

Trading Accuracy for Power with an Under-designed Multiplier Architecture

Parag Kulkarni*¹, Puneet Gupta¹, Milos Ercegovac²

¹Electrical Eng., UCLA

²Computer Sci., UCLA

(paragk@ucla.edu)

NanoCAD Lab - <http://nanocad.ee.ucla.edu/>

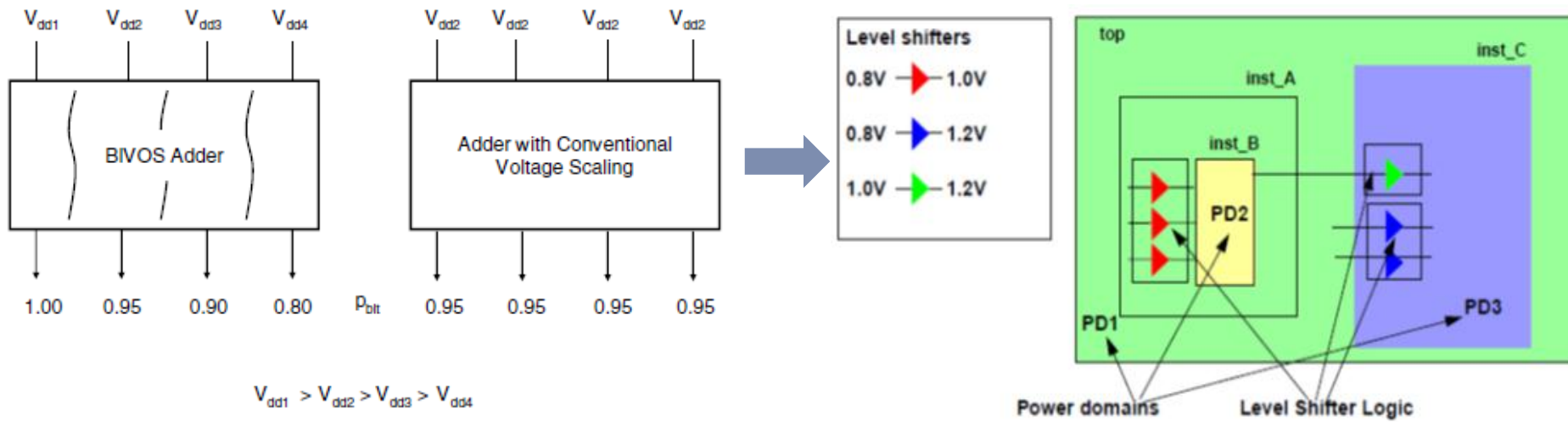
Outline

- Introduction
- Inaccurate Multiplier Design
- Hardware vs. Software Tradeoff
- Conclusions & Future Work

Motivation

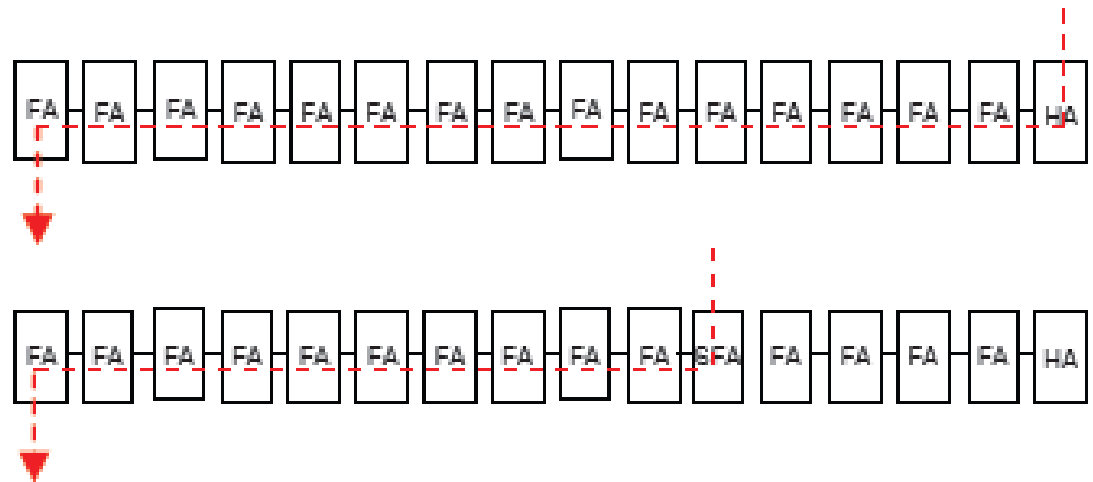
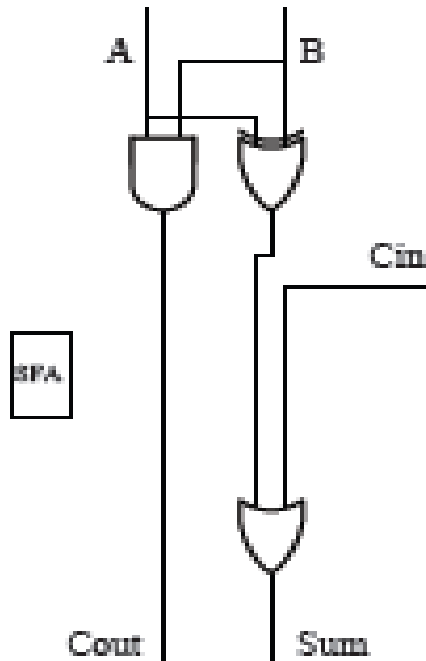
- Accuracy can be used as a design metric, traded for :
 - Area – Nearly half the hardware in an array multiplier is there to get the last bit right
 - Timing – in n-bit addition, on average the carry propagates $\log_2 n$ positions
 - Power – a probabilistic switch's energy consumption can be as low as $kt \cdot \log_2^*(1-p)$ Joules per transition
- Especially useful in applications that are capable of absorbing errors

Voltage Over-scaling



- Introduce inaccuracy by intelligently over-scaling voltage(s):
 - Saving power, at the expense of incomplete computations
 - Use multiple voltage levels to generate acceptable SNRs
- Ignore overhead of level conversion, routing etc.
- Not always feasible in compact arithmetic layouts

Under-design



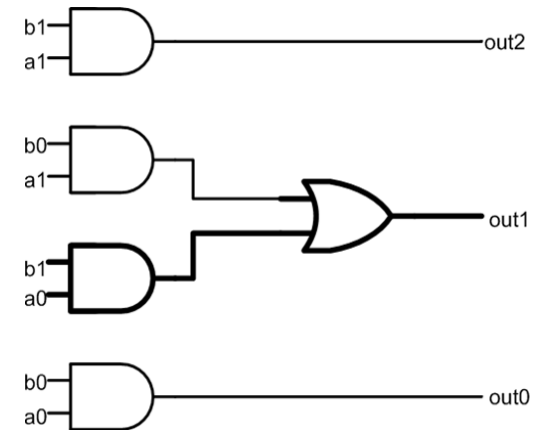
- Focus on inaccurate adders
- $Carry_{Out}$ independent of $Carry_{In}$, shortens critical path
- Accuracy traded for speed or yield
- No easy correction methodology

Outline

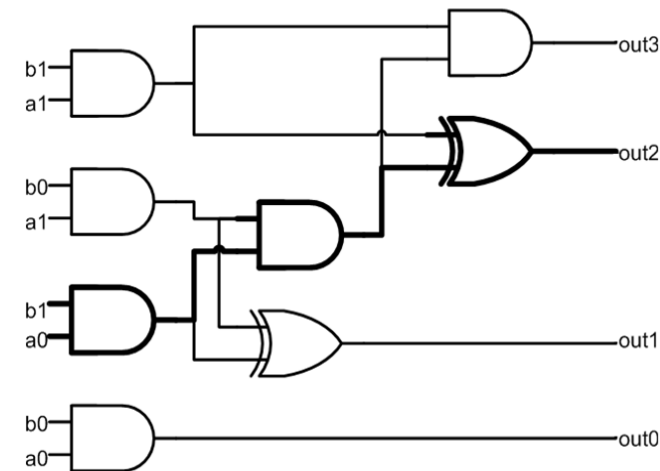
- Introduction
- Inaccurate Multiplier Design
 - Error rates and Power savings
 - Application : Image Filtering
 - Correct Mode of operation
- Hardware vs. Software Tradeoff
- Summary, Conclusions & Future Work

Inaccurate Multiplier

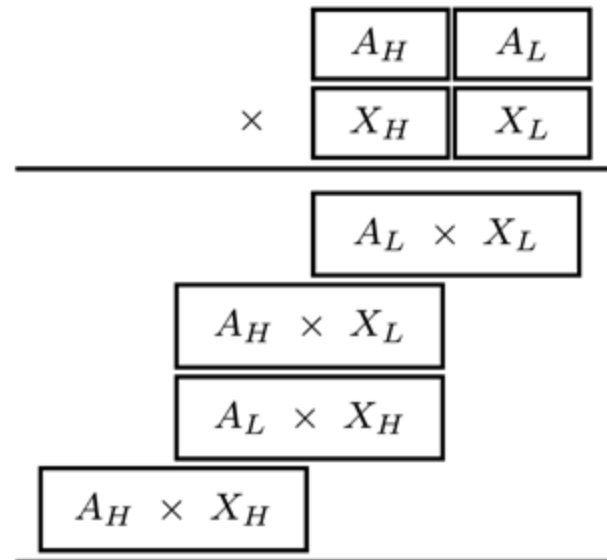
		B_1B_0			
		00	01	11	10
A_1A_0	00	000	000	000	000
	01	000	001	011	010
	11	000	011	111	110
	10	000	010	110	100



- Introduce error by manipulating the logic function, using the 2x2 multiplier as a building block
- Correct for 15/16 input combinations
- The modified K-Map allows for a simpler implementation:
 - Half the area
 - Shorter critical path
 - Lesser switching capacitance



Building Larger Multipliers



- A_H and X_H upper two bits, A_L and X_L lower two bits
- Use the inaccurate 2x2 block to generate partial products, add shifted partial products to get final result
- Inaccuracy introduced through the partial products alone, the adder network remains accurate

Error Rates

Bit-Width	Error Probability	Mean-Error	Max-Error
2	0.06	1.39%	22.22%
4	0.19	2.60%	22.22%
8	0.46	3.25%	22.22%
12	0.67	3.31%	22.22%
16	0.81	3.32%	22.22%

- Building block error-probability = $1/16$
- Used C++ simulation models for higher bit widths
- Relatively large but constant max-error of 22.22%
- Mean-Error increases with bit-width saturating around $\sim 3.3\%$

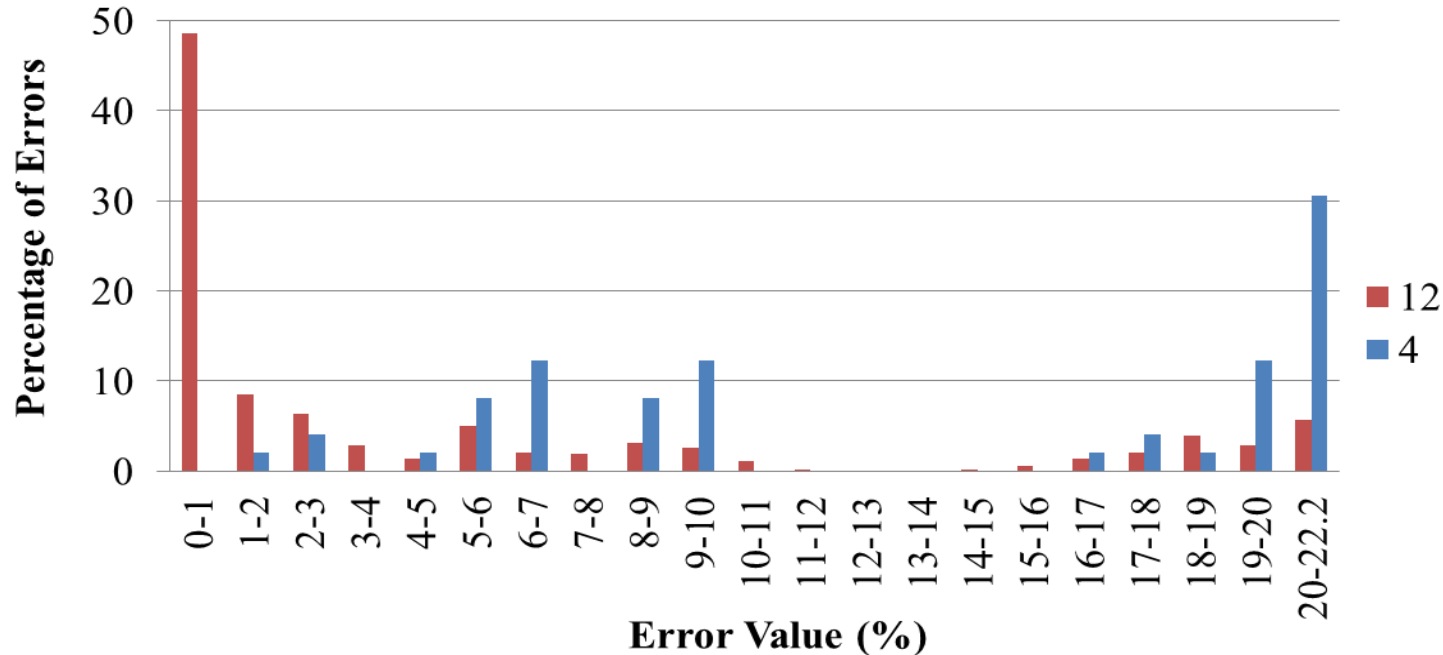
Max Error

$$\begin{array}{r}
 1111 \\
 1111 \\
 \hline
 0111 \\
 11100 \\
 11100 \\
 1110000 \\
 \hline
 11001111
 \end{array}
 \begin{array}{l}
 \leftarrow 15 \\
 \leftarrow 15 \\
 \\
 \\
 \\
 \leftarrow 175, \text{ error} = 50/225 = 22.22\%
 \end{array}
 \begin{array}{r}
 11 \\
 11 \\
 \hline
 111
 \end{array}
 \leftarrow 7, \text{ error} = 2/9 = 22.22\%$$

- Max-Error remains constant at 22.22%
- Property of the building block, transferred to higher bit-widths
- Probability of hitting max-error reduces rapidly with increasing bit-width

Bit-Width	Max-Error Probability
2	0.0625
4	0.0351
8	0.0243
12	0.0226
16	0.0225

Saturating Mean-Error

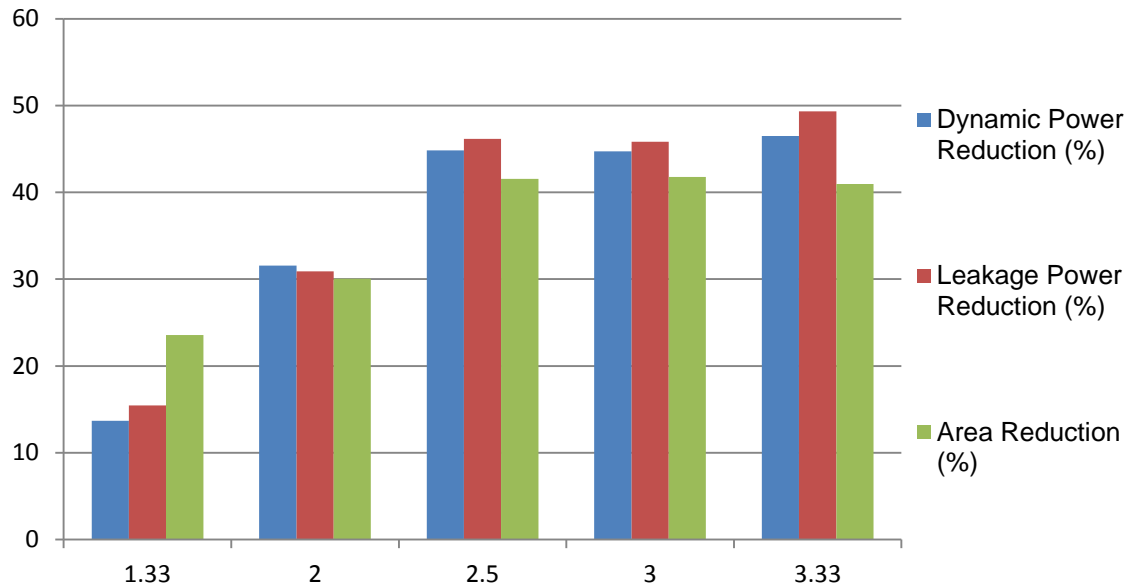


- Relatively small mean error of between $\sim 1.4\%$ - 3.3%
- This compares well with other approaches
- As bit width increases the percentage of larger errors reduces, leading to saturation

Experimental Setup

- Architectures written in Verilog and synthesized using RTL-Compiler to 45nm Nangate open cell library
- Inaccurate Multiplier :
 - 2x2 building blocks to generate partial products
 - Accurate adder tree to produce final solution
- Accurate Multiplier, best of either :
 - Generate accurate partial products and add
 - Architecture selection and optimization by the tool
- Simulation in NCSIM → VCD file → back-annotate for power analysis

Power Savings



- Power and area savings between 31%-45%
- Power benefits of building block get transferred to larger bit-widths

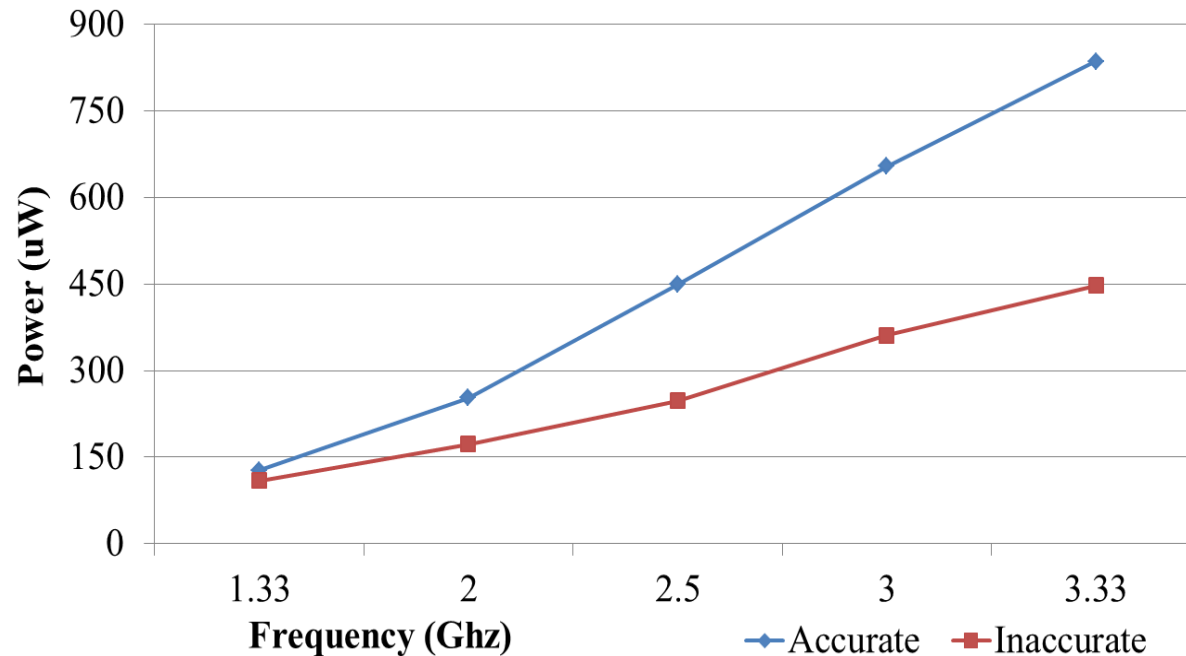
Bit Width	F	1.25F	1.5F	1.75F	2F	Avg.
2	44.9%	42.1%	42.1%	48.9%	48.9%	45.4%
4	13.7%	31.6%	44.8%	44.7%	46.5%	36.3%
8	33.1%	40.4%	26.3%	48.8%	58.9%	41.5%
16	25.6%	29.6%	32.4%	33.8%	37.4%	31.8%

Design Level Savings

Design	# of Multipliers	# of Gates	Multiplier Power Reduction	Total Power Reduction
FFT	32	158K	25.16%	13.98%
FIR	4	1.1K	31.09%	18.30%
Mini-RISC	1	10K	28.04%	1.51%

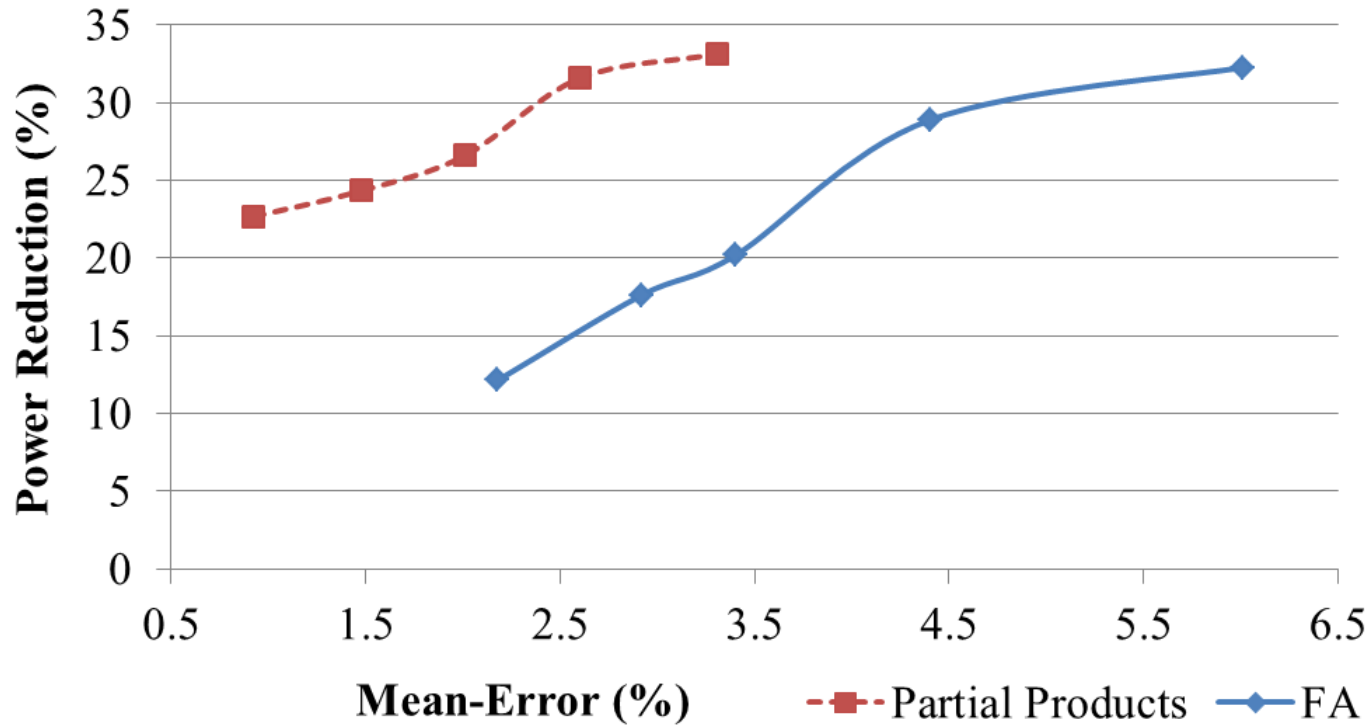
- Power savings reflect best in multiplier intensive designs like the FIR filter
- Less pronounced for other designs
- Benefits of approximate arithmetic are very design specific

Power Savings vs. Frequency



- Savings increase as desired frequency of operation increases
- The inaccurate multiplier is inherently faster, needing less aggressive gate sizing to meet increasing frequency constraints

Tunable Error



- Replacing individual 2x2 components with accurate versions allows a designer to exploit other points on the power-accuracy curve
- Inaccurate partial products yield a better trade-off than an inaccurate adder tree

Application : Image Filtering



Accurate Baseline



Inaccurate: 41.5% power reduction, 20.3dB

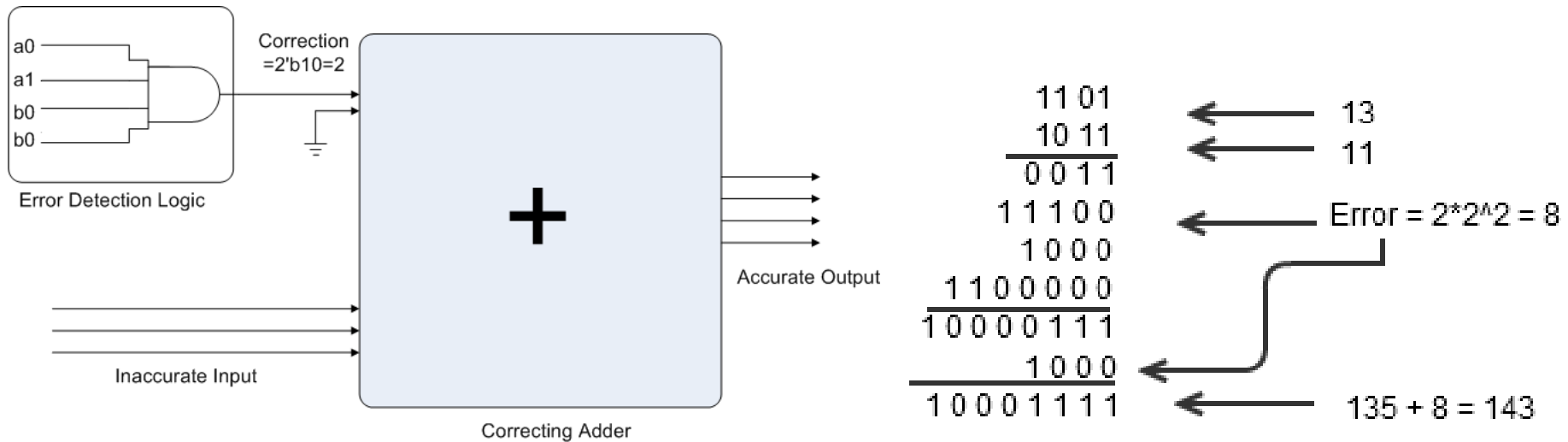


Over-scaling: 30% power reduction, 9.16dB



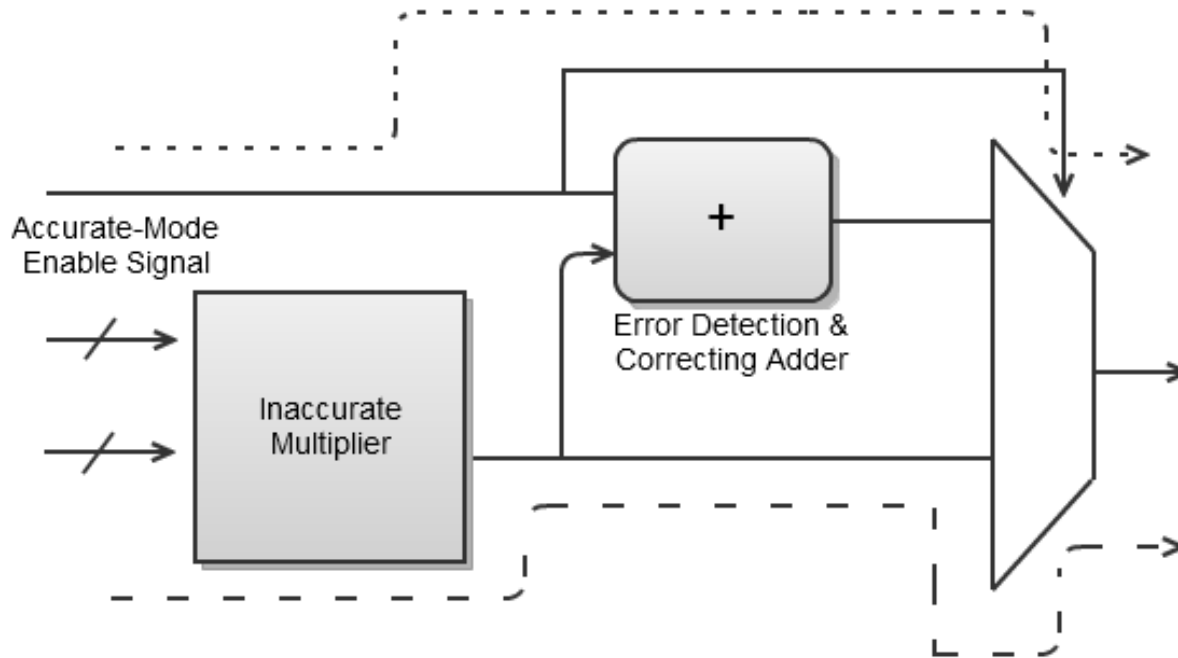
Over-scaling: 50% power reduction, 2.64dB

Error Detection/Correction



- Simple decoder logic used to detect presence and value of errors
- *Residual Adder* can be used to add the correction amount
- Can produce correct behavior when needed

Accurate Mode of Operation

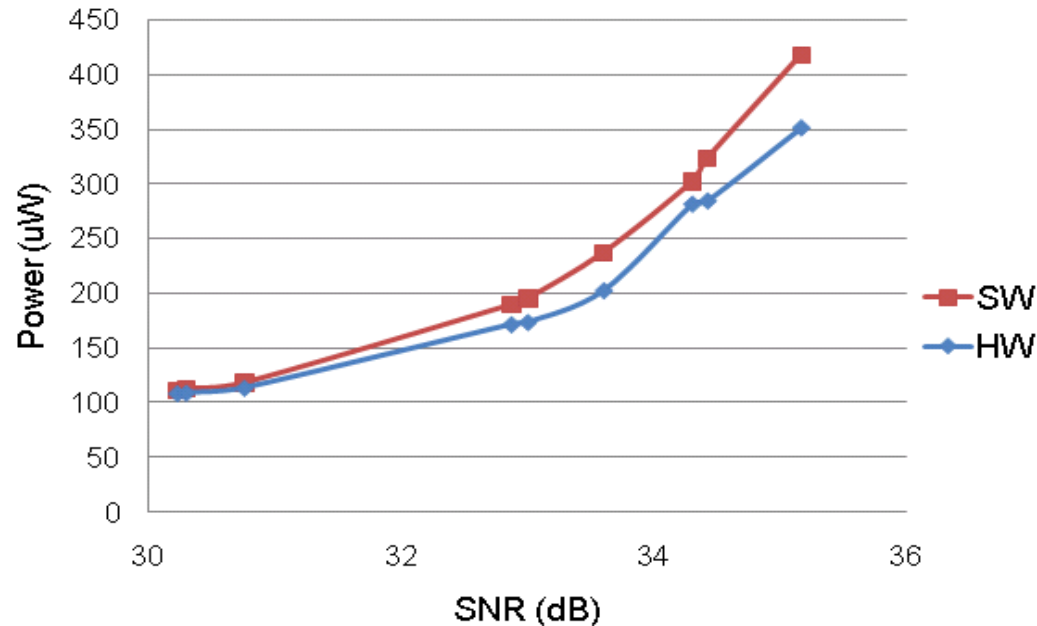


- 4.6% - 10.5% area & 4.8% - 8.56% power overhead (over baseline accurate)
- Envision a system with two modes of operation :
 - Regular : inaccurate, power-saving mode with correction unit turned off
 - Critical : accurate, correction unit ON leading to slower and more power hungry operation

Outline

- Introduction
- Inaccurate Multiplier Design
- Hardware vs. Software Tradeoff
- Conclusions & Future Work

Comparison with Software Tradeoff



- Software allows for a tradeoff between accuracy and runtime/power for applications such as JPEG
- Hardware approach still consumes less power for the same SNR if it lies on the critical path
- Savings are smaller than stand-alone inaccurate vs. baseline

Outline

- Introduction
- Inaccurate Multiplier Design
- Hardware vs. Software Tradeoff
- Conclusions & Future Work

Conclusions (1/2)

- We propose an inaccurate multiplier architecture that :
 - Leverages a 2x2 building block to trade accuracy for power
 - Displays a mean-error of 1.39% - 3.35%
 - Results in power savings between 30% - 50%
- For a simple image filtering application the inaccurate multiplier :
 - Achieves 2X - 8X better SNR than simple voltage scaling
 - Does not suffer from multiple voltage domain overheads of advanced over-scaling techniques
- The architecture is extended to allow for a correct mode of operation

Conclusions (2/2)

- For inaccurate multipliers, introducing errors via partial products offers a better power vs. error tradeoff than through the adders
- Designing for error avoids the overheads of multiple power domains and can be easily integrated into the ASIC design flow
- Benefits of inaccurate arithmetic are very design specific
- Software based tradeoff can offer comparable benefits for some applications

Future Work

- Extension of methodology to other arithmetic units – adders, dividers etc.
- Comparison with bit-width reduction based techniques
- Simpler decoder units, that would reduce cost of correction