Design and Applications of Approximate Circuits by Gate-Level Pruning

Jeremy Schlachter, *Student Member, IEEE*, Vincent Camus, *Student Member, IEEE*,
Krishna V. Palem, *Fellow, IEEE*, and Christian Enz, *Senior Member, IEEE*

***Abstract*—Energy-efficiency is a critical concern for many systems, ranging from IoT objects and mobile devices to high-performance computers. Moreover, after 40 years of prosperity, Moore's law is starting to show its economic and technical limits. Noticing that many circuits are over-engineered and that many applications are error-resilient or require less precision than offered by the existing hardware, approximate computing has emerged as a potential solution to pursue improvements of digital circuits. In this regard, a technique to systematically trade off accuracy in exchange for area, power and delay savings in digital circuits is proposed: Gate-Level Pruning. A CAD tool is build and integrated into a standard digital flow to offer a wide range of costs-accuracy tradeoffs for any conventional design. The methodology is first demonstrated on adders, achieving up to 78 % energy-delay-area reduction for 10 % mean relative error. It is then detailed how this methodology can be applied on a more complex system composed of a multitude of arithmetic blocks and memory: the Discrete Cosine Transform (DCT), which is a key building block for image and video processing applications. Even though arithmetic circuits represent less than 4 % of the entire DCT area, it is shown that the Gate-Level Pruning technique can lead to 21 % energy-delay-area savings over the entire system for a reasonable image quality loss of 24 dB. This significant saving is achieved thanks to the pruned arithmetic circuits which sets some nodes at constant values, enabling the synthesis tool to further simplify the circuit and memory.**

***Index Terms*—Approximate computing, approximate adders, approximate circuit design, low-power digital circuits, IoT.**

I. Introduction

IMPROVING energy efficiency of modern computing systems is the main challenge in today's digital design. Computing capabilities of mobile devices such as smartphones has grown exponentially in the past decades, however battery technology did not follow the same evolution and autonomy is becoming a critical point. Additionally, the number of IoT (Internet of Things) devices is expected to reach 21 Billion by 2020 [1]. The latter not only require to operate for several years without user intervention, but will also produce a gigantic amount of data that will have to be processed in data centers which are extremely power hungry and need complex cooling systems.

In the past four decades, technology scaling has been leading integrated circuits' advancement. Unfortunately, the growing complexity of deeply-scaled technology combined with increasing PVT (Process-Voltage-Temperature) variations and the poor scaling of V_{th}, Moore's Law is starting to show its limits. Nonetheless, *approximate computing*—which can be applied through different abstraction layers ranging from technology, hardware design, up to algorithm or software level—is a potential solution to pursue the challenge of computing advancement and overcome the physical and economical limitations encountered with technology scaling.

The first attempts to trade exactness of computation against energy were published in the early 2000s and were referred as Probabilistic CMOS [2]–[4]. This theoretical approach relied on the fact that noise levels in future technologies would become significant, and would lead to an energy-correctness relationship where the amount of energy consumed to get a correct result grows exponentially. Hence, allowing a slight accuracy degradation would lead to significant energy savings. However, this approach is valid only if the power supply voltage is scaled down to the noise level, i.e. $\frac{kT}{q}$, which is currently not the case with any CMOS technology.

A different approach to potentially save energy is to exploit the quadratic relationship between supply voltage and power consumption, which consists in reducing the voltage below the critical point where timing errors start to occur [5]. This aggressive voltage scaling can be applied to data-path as well as memory, but the resulting errors are extremely difficult to predict and generally lead to an abrupt loss of functionality above a critical threshold. To overcome this issue, some authors proposed to apply non-uniform voltage scaling [6], [7] where most significant stages are powered with a higher voltage than least significance stages. However, the possible savings would be masked by the overhead of multiple voltage panes and by the deterioration of the carry-chain computation.

Due to the ever-increasing PVT variations, huge safety margins are required to guarantee the circuit's functionality among all corners, in particular for the worst case. This leads to drastic area and energy penalties. One of the most famous ways to get rid of these margins and gain back the wasted performances, is to use dual-latching methods such as the Razor flip-flop [8] or Adaptive Voltage Over-Scaling (AVOS) [9] to detect timing errors, and allow for an extra clock cycle to ensure error free operation if needed. A slightly different approach consists in preventing timing errors rather than detecting them by making critical paths rare and predictable, and by allowing a two-cycle operation when the critical path is activated [10].

The main drawback of the previously mentioned techniques is that they require hardware overhead, such as error recovery

This work was supported by the Swiss National Science Foundation grant No 200021-144418.

J. Schlachter, V. Camus and C. Enz are with the Integrated Circuits Laboratory (ICLAB), Ecole Polytechnique Fédérale de Lausanne (EPFL), Neuchatel, Switzerland, (e-mails: jeremy.schlachter@epfl.ch; vincent.camus@epfl.ch; christian.enz@epfl.ch).

K. V. Palem is with the Department of Computer Science, Rice University, Houston, TX, USA.

circuitry and additional voltage domains. A different approach consists in modifying the functionality of the circuit to trade a limited amount of accuracy against significant power, area and delay savings without any overhead. Arithmetic circuits are particularly good candidates for this kind of approach thanks to the notion of bit significance. For instance, Gupta *et. al* reduced the number of transistors of the mirror adder to build approximate full-adder cells [11]. For multiplier circuits, Karnaugh maps of 2x2 multipliers [12] and 4:2 compressors [13] can be simplified, reducing the cell area and energy consumption while leading to rare and limited errors.

At architectural level, the bio-inspired adder [14] simplifies the LSB stages by simple OR gates. Another method consists in relaxing the timing constraints on the critical path of the adder, by splitting the carry chain like in speculative adders [15]–[17] or by transforming it into a false path as in the Carry Cut-Back (CCB) adder [18]. The Dynamic Range Unbiased Mulitiplier (DRUM) [19] features a dynamic-range selection scheme, which is essential for general purpose circuits.

More systematic approaches have also been proposed, for instance *Probabilistic Logic Minimization* [20] where bit-flips are introduced in Karnaugh maps to simplify logic functions, or *Probabilistic Pruning* where full adder cells are pruned out of adders. Nevertheless, for all these techniques the amount of inaccuracy is set at design time and cannot be changed. However, none of those techniques have been automatized and fully integrated in a standard digital flow, allowing the designer to choose among a multitude of energy-accuracy tradeoffs by adding only one step in a standard digital flow.

This paper further investigates the pruning methodology [21] by applying it at gate-level and by automatizing and integrating it in a standard digital flow as presented in [22]. Section II introduces the Gate-level Pruning (GLP) technique and describes the tools that have been built to automatize the pruning process. Section III evaluates the proposed methodology on adders, which are key building blocks of computing systems, and investigates the errors resulting from the pruning. Finally, section IV demonstrates how the GLP technique can be applied the Discrete Cosine Transform, a hardware accelerator used in many image and video processing application. In this work, all circuits are synthesized with the same UMC 65 nm technology.

II. Automatic Gate-Level Pruning

Probabilistic pruning is a design technique that consists of removing circuits blocks and their associated wires in order to trade exactness of computation against power, area and delay savings without any overhead. The amount of pruning is dictated by the application's error tolerance. A formal definition of probabilistic pruning, as well as the proof of concept, have already been addressed in [21]. The following paragraphs expose the key points necessary to build an automatic Gate-Level Pruning tool using existing CAD software, and compares different pruning criteria.

A. Significance and Activity ranking

A circuit netlist as depicted in Fig. 1 can be represented by a directed acyclic graph, where the nodes are components such as gates, and whose edges are wires. The decision to prune a node is generally based on two criteria: the significance, which is a structural parameter, and the activity or toggle count. The nodes with the lowest significance-activity product (SAP) are pruned first. By doing so, the error magnitude grows with the amount of pruning. Alternatively, depending on the application's requirements, the designer may choose to prune nodes according to the activity only in order to minimize the error rate, or by significance only in order to shorten design time by skipping the gate-level simulation process.

The activity of each wire is extracted from the .SAIF file (Switching Activity Interchange Format) obtained through gate-level hardware simulations. This file contains the toggle count (TC) of each wire, as well as the time spent at the logic levels 0 and 1 (T0 and T1) respectively. While TC is used to rank the nodes, T0 and T1 are used later in the pruning process to set unconnected gate inputs to a specific value. Note that to get an accurate activity estimation, the system should be simulated with an input stimulus representative of the *real operation* of the circuit.

The significance of each primary output can be set by the designer depending on the application's requirement. However, the experiments performed on adders and multipliers in this paper assume an automatic weighted significance attribution, where each bit position has a significance 2 times higher than the previous when moving from the LSB to the MSB. Reverse topological graph traversal is then performed to compute each nodes' significances as follows:

$$\sigma_i = \sum \sigma_{desc(i)} \tag{1}$$

where σ_i is the significance of the node i and $\sigma_{desc(i)}$ is the significance of the direct descendants of node i. An example of weighted significance attribution is shown in Fig. 1.

B. Pruning

Once the nodes are ranked according to their significance-activity product, significance only or activity only, the gate-level netlist is modified in order to remove *unessential* nodes from the design. For the sake of simplicity, and in order to maximize the use of the existing EDA tools, the probabilistic pruner does not literally remove the gates form the netlist, but it disconnects the corresponding wires. Gates whose outputs are unconnected will automatically be removed by the synthesis tool. However, leaving gate inputs unconnected would fail the re-synthesis of the design. For this reason, and in order to minimize the error, those inputs are set to 0 if they statistically spend most of the time at 0 (i.e. T0 $\geq$ T1). Otherwise they are connected to 1 (i.e. T0 $<$ T1). This should allow to statistically reduce the error magnitude. The synthesis of the modified netlist therefore improves the design in two ways:

- One or more gates having their outputs unconnected are removed, allowing direct area, power and delay savings.
- Gates having their inputs set to 1 or 0 can then be replaced by lower complexity ones.

Furthermore, the resulting circuit is optimized for the timing and area constraints set by the designer. Fig. 2 shows the functional diagram of the presented pruning tool. The initial

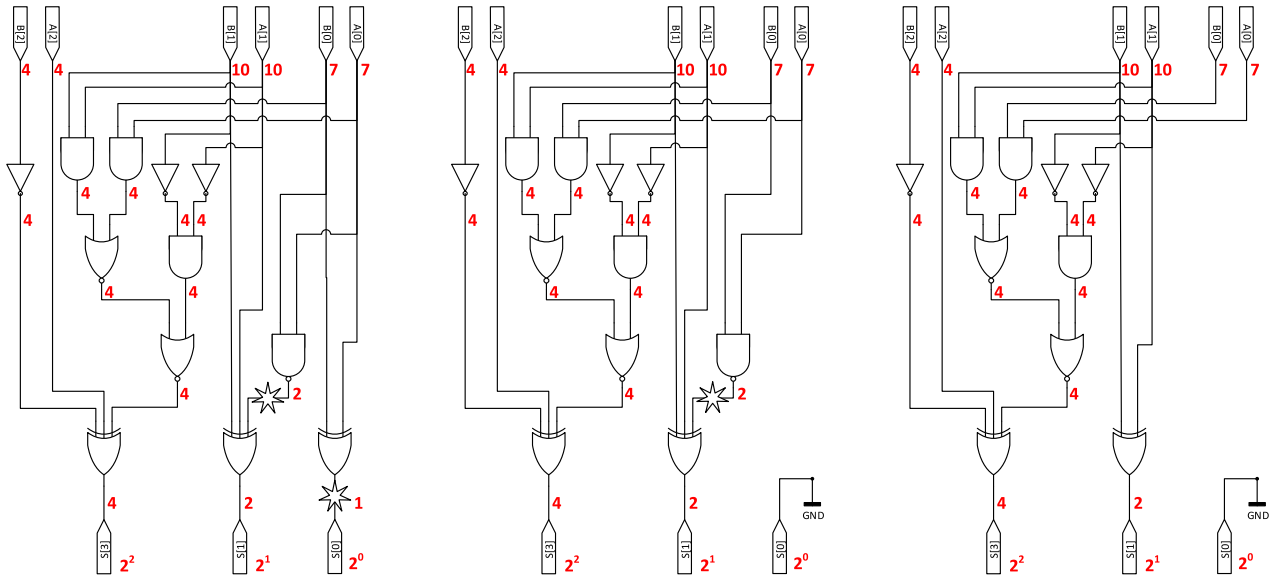


Fig. 1: Gate-level netlists of a 3-bit adder and the associated significance attribution (a), the stars indicate the nets that are pruned first. The same netlist with one pruned node (b), and two pruned nodes (c).

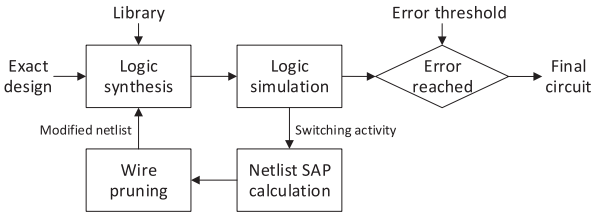


Fig. 2: CAD framework for Gate-Level Pruning.

design is synthesized and mapped to a technology in order to get the gate level netlist. This netlist then enters a pruning loop composed of four steps:

1) Hardware simulation to monitor the activity of the circuit and to check if the amount of error introduced by the pruned netlist can still fit the application.
2) The significance-activity product is calculated depending on the designer's requirements (weighted or uniform pruning).
3) Wires are pruned according to the ranking of the nodes.
4) Re-synthesis of the netlist is performed in order to remove or replace non-essential gates.

Synthesis and hardware simulations are performed using existing software, whereas scripting languages are used for SAP calculation and wire pruning. This framework outputs all the gate-level netlists ranked by growing order of inexactness, i.e., by decreasing energy-delay-area product. A significant advantage of the proposed tool and methodology is that they can be embedded in an existing standard digital flow, making them fully compatible with any synthesizable HDL code. Moreover, that same flow can be used indifferently for inexact ASIC or FPGA design.

Fig. 1 illustrates the netlists provided by the automatic pruning tool for a 3-bit adder. Fig. 1a is the conventional circuit where the significance of each node is indicated in red. The two first wires to be pruned, i.e. the ones with the lowest significance, are indicated by stars. The approximate circuit with one pruned node is obtained as follow: the wire with a significance of '1' is disconnected and the primary output S[0] is connected to ground assuming it statistically spends most of the time at the logic level '0'. As shown in Fig. 1b, the XOR gate preceding the output S[0] can be removed as it becomes useless. Similarly, the circuit with 2 pruned nodes is obtained by disconnecting the net having a significance of '2' from the circuit 1b. This operation has multiple advantages: the NAND gate can be removed and the 3-inputs XOR gate can be replaced by a 2-inputs XOR. Since the least significant output is connect to ground, this approach could be mixed up with truncation or bit-width reduction, however the main difference here is that the least significant inputs are still used to the calculation, and the carry chain remains intact so that the MSBs of the sum remain exact.

III. Pruned Arithmetic Circuits

A. Error characterization and metrics

In order to get an accurate error characterization of arithmetic circuits, extensive simulations need to be performed. To cover all the possible cases, a 64-bit adder would have to be simulated with 2^{128} different input combinations which is not possible within a reasonable time. Moreover, the simulation time would need to be multiplied by the number of approximate operators generated by the pruning tool.

Approximate adders are commonly characterized and validated through the simulation of random sets of inputs. Hence, using a set of five million uniformly distributed random inputs allows to get a fairly good estimate of the error characteristics

within a reasonable simulation time, but the presented results are statistical estimations depending on the random sample distribution.

The metrics used to characterize approximate adders in this work are based on the relative error (*RE*), defined as:

$$RE = \left| \frac{S_{approx} - S_{correct}}{S_{correct}} \right| \quad (2)$$

where S_{approx} and $S_{correct}$ are respectively the approximate and correct sums of an addition. Two interesting metrics are considered:

- *Error Rate* – The error rate corresponds the ratio of erroneous computations over the entire set of computations and is defined as follow:

$$Error\ Rate = \frac{Number\ of\ erroneous\ computations}{Total\ number\ of\ computations} \quad (3)$$

- *Mean Relative Error* (*MRE*) – The mean of *RE* is a good estimator of accuracy over a given set of inputs and is interesting at the application level, where for example a few erroneous pixels over an image might have a limited visual impact. It is defined as:

$$MRE = \frac{1}{N} \sum_{n=1}^{N} RE \quad (4)$$

where N is the total number of computations.

The performances of each implementation are evaluated based on energy consumption, silicon area and critical path delay. The Energy Delay Area Product (EDAP) is used as a figure of merit to compare each circuit implementation.

B. Adders implementation

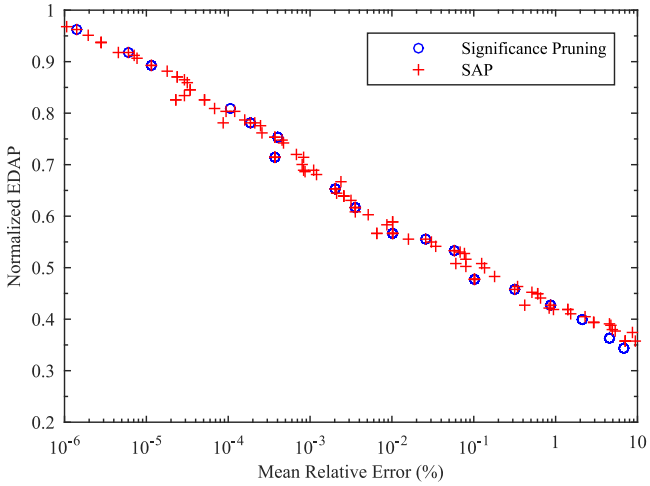
In previous works [21], *Probabilistic Pruning* has been applied manually on several traditional 64-bit adder architectures such as Kogge-Stone and Han-Carlson. However, it is very rare that the designer selects one of these specific architectures. In fact, arithmetic operations are implemented with high-level description languages (HDL) and the designer does not specify the architecture. Low-level structural details are handled by the synthesis tool which selects the optimal architecture based on many optimization scripts and arithmetic IP libraries to fit the given design constraints.

One of the key strength of the proposed tool is that it is able to prune *any digital circuit*, particularly those produced by behavioral description in HDL codes, the only condition being that the HDL code is synthesizable.

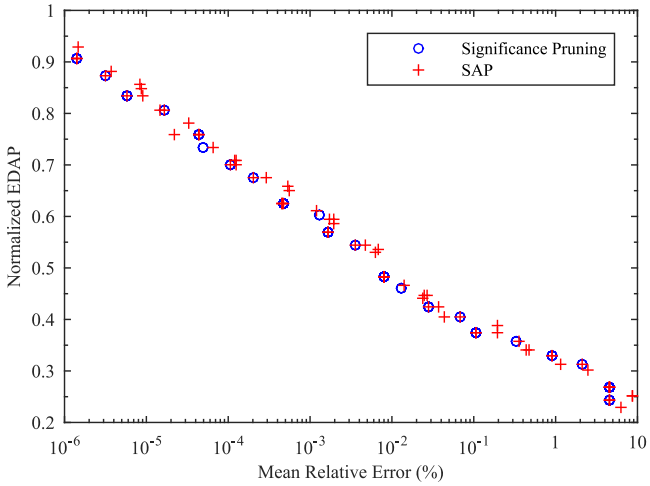
In addition, the previous work [21] exposes the pruning of 64-bit adders, but this approach is a bit too optimistic since highly pruned 64-bit adders could certainly be replaced by 32-bit adders. Moreover, a random uniform distribution over 64 bits features mostly very large numbers and even errors at the bit position 32 almost have no impact on the MRE. The two techniques, GLP and the previous work are compared on a 64-bit adder basis in Table I. It is shown that the finer granularity of the GLP enables much higher savings for equivalent Mean Relative Error.

TABLE I: Comparison of the two pruning techniques for 10 % MRE

Pruning Technique	Area gains	Energy gains	Delay gains	EDAP gains
Gate-level	21X	7.82X	1.07X	175X
Previous work [21]	1.8X	1.8X	2.3X	7.5X



(a) Pruned 32-bit adders at 3.3 GHz



(b) Pruned 32-bit adders at 1.25 GHz

Fig. 3: Normalized savings of pruned 32-bit adders synthesized at (a) 3.3 GHz and (b) 1.25 GHz

C. Significance and SAP-based pruning

Since 32-bit adders are more common and wide spread, this work focuses on this bit-width. Fig. 3 shows the savings of pruned 32-bit adders, for two different timing constraints: 3.3 GHz and 1.25 GHz. Here, the synthesis tool generates the best architecture for each timing constraint, providing an optimized netlist. The two most efficient types of pruning for arithmetic circuits are compared: Significance pruning (circles) and SAP pruning (crosses). Its is shown in Fig. 3 that both pruning types can provide similar savings for a Relative Error Magnitude of 10 %: 64 % EDAP reduction at 3.3 GHz and up to 78 % EDAP reduction at 1.25 GHz. In some cases, the SAP driven pruning and the Significance driven pruning lead to exactly the same circuit. However, SAP pruning offers a larger

range of tradeoffs compared to Significance based pruning, i.e. there is a higher number of pruned designs satisfying a similar error specification when using SAP pruning. This is particularly true for larger circuits. This means that Significance based pruning can be used for a fast first design, and SAP pruning can be used for fine tuning if required. It should be noted that estimating the switching activity of each gate is particularly time consuming as it requires gate-level simulations. Obtaining the SAP pruned netlists for a single 32-bit adder can therefore take up to 15-20 minutes with a set of 5 million inputs, whereas the Significance pruned netlists can be obtained in less than 30 seconds on an Intel Core i7-4770 processor equipped with 16 GB of memory.

Gate-Level Pruning can be applied to circuit synthesized under any frequency constraint, but this parameter influences a lot the savings obtained. Indeed, high frequency adders, such as those presented in Fig. 3, are generally large circuits featuring expensive parallelism. Removing a small portion of this kind of circuit poorly affects the correctness of the results and can lead to significant savings. On the other hand, adders synthesized at low frequency, which turn out to be ripple carry adders built from a chain of full adder cells, are bad candidates for Gate-Level Pruning. Those adders can be found in the 300 to 500 MHz range for this technology. Due to their serial architecture, pruning would rapidly break the carry chain and lead to large errors. Even though a few percent power and area savings are possible, it does not really make sense since the ripple carry adder is one of the most power efficient architecture. Moreover, the savings achieved would be imperceptible at the system level due to their small size. It should be noted that among high frequency adders, the best pruned circuit is not always the one with a higher frequency, and Fig. 3 illustrates this fact: the pruned adders at 1.25 GHz have a slightly lower normalized EDAP than the ones at 3.3 GHz.

D. Error distributions

A key property of approximate adders in order to enable their wide-spread use is that their failures have to remain small, at least relative to the expected exact results. In this regard, the error distributions of pruned adders have been investigated with a set of five million uniformly distributed random inputs. Fig. 4 plots the error distribution of 32-bit adders synthesized at a frequency of 1.25 GHz with 10, 20 and 30 nodes pruned. Those pruning levels correspond to 12.5 %, 14.3 % and 37.6 % EDAP savings respectively. It is of particular interest to note that the error with the highest occurrence are low relative errors. In other words, the highest relative errors are the ones with the lowest occurrence, ensuring a *fail safe* behaviour. It can be observed that for a small number of pruned nodes Fig. 4a, most of the errors remain below 10^{-2} %, and as the number of pruned nodes increases, the shape of the distribution remains the same but errors are shifted towards higher magnitudes (Fig. 4b and 4c).

E. Activity based pruning

For some applications, the error rate might be more important than the error magnitude. That is to say that only the number of

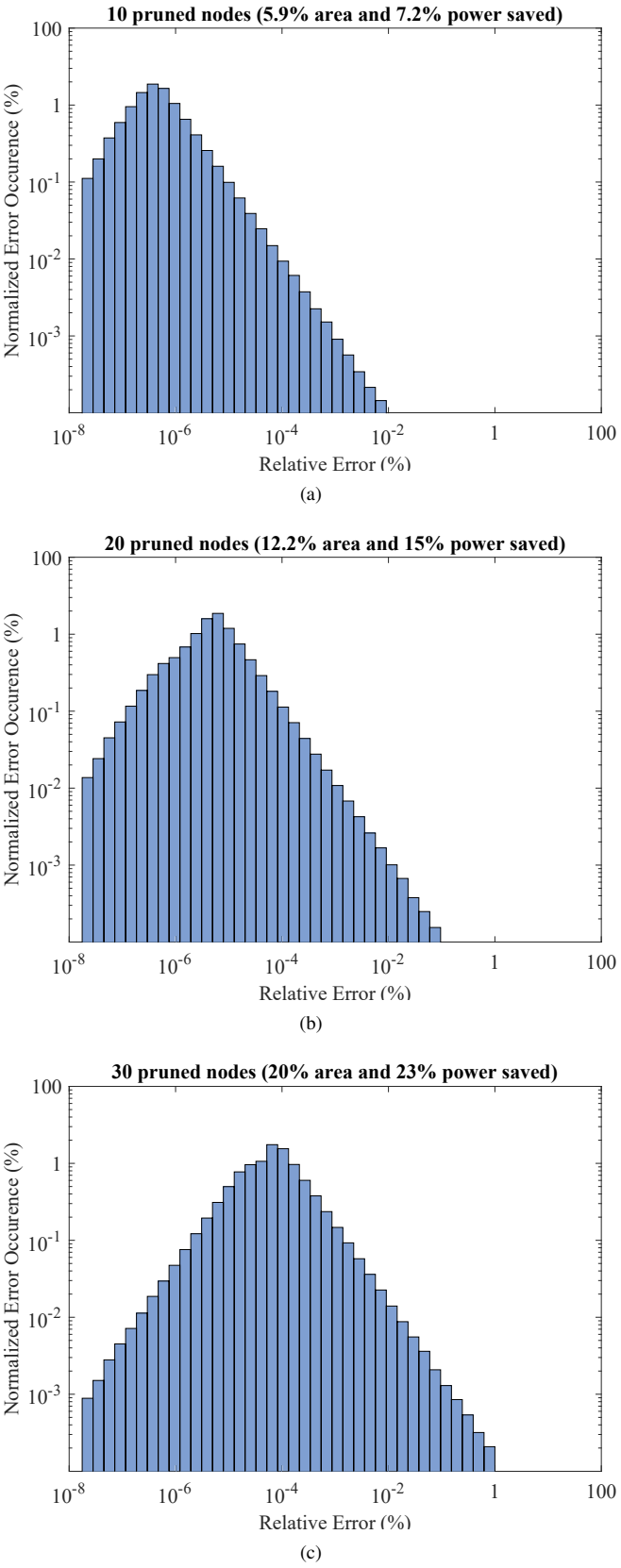


Fig. 4: Error distribution of 32-bit adders at 1.25 GHz

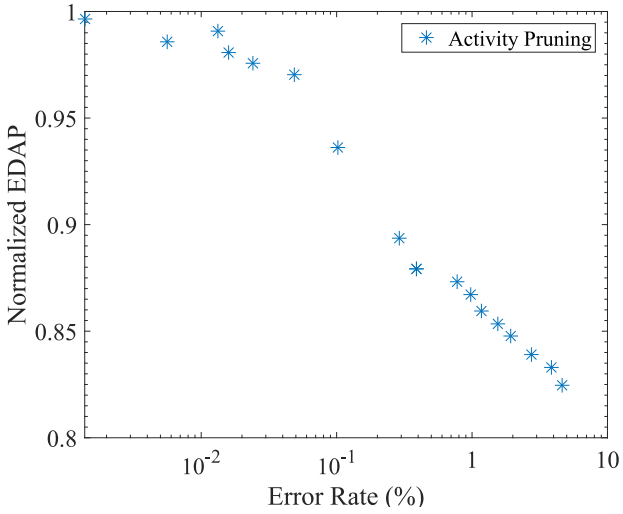


Fig. 5: Activity based pruning of 32-bit adders at 3.3 GHz

errors matters, regardless of their magnitude. For this reason, the error rate is used to characterize activity based pruned adders. In this case, only activity should be considered to rank the nodes for pruning. Fig. 5 shows an activity based pruning of a 32-bit adder at a frequency of 3.3 GHz. The number of possible energy-accuracy tradeoffs is much smaller compared to SAP and significance pruning, but it is still possible to save up to 18 % EDAP for an error rate inferior to 10 %. In the specific case of an adder, the gates close to the MSB are generally pruned first since they are the ones with the lowest activity as they are at the end of the carry chain. For this reason, this pruning methodology leads to very high error magnitudes when applied on arithmetic circuits. Nevertheless, it could be useful for circuits where there is no notion of bit significance.

F. Remarks

This automatic Gate-Level pruning tool is fully integrated in the standard digital flow. This provides the designer a wide range of energy-accuracy tradeoffs for arithmetic circuits and more generally for any combinational circuit as demonstrated in [23], [24]. This work however does not address formal verification, which is generally very challenging for any approximate circuit. The functionality of the circuit is tested by gate-level simulations, which is a cumbersome and time consuming process. To enable the industrial use of approximate circuits and to speed-up design time, new verification techniques would have to be developed.

IV. Pruned Discrete Cosine Transform for Image Processing

The previous section has shown that the GLP enables large power and area savings when applied to arithmetic circuits. However, one single adder generally only represents a tiny fraction of the area and power consumption of the system it is placed in. For this reason, even 50 % power and area savings achieved on a single adder could turn out to be insignificant at system level, and would not justify the quality loss. Nevertheless, this approach becomes more interesting at the level of a hardware accelerator dedicated to one specific task and which is built out of multiple arithmetic circuits. In this regards, this section analyses how the GLP can be applied simultaneously on several adders and subtractors used to build a Discrete Cosine Transform (DCT), which is one of the most computationally intensive element for many image and video processing compression algorithms such as JPEG or MPEG. This work does not claim or present a novel type of DCT, but it demonstrates how energy-quality tradeoffs can be achieved by applying inexact design techniques, such as GLP, on existing state-of-the-art architectures.

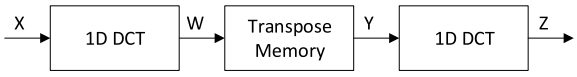


Fig. 6: 2D DCT architecture based on 1D stages

A. Conventional DCT

DCT algorithms and architectures have been extensively studied in the literature. Even error resilient DCTs have already been proposed [25], [26]. Image encoding algorithms used for instance in JPEG encoding generally compute the DCT per pixel blocks. The following work considers the example of 8x8 pixel blocks DCT, but could be extended to other block sizes and architectures. Efficient implementations are generally based on distributed arithmetic computations [27], and is taken as starting point for the following example, but the proposed methodology could be applied to any existing architecture to trade accuracy of computation against significant area and power savings.

A 2D DCT used in image encoding can be split in two single stage DCTs interleaved with transpose memory as shown in Fig. 6. The 8-point 1D-DCT w_k of a data sequence x_i is defined by

$$w_k = \frac{a_k}{2} \sum_{i=0}^{7} x_i \cos\left[\frac{(2i+1)k\pi}{16}\right] \quad (5)$$
$$\text{with } a_k = \begin{cases} 1/2, & k = 0 \\ 1, & k = 1...7 \end{cases}.$$

This can also be expressed in its matrix form as

$$W = T \cdot X, \quad (6)$$

where T is an 8 x 8 matrix in the case of an 8 point DCT and X and W are row and column vectors. Using the symmetry property of T, (6) can be decomposed as follow for even / odd 1D DCT calculations

$$\begin{bmatrix} w_0 \\ w_2 \\ w_4 \\ w_6 \end{bmatrix} = \begin{bmatrix} c_4 & c_4 & c_4 & c_4 \\ c_2 & c_6 & -c_6 & -c_2 \\ c_4 & -c_4 & -c_4 & c_4 \\ c_6 & -c_2 & c_2 & -c_6 \end{bmatrix} \begin{bmatrix} x_0 + x_7 \\ x_1 + x_6 \\ x_2 + x_5 \\ x_3 + x_4 \end{bmatrix} \quad (7)$$

$$\begin{bmatrix} w_1 \\ w_3 \\ w_5 \\ w_7 \end{bmatrix} = \begin{bmatrix} c_1 & c_3 & c_5 & c_7 \\ c_3 & -c_7 & -c_1 & -c_5 \\ c_5 & -c_1 & -c_7 & c_3 \\ c_7 & -c_5 & c_3 & -c_1 \end{bmatrix} \begin{bmatrix} x_0 - x_7 \\ x_1 - x_6 \\ x_2 - x_5 \\ x_3 - x_4 \end{bmatrix} \quad (8)$$

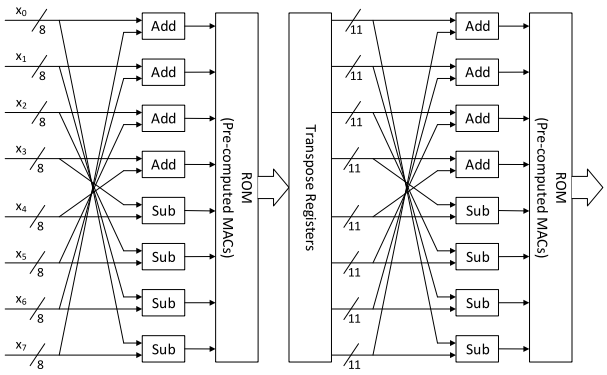


Fig. 7: Architecture of the 8 x 8 2D DCT.

where $c_k = \cos(\frac{k\pi}{16})$. It can be seen from (5) that the DCT is computationally intensive, and requires a large amount of multiplications which are power hungry. Plenty of DCT architectures have been proposed in the literature. However, since the scope of the paper is to improve energy-efficiency, a low power multiplier-less DCT architecture based on row-column parallel distributed arithmetic has been chosen. Fig. 7 shows this implementation of the 8 x 8 2D DCT where only 4 adders and 4 subtractors are required to compute the right part of (7) and (8). The final 1D DCT is obtained by looking-up pre-computed multiply and accumulate (MAC) coefficients stored in a Read-Only Memory (ROM).

B. Quality testing

Fig. 8 sketches the test setup used to characterize the DCT for image processing. First, the DCT of an image sample is computed with the hardware under test. Image is then reconstructed using a behavioral inverse transform, i.e. with infinite precision. The quality of the reconstructed image compared to the original image is evaluated by calculating the Peak Signal-to-Noise Ratio (PSNR) between the two images as follow:

$$\text{PSNR} = 10\log_{10}\left(\frac{D^2}{\text{MSE}}\right) \tag{9}$$

where MSE is the mean squared error between the original and the reconstructed image and D is the maximum possible pixel value, here 255, considering 8-bit pixel representation. With a sample *Lena* picture transformed by the conventional 2D DCT shown in Fig. 7, the PSNR is equal to 48 dB. Image quality is limited mainly due to the use of fixed point arithmetic. As conventional designs are already lossy, it can be acceptable to trade some more accuracy in exchange for power and silicon area savings.

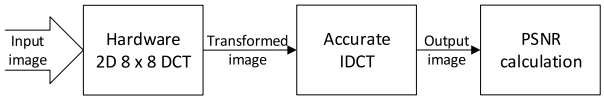


Fig. 8: Test setup for quality measurement

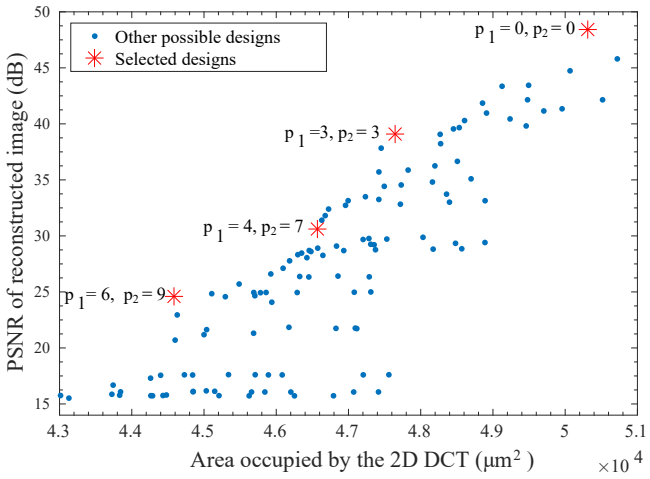


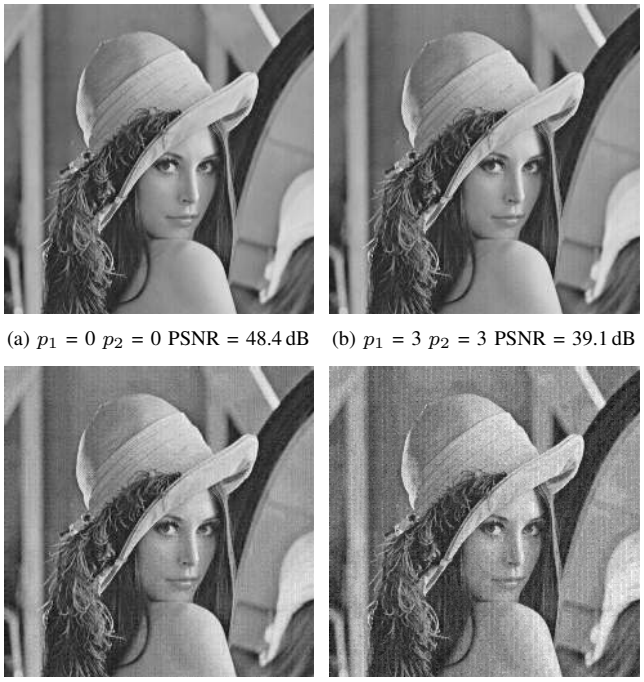
Fig. 9: Image quality versus circuit area

C. Pruning methodology

The 2D DCT described in section IV-A has been synthesized with an industrial 65 nm technology at clock frequency of 1.25 GHz. The resulting circuit is used as a reference to apply the Gate-Level pruning to each of the 16 adders and subtractors. Seeing that each of these components have slightly different architectures due to differences in timing paths, and considering that the switching activity differs from one to another, pruning is applied individually on each of the 16 operators. Besides, each can have a different impact on the final error bound. It is consequently required to explore the design space to find out the best possible combination of inexact adders in order to minimize the quality loss and maximize the savings. The synthesized adders and subtractors are built out of 45 standard cells in average. It is therefore worth pruning up to 10 nodes for fine-tuning the accuracy. Higher pruning would dramatically degrade the image quality. For 10 levels of pruning considered per adder and subtractor (the exact operator plus 10 pruned ones), there are 11^{16} possible design combinations. For practical reasons such as computing resources, it is clearly not possible to run 11^{16} synthesis and hardware simulations to find out the optimal design.

A good solution to narrow the design space is to apply the same level of pruning p_i to each adder and subtractor inside a given stage i. As the bit-width is the same within a stage, the degradation of arithmetic accuracy is progressive. With this approach, there are $11^2 = 121$ possible combinations left.

Synthesis shows that the area occupied by the 16 adders and subtractors depicted in Fig. 7 represent a small part of the entire conventional 2D DCT area. Hence, a simple swap between exact and approximate operators would lead to very limited savings. Nevertheless, re-synthesizing the full design with pruned operators eliminates unused ROM and un-necessary registers thanks to logic simplification and constant propagation implemented in the synthesis tool. This results in attractive power and area savings.



(a) p_1 = 0 p_2 = 0 PSNR = 48.4 dB (b) p_1 = 3 p_2 = 3 PSNR = 39.1 dB

(c) p_1 = 4 p_2 = 7 PSNR = 30.6 dB (d) p_1 = 6 p_2 = 9 PSNR = 24.6 dB

Fig. 10: Pictures of Lena resulting from the test setup using the conventional DCT (a) and the approximate versions (b,c,d). p_i denotes the number of pruned nodes per adder and subtractor in stage i.

D. Results

Fig. 9 shows the image quality versus area savings for the implemented DCTs. Each point corresponds to a combination (p_1, p_2) in $[0, 10]^2$. This figure highlights the broad diversity of design options offered using this methodology. For a given image quality requirement, pruning of operators in such a complex system allows to precisely match design specifications with an optimal circuit efficiency.

Keeping in mind that the goal of approximate circuits is to trade a little accuracy for the maximum area and power savings, only designs along the upper envelope of the plot in Fig. 9 are of interest since they maximize the gains with minimum quality loss.

Fig. 10 shows reconstructed *Lena* pictures obtained from four selected DCT implementations (the red stars in Fig. 9 highlight those designs). Conventional DCT has been used for Fig. 10a, while the three others have been obtained using three pruned designs representative of the area-accuracy tradeoff plotted in Fig. 9. On the one hand, it is possible to save up to 12 % area at the cost of almost imperceptible errors. On the other hand, for designs achieving the highest area reductions, artefacts start to appear on the edges of the 8x8 pixel blocks.

For the selected designs, power consumption is estimated based on gate-level simulations monitoring switching activity of the *Lena* picture processing. Results are summarized in Table II. Despite adders and substractors represent less than 4 % of the overall DCT area, re-synthesis of the design with pruned operators enables larger savings over the entire system, as explained in Section IV-C. For the case ($p_1 = 6$, $p_2 = 9$), the arithmetic area is reduced by 28 % and the arithmetic power consumption is reduced by 46 %. Finally, the re-synthesis of the design with the pruned operators leads to 21 % energy-delay-area savings over the entire DCT. This significant overall saving is obtained thanks to the pruned arithmetic circuits which sets some nodes at constant values, enabling the synthesis tool to further simplify the circuit and memory.

TABLE II: Power, area and quality of the 4 selected DCTs

Pruning level	PSNR (dB)	Normalized area			Normalized Power
		Arithmetic	Memory	**Total**	
p_1 = 0 p_2 = 0	48.4	1	1	**1**	**1**
p_1 = 3 p_2 = 3	39.1	0.94	0.95	**0.94**	**0.96**
p_1 = 4 p_2 = 7	30.6	0.82	0.93	**0.92**	**0.94**
p_1 = 6 p_2 = 9	24.6	0.72	0.89	**0.88**	**0.90**

V. Conclusion

This paper presented a methodology and a CAD tool integrated in a standard digital flow to automatically trade a determined amount of accuracy in exchange for area power and delay savings. This Gate-Level Pruning method can be applied to any combinational circuit, but more particularly on arithmetic circuits in which there is a notion of bit significance. While gains achieved on adder circuits are already interesting, up to 78 % EDAP reduction at 10 % MRE for 32-bit adders, those could be insignificant since an adder generally only represents a small fraction of the system it is placed in. It is therefore interesting to apply the Gate-Level Pruning to a hardware accelerator such as the DCT with the state-of-the-art distributed arithmetic architecture, which is built out of multiple arithmetic circuits and memory. In this case the EDAP gains achieved for the specific adders and subtractors reach 46 % for an image quality loss of 24 dB. Despite the latter arithmetic circuits occupy less than 4 % of the total DCT area, the re-synthesis of the entire DCT with pruned operators leads to 21 % EDAP savings over the entire accelerator. This significant overall saving is obtained thanks to the pruned arithmetic circuits which sets some nodes at constant values, enabling the synthesis tool to further simplify the circuit and memory. This Pruning technique is therefore well suited to the design of VLSI circuit for IoT applications where silicon costs and energy consumption are the main targets.

References

[1] S. Jankowski, J. Covello, H. Bellini, J. Ritchie, and D. Costa, "The internet of things: Making sense of the next mega-trend," *Goldman Sachs*, 2014.

[2] K. V. Palem, "Energy aware computing through probabilistic switching: a study of limits," *IEEE Transactions on Computers*, vol. 54, no. 9, pp. 1123–1137, Sept 2005.

[3] S. Cheemalavagu, P. Korkmaz, K. V. Palem, B. E. Akgul, and L. N. Chakrapani, "A probabilistic cmos switch and its realization by exploiting noise," in *IFIP International Conference on VLSI*, 2005, pp. 535–541.

[4] P. Korkmaz, B. E. Akgul, K. V. Palem, and L. N. Chakrapani, "Advocating noise as an agent for ultra-low energy computing: probabilistic complementary metal–oxide–semiconductor devices and their characteristics," *Japanese journal of applied physics*, vol. 45, no. 4S, p. 3307, 2006.

[5] G. Karakonstantis and K. Roy, "Voltage over-scaling: A cross-layer design perspective for energy efficient systems," in *Circuit Theory and Design (ECCTD), 2011 20th European Conference on*. IEEE, 2011.

[6] J. George, B. Marr, B. E. S. Akgul, and K. V. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in *Proceedings of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, ser. CASES '06. New York, NY, USA: ACM, 2006, pp. 158–168. [Online]. Available: http://doi.acm.org/10.1145/1176760.1176781

[7] J. George, B. Marr, B. E. Akgul, and K. V. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in *Proceedings of the 2006 international conference on Compilers, architecture and synthesis for embedded systems*. ACM, 2006, pp. 158–168.

[8] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Microarchitecture, 2003. MICRO-36. 36th Annual IEEE/ACM International Symposium on*. IEEE, 2003.

[9] P. K. Krause and I. Polian, "Adaptive voltage over-scaling for resilient applications," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 2011, pp. 1–6.

[10] S. Ghosh, S. Bhunia, and K. Roy, "Crista: A new paradigm for low-power, variation-tolerant, and adaptive circuit synthesis using critical path isolation," vol. 26, no. 11. IEEE, 2007, pp. 1947–1956.

[11] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 32, no. 1, pp. 124–137, 2013.

[12] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in *VLSI Design (VLSI Design), 2011 24th International Conference on*. IEEE, 2011.

[13] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," *Computers, IEEE Transactions on*, vol. 64, no. 4, pp. 984–994, 2015.

[14] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient vlsi implementation of soft-computing applications," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 4, pp. 850–862, 2010.

[15] N. Zhu, W. L. Goh, and K. S. Yeo, "An enhanced low-power high-speed adder for error-tolerant application," in *Integrated Circuits, ISIC'09. Proceedings of the 2009 12th International Symposium on*. IEEE, 2009.

[16] V. Camus, J. Schlachter, and C. Enz, "Energy-efficient inexact speculative adder with high performance and accuracy control," in *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*. IEEE, 2015.

[17] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012, pp. 820–825.

[18] V. Camus, J. Schlachter, and C. Enz, "A low-power carry cut-back approximate adder with fixed-point implementation and floating-point precision," in *Design Automation Conference (DAC), 2016 53rd ACM/EDAC/IEEE*, 2016, pp. 127:1–127:6.

[19] S. Hashemi, R. Bahar, and S. Reda, "Drum: A dynamic range unbiased multiplier for approximate applications," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2015.

[20] A. Lingamneni, C. Enz, K. Palem, and C. Piguet, "Parsimonious circuits for error-tolerant applications through probabilistic logic minimization," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization, and Simulation*. Springer, 2011, pp. 204–213.

[21] A. Lingamneni, C. Enz, J. L. Nagel, K. Palem, and C. Piguet, "Energy parsimonious circuit design through probabilistic pruning," in *Design, Automation Test in Europe Conference (DATE), 2011*, March 2011.

[22] J. Schlachter, V. Camus, C. Enz, and K. Palem, "Automatic generation of inexact digital circuits by gate-level pruning," in *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, May 2015.

[23] J. Schlachter, V. Camus, and C. Enz, "Near/sub-threshold circuits and approximate computing: The perfect combination for ultra-low-power systems," in *2015 IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2015, pp. 476–480.

[24] V. Camus, J. Schlachter, C. Enz, M. Gautschi, and F. K. Gurkaynak, "Approximate 32-bit floating-point unit design with 53% power-area product reduction," in *European Solid-State Circuits Conference, ESSCIRC Conference 2016: 42nd*. IEEE, 2016, pp. 465–468.

[25] G. Karakonstantis, N. Banerjee, and K. Roy, "Process-variation resilient and voltage-scalable dct architecture for robust low-power computing," vol. 18, no. 10, Oct 2010, pp. 1461–1470.

[26] V. A. Coutinho, R. J. Cintra, F. M. Bayer, S. Kulasekera, and A. Madanayake, "A multiplierless pruned dct-like transformation for image and video compression that requires ten additions only," *Journal of Real-Time Image Processing*, pp. 1–9, 2015.

[27] S. Yu and J. Swartziander, E.E., "Dct implementation with distributed arithmetic," vol. 50, no. 9, Sep 2001, pp. 985–991.

Jeremy Schlachter (S'15) received the B.Sc. degree in physics and electrical engineering and the M.Sc in micro and nanoelectronics from the University of Strasbourg, France, in 2011 and 2013, respectively.

He joined the Integrated Circuits Laboratory (ICLAB) at the Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland, in 2013 to pursue a Ph.D. degree in electrical engineering. His current research focuses on approximate digital circuit design for energy-efficient and error-tolerant computing systems. His interests include low-power and high-performance circuit design, embedded systems, electronic design automation and energy-aware programming.

Vincent Camus (S'15) received the B.Sc. in physics and electrical engineering and the *Diplôme d'Ingénieur* from the Grenoble Institute of Technology (Grenoble INP), France, in 2010 and 2013, respectively. He also received the M.Sc. in micro and nanotechnologies for integrated systems delivered by the Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland, the Polytechnic University of Turin (Polito), Italy, and Grenoble INP, France.

He joined the Integrated Circuits Laboratory (ICLAB) at the EPFL, Switzerland, in 2013 and is currently pursuing a Ph.D. in approximate computing and error-resilient circuits for energy-efficient computing systems. His interests include digital circuits and design automation for low-power and high-performance electronics.

Krishna Palem (S'80-M'86-F'04) received the M.S. degree in electrical and computer engineering (biomedical engineering) and the Ph.D. degree from the University of Texas, Austin, TX, USA, in 1981 and 1986, respectively.

He is the Kenneth and Audrey Kennedy Professor with Rice University, Houston, TX, USA, with appointments in Computer Science, in Electrical and Computer Engineering and in Statistics. He was a Founder and the Director of the NTU-Rice Institute on Sustainable and Applied Infodynamics. He is a Scholar with the Baker Institute for Public Policy, Rice University. He was a Moore Distinguished Faculty Fellow with Caltech, Pasadena, CA, USA, from 2006 to 2007, and a Schonbrunn Fellow with the Hebrew University of Jerusalem, Jerusalem, Israel, in 1999, where he was recognized for excellence in teaching. In 2002, he pioneered a novel technology entitled probabilistic CMOS (PCMOS) for enabling ultralow-energy computing.

Prof. Palem was a recipient of the IEEE Computer Societys 2008 W. Wallace McDowell Award. In 2012, Forbes (India) ranked him second on the list of 18 scientists who are some of the finest minds of the Indian origin. He is a fellow of ACM and American Association for the Advancement of Science.

Christian Enz (S'83-M'84-SM'11) received the Ph.D. degree from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1989.

He is currently a Professor with the EPFL, Director of the Institute of Microengineering, and Head of the Integrated Circuit (IC) Laboratroy. Until April 2013, he was Vice President (VP) of the Swiss Center for Electronics and Microtechnology (CSEM), Neuchtel, Switzerland, where he headed the Integrated and Wireless Systems Division. Prior to joining CSEM, he was Principal Senior Engineer with Conexant (formerly Rockwell Semiconductor Systems), Newport Beach, CA, USA, where he was responsible for the modeling and characterization of MOS transistors for RF applications. His technical interests and expertise are in the field of ultra-low-power analog and RF integrated circuit (IC) design, wireless sensor networks, and semiconductor device modeling. He was a codeveloper of the EKV MOS transistor model. He has authored or coauthored more than 200 scientific papers and has contributed to numerous conference presentations and advanced engineering courses.

Dr. Enz is an Individual Member of the Swiss Academy of Engineering Sciences (SATW). He was an elected Member of the IEEE Solid-State Circuits Society (SSCS) Administrative Committee (AdCom) from 2012 to 2014. He is the Chair of the IEEE SSCS Chapter of Switzerland.