Probabilistic CMOS Technology*
Probabilistic System-on-a-Chip Architectures**

Sebastian Schiessl

Technische Universität München

sebastian.schiessl@mytum.de

* Akgul et al., Probabilistic CMOS Technology: A Survey and Future Directions,

** Chaprakani et al., Probabilistic System-on-a-Chip Architectures

Probabilistic CMOS Technology
Probabilistic System-on-a-Chip Design

- Motivation
- Foundational Model
- Technology of Probabilistic CMOS
- Application Scenarios
  - (i) Applications that harness probabilistic behaviour
  - (ii) Applications that can tolerate probabilistic behaviour
  - (iii) Applications which can not tolerate probabilistic behaviour
- Conclusion

Motivation

- Scaling into the nanometer regime poses several problems
  - Noise
  - Parameter Variations
  - Device Perturbations
- Overcoming those hurdles leads to increased power consumption
- Devices already operate close to the thermal limit
- Rigorous testing methodology is required

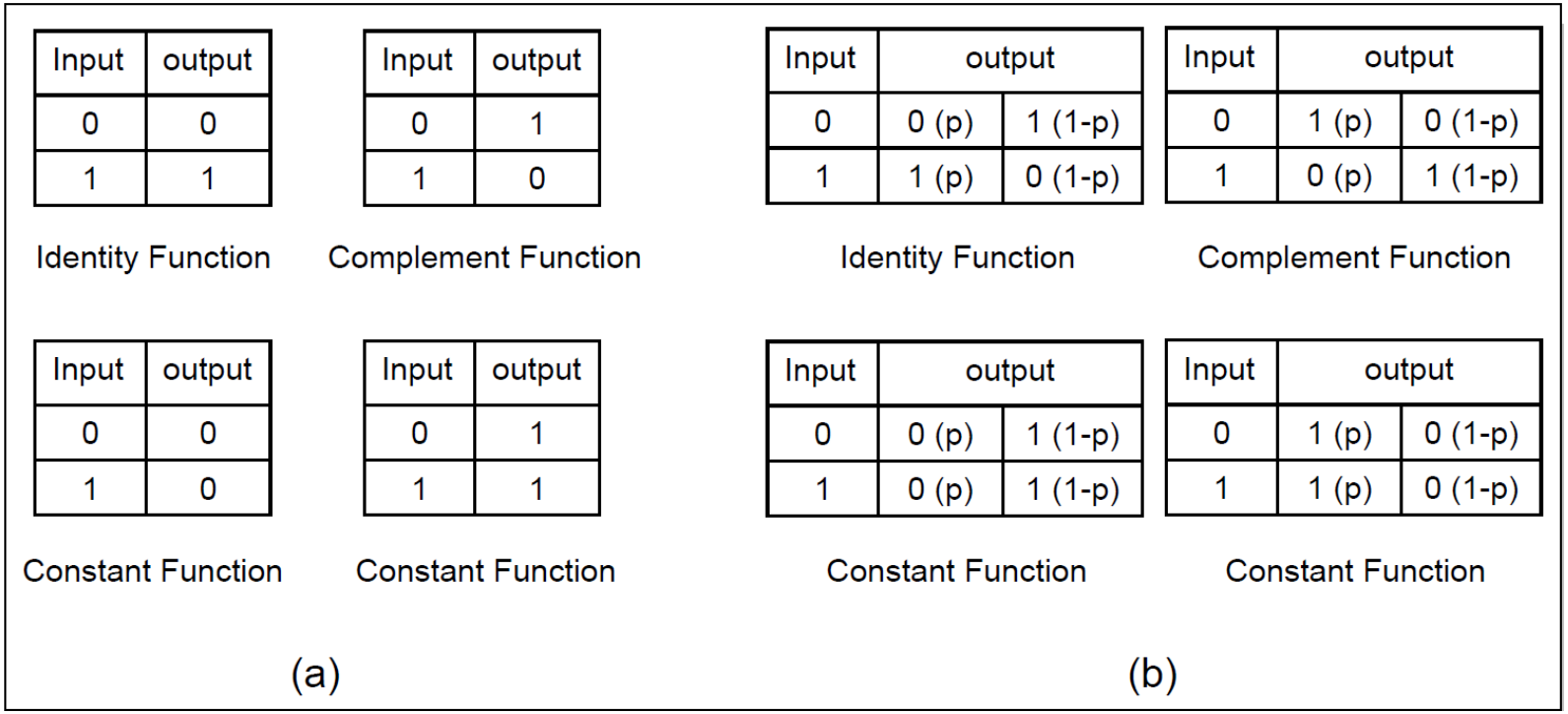
➔ A shift in design paradigm towards probabilistic design seems to be inevitable

Harnessing probabilistic behaviour

Noise can also be used to perform useful computations. Applications include:

- Bayesian Inference (BN)
- Probabilistic Cellular Automata (PCA)
- Random Neural Networks (RNN)
- Hyper Encryption (HE)
- The celebrated test for primality

Foundational Model: Probabilistic Switch



Input	output
0	0
1	1

Identity Function

Input	output
0	1
1	0

Complement Function

Input	output	
0	0 (p)	1 (1-p)
1	1 (p)	0 (1-p)

Identity Function

Input	output	
0	1 (p)	0 (1-p)
1	0 (p)	1 (1-p)

Complement Function

Input	output
0	0
1	0

Constant Function

Input	output
0	1
1	1

Constant Function

Input	output	
0	0 (p)	1 (1-p)
1	0 (p)	1 (1-p)

Constant Function

Input	output	
0	1 (p)	0 (1-p)
1	1 (p)	0 (1-p)

Constant Function

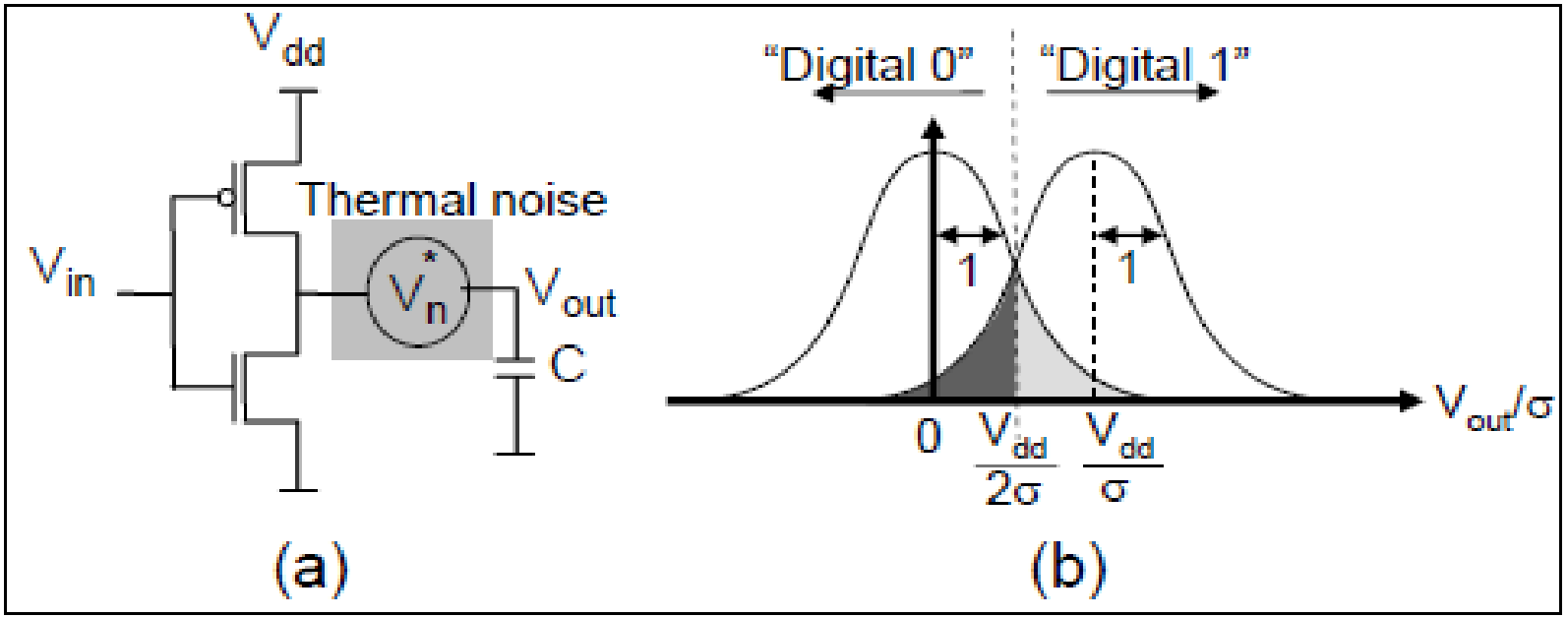
(a)

(b)

(a) Deterministic Switches [Akgul et al., p.2]

(b) Probabilistic Switches

Technological Considerations



Lower Error Probability means significantly higher Energy Comsumption, as $E \sim V_{dd}^2$

[Akgul et al., p.2]

Technological Considerations

Probability of Correctness: $p = 1 - \frac{1}{2} erfc\left(\frac{V_{dd}}{2\sqrt{2}\sigma}\right)$

with $erfc(x) = \frac{2}{\sqrt{\pi}} \int_{x}^{\infty} e^{-t^2} dt$

Using upper bound for erfc(x): $p < 1 - 0.28\, e^{-1.275 \frac{V_{dd}^2}{8\sigma^2}}$

Switching Energy E: $E = \frac{1}{2} C V_{dd}^2$

$$E(p, C, \sigma) > C\sigma^2 \left(\frac{4}{1.275}\right) \ln\left(\frac{0.28}{1-p}\right)$$

Technological Considerations

Law 1: Energy-probability Law: For any fixed technology generation (which determines the capacitance $C = \hat{C}$) and constant noise magnitude $\sigma = \hat{\sigma}$, the switching energy $\hat{E}_{\hat{C},\hat{\sigma}}$ consumed by a probabilistic switch grows with p. Furthermore, the order of growth of $\hat{E}_{\hat{C},\hat{\sigma}}$ in p is asymptotically bounded below by an exponential in p since $\hat{E}_{\hat{C},\hat{\sigma}}(p) = \Omega_r\left(E^e_{\hat{C},\hat{\sigma}}(p)\right)$.

Law 2: Energy-noise Law: For any fixed probability $p = \hat{p}$ and a fixed technology generation (which determines the capacitance $C = \hat{C}$), $\tilde{E}_{\hat{C},\hat{p}}$ grows quadratically with σ, since $\tilde{E}_{\hat{C},\hat{p}}(\sigma) = \Omega_r\left(E^q_{\hat{C},\hat{p}}(\sigma)\right)$.

[Akgul et al.]

- Applications that harness probabilistic behaviour

Noise can also be used to perform useful computations. Applications include:

- Bayesian inference (BN)
- Probabilistic Cellular Automata (PCA)
- Random Neural Networks (RNN)
- Hyper Encryption (HE)
- The celebrated test for primality [Rabin 1976; Solovay and Strassen 1977]

Key probabilistic steps ("toin cosses") of those algorithms are identified and built with PCMOS technology

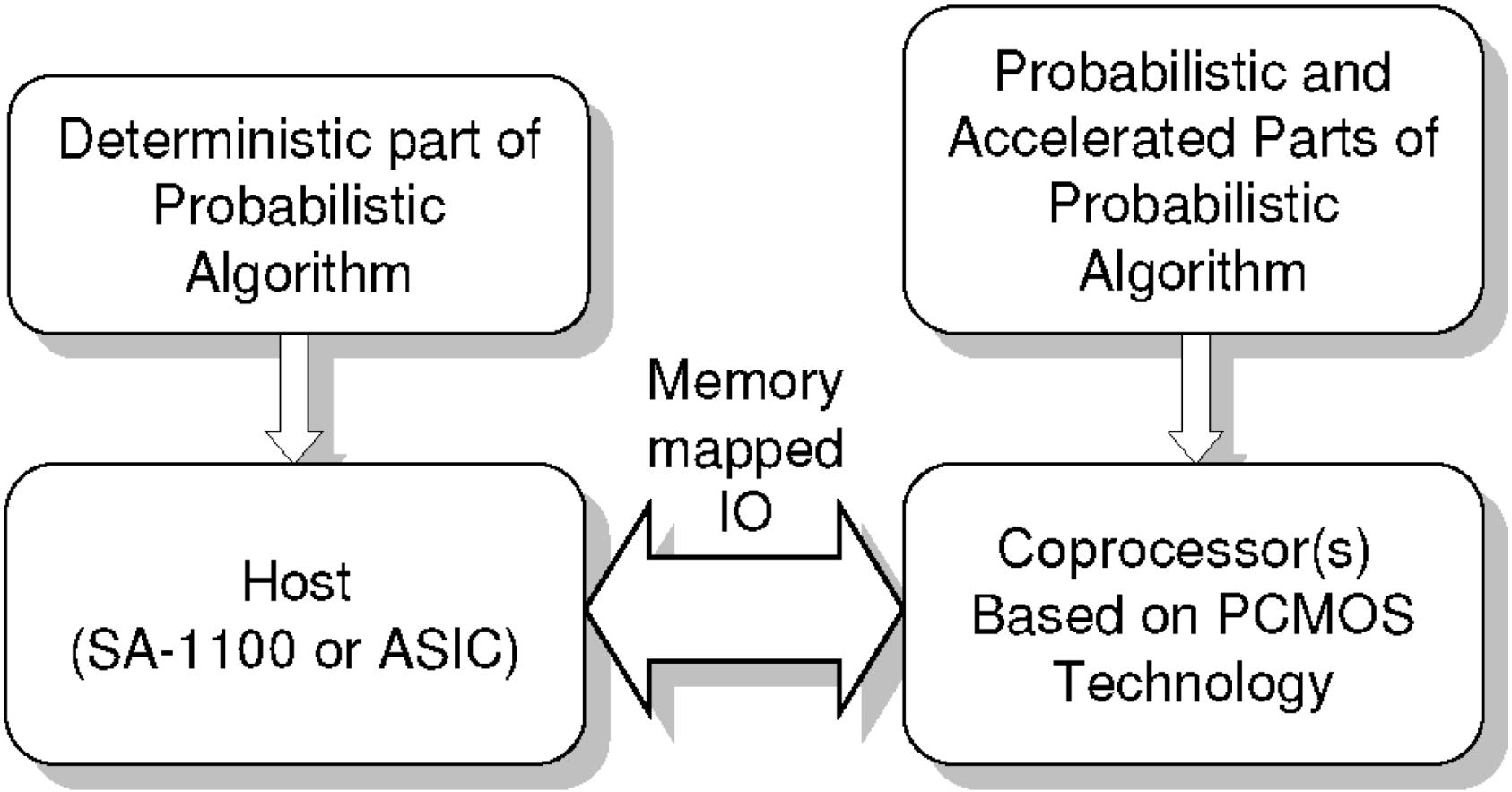
Algorithm	Application Scenarios	Implemented(s) Application(s)	Core Probabilistic Step
Bayesian Inference [MacKay 1992]	SPAM Filters, Cognitive applications, Battlefield Planning [Pfeffer 2000]	Windows printer trouble shooting, Hospital Patient Management [Beinlich et al. 1989]	Choose a value for a variable from a set of values based on its conditional probability
Random Neural Network [Gelenbe 1989]	Image and pattern classification, Optimization of NP-hard problems	Vertex cover of a graph	Neuron firing modeled as a Poisson process
Probabilistic Cellular Automata [Wolfram 2002]	Pattern classification	String classification [Fuks 2002]	Evaluating the probabilistic transition rule
Hyper-Encryption [Ding and Rabin 2002]	Security	Message encryption	Generation of a random string and encryption pad generation from this string

[Chakrapani et al.]

Applications that harness probabilistic behaviour

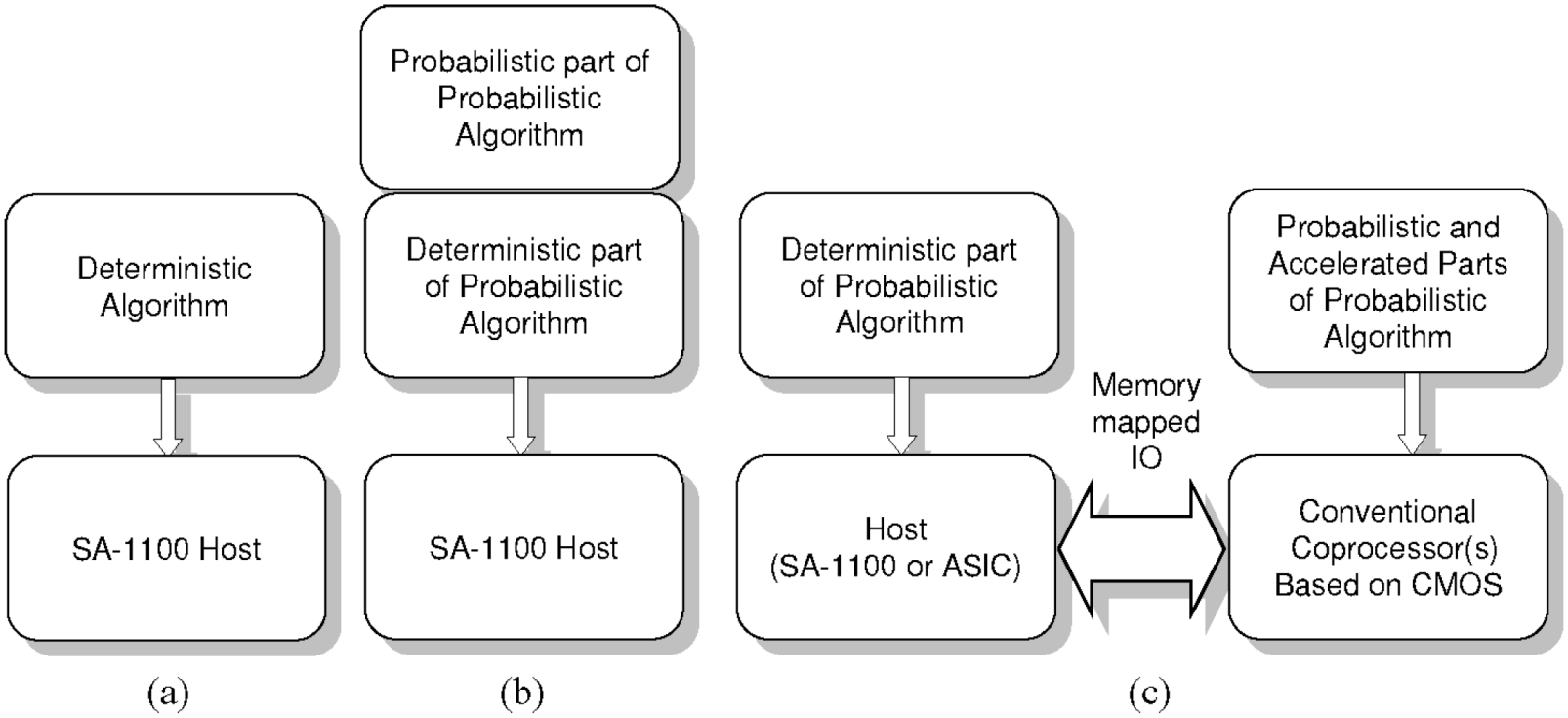
- Energy savings not only on the device level (due to reduced operating voltage) but mainly on the application level
  - Useful for those applications that are using probabilistic algorithms
  - Reason: no more need for intensive calculation of pseudorandom numbers
- Motivation: Reduction of energy consumption and execution time.
- Quality of the probabilistic output (randomness; statistical independence) must also be taken into account

Probabilistic System-on-a-Chip Architectures



[Chakrapani et al., p.6]

Competitors: NON-probabilistic architectures



[Chakrapani et al., p.6]

Comparison between different implementations

- Energy Performance Product:

  EPP = Energy Consumption x Execution Time

- Energy performance product gain:

  Ratio of the EPP of the baseline (β) to the EPP of the particular implementation (I)

$$\Gamma_{\mathcal{I}} = \frac{Energy_{\beta} \times Time_{\beta}}{Energy_{\mathcal{I}} \times Time_{\mathcal{I}}}$$

Application Level EPP gains Γ_I for different applications

Algorithm	$\Gamma_{\mathcal{I}}$	
	Min	Max
BN	3	7.43
RNN	226.5	300
PCA	61	82
HE	1.12	1.12

[Chakrapani et al., p10]

- Bayesian inference (BN)
- Probabilistic Cellular Automata (PCA)
- Random Neural Networks (RNN)
- Hyper Encryption (HE)

Factors influencing EPP gains

1. "Amount of opportunity", number of probabilistic steps in the application that use the PCMOS-based coprocessor.

$$Probabilistic\ Flux\ F = \frac{Number\ of\ probabilistic\ steps}{Total\ number\ of\ operations}$$

(application dependent)

2. Amount of gain afforded "per unit of opportunity"
(dependent on architecture and technology)

Analyzing EPP gains

$$
\begin{aligned}
Energy_{\beta} &= Energy_{det,\beta} + Energy_{prob,\beta} \\
&= Cycles_{det,host} \times Energy_{cycle,host} + Energy_{prob,\beta} \\
&= Cycles_{det,host} \times Energy_{cycle,host} + \mathcal{F} \times Cycles_{det,host} \times Energy_{flux0,\beta}
\end{aligned}
$$

$$
\begin{aligned}
Energy_{\mathcal{I}} &= Cycles_{det,host} \times Energy_{cycle,host} + \mathcal{F} \times Cycles_{det,host} \times Energy_{flux,\mathcal{I}} \\
&\approx Cycles_{det,host} \times Energy_{cycle,host}.
\end{aligned}
$$

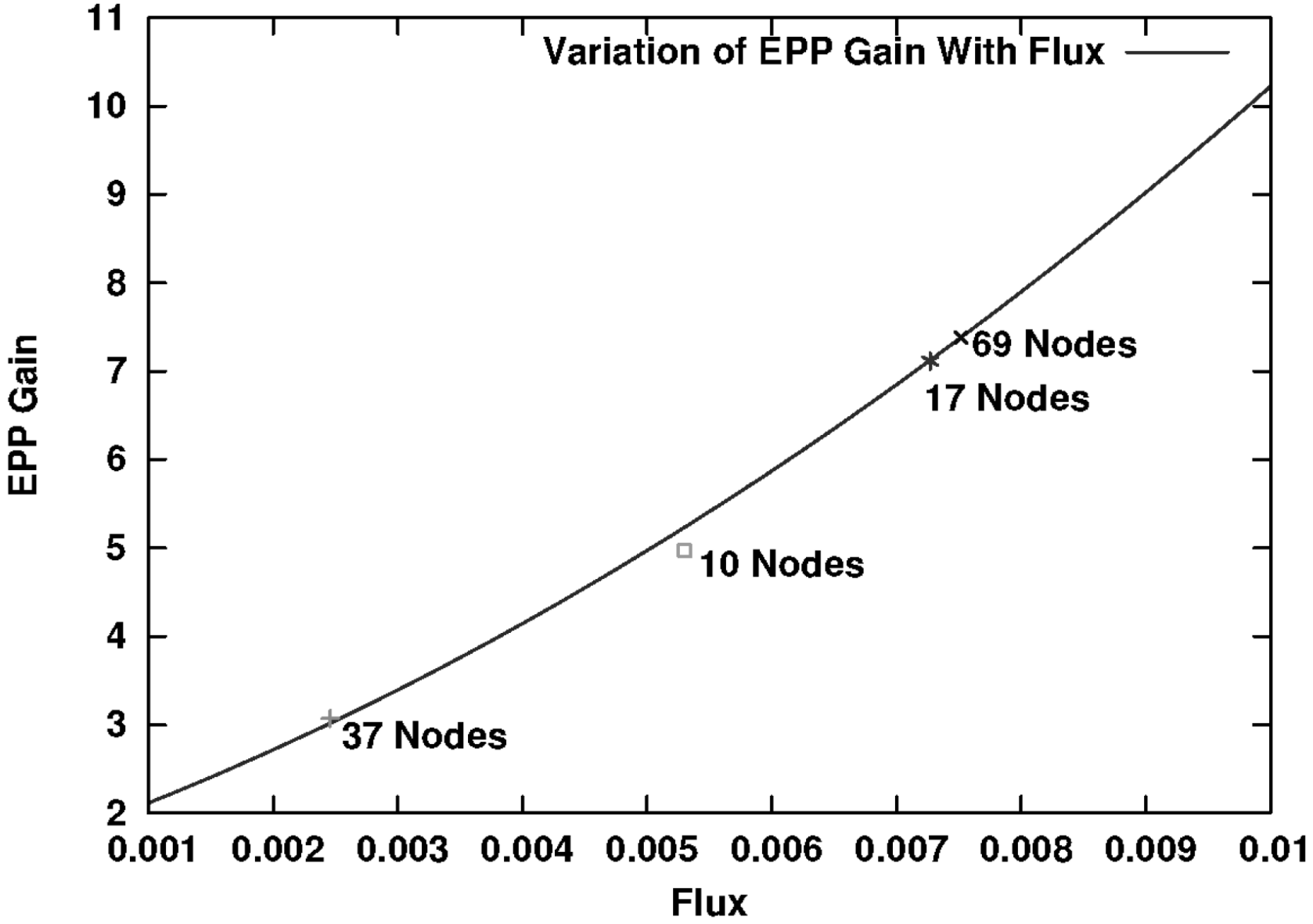
Amount of gain per core probabilistic Step [Chakrapani et al., p.13]:

Application	Gain Over SA-1100	Gain Over CMOS
BN	9.99×10^7	2.71×10^6
RNN	1.25×10^6	2.32×10^4
PCA	4.17×10^4	7.7×10^2
HE	1.56×10^5	2.03×10^3

Analyzing EPP gains

$$\Gamma_{\mathcal{I}} = \frac{Energy_{\beta} \times Time_{\beta}}{Energy_{\mathcal{I}} \times Time_{\mathcal{I}}}$$

$$= \left(1 + \frac{\mathcal{F} \times Energy_{flux,\beta}}{Energy_{cycle,host}}\right) \times \left(1 + \frac{\mathcal{F} \times Time_{flux,\beta}}{Time_{cycle,host}}\right)$$



(for a Bayesian Network) [Chakrapani et al., p.14]

Energy and Performance characteristics per core probabilistic step

- **PCMOS energy-efficiency** (0.4 pJ for a logical NOT)
- PCMOS specialization (avoiding Dilation)
- Replication factor R
  - Operating frequencies of PCMOS devices are at 1MHz
  - If a probabilistic algorithm needs a higher rate of random bits, the PCMOS building block has to be replicated
  - Optimization on application level is possible
  - Building blocks can't be turned off, take this into account when considering the EPP
- Communication Costs

Energy and Performance characteristics per core probabilistic step

- Spread Factor S: number of distinct probability parameters needed for an application.
  - Problem: Every distinct probability needs a distinct operating voltage:

$$p < 1 - 0.28\,e^{-1.275\frac{V_{dd}^2}{8\sigma^2}}$$

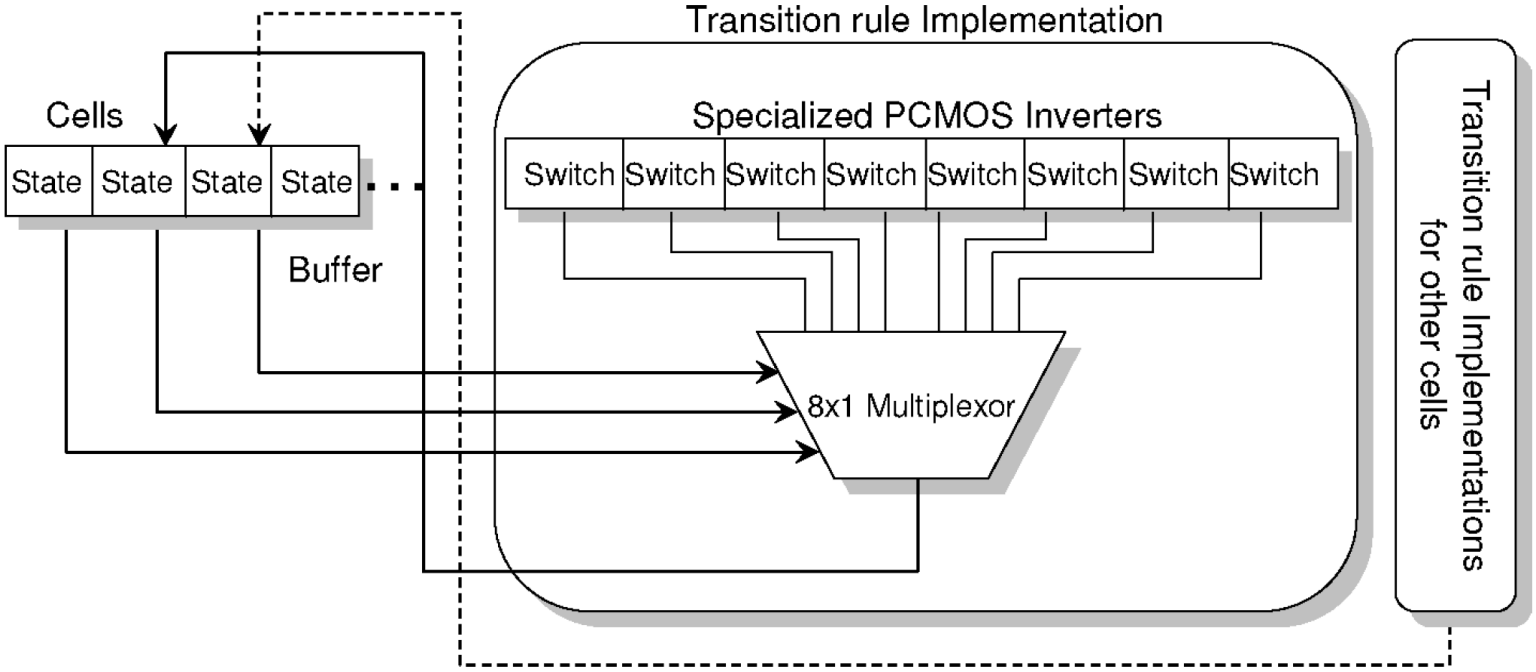
  - Optimization on application level: reducing the number of necessary distinct probability parameters (p=0.75, p=0.80, ~~p=0.85~~)
  - Optimization on architecture level: choosing a nonspecialized implementation if the spread factor is too high (p=0.75, p=0.80, ~~p=0.60~~)

Example for nonspecialized implementation

Application-Required Probability Parameters	Composition Tree
0.05	[[(0.4)AND(0.5)]AND(0.5)]AND(0.5)
0.10	[(0.4)AND(0.5)]AND(0.5)
0.15	[(0.5)AND[NOT(0.4)]]AND(0.5)
0.20	(0.4)AND(0.5)
0.25	(0.5)AND(0.5)
0.30	(0.5)AND[NOT(0.4)]
0.35	[NOT[(0.5)AND[NOT(0.4)]]AND0.5
0.40	0.40
0.45	[NOT[[(0.4)AND(0.5)]AND(0.4)]]
0.50	0.50

[Chakrapani et al., p.25]

Example for a probabilistic step: Probabilistic Cellular Automata



Next state (0 or 1) of a cell is probabilistic and depends on its current state and the state of the nearest two neighbors. (2^3=8 distinct probabilities) [Chakrapani et al., p.21]

Statistic Quality (Randomness)

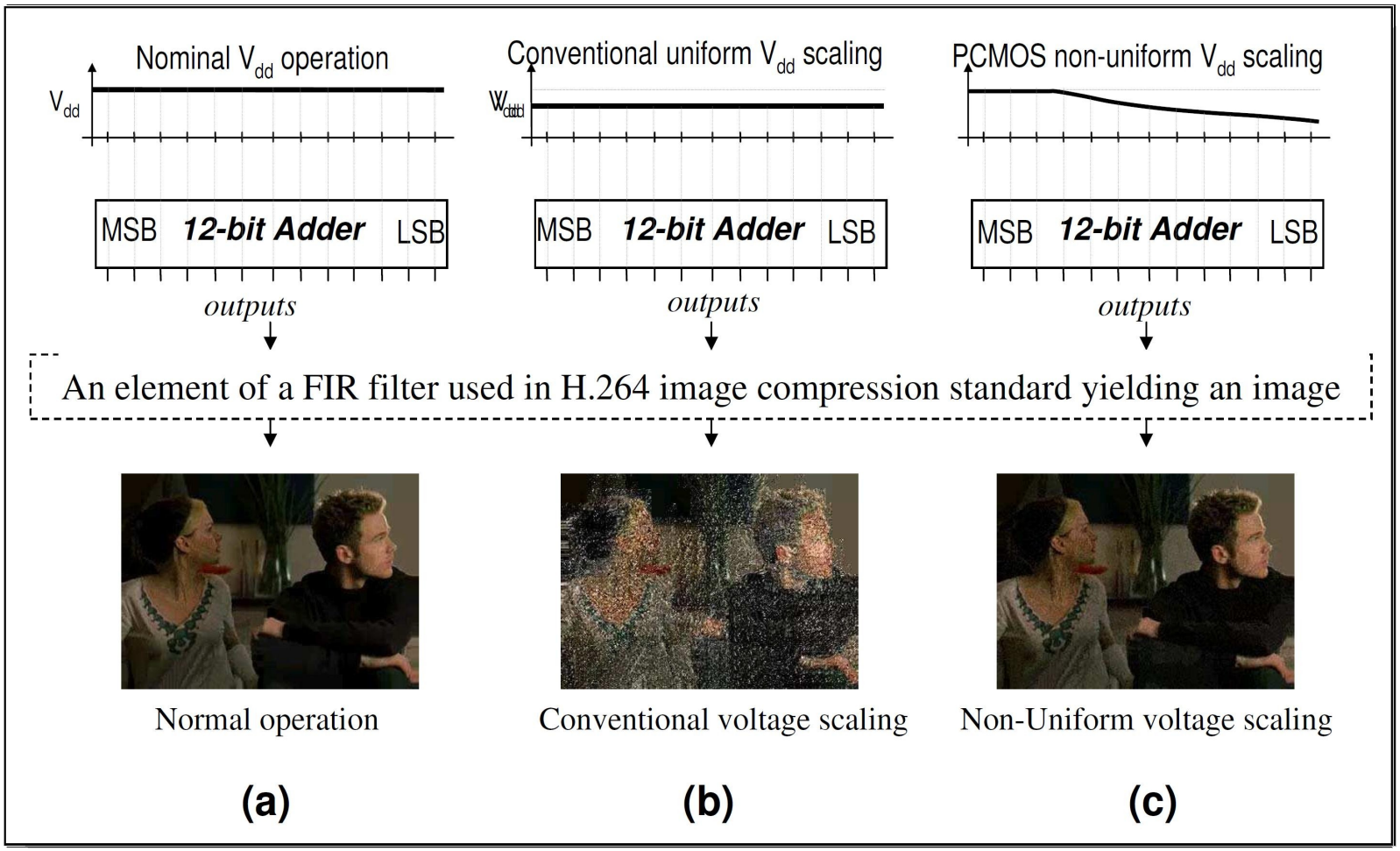
- 10101010101010101010101010101010
- 11101010101010101010101010101010
- Low quality bits affect applications (weaker encryption etc.)
- Theoretical approaches to examine the statistic quality are not practical
- But the quality of random bits generated with PCMOS devices has been tested with the randomness test from NIST suite with very favorable results

- Applications that tolerate probabilistic behavior

Those applications allow a Trade-off between Energy and Performance <=> Quality of the Solution

Example: Applications in Digital Signal Processing, because they already have to do a natural trade-off between Energy and Quality in the form of Signal-to-Noise-Ratio (SNR)

- Applications that tolerate probabilistic behavior



PCMOS in H.264 decoding [Akgul et al.]

Applications that do NOT tolerate probabilistic behavior

- Most Approaches are based on **redundancy** with **reliable arbitrators**
- Also: **Speculative execution** on faster (but less reliable) logic elements and verification by slower (and more reliable) logic elements

45 / 5 = 9 (on unreliable logic)

5 * 9 = 45 (correct)

Conclusion

- Modelling of probabilistic devices with regards to probabilistic behavior and energy consumption
- Probabilistic behavior can be harnessed for some applications with massive reduction of energy consumption and execution time