Approximate Computing: An Emerging Paradigm For Energy-Efficient Design

Jie Han and Michael Orshansky

University of Alberta and University of Texas at Austin

Embedded Tutorial, ETS'13, Avignon, France

Please use with attribution to the authors. © 2013 Han and Orshansky

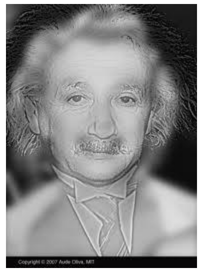
Outline

- MOTIVATION
- OVERVIEW OF ERROR-RESILIENT PARADIGMS
- APPROXIMATE ARITHMETIC CIRCUITS
  - Approximate adders
  - Approximate multipliers
  - Metrics for approximate computing
  - Quality-energy optimal designs
  - Approximate logic synthesis
- ALGORITHM-LEVEL APPROXIMATE COMPUTING TECHNIQUES

 Please use with attribution to the authors. © 2013 Han and Orshansky

Motivation

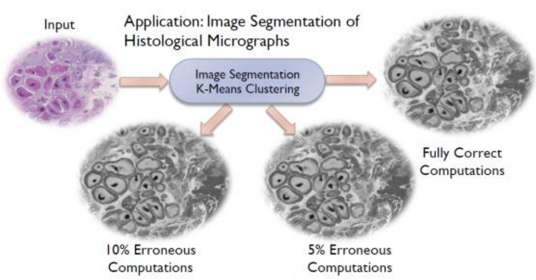
- Energy-efficiency is of paramount concern in digital system design
- Computing becomes increasingly heavy with media processing (audio, video, graphics, and image), recognition, and data mining
- A common characteristic: a perfect result is not necessary and an approximate or less-than-optimal result is sufficient
  - Signal processing: image, video, speech
    - Human perception is not sensitive to high frequency changes
    - Natural noise floor due to quantization noise
  - Search, machine learning, data mining
  - Optimization



© 2013 Han and Orshansky

Sources of Imprecision Tolerance

- Perceptual limitations: these are determined by the ability of the human brain to 'fill in' missing information and filter out high-frequency patterns
- Redundant input data: this redundancy means that an algorithm can be lossy and still be adequate
- Noisy inputs



Source: A. Raghunathan, Dagstuhl Seminar 2012

© 2013 Han and Orshansky

Error-Resilient Paradigms

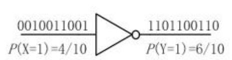
- How can we exploit system's ability for imprecision-tolerance for energy reduction?
- ***Approximate Computing***
  - *Does not involve* assumptions on the stochastic nature of any underlying processes *implementing* the system. Utilizes *statistical* properties of data and algorithms to trade quality for energy reduction.
- ***Stochastic Computing***
  - Real numbers are represented by random binary bit streams that are usually implemented in *series* (or *parallel*) and in *time (*or *space)*. Information is carried on the statistics of the binary streams.
- ***Probabilistic Computing***
  - Exploits intrinsic probabilistic behavior of the underlying circuit fabric, most explicitly, of the stochastic behavior of a binary switch under the influence of thermal noise.

Embedded Tutorial, ETS'13, Avignon, France

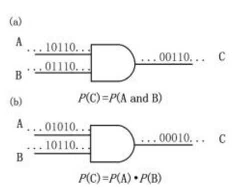
© 2013 Han and Orshansky

Stochastic Computing: Basics

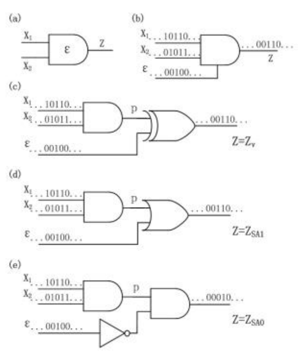
- In stochastic computing, real numbers in [0, 1] are represented by random binary bit streams
- Information is carried in the statistics of the binary streams, e.g., the proportion of 1's



An inverter and stochastic encoding



Stochastic AND logic:
(a) the general model;
(b) the special case of multiplication with independent inputs.



Stochastic logic models: (a) An unreliable AND gate; (b) A general stochastic implementation; (c) for soft errors; (d) for stuck-at-1 fault; (e) for stuck-at-0 fault [Han14].

Embedded Tutorial, ETS'13, Avignon, France

© 2013 Han and Orshansky

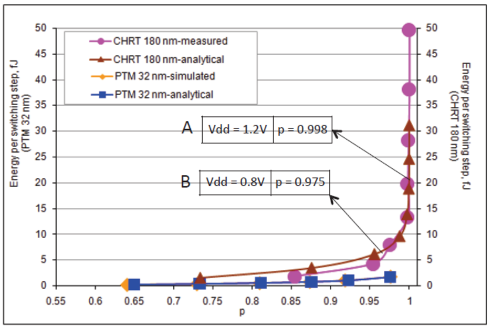
Stochastic Computing: Milestones

- Probabilistic logic and multiplexing [vonNeumann52]
  - Renaissance in 2000s for nanoelectronics [Han05]
- Stochastic computing systems [Poppelbaump67, Gaines69]
- Stochastic neural computation [Brown01]
- Stochastic LDPC decoders [Gaudet03, Tehrani08]
- General-purpose computing [Qian11, Li12]
- Reliability analysis [Han14, Aliee13]
- Spectral transform analysis [Alaghi12]
- Recent review in [Alaghi12]
  - Extensions to error-resilient systems [Shanbhag10, Sartori11, Cho12]

© 2013 Han and Orshansky

Probabilistic Computing

- A proposal for using *physically unreliable* devices
  - Probabilistic switches and probabilistic Boolean logic developed from a thermodynamic perspective
  - Probabilistic CMOS (PCMOS) family of circuits
  - A recent philosophical introduction in [Palem12]



A tradeoff between switching probability and associated energy: energy-probability relationship of an XOR gate [Palem12]

© 2013 Han and Orshansky

Approximate Computing

- Employs *deterministic* designs that produce *imprecise* results
- Key idea: trade small quality degradation for improved design metrics, esp. energy
- Accurate (optimal) computation is expensive in terms of energy
  - Typical behavior often much better than rare worst-case behavior
  - Performance or Timing
    - E.g. for N-bit ripple carry adders, worst-case carry-length (delay) ~ N
    - Expected carry-length is ~ log N
- This tutorial is focused on how hardware is re-designed for approximate computing applications

© 2013 Han and Orshansky

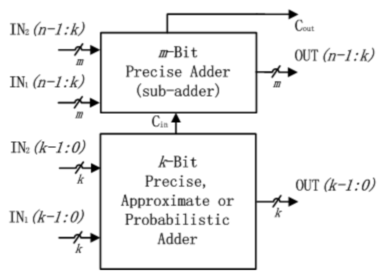
Outline

- MOTIVATION
- OVERVIEW OF ERROR-RESILIENT PARADIGMS
- APPROXIMATE ARITHMETIC CIRCUITS
  - Approximate adders
  - Approximate multipliers
  - Metrics for approximate computing
  - Quality-energy optimal designs
  - Approximate logic synthesis
- ALGORITHM-LEVEL APPROXIMATE COMPUTING TECHNIQUES

© 2013 Han and Orshansky

Approximate *n*-bit Adders

- In an approximate implementation, *n*-bit adders can be divided into two modules: the (accurate) upper part of more significant bits and the (approximate) lower part of less significant bits.
- For each lower bit, a single-bit approximate adder implements a modified, thus inexact, function of the addition.

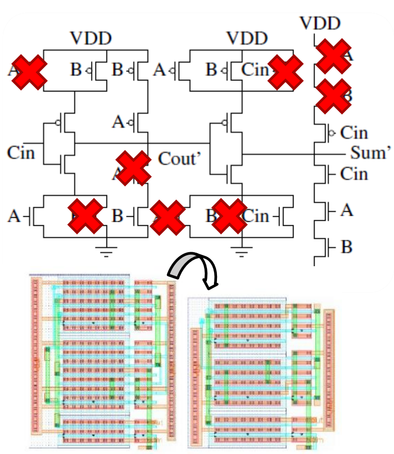


A general architecture for an approximate adder divided into two modules: the accurate MSBs and approximate LSBs.

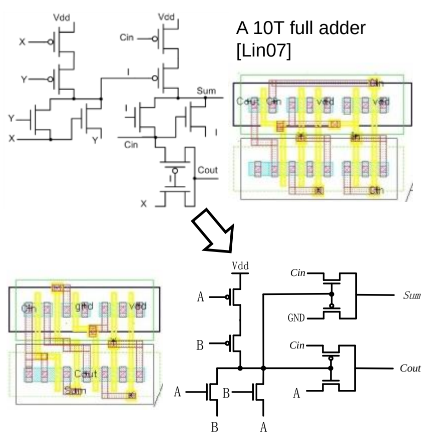
© 2013 Han and Orshansky

Approximate Full Adders

- Approximate Mirror Adders (AMAs) [Gupta13]



- Approximate XOR/XNOR Adders (AXAs) [Yang13]



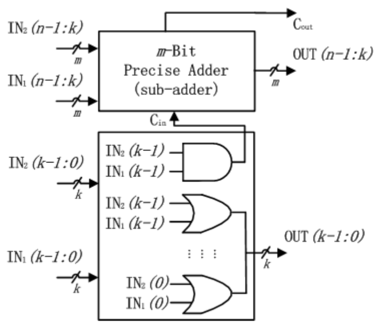
An 8T approximate adder [Yang13]

© 2013 Han and Orshansky

Approximate and Probabilistic Adders

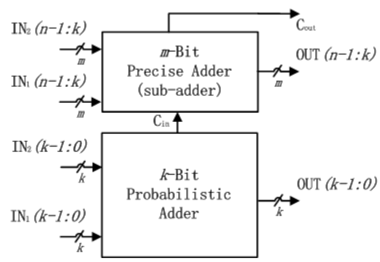
Approximate logic design

- ❑ Lower-part OR adder [Mahdiani10]



Probabilistic adder

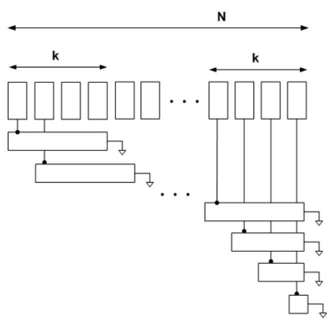
- ❑ PCMOS-based design [Cheemalavagu05]



© 2013 Han and Orshansky

Approximate Speculative Adders (1)

- ❑ The critical path delay of a parallel adder (such as a carry look ahead) is asymptotically proportional to log(*N*) for an *N*-bit adder.
- ❑ Sub-logarithmic delays can however be achieved by the so-called speculative adders [Lu04, Verma08].
- ❑ A speculative adder exploits the fact that the typical carry propagation chain is significantly shorter than the worst-case carry chain by using a limited number of previous input bits to calculate the sum (e.g. look-ahead *k* bits) [Lu04].
- ❑ It can be developed into a reliable variable latency speculative adder (VLSA) with error detection and recovery [Verma08].

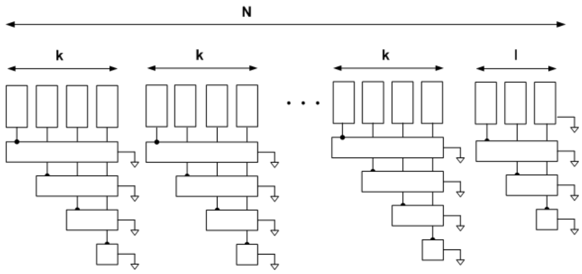


A speculative adder as an almost correct adder (ACA).

© 2013 Han and Orshansky

Approximate Speculative Adders (2)

- An error tolerant adder truncates the carry propagation chain by dividing the adder into several sub-adders (ETAII); its accuracy can be improved by connecting carry chains in a few most significant sub-adders (ETAIIM) [Zhu09].
- An alternating carry select process can be used in the sub-adder chain to enhance the design (ETAIV) [Zhu10].



A general architecture of an error tolerant adder (ETA).

© 2013 Han and Orshansky

Approximate Speculative Adders (3)

- A reliable variable latency carry select adder (VLCSA) employs carry chain truncation and carry select addition as a basis in a speculative adder [Du12].
- An accuracy-configurable adder (ACA) enables an adaptive operation, either approximate or accurate, that is configurable at runtime [Kahng12].
- In a dithering adder, subsequent additions produce opposite-direction errors such that the final result has a smaller overall error variance [Miao12].
- More details discussed later.

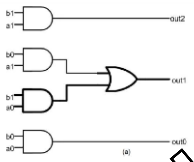
© 2013 Han and Orshansky

Approximate Multipliers (1)

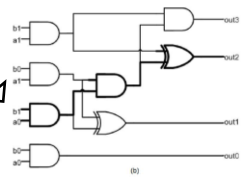
- A multiplier usually consists of three stages: partial product generation, partial product accumulation and a carry propagation adder at the final stage.
  - The use of speculative adders in an approximate multiplier to compute the sum of partial products is not efficient in terms of trading off accuracy for energy and area savings [Lu04, Huang12].
- In [Kulkarni11], inaccurate 2 × 2 multiplier blocks are used to compute approximate partial products that are accumulated using accurate adders.

A_1A_0 \ B_1B_0	00	01	11	10
00	000	000	000	000
01	000	001	011	010
11	000	011	111	110
10	000	010	110	100

Truth table for the approximate 2 × 2 multiplier



(a) Inaccurate 2 × 2 multiplier

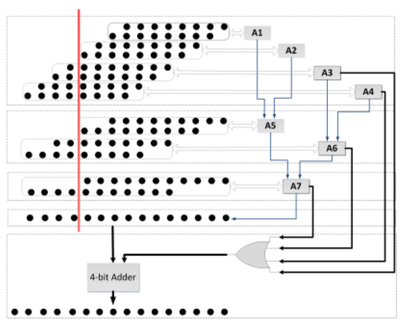


(b) Accurate 2 × 2 multiplier

© 2013 Han and Orshansky

Approximate Multipliers (2)

- A significant design aspect is to reduce the critical path delay in an approximate multiplier.
- A high-performance approximate multiplier with configurable partial error recovery is proposed for DSP applications [Liu14].
  - This multiplier leverages a newly-designed approximate adder that limits its carry propagation to the nearest neighbors for fast partial product accumulation.
  - Different levels of accuracy can be achieved through a configurable error recovery by using different numbers of MSBs for error reduction.
  - Similar performance as exact multipliers in image processing applications.



An approximate multiplier with partial error recovery

© 2013 Han and Orshansky

New Metrics for Approximate Circuits

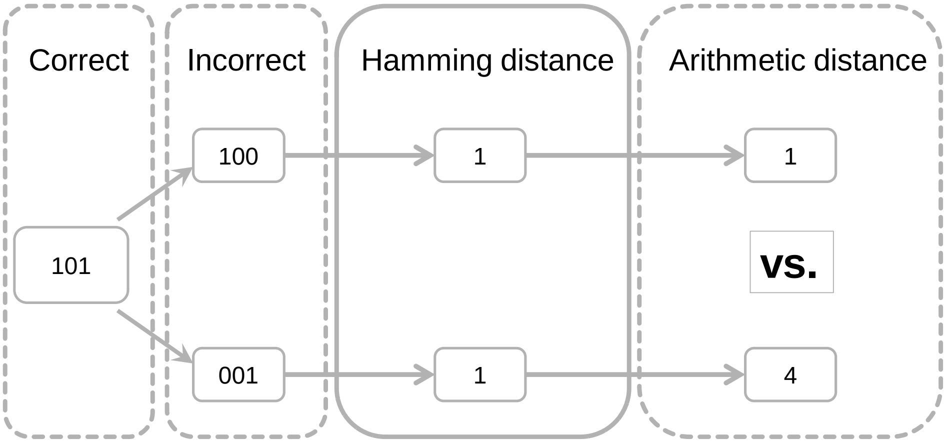
- The traditional metric of reliability is defined as the probability of correct circuit function:
  - Reliability of any approximate circuit is 0 for some inputs.
- New metrics are needed to assess the reliability of approximate circuits.
  - *Error rate* (ER) or *error frequency* is the fraction of incorrect outputs out of a total number of inputs in an approximate circuit [Breuer04].
  - *Error significance* (ES) refers to the degree of error severity due to the approximate operation of a circuit [Breuer04], as
    - the numerical deviation of an incorrect output from a correct one [Shin10],
    - the Hamming distance of the two vectors [Kahng12],
    - the maximum *error magnitude* of circuit outputs [Miao12].
  - A composite quality metric is the product of ER and ES [Shin11, Chong06].
  - Other common metrics include the relative error, average error and error distribution.

Embedded Tutorial, ETS'13, Avignon, France

© 2013 Han and Orshansky

Error Distance for Approximate Circuits

- Error distance is defined as the arithmetic distance between an inexact output and the correct output for a given input [Liang13].



Embedded Tutorial, ETS'13, Avignon, France

© 2013 Han and Orshansky

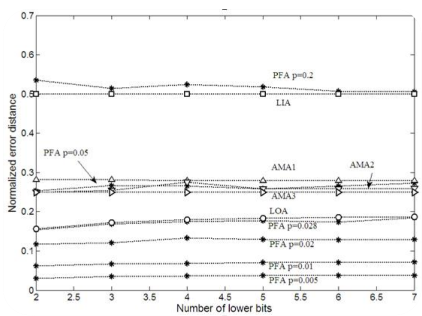
Mean and Normalized Error Distances

- *Mean error distance* (MED) considers the averaging effect of multiple inputs.
  - The MED is useful in measuring the implementation accuracy of a multiple-bit adder, but its value increases exponentially with the number of approximate bits in an adder.
- *Normalized error distance* (NED) is the normalization of MED for multiple-bit adders.
  - The NED is a nearly invariant metric independent of the size of an adder, so it is useful when characterizing the reliability of a specific design of full adders.
- MED or NED can be used with power or energy for evaluating the tradeoff between power consumption and precision in an approximate circuit.

© 2013 Han and Orshansky

NED as a Metric for Approximate adders

- The normalized error distance (NED) is almost independent of the number of approximate bits.



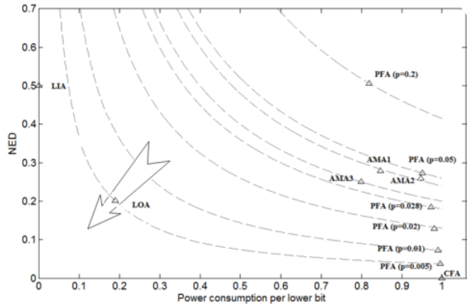
Normalized error distance (NED) vs. the number of approximate bits in an adder.

- It provides an effective alternative to an application-specific metric such as the peak signal-to-noise ratio (PSNR).

© 2013 Han and Orshansky

Power and Accuracy Tradeoffs

- The product of power and NED can be utilized for evaluating the tradeoff between power consumption and precision.
  - To emphasize the significance of a particular metric (such as the power or precision), a different measure with more weight on this metric can be used for a better assessment of a design according to the specific requirement of an application.



Power and precision tradeoffs: the product of power per bit and NED is shown by a dashed curve. The arrow points to the direction for a better design with a more efficient power and accuracy tradeoff.

© 2013 Han and Orshansky

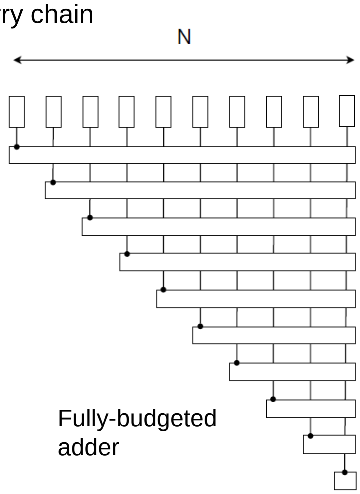
Outline

- MOTIVATION
- OVERVIEW OF ERROR-RESILIENT PARADIGMS
- APPROXIMATE ARITHMETIC CIRCUITS
  - Approximate adders
  - Approximate multipliers
  - Metrics for approximate computing
  - Quality-energy optimal designs
  - Approximate logic synthesis
- ALGORITHM-LEVEL APPROXIMATE COMPUTING TECHNIQUES

© 2013 Han and Orshansky

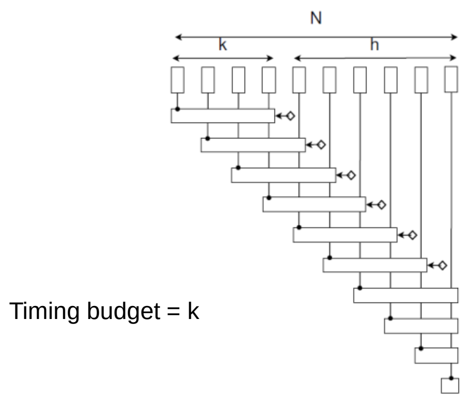
Formally Modeling Timing Starved Addition

- An effective way of energy saving: Vdd scaling
  - Can use timing-starved adders to reduce energy
  - Different ways of dealing with the carry chain
- Key question: what is the energy-optimal design strategy for timing-starved approximate adders?
- Need a tool for analysis of impact of internal values when starvation occurs
  - Formalizes error frequency-magnitude trade-offs
- Model: rightmost point of a segment shows farthest accessible internal carry (at a given budget)



© 2013 Han and Orshansky

Modeling Timing Starved Adder (TSA)



- Under reduced budget, some bits do not have access to primary carry-in
  - In a TSA, actual accessible carry depends on previous cycle and is unknown
- Error magnitude depends on pattern of T and F bits in starved operation
- In above TSA, any bit up to MSB can be false, max error magnitude is 2^{N-1}

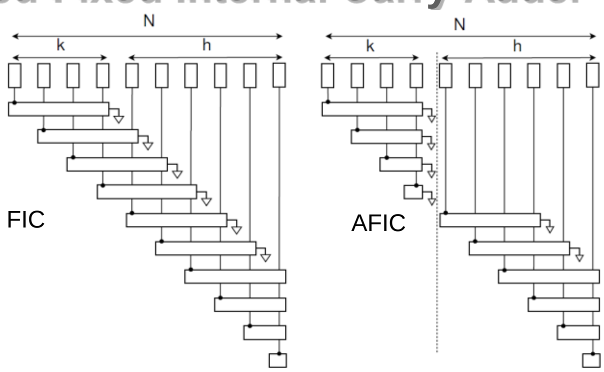
© 2013 Han and Orshansky

Error Pattern and Max Possible Error

- Goal: reduce error magnitude via two mechanisms
  - Prohibit ‘bad’ patterns of false bits
  - Convert true bits into false bits
- Statement: An *F** pattern with a bitwidth of *m*, with a right-most bit in the pattern rooted at bit position *r*, can result in errors with only two magnitudes: $2^{m+r}-1$ or 2^r
- Example: consider a 16b adder, k=8 (timing sufficient for 8 bits)
  - Suppose, MSB bits has TTFFFTTT pattern
  - Error could be either 2^{11} or $2^{14}-1$ depending on F direction (F_+ /F_-)
- Statement: An *F** pattern produces only small error 2^r if all internal carries are fixed to the same value (1 or 0)
  - Suggests a Fixed Internal Carry (FIC) structure

© 2013 Han and Orshansky

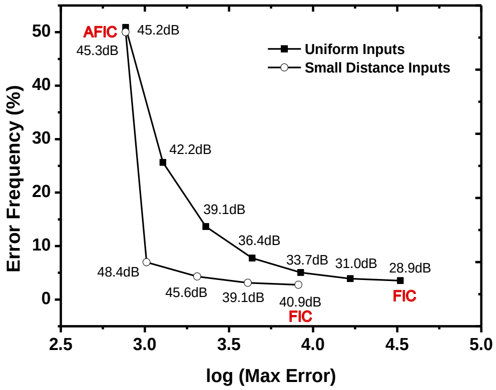
Aligned Fixed Internal Carry Adder



- Location of leftmost *possible* FT transition bounds *max* error magnitude
  - Max error is reduced if a false bit is followed by a string of false bits
  - To shift FT transition, convert as many following T bits to F bits
    - Convert TTFFFTTT pattern into TTFFFFFF → error = 2^{11} -> 2^8
- Realized via Aligned FIC (AFIC)
  - All bits > h must depend on same (in)correct carry
  - If bit j is F, j-1 is also F since both depend on same incorrect carry (fixed at 0)

© 2013 Han and Orshansky

Error Frequency-Magnitude Curve



- Alignment of segments means that effective carry chains are reduced for the sum-bits below MSB
  - Increases probability of individual and thus overall error
- For quadratic quality metrics, minimization of error magnitude is more important
  - As long as timing budget > 4bits, AFIC is better than FIC

© 2013 Han and Orshansky

Reducing Max Error via LSB Bounding

- AFIC always under (over) estimates true result depending on fixed (controlled) carry
- Quality-optimal adder is achieved by further reducing the chances to trigger the max error
- Optimal LSBs bounding logic should
  - Produce a correct result when C = 0, and
  - Produce largest possible value (i.e. 11 : : : 1) when C = 1 to compensate for MSBs errors
- Conditional Upper Bounding (CUB) logic
  - Fixed internal carries to 0
  - Under-estimating approximate adders

$$S_i^{'} = S_i \vee C$$

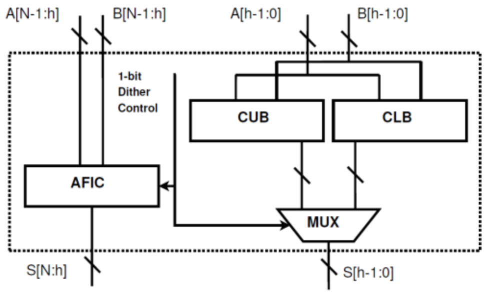
- Conditional Lower Bounding (CLB) logic

$$S_i^{'} = S_i \wedge C$$

© 2013 Han and Orshansky

Using Both Bounding Directions: Dithering Adder

- Can also employ *dithering approximate adder*
  - Alternates between CLB and CUB logic

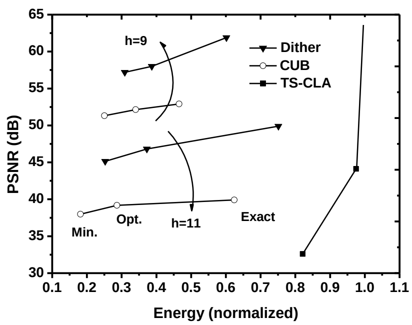


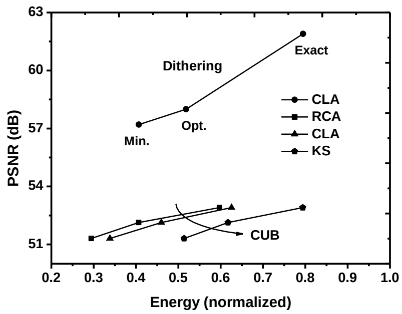
- Produces zero-centered error distributions and a reduced-variance error
- In experiments we use A[h-1] as the dithering signal

$$S_i' = (A_{h-1} \wedge S_i \wedge C) \vee (\overline{A}_{h-1} \wedge (S_i \vee C))$$

© 2013 Han and Orshansky

Adder Quality-Energy Design-Space Exploration





- 16-bit CLAs and TS-CLA
- Conventional timing-starved adder experiences a sharp drop in quality once their timing budget is exceeded.
- Different type AFIC adders with h=9

© 2013 Han and Orshansky

Timing Starved Adders in Image Processing

- IDCT based image processing system
  - Left: a truncated adder: PSNR=16.90dB, energy savings = 40%
  - Center: an inexact CB logic adder: PSNR=33.15dB, energy savings = 38%
- Sharpening filter
  - Right: an inexact CB logic adder: PSNR = 23.7dB (original 23.9dB), energy savings = 40%


IDCT, Truncated


IDCT, AFIC+CB


Filter, AFIC+CB

© 2013 Han and Orshansky

Outline

- MOTIVATION
- OVERVIEW OF ERROR-RESILIENT PARADIGMS
- APPROXIMATE ARITHMETIC CIRCUITS
  - Approximate adders
  - Approximate multipliers
  - Metrics for approximate computing
  - Quality-energy optimal designs
  - Approximate logic synthesis
- ALGORITHM-LEVEL APPROXIMATE COMPUTING TECHNIQUES

© 2013 Han and Orshansky

Approximate Logic Synthesis (ALS)

- Need new formal tools for synthesizing approximate versions of combinational Boolean functionality
  - E.g., use it to synthesize approximate versions of conditional bounding logic
- Approximate Boolean function has a modified truth table compared to exact function
  - Reduced complexity, delay, and energy
- Prior work
  - ALS with error frequency constraint only [Shin10, Palem10]
    - High runtime for large error frequencies
    - Does not consider error magnitude
  - ALS with error magnitude constraint only [Miao12, Roy12]
    - Based on unmodified conventional logic synthesis flow
    - Does not consider error frequency

© 2013 Han and Orshansky

General ALS Problem Formulation

- Goal: develop a general algorithm to generate approximate Boolean function from an exact function
  - Handling arbitrary constraints on error structure

$$\begin{aligned} \min \quad & L(F_{m,r}) \\ \text{s.t.} \quad & r \leq R, \\ & |F(x) - F_{m,r}(x)| \leq M \quad \forall x \in B^n \end{aligned} \tag{1}$$

- Our contributions:
  - Identify isomorphism of ALS w/ un-constrained error frequency and Boolean Relation (BR) minimization problem
  - Develop an efficient algorithm to refine error magnitude-constrained solution to satisfy error frequency constraint

© 2013 Han and Orshansky

ALS Constrained by Error Magnitude Only

- ALS constrained by error magnitude only

$$\begin{aligned} \min \quad & L(F_m) \\ \text{s.t.} \quad & |F(x) - F_m(x)| \leq M \quad \forall x \in B^n \end{aligned} \tag{2}$$

- Rewrite Equation 2 minterm-wise

$$\begin{aligned} \min \quad & L(F_m) \\ \text{s.t.} \quad & F_m(x_i) \in F(x_i) \cup E_i(x_i) \quad \forall x_i \in B^n \end{aligned} \tag{3}$$

E_i represents the additional values that F_m can take while satisfying the error magnitude constraints

- Error magnitude constrained ALS problem is isomorphic with the Boolean Relation (BR) problem

© 2013 Han and Orshansky

ALS Constrained by Error Magnitude Only

- Boolean Relation (BR)
  - A Boolean relation is a one-to-many, multi-output Boolean mapping, $R: B^n \rightarrow B^k$
- Boolean relation problem finds the min-cost 2-level function satisfying the given BR
  - Exact algorithms [Brayton89], [Lin90], [Jeong92]
  - Heuristic algorithms Herbs [Ghosh90], Gyocro [Watanabe93], BREL [Baneres04]
- Example: ALS with error magnitude $M = 1$

F	
a, b	c, s
00	{00}
01	{01}
10	{01}
11	{10}

F_m	
a, b	c, s
00	{00, 01}
01	{01, 00, 10}
10	{01, 00, 10}
11	{10, 01, 11}

© 2013 Han and Orshansky

Frequency-Constrained ALS: Formulation

- The frequency of errors of F_M solved by BR solver may not satisfy the constraint R
- For any function $F_{M,r}$

$$L(F_{M,r}) \geq L(F_{M,k}), \quad \textit{for any } r < k.$$

- Allows converting the problem to the min-cost increase problem

$$\begin{aligned} \min \quad & L(F_{M,R}) - L(F_M) \\ \text{s.t.} \quad & |F_{M,R} - F| \leq M \end{aligned} \tag{4}$$

- Goal: find $F_{M,R}$ by identifying the minterms of the function on which the *correctness* should be enforced with the minimum literal increase

© 2013 Han and Orshansky

Frequency-Constrained ALS: Strategy

- **Theorem**: For a single-output function F, the optimal set of minterms to add to the ON/OFF-set at the minimum literal increase in the cover of function $F_{M,r}$, lies among the prime implicants of the minimum cover of the minterms with identical error structure
- Definitions
  - DIFF minterm
    - A minterm x on which $F(x) \neq F_M(x)$
  - DIFF prime
    - Prime implicants of the minimum cover of each group

DIFF prime	CET	Group
x_1x_0	y_1y_0	#
00	$\{ET = 01, ET = 10\}$	1
01	$\{ET = 01, ET = 01\}$	2
1-	$\{ET = 10, ET = 00\}$	3

© 2013 Han and Orshansky

Example: Logically Approximate 2-bit Adder

$$F_1 = \begin{cases} C = a_1 b_1; \\ S_1 = a_1 b_1' + a_1' b_1; \\ S_0 = 1; \end{cases}$$

$$F_{1,\frac{2}{16}} = \begin{cases} C = a_1 b_1; \\ S_1 = a_1 b_1' + a_1' b_1 + a_0 b_0; \\ S_0 = a_1 b_1' b_0 + a_1' b_1 b_0 + a_0 b_0' + a_0' b_0; \end{cases}$$

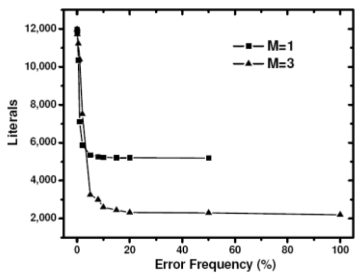
$$F_{1,\frac{1}{16}} = \begin{cases} C = a_0 b_1 b_0 + a_1 b_1; \\ S_1 = a_1' b_1 b_0' + a_1' a_0' b_1 + a_0 b_1' b_0 \\ \quad + a_1 a_0 b_0 + a_1 b_1'; \\ S_0 = a_1 b_1' b_0 + a_0 b_0' + a_0' b_0; \end{cases}$$

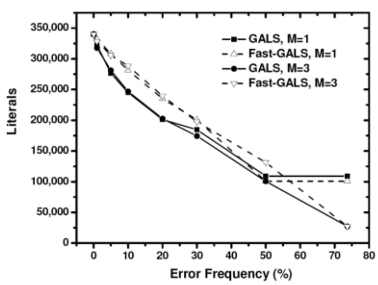
$$F = \begin{cases} C = a_0 b_1 b_0 + a_1 a_0 b_0 + a_1 b_1; \\ S_1 = a_1' a_0 b_1' b_0 + a_1 a_0 b_1 b_0 + a_1 b_1' b_0' \\ \quad + a_1' b_1 b_0' + a_1 a_0' b_1' + a_1' a_0' b_1; \\ S_0 = a_0 b_0' + a_0' b_0; \end{cases}$$

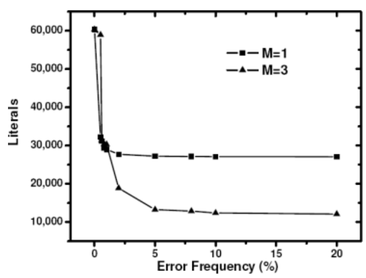
© 2013 Han and Orshansky

ALS Experiments: Approximate Arithmetic Circuits

- Adder:
  - 8b: runtime 2s~5m
  - 10b: runtime 30s~3h
- Multiplier:
  - Truncated 8b: 20m~3.3h







© 2013 Han and Orshansky

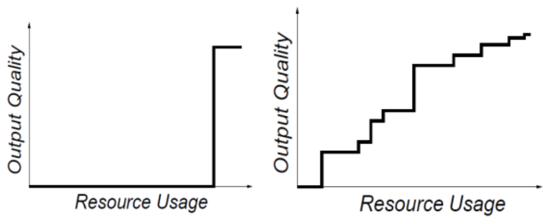
Outline

- MOTIVATION
- OVERVIEW OF ERROR-RESILIENT PARADIGMS
- APPROXIMATE ARITHMETIC CIRCUITS
  - Approximate adders
  - Approximate multipliers
  - Metrics for approximate computing
  - Quality-energy optimal designs
  - Approximate logic synthesis
- ALGORITHM-LEVEL APPROXIMATE COMPUTING TECHNIQUES

© 2013 Han and Orshansky

Incremental Refinement Property

- *Incremental refinement* property: iterations of an algorithm can be terminated earlier to save energy in exchange for incrementally lower quality.
- FFT-based maximum-likelihood detection algorithm [Nawab *et al.* 1997]
  - Can terminate FFT at an intermediate stage of computation
  - Detection performance grows monotonically with number of stages
    - Converges to that of the exact ML detector
- Support vector machines [Chipa *et al.* 2010]
  - Number of support vectors correlates well with algorithm quality
    - Also determines algorithm's energy



Source: Nawab *et al.* 1997

© 2013 Han and Orshansky

Dynamic Bit-Width Adaptation

- Run-time adaptation of bit-width is an effective tool
  - Powerful in changing energy-quality trade-off; easily available
- Ex. Discrete-cosine transform algorithm [Park *et al.* 2011]
  - High-frequency DCT coefficients are small after quantization
    - Less impact on image than low-frequency coefficients
  - Lower bit-width can be used for high frequency coefficients
    - Use carry save adder tree implementation and turn off un-used adders
    - 60% power savings at slight image quality degradation (3dB)

	Row Coefficients							
	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7
Normal Operation	9	9	9	9	9	9	9	9
Level 1	9	9	6	6	6	4	0	0
Level 2	9	6	4	4	0	0	0	0
Level 3	9	4	4	0	0	0	0	0

	original	level 1	level 2	level 3
lena	34.97	34.85	33.79	31.51
peppers	36.16	35.55	33.02	30.60
monarch	36.05	35.88	34.00	31.08
sail	34.40	34.15	32.75	30.02

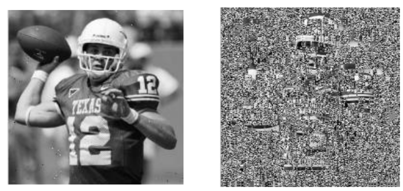
	Normal operation	Trade-off level 1	Trade-off level 2	Trade-off level 3
power (mW)	94.05	60.54	37.70	23.8
percentage	100%	64%	40%	25%

Source: Park *et al.* 2011.

© 2013 Han and Orshansky

Adaptive Voltage-Overscaling

- Conventional design aims to ensure timing correctness
- At circuit level, can reduce energy by accepting *some timing* errors
- Timing-error acceptance philosophy
  - Worst-case timing is not guaranteed
  - Prevent severe quality loss



Direct Vdd reduction

Controlled Timing-Error Acceptance (same Vdd reduction as above)

Embedded Tutorial, ETS'13, Avignon, France
© 2013 Han and Orshansky

Reducing Energy via Controlled Timing Error Acceptance

- Scaling of Vdd leads to non-uniform timing errors
  - Eliminate sources of worst and earliest timing errors

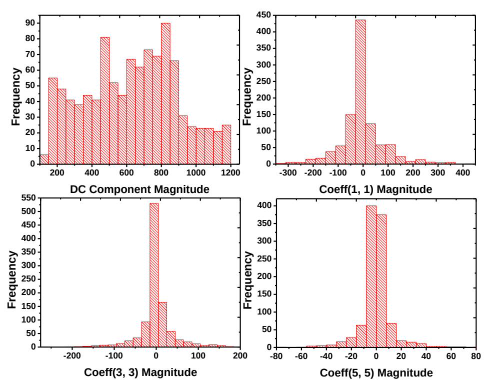
Principles

- (1) Controlling large magnitude errors in operations by exploiting the knowledge of statistics of operands
- (2) Controlling the frequency of error-generating operations by dynamically re-arranging the sequence of operations, i.e. in accumulation
- (3) Timing errors in early algorithm steps tend to cause larger final errors due to incorrect data reuse

© 2013 Han and Orshansky

Timing Error Dependence on Operand Values

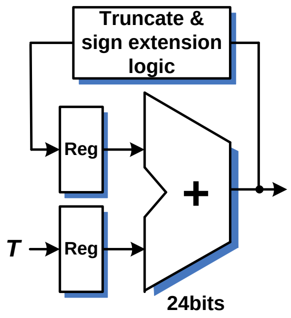
- Small positive + small negative addition leads to early and large errors
- Observation: these patterns can be processed on smaller adder
  - Smaller adder -> smaller worst-case delay -> can reduce Vdd w/o errors
- Use small-width adder *selectively* for small opposite sign operands
  - Adder 1: full width (24-bit)
  - Adder 2: reduced width (use sign extension)
    - In practice, use variable–width adder
- 2D-IDCT coefficient matrix components distribution [Goodman, 2000]

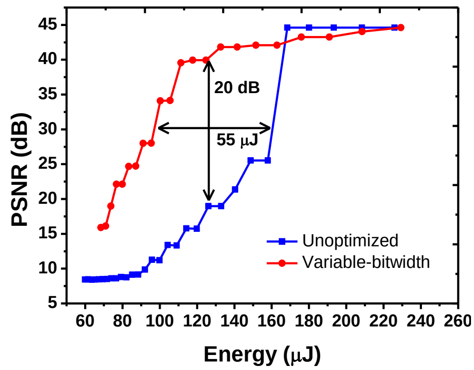


© 2013 Han and Orshansky

Error Control Based on Operand Statistics

- Use reduced-width adder for small operands
  - Keep a full-width adder for larger operands
  - In practice, use single variable-width adder
- For a PSNR loss of 5dB, we get 32% energy reduction

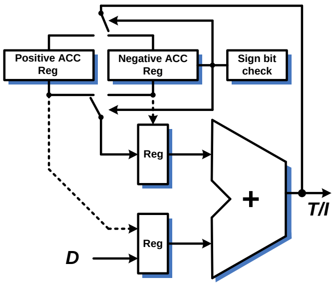


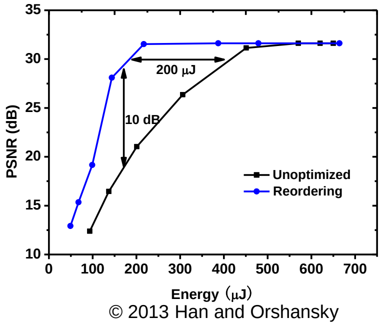


© 2013 Han and Orshansky

Controlling Error Frequency by Dynamically Rearranging Operation Sequence

- In error-accepting paradigm, frequency of errors determines quality loss
  - Reduce Vdd to cause violations (and reduce energy)
  - Reduce the frequency of error occurrence
- Accumulation on opposite-sign numbers causes large error
  - Case 1: 11111111+00000001+11111111+00000001
  - Case 2: (11111111+11111111)+(00000001+00000001)
- Solution: separate accumulation registers for positive/negative numbers

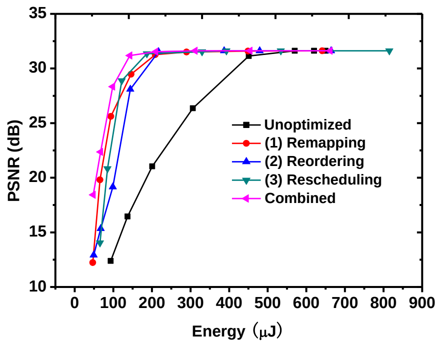




© 2013 Han and Orshansky

Error Acceptance Techniques in IDCT

- Combination of techniques is effective
  - Q-E profile is significantly improved
- Energy savings: 60% at PSNR = 30dB
  - Area overhead: ~3%
  - Acceptable image quality



(a) Nominal: Energy=570μJ PSNR=31.6dB

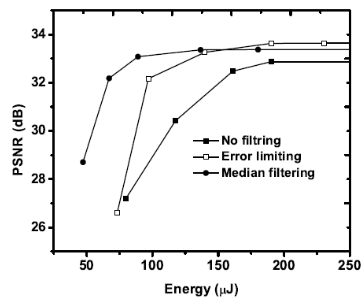
(b) Nominal: Energy=137μJ PSNR=16.5dB

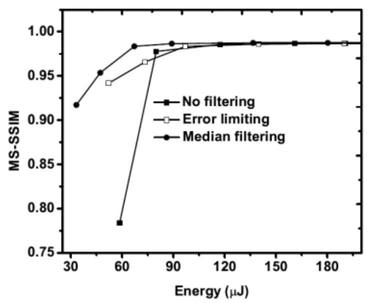
(c) TERRA: Energy=143μJ PSNR=31.2dB

(d) TERRA: Energy=98.5μJ PSNR=28.3dB

© 2013 Han and Orshansky

Error Post-Processing in 2D-IDCT





(a) Test image with error limiting: Energy=140μJ PSNR=33.08dB MS-SSIM=0.9860

(b) Test image with median filtering: Energy=137μJ PSNR=33.3dB MS-SSIM=0.9866

(c) Test image with error limiting: Energy=97μJ PSNR=32.2dB MS-SSIM=0.9834

(d) Test image with median filtering: Energy=89μJ PSNR=32.2dB MS-SSIM=0.9835

© 2013 Han and Orshansky

Summary

- Review of error-permissive paradigms: stochastic, probabilistic, and approximate computing
- Approximate arithmetic circuits (adders, multipliers)
  - Metrics for approximate computing
  - Introduced a formal model for timing starved addition
- Approximate logic synthesis
  - Boolean-relations based algorithm
- Algorithm-level approximate computing
  - Incremental refinement principle
  - Dynamic bit-width adaptation
  - Developed several strategies for circuit-level timing error acceptance

© 2013 Han and Orshansky

Conclusions

- Approximate computing shows promise
  - Large number of error-permissive applications
  - Does not seem suitable for general-purpose computing
  - Better understanding of scope of application is needed
- Design considerations
  - Identification of algorithm phases that allow errors
    - E.g. control vs. data, error tolerant vs. critical computations
  - Identification of relevant error metric
    - Error frequency vs. magnitude
- Tool support needed
  - Early work on automated approximate synthesis
- Open question:
  - When will approximate computing principles be used in practice?

© 2013 Han and Orshansky