multiplier



Odd DCT

$$\begin{bmatrix} z_1 \\ z_3 \\ z_3 \\ z_7 \end{bmatrix} = \begin{bmatrix} a & c & e & g \\ c & -g & -a & -e \\ e & -a & g & c \\ g & -e & c & -a \end{bmatrix} \begin{bmatrix} x_0 - x_7 \\ x_1 - x_6 \\ x_2 - x_5 \\ x_3 - x_4 \end{bmatrix}$$

- **The Shared Multiplier can be effectively used to implement matrix multiplication**

Approximation: Modification of DCT Coefficients

The number of *alphabets* can be reduced by modifying the coefficients in DCT matrix.

Only *1x & 3x* are required for the Precomputer bank.

Performance and Power improvement in Precomputer bank and Select unit.

8bit DCT Coefficients

Original 8-bit DCT coefficient			
Coefficient	Value	Binary code	Pre-computer bank Needed
a	0.49	0011 1111	3x, 15x
b	0.46	0011 1011	3x, 11x
c	0.42	0011 0101	3x, 5x
d	0.35	0010 1101	1x, 13x
e	0.28	0010 0100	1x
f	0.19	0001 1000	1x
g	0.10	0000 1100	3x

Modified 8-bit DCT coefficient			
Coefficient	Value	Binary code	Pre-computer bank Needed
a'	0.50	0100 0000	1x
b'	0.47	0011 1100	3x
c'	0.41	0011 0100	1x, 3x
d'	0.34	0010 1100	1x, 3x
e'	0.28	0010 0100	1x
f'	0.19	0001 1000	1x
g'	0.12	0000 1100	3x

Prof. Kaushik Roy

@ Purdue Univ.

DCT using Shared Multiplier

$$X_{kl} = \frac{c(k)c(l)}{4}\sum_{i=0}^{7}\sum_{j=0}^{7} x_{ij}\cos\left(\frac{(2i+1)k\pi}{16}\right)\cos\left(\frac{(2j+1)l\pi}{16}\right)$$

$$Z = Tx^t \ , \ X = TZ^t$$

Even DCT

$$\begin{bmatrix} z_0 \\ z_2 \\ z_4 \\ z_6 \end{bmatrix} = \begin{bmatrix} d & d & d & d \\ b & f & -f & -b \\ d & -d & -d & d \\ f & -b & b & f \end{bmatrix} \begin{bmatrix} x_0 + x_7 \\ x_1 + x_6 \\ x_2 + x_5 \\ x_3 + x_4 \end{bmatrix}$$

Odd DCT

$$\begin{bmatrix} z_1 \\ z_3 \\ z_3 \\ z_7 \end{bmatrix} = \begin{bmatrix} a & c & e & g \\ c & -g & -a & -e \\ e & -a & g & c \\ g & -e & c & -a \end{bmatrix} \begin{bmatrix} x_0 - x_7 \\ x_1 - x_6 \\ x_2 - x_5 \\ x_3 - x_4 \end{bmatrix}$$

8bit DCT Coefficients

Original 8-bit DCT coefficient			
Coefficient	Value	Binary code	Pre-computer bank Needed
a	0.49	0011 1111	3x, 15x
b	0.46	0011 1011	3x, 11x
c	0.42	0011 0101	3x, 5x
d	0.35	0010 1101	1x, 13x
e	0.28	0010 0100	1x
f	0.19	0001 1000	1x
g	0.10	0000 1100	3x

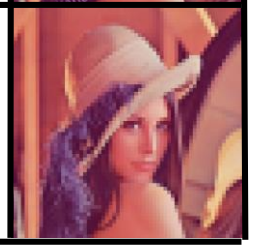
Modified 8-bit DCT coefficient			
Coefficient	Value	Binary code	Pre-computer bank Needed
a'	0.50	0100 0000	1x
b'	0.47	0011 1100	3x
c'	0.41	0011 0100	1x, 3x
d'	0.34	0010 1100	1x, 3x
e'	0.28	0010 0100	1x
f'	0.19	0001 1000	1x
g'	0.12	0000 1100	3x

- Only *1x* & *3x* are required in the Modified 8-bit DCT Coefficient

Prof. Kaushik Roy

@ Purdue Univ.

Effect of V_{dd} Scaling

	Conventional WTM DCT	CSHM DCT (2 alphabet)	Proposed DCT
1.0 V			
0.9 V	FAILS	FAILS	
0.8 V	FAILS	FAILS	

Different Architectures at Nominal Voltage

1.0V	CSHM DCT (2 alphabets)	DCT with WTM	Proposed DCT
Power (mW)	25.1	29.8	26
Delay (ns)	3.2	3.64	3.57
Area (um^2)	80490	108738	90337
PSNR (dB)	21.97	33.23	33.22

Proposed Architecture at Reduced Voltage

	Proposed DCT Vdd=0.9V	Proposed DCT Vdd=0.8V
Power (mW)	17.53(41.2%)	11.09(62.8%)
PSNR (dB)	29	23.41

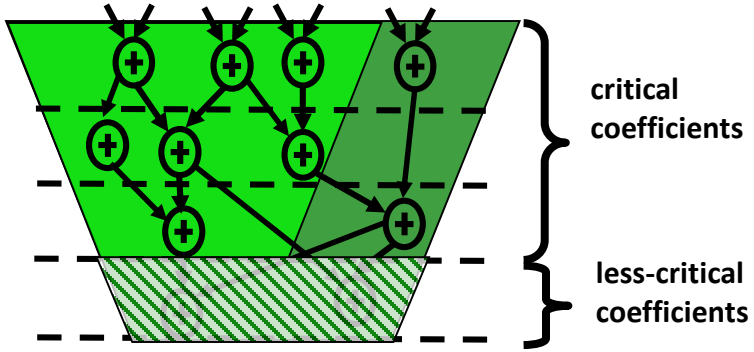
- Graceful degradation of proposed DCT architecture under Vdd scaling (Vdd can be scaled to 0.75V)
- Conventional architectures fails

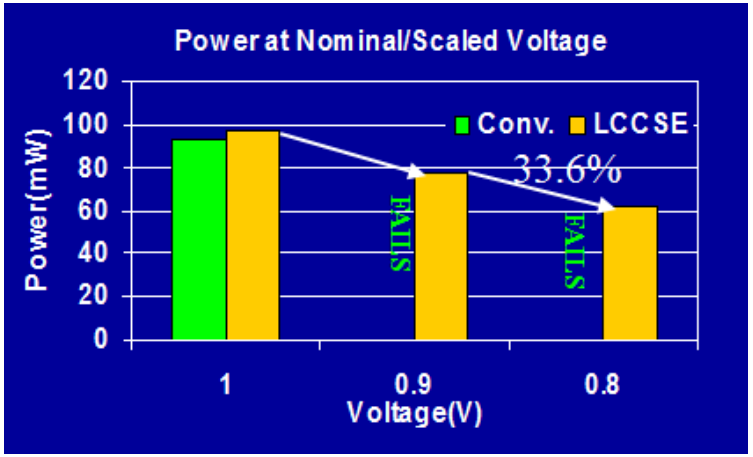
Prof. Kaushik Roy

@ Purdue Univ.

Other DSP Systems

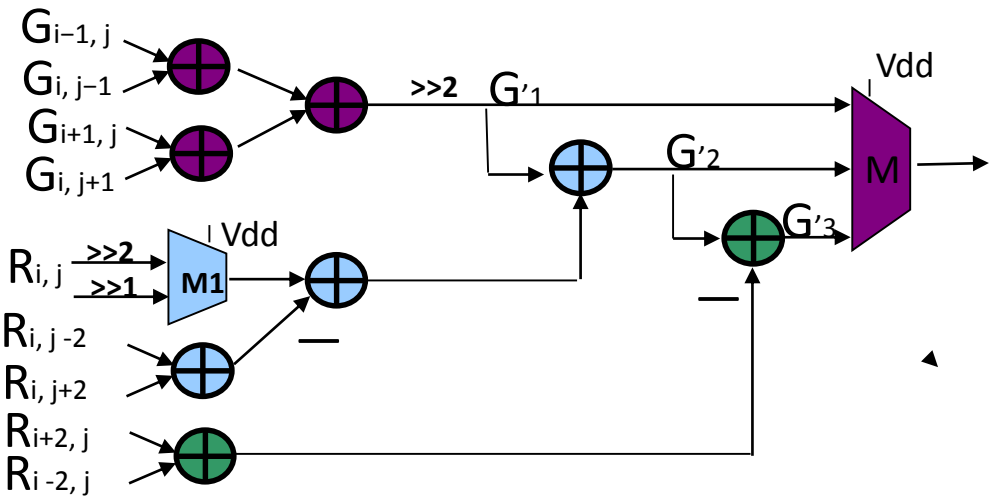
2. Finite Impulse Response (FIR)





3. Color Interpolation

- Bilinear component is critical and gradient component is less-critical
- Design architecture such that failures can only occur in gradient term

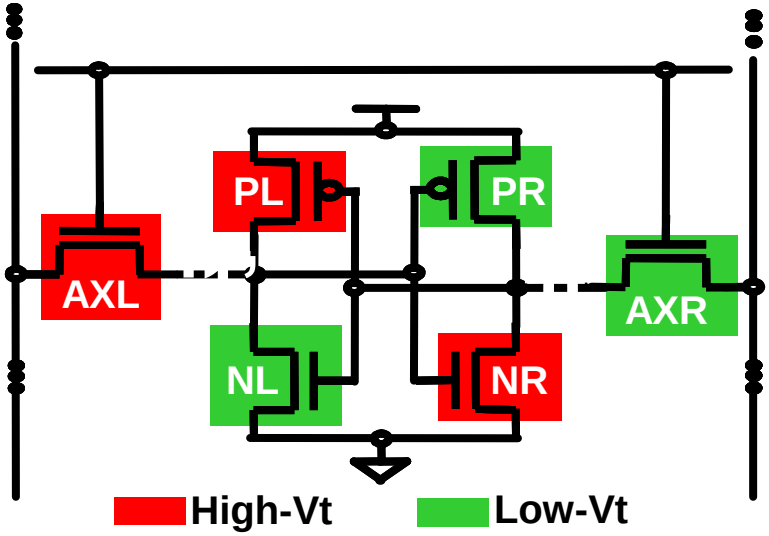


	Conv	Proposed
1.0 V		
0.9 V	FAILS	
0.8 V	FAILS	

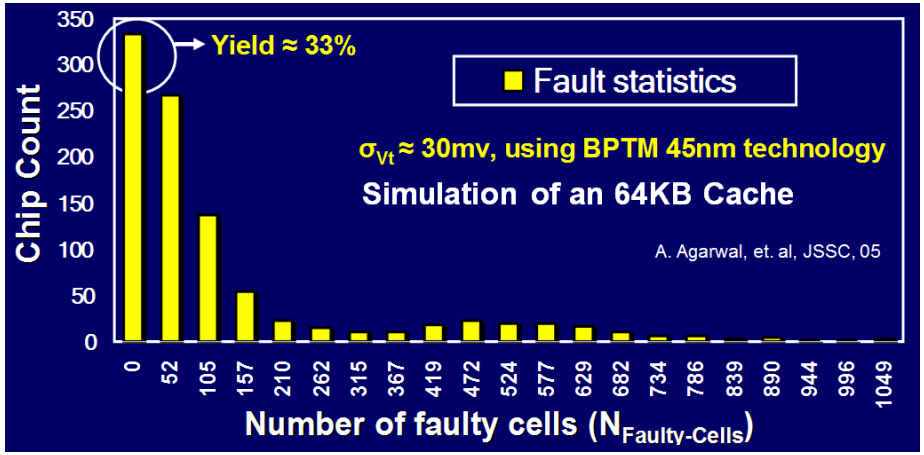
Approximate Memories

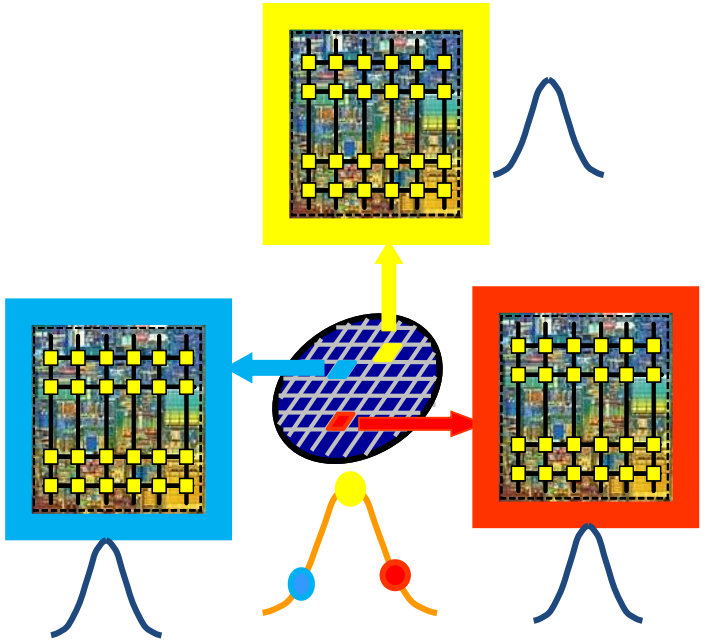
- Failures under parameter variations
- Energy Vs. Quality Trade-off

Low Voltage SRAM Operation: Issues



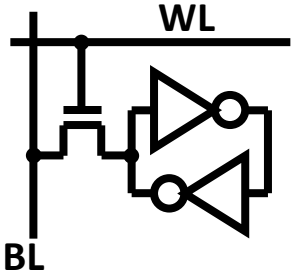
- **Parametric failures**
  - -Read, Write, Access, Hold





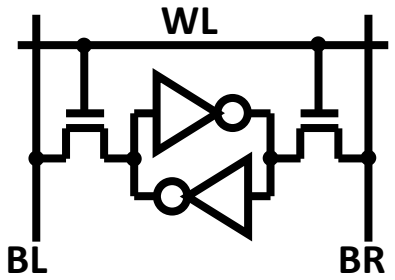
Parametric failures can degrade SRAM yield

Other SRAM Bit-Cells: Separating Read/Write

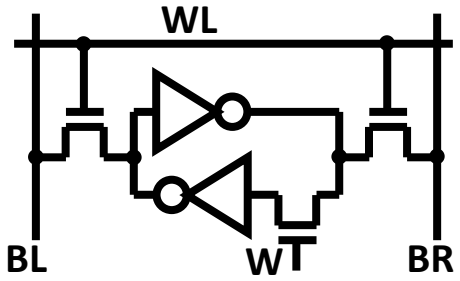


5T

I. Calson et. al., ESSCIRC05

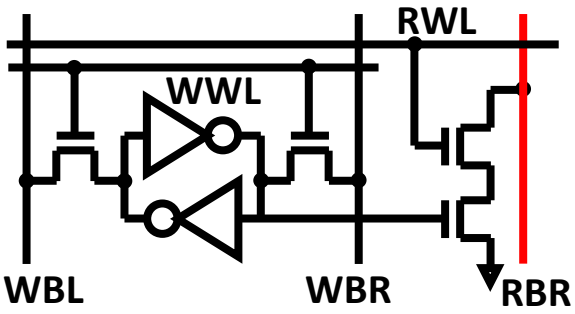


6T



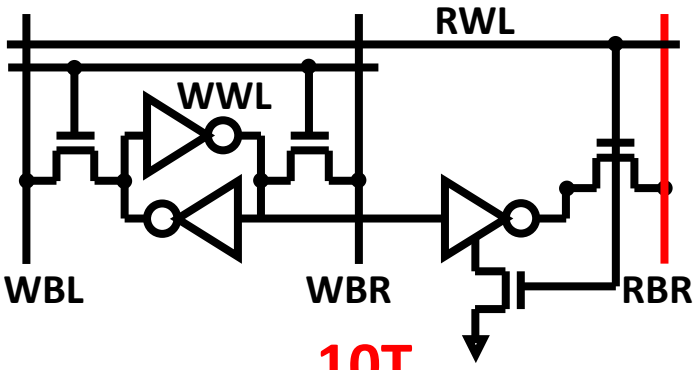
7T

K. Takeda et. al., ISSCC05



8T (Register File)

L. Chang et. al., VLSI Tech.'05

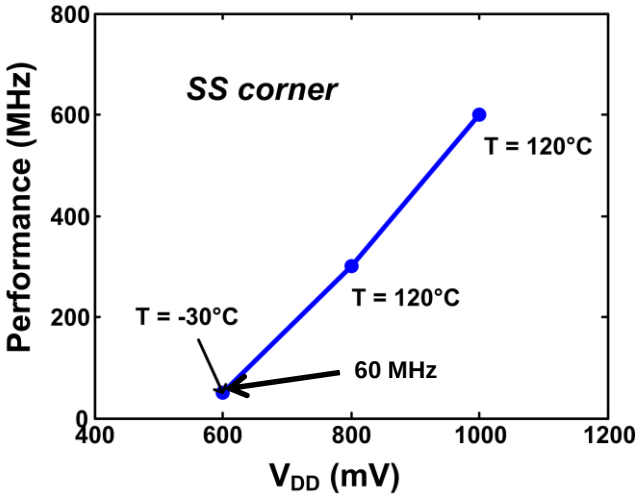


10T

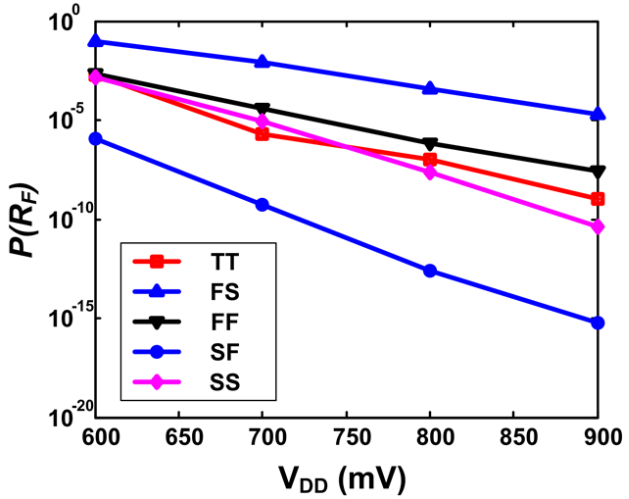
B. Calhoun et. al., ISSCC06

- 5T, 8T, 10T cells - single ended.
- 8T/10T decoupled read and write operation.
- No in-built process variation tolerance

Low Performance Case (CIF / QCIF)



Performance Simulation Results at the worst PT corner (65nm CMOS)

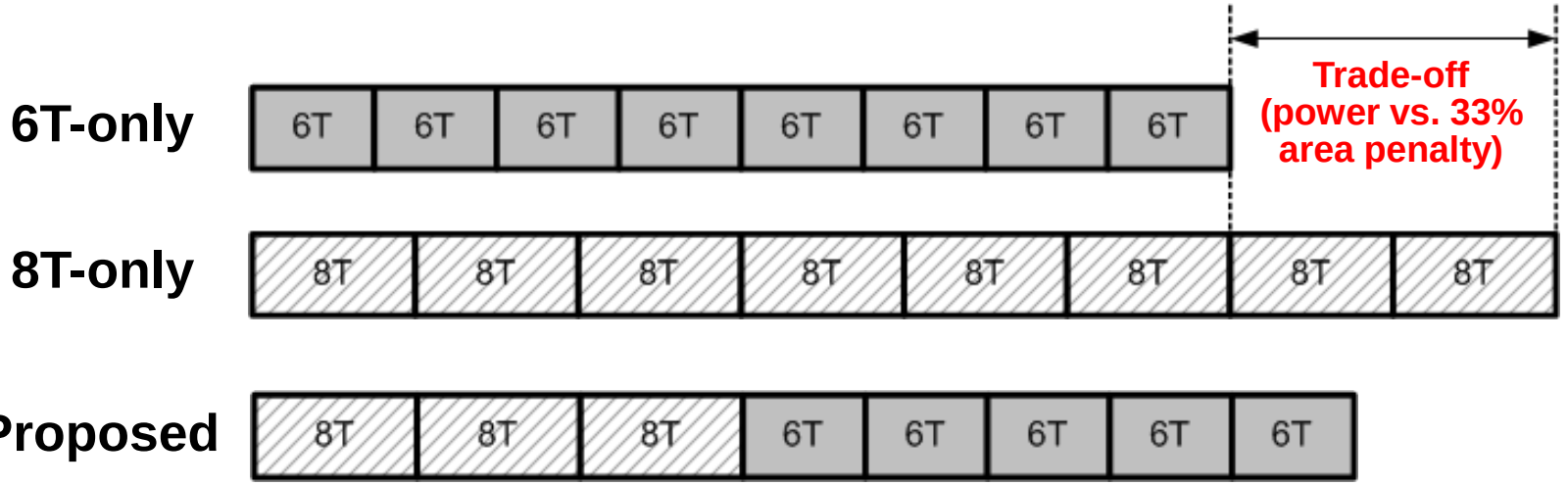


Read Failure Prob. of a 6T bit-cell (@ T= 25ºC, 65nm CMOS)

- CIF/QCIF display format operates at low frequency (less than 10Mhz)
  - Easily satisfied at 65nm CMOS (even at 600mV V_{DD})
- Memory stability issue still impedes V_{DD} scaling
  - Read stability is one of the major obstacles

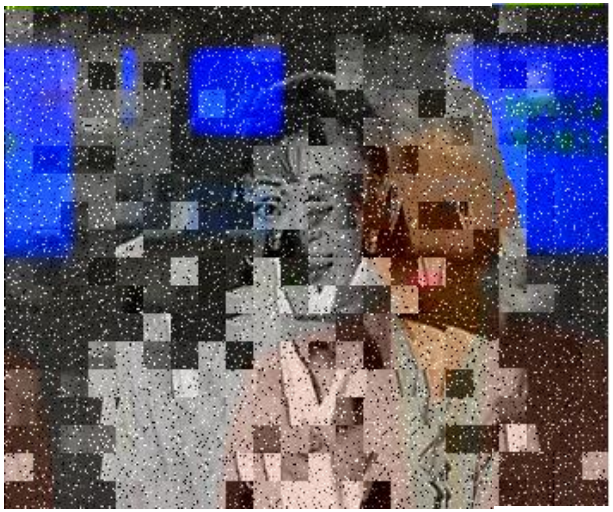
Hyndrid-Memory for Lower-Vmin

Eight luma bits of an image pixel



- 6T: Small Area but, Large Power
- 8T: Large Area (33% penalty) but, Small Power (more V_{DD} scaling)
- Our innovation is mixture of 6T and 8T bit-cells
  - Critical MSB bits: 8T, Non-critical LSB bits: 6T
  - Small area penalty (11.5%) and aggressive V_{DD} scaling

Video Image Simulation



Fully 6T (FS) @ 600mV

PSNR = 12.83dB

Fully 6T (FS) @ 800mV

PSNR = 23.38 dB

Hybrid SRAM (FS)

@ 600mV PSNR = 22.80 dB

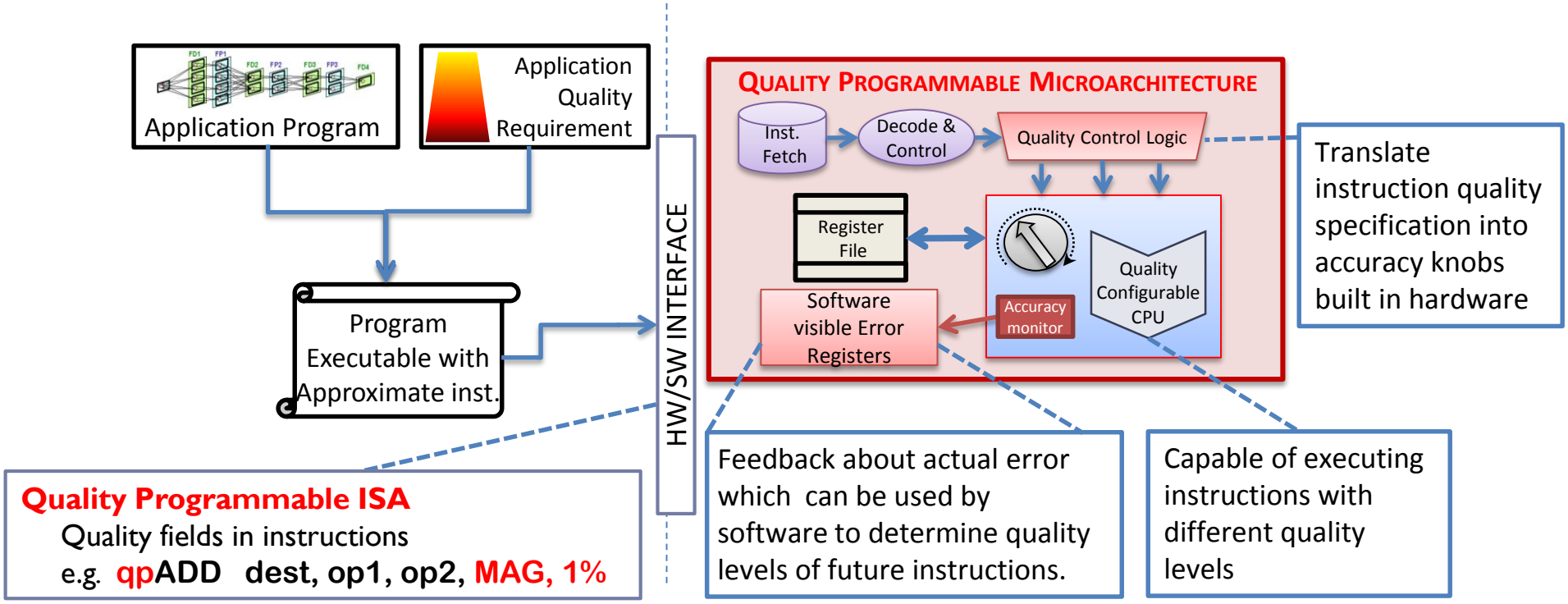
Hybrid SRAM (SF)

@ 600mV PSNR = 23.04 dB

- Assumption: MV is stored in fully 8T (0.7 ~ 0.8% of luma bits)
- Overall area penalty is 11.64%
- Despite 200mV over-scaling, output image quality is comparable (0.58db degradation)

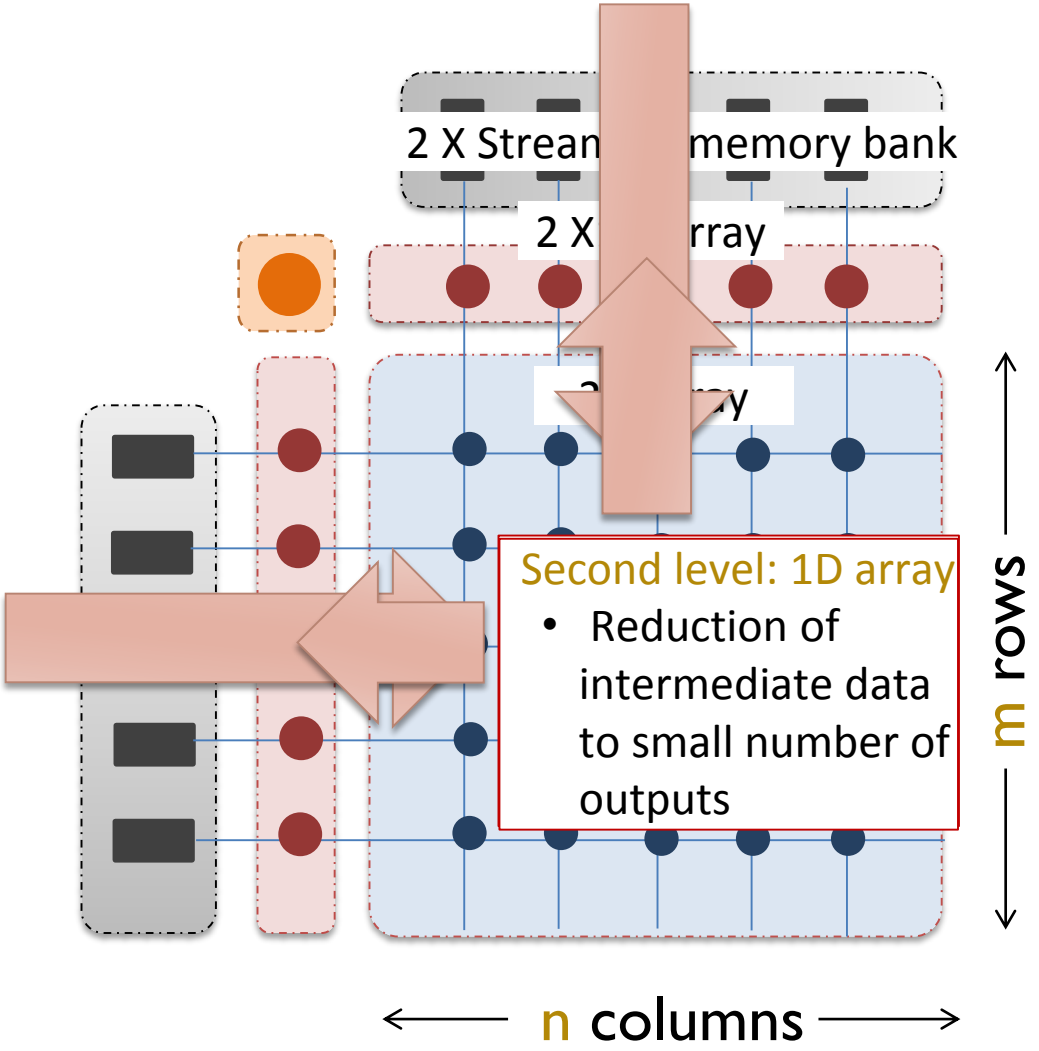
Quality Programmable Processors

- Broader adoption of approximate computing requires programmable platforms!
- Software expresses accuracy bounds/expectations at the outputs of individual instruction
- Hardware guarantees that instruction accuracy bounds are met

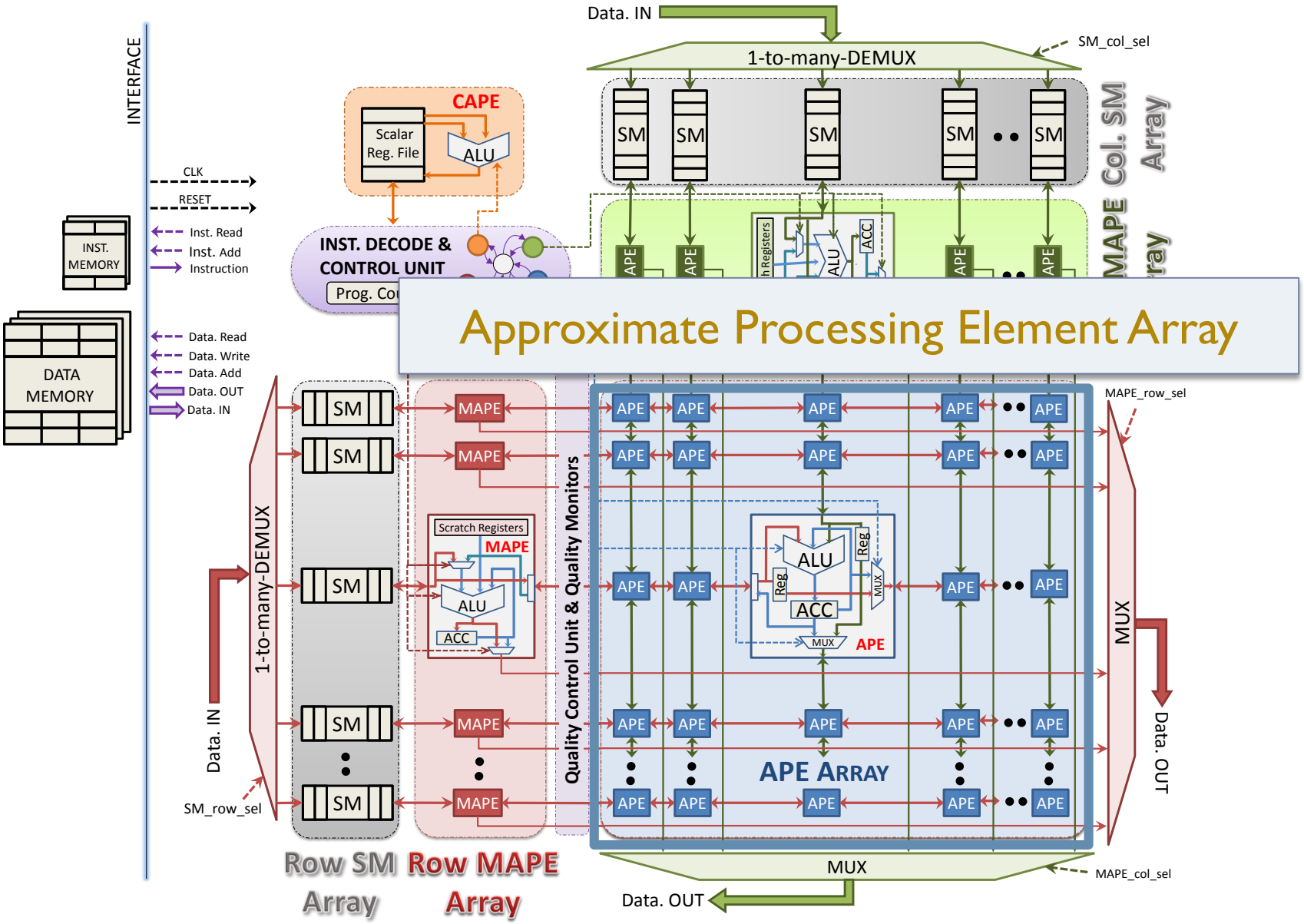


QP-Vec 1D/2D Vector Processor

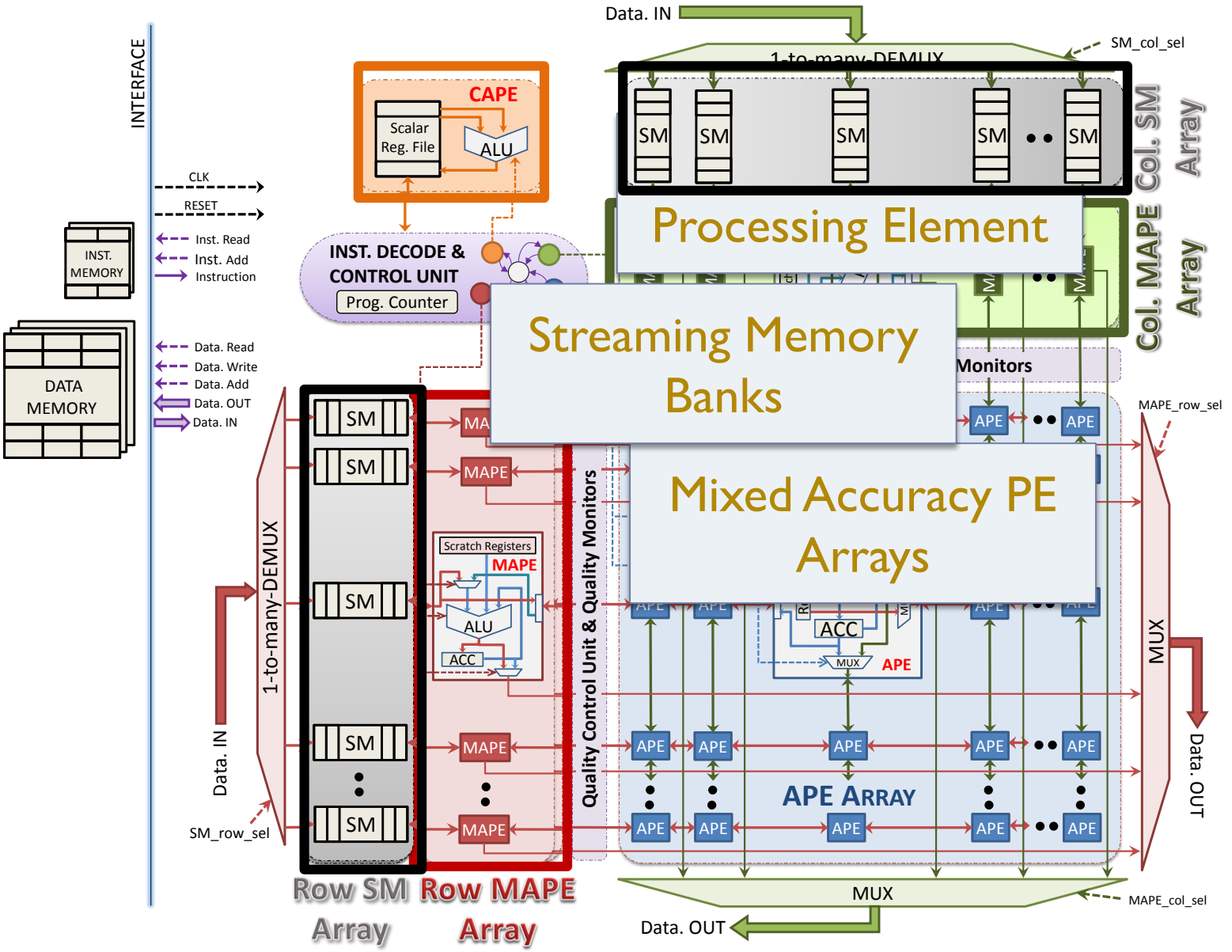
- 3-tier processing element hierarchy
  - 2D array PEs
  - 2 sets of 1D array PEs
  - One scalar PE
- 2 streaming memory banks along the array borders
- Computation pattern: 2-level vector reduction



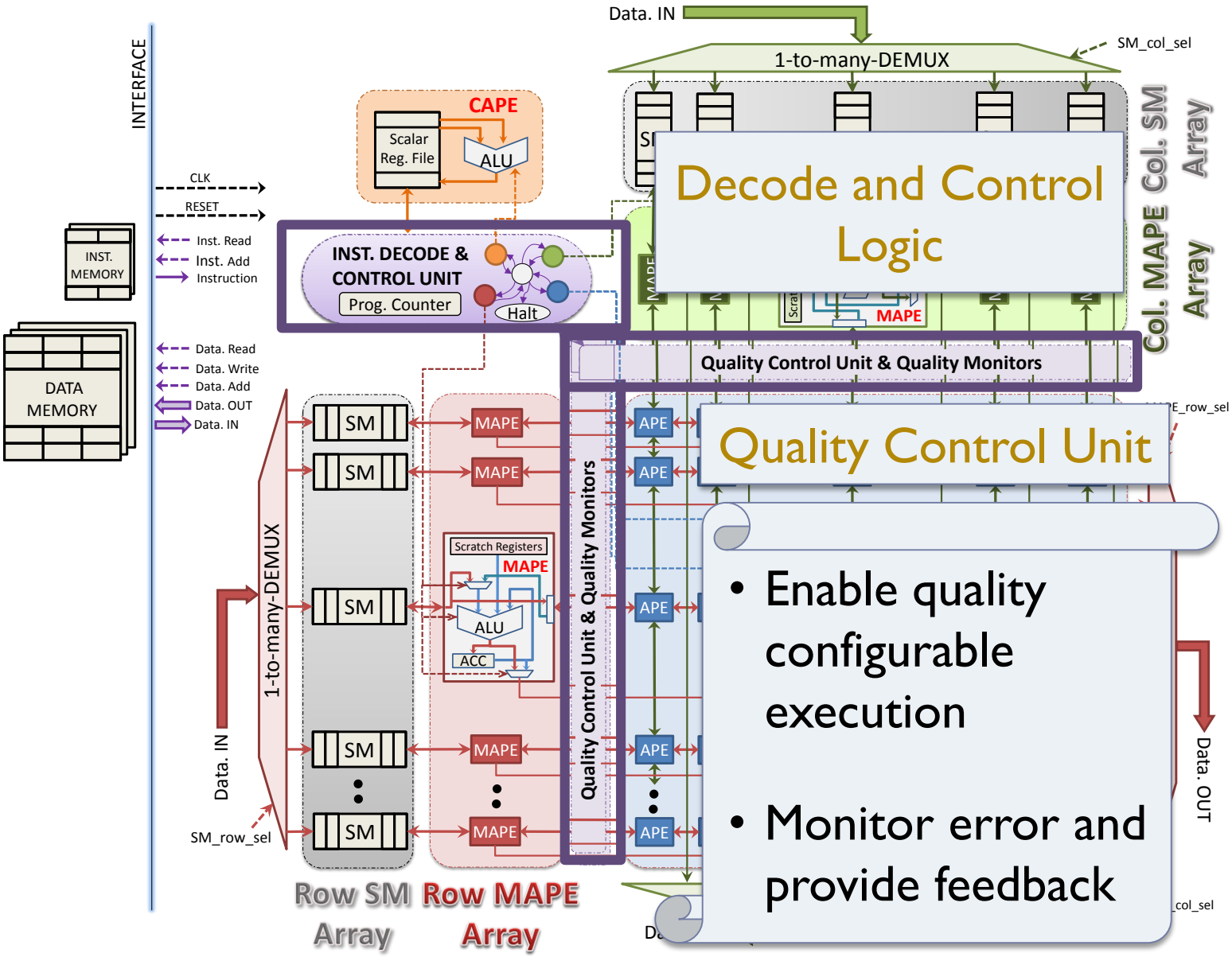
QP-VEC 1D/2D VECTOR PROCESSOR



QP-VEC 1D/2D VECTOR PROCESSOR



QP-Vec 1D/2D Vector Processor



QUORA: INSTRUCTION SET ARCHITECTURE

- 47 Instructions – 9 APE, 22 MAPE, 13 CAPE, 3 SM

Inst. Type	Instruction
Scalar Instructions	LDRI Rd, value
	ADDR Rd,Rs1,Rs2
	BEZ Rs, Rel. address
	HALT
Streaming Memory instructions	LDSM R_length, stride, burst, R_st_add
2D Array Instructions	**qpMAC** R_length, R_row_enb, R_col_enb, ***R_q_type, R_q_amt***
	qpMOD2 R_length, R_row_enb, R_col_enb, ***R_q_type, R_q_amt***
	STR <r/c>, R_stride, R_burst, R_st_add, R_row_enb, R_col_enb

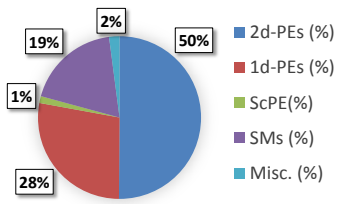
Inst. Type	Instruction
1D Array Reduction Instructions	**qpACC** <r/c>, R_row_enb, R_col_enb, ***R_q_type, R_q_amt***
	qpMIN <r/c>, R_row_enb, R_col_enb, ***R_q_type, R_q_amt***
1D Array Streaming Instructions	SEQ R_length, SReg, R_row_enb, R_col_enb
1D Array Self-Operand Instructions	MVASR <r/c>, R_<r/c>_enb, SReg
	qpADDX <r/c>, R_<r/c>_enb, Sreg, ***R_q_type, R_q_amt***
	qpMUL <r/c>, R_<r/c>_enb, Sreg, **R_q_type, R_q_amt**
	STMCG <r/c>, R_<r/c>_enb, SReg

QUORA: EVALUATION METHODOLOGY

- RTL implementation of QUORA (289 cores) synthesized to IBM 45nm tech. node
  - Design flow: *Synopsys Design Compiler, ModelSim, Synopsys Power Compiler*

Micro-architectural Parameters	Value
Array Dimensions	16 X 16
No. of PEs (2d-PEs + 1d-PEs+ ScPE)	289 (256 + 32 + 1)
Size of Register File – ScPE / 1d-PE	32 / 8
No. of SM elements	32
Depth of SM elements	64
Operating Frequency	250 MHz

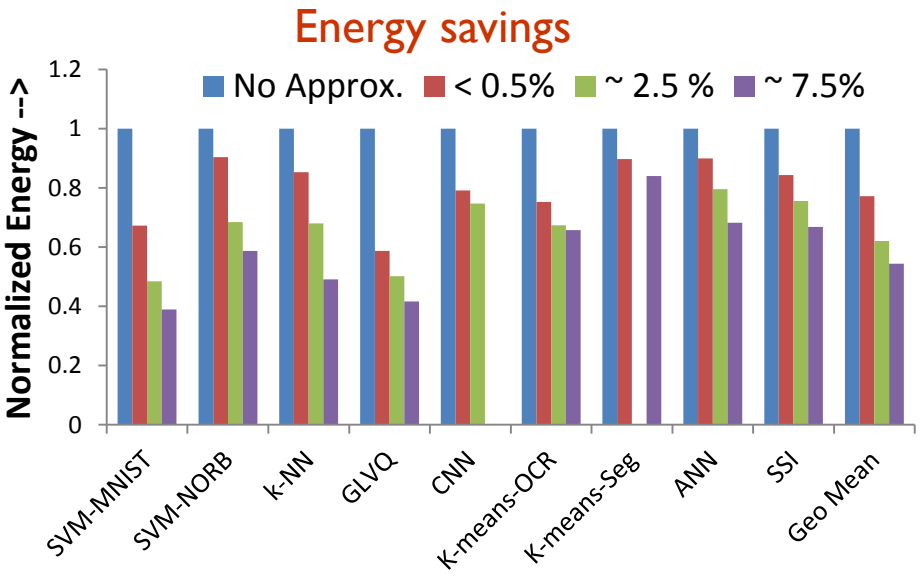
Circuit Parameters	Value
Technology Library	IBM 45nm
Area	2.6 mm^2
Power	367.8 mW
Gate Count	502042

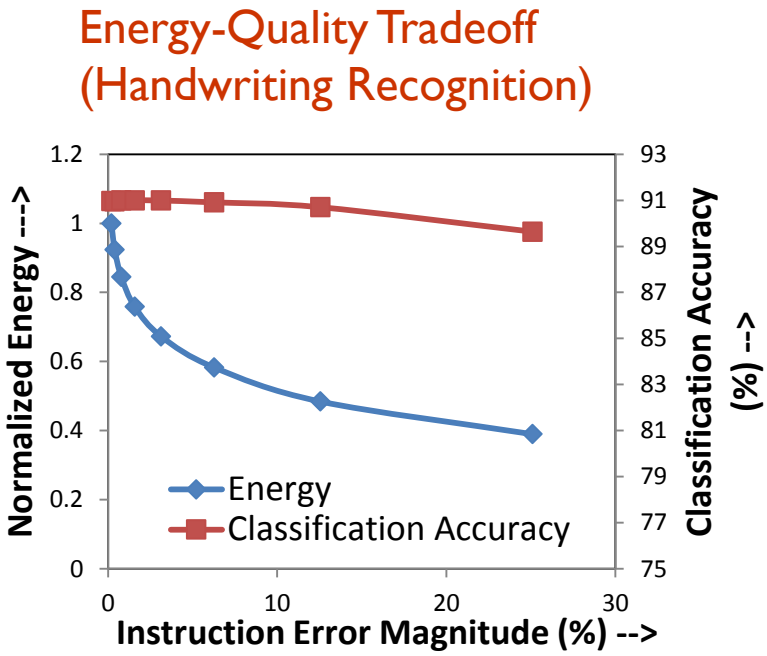


Benchmarks:

Applications	Algorithm	Dataset
Handwritten Digit Recognition (SVM-MNIST)	Support Vector Machines	MNIST
Object Recognition (SVM-NORB)	Support Vector Machines	NORB
Digit Classification (CNN)	Convolutional Neural Networks	MNIST
Eye Detection (GLVQ)	Generalized Learning vector Quantization	Image set from NEC labs.
Optical Character Recognition (k-NN)	K-nearest Neighbors	OCR digits
Image Segmentation (K-Means-Seg)	K-means Clustering	Berkeley dataset
Optical Character Clustering (K-Means-OCR)	K-means Clustering	OCR digits

QUORA: RESULTS





Approximate Computing @ Purdue

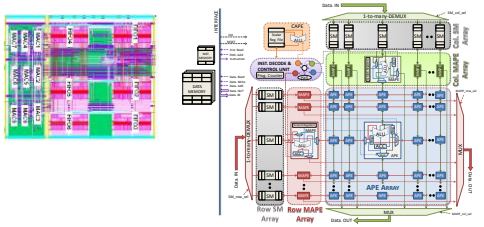
Approximate Computing in Software

- Best-effort parallel computing (DAC 2010)
- Dependency relaxation (IPDPS 2010)
- Analysis and characterization of inherent application resilience (DAC 2013)
- Approximate Neural Networks (ISLPED 2014)

- Improve parallel scalability / skip computations
- Exploit domain specific properties to reason about computations

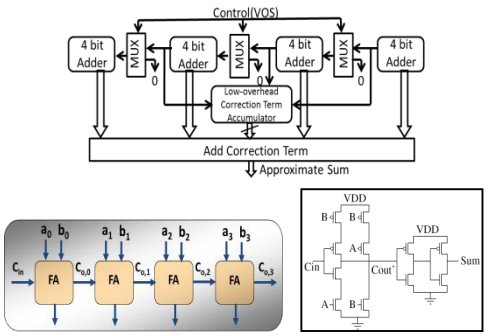
Approximate Architecture & System Design

- Scalable Effort Hardware (DAC 2010, DAC 2011, CICC 2013)
- Significance Driven Computation: MPEG, H.264 (DAC2009, ISLPED 2009)
- QUORA: Quality Programmable vector processor (MICRO 2013)



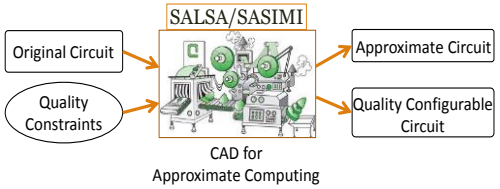
Approximate Circuit Design

- Voltage Scalable meta-functions (DATE 2011)
- Energy-quality tradeoff in DCT (DATE 2006)
- Approximate memory design (DAC 2009)
- IMPACT: Imprecise Adders for low power approximate computing (ISLPED 2011)

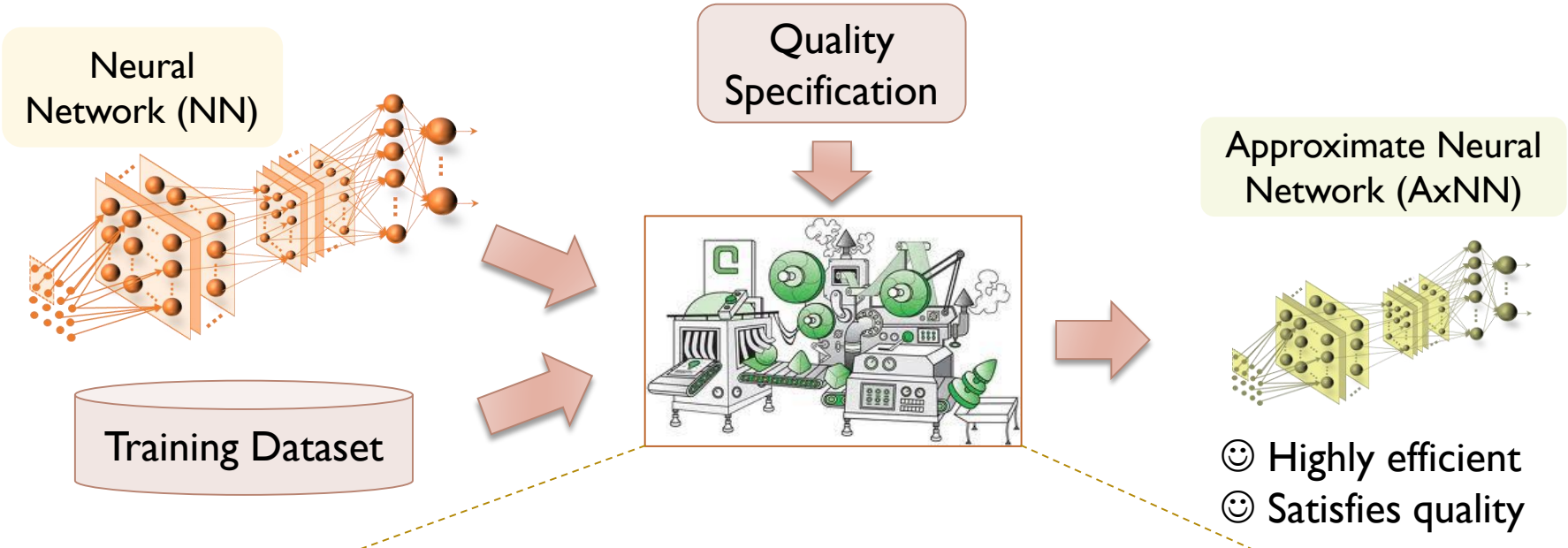


Design Automation for Approximate Computing

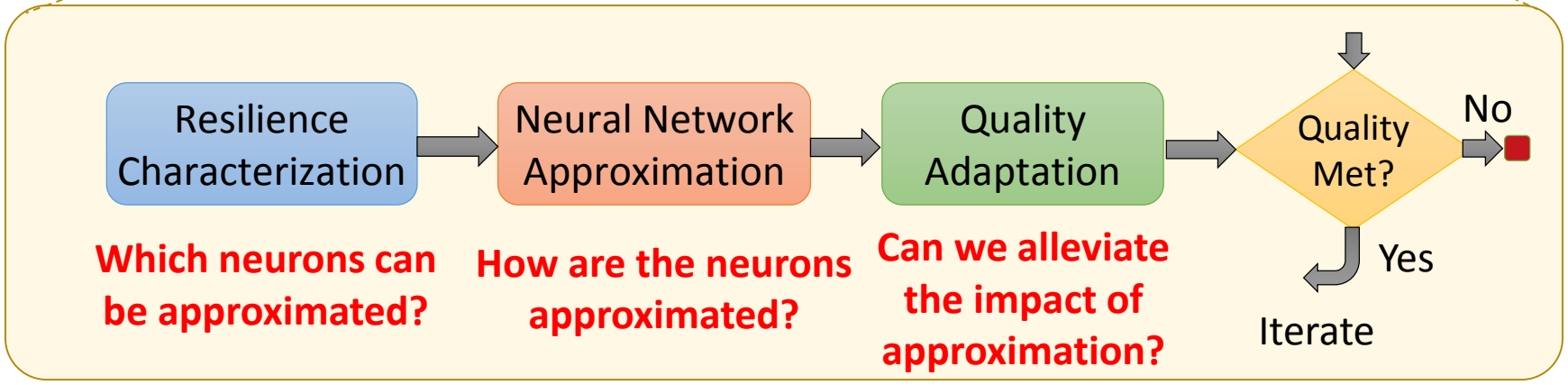
- SALSA: Systematic Logic Synthesis for Approximate Circuits (DAC 2012)
- Substitute-and-Simplify: Design of quality configurable circuits (DATE 2013)
- MACACO: Modeling and Verification of Circuits for Approximate Computing (ICCAD 2011)



AxNN: Approximate Neuromorphic Systems



AxNN Transformation

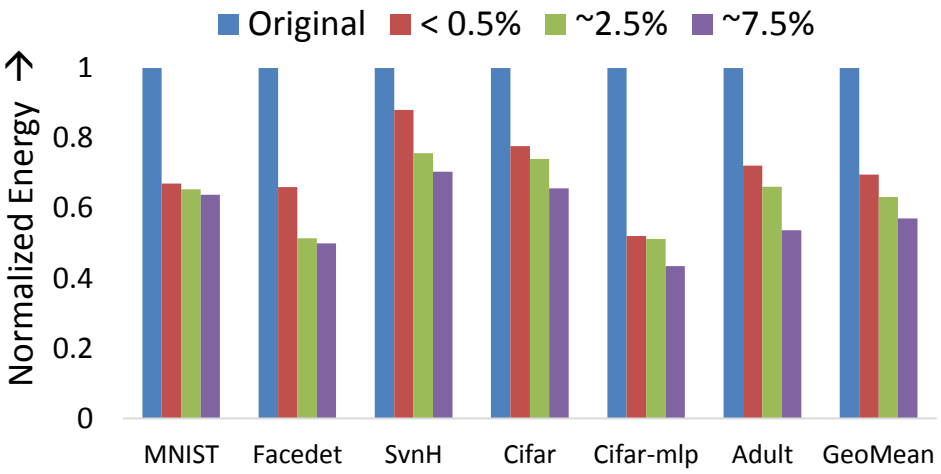


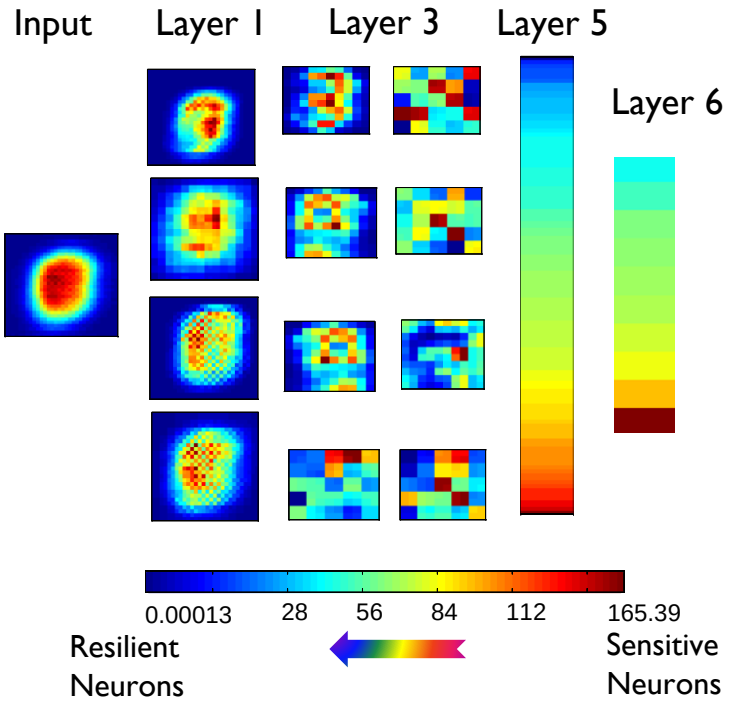
AxNN: Results

Applications	Layers	Neurons	Parameters
House Number Recognition	8	47818	847434
Object Classification	6	38282	846890
Digit Recognition	6	8010	51046
Face Detection	4	13362	25634
Object Recognition MLP	2	1034	3157002
Census Data Analysis	2	12	172

Neuron Resilience: Insights

Energy savings





Takeaways

- Approximate computing taps into intrinsic resilience of applications
  - Computing efficiently with good-enough results – large improvement in energy consumption.
- Approximate computing techniques at various layers of computing stack
  - Circuits, Architecture, Software
- Intrinsic resilience can also be leveraged for
  - Designing with error-prone devices (unequal error protection)
  - New computing models for Post-CMOS devices