Near/Sub-Threshold Circuits and Approximate Computing: the Perfect Combination for Ultra-Low-Power Systems

Jeremy Schlachter†, Vincent Camus†, Christian Enz
Integrated Circuits Laboratory (ICLAB)
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
jeremy.schlachter@epfl.ch, vincent.camus@epfl.ch
†these authors contributed equally to this work

***Abstract*—While sub/near-threshold design offers the minimal power and energy consumption, such approach strongly deteriorates circuit performances and robustness against PVT (process/voltage/temperature) variations, leading to gigantic speed penalties and large silicon areas. Inexact and approximate circuit design can address these issues by trading calculation accuracy for better silicon area, circuit speed and even better power consumption. This paper reviews and proposes improvements for two approximate computing techniques applicable to arithmetic circuits: gate-level pruning and carry speculation. A critical study is then carried out considering several error metrics, and for the first time, those techniques are combined to produce approximate adders showing even higher gains at similar error levels. It is then shown that those techniques can be applied to a sub-threshold library to mitigate the large variability.**

Keywords*—*Approximate computing, inexact circuits, speculative adder, pruning, sub-threshold.

I. Introduction

Sub/near-threshold circuit are capable of ultra-low power and the best possible energy-efficiency compared to conventional super-threshold circuits. However, with this approach, the Process Voltage and Temperature (PVT) variations become huge and thus, significantly complexify the design process to meet timing and area constraints. Moreover, the circuit speed is drastically reduced and memories require a large area to be functional in the sub-threshold domain. It would therefore only be possible to execute low-complexity applications on a sub/near-threshold hardware. A potential solution to overcome these limitations is the use of inexact circuits. Indeed, approximate circuits are capable of significant power, area and delay reductions compared to their conventional counterparts. Applying inexact design techniques on sub/near-threshold hardware, can significantly relax the design constraints, speed up the nominal frequency of operation and eventually further increase the energy efficiency at the cost of some occasional errors.

This article reviews two inexact design techniques that can easily be implemented in a standard near/sub-threshold digital flow: probabilistic pruning and carry speculation. A critical comparison is carried out and finally, the demonstration is made that the two techniques can be combined to further reduce power consumption, critical path delay and silicon area. The remainder of this paper is organised as follow: section II reminds the state-of-the-art for the two design techniques, section III re-defines the error metrics and provides a functional description of the pruning and the speculation technique, section IV carries out a comparative study considering several error metrics and depicts how both design techniques can be combined to save even more silicon area, power and critical path delay. Finally, section V demonstrates that inexact design can help to overcome the limitations of sub-threshold circuits.

This work was supported by the NanoTera "IcySoC" project and the Swiss National Science Foundation grant No 200021-144418.

II. State-of-the-Art

A. Probabilistic Pruning

Probabilistic pruning is a design technique consisting in removing circuit blocks or elements such as full adder cells, gate clusters or single gates in order to trade exactness of computation against power, area and delay savings without any overhead. The decision of pruning one of these elements is based on two parameters: the significance, which is a structural parameter, and the activity determined by hardware simulations. The amount of error is simply proportional to the number of pruned elements. This technique was first introduced in [1], where full adder cells were removed from various adder architectures, resulting in gains of up to 7.5 X in Energy-Delay-Area Product (EDAP), for a 10 % relative error magnitude.

Later on, this technique has been improved by integrating it in the standard digital flow using existing tools, and by applying pruning at the gate level [2]. This finer granularity enables an order of magnitude area and power savings for a 64-bit adder with 10 % relative error magnitude. It has also been shown that 25 % power and area reduction can be achieved for 16-bit multipliers.

B. Speculative Adders

Speculative adders [3] exploit the fact that carry propagate sequences in additions are typically short, making it possible to estimate intermediate carries using a limited number of previous stages. They split the binary addition into several subpaths executed concurrently for higher execution speed and energy efficiency, but at the risk of generating occasionally incorrect results. Thus, the critical path of the adder can be divided in two or more shorter paths, relaxing constraints over the entire design and improving the speed, area and power beyond the theoretical bounds of exact adders.

A number of speculative adders have been proposed in literature with different approaches in order to reduce the error frequency or magnitude. The ETAII adder [4] consists of regular sub-adder blocks with input carries speculated from Carry Look Ahead (CLA) blocks of the same length. In the ETAIIM version, several of the most significant CLA blocks are chained to increase accuracy. The ETBA adder [5], direct descendent of the ETAIIM, adds variable speculation signs and sub-adder sum balancing multiplexed blocks to mitigate relative errors. The ETAIV [6] and CSA [7] adders have enhanced accuracy by considering two prior carry speculation blocks instead of one, coupled respectively with a carry select or a carry skip technique, with the latter also using sum balancing over several sub-adder blocks. On the other side, ISA [8] and CSC [9] adders have recently improved circuit performance and efficiency by introducing off-critical path error reduction techniques. The ISA adder concept [8] has also proposed an optimal and generalized approach of speculative compensated adders, encompassing aforementioned adders, and has introduced a simple methodology to allow designers to generate efficient architectures from a delay-accuracy specification.

III. Inexact Arithmetic Circuit Design

A. Gate-Level Pruning

Gate-Level Pruning is a CAD technique to automatically generate inexact circuits starting from a conventional design by adding only one small step in the digital design flow. The CAD framework is presented in Fig. 1.

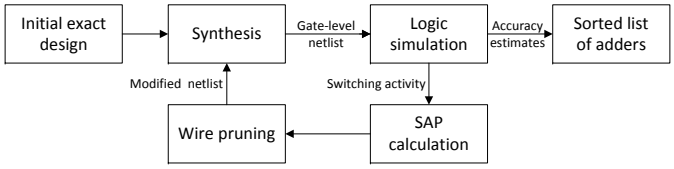


Fig. 1: CAD framework for Gate-Level Pruning.

Any exact circuit can be represented by a directed acyclic graph as depicted in Fig. 2, where the nodes are components such as gates, and whose edges are wires. The decision to prune a node is 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 requirement, the designer may choose to prune nodes according to the activity only, in order to minimize the error rate.

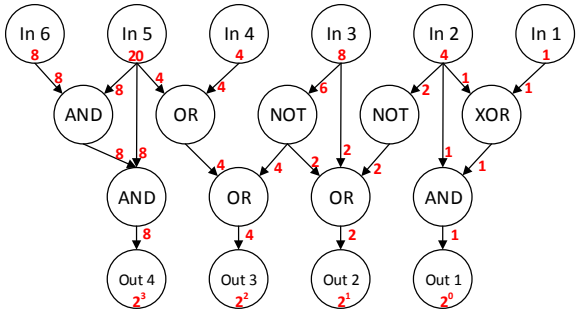


Fig. 2: Directed acyclic graph representation of a gate level netlist and the associated significance attribution

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 of each wire, as well as the time spent at the logic levels 0 and 1 respectively. In order to get an accurate activity estimation, the system should be simulated with an input stimulus representative of the *real operation* of the circuit. The more the simulation is realistic, the more the toggle count is accurate and leads to an efficient pruning.

The significance of each primary output is set by the designer depending on the application's requirement. In this paper, pruning is applied on several arithmetic circuits where each primary output is weighted by a power of two. It is therefore worth applying a weighted significance attribution, where each bit position has a significance two times higher than the previous when moving from the Least Significant Bit (LSB) to the Most Significant Bit (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. 2.

Once the significance and activity is determined, the nodes, i.e. gates and their corresponding wires, are ranked according to their SAP. The ones with the lowest SAP are disconnected from the verilog netlist, and an incremental re-synthesis is performed in order to remove or replace the unconnected gates.

B. Inexact Speculative Adders

1) General Concept

The general block diagram of an Inexact Speculative Adder (ISA) adder is depicted in Fig. 3. An ISA splits the carry propagation chain in multiple paths executed concurrently. Each path consists of a carry speculator block (SPEC), a sub-adder block (ADD) and an error compensation block (COMP). For each of these SPEC-ADD-COMP paths, the different blocks have the following functions:

- *SPEC* – The speculator block generates a partial carry signal from a limited number of operand bits in a carry look-ahead approach and sourced by either a static or a dynamic input. When a propagate chain covers the full SPEC block, the exact carry cannot be speculated from the partial product and the output carry is guessed at the

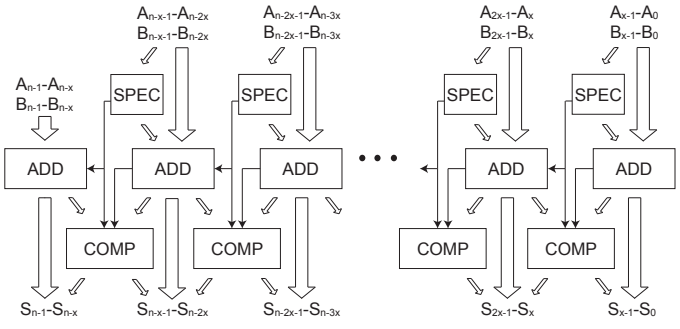


Fig. 3: General block diagram of an Inexact Speculative Adder (ISA). Each speculative segment consists of a carry speculator (SPEC), a regular adder (ADD) and an error compensation block (COMP).

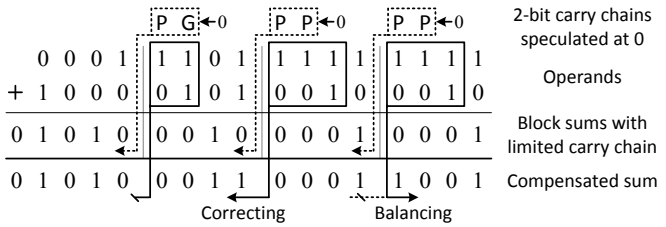


Fig. 4: Example of ISA addition arithmetic with 2-bit speculation, 1-bit correction and 1-bit error reduction. Faults only occur in the two right-hand paths. The 1st LSB of the central path can be corrected. The 1st LSB of the right path cannot be corrected, so the 1st MSB of the preceding sum is flipped.

input value. As long propagate sequences are uncommon in uniform input distribution [4], the probability of fault decreases when increasing the size of this block.

- *ADD* – The sub-adder block calculates local sums from the speculated carry of the SPEC block.
- *COMP* – Without compensation, an internal overflow caused by an inconsistent carry could lead to a massive error. Therefore, the COMP block detects those speculation faults by comparing the carry generated from the SPEC with the carry-out coming from the prior ADD block. It then compensates faulty sums either by attempting to correct a few bits of the local sum or by reducing relative error over a few bits of the preceding sum.

The first speculative path, operating on the LSBs of the adder, does not have SPEC nor COMP blocks since it uses directly the adder carry-in. The achieved addition arithmetic is illustrated in Fig. 4.

2) Error Compensation

The COMP's error correction technique, introduced in [8], consists in incrementing or decrementing only a small group of LSBs of the local sum in order to compensate the erroneous speculated carry. In most cases, this can fully resolve carry errors. In the case where those stages are all in propagate modes, correction is impossible as it would lead to an internal overflow. In that case, the uncorrected bits, having a higher significance than the error bit, ensure a low relative error of the result. Using the COMP's error correction technique thus reduces both error rate and relative error. The correction hardware is executed concurrently to the local addition, thus this technique impacts minimally the critical path of the adder.

The COMP's error reduction technique consists in balancing a group of MSBs of the preceding sub-adders in opposite direction than the error. This technique, similarly as in [5], has been intensively employed in literature. But to avoid high relative errors and better control the worst-case error (RE_{MAX}), it relies on large SPEC block directly lying in the critical path of the adder.

3) Design Strategy

The ISA offers a general topology of speculative compensated addition inclusive of the state-of-the-art and that allows an optimal balance between circuit and accuracy specifications.

A design methodology through a delay-accuracy approach is presented in Fig. 5. The adequate delay tradeoff is mainly obtained by sizing SPEC and ADD blocks, principal slack elements of the ISA. Then, the COMP's error correction and error reduction techniques enable to tune and fit the accuracy

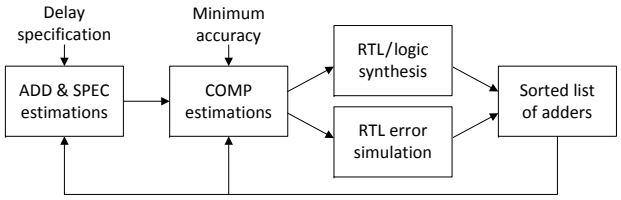


Fig. 5: CAD framework for ISA design.

requirements at the cost of hardware overhead and with a minimum delay penalty for multiplexing the result on a few compensated bits.

Adders in literature describe particular cases of implementation excessively considering either performances or errors. In the ISA architecture, the speculation overhead can be traded for longer sub-adders while fitting the same delay requirement. It is then possible to use fewer speculative paths and limit the in-critical path speculation-compensation overhead to a few bits of each path while fitting the accuracy requirement. This approach allows notable improvements in circuit performances [8].

IV. Results and Comparative Study

A. Accuracy Metrics

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

$$RE = \left| \frac{S_{approx} - S_{correct}}{S_{correct}} \right| \tag{2}$$

where S_{approx} and $S_{correct}$ are respectively the approximate and correct sums of an addition. In [1], three interesting metrics are defined:

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

$$Error\ Rate = \frac{Number\ of\ erroneous\ computations}{Total\ number\ of\ computations} \tag{3}$$

- *Relative Error RMS* (RE_{RMS}) – The root mean square (RMS) of *RE* is a good estimator of accuracy and is interesting for many applications, particularly in image and video processing. It is defined as:

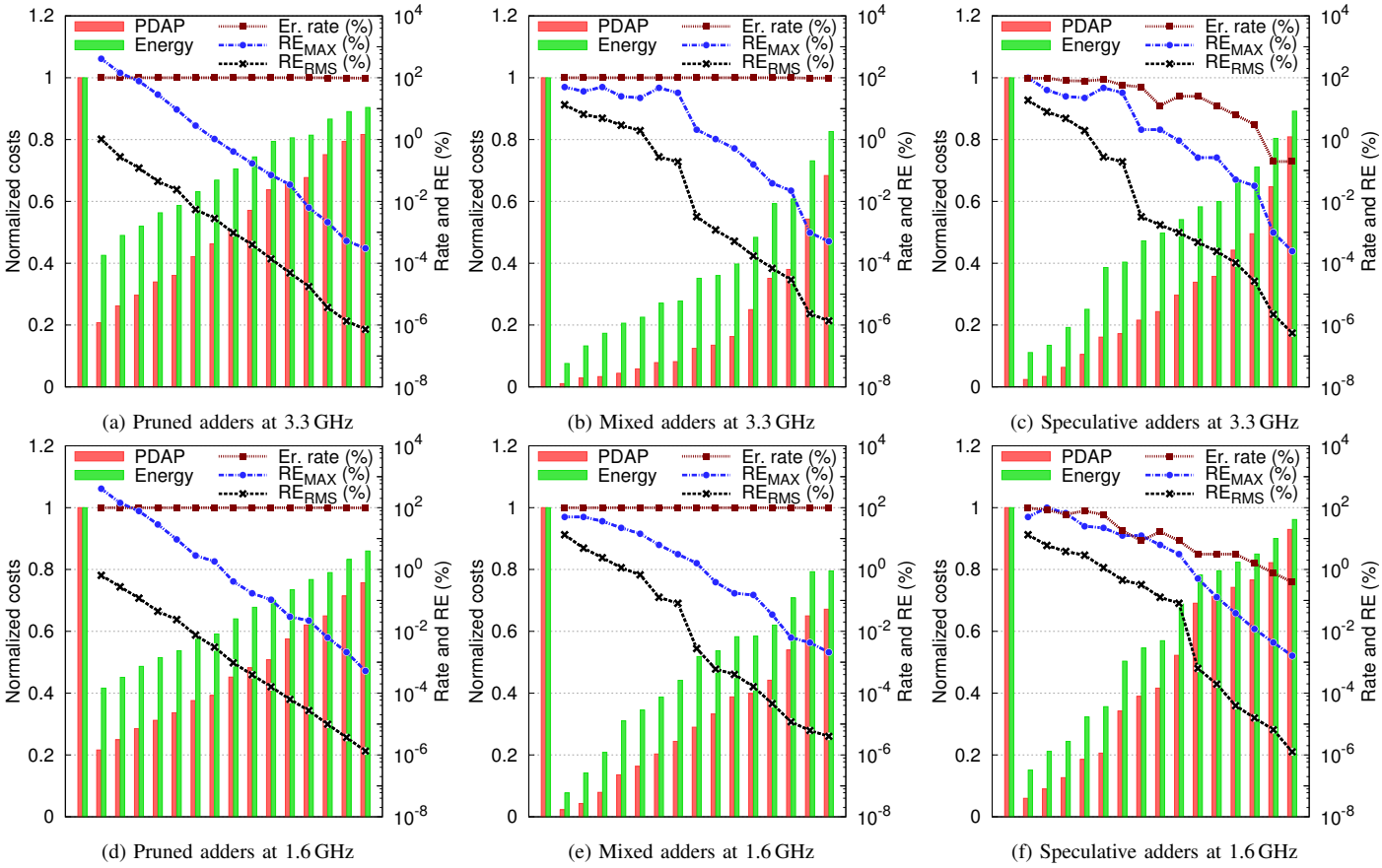
$$RE_{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} {RE_n}^2} \tag{4}$$

- *Maximum Relative Error* (RE_{MAX}) – RE_{MAX} represents the largest relative error of an adder and defines its worst case accuracy. It is here obtained over a set of computations.

B. Pruning or Speculation Applied Individually

In order to perform a comparative study, both techniques have been applied on 32-bit adders synthesized in a 65 nm UMC technology library and all the designs have been simulated with a set of five million uniformly distributed random inputs. Comparisons of error characteristics and normalized costs in terms of energy and Power-Delay-Area Product (PDAP) are shown for a selection of pruned and speculative adders synthesized at 3.3 GHz in Fig. 6a and 6c and synthesized at 1.3 GHz in Fig. 6d and 6f.

Only speculative adders with regular structures have been synthesized (i.e. 2x16, 4x8, 8x4 and 16x2 bit concurrent



(a) Pruned adders at 3.3 GHz (b) Mixed adders at 3.3 GHz (c) Speculative adders at 3.3 GHz

(d) Pruned adders at 1.6 GHz (e) Mixed adders at 1.6 GHz (f) Speculative adders at 1.6 GHz

Fig. 6: Error characteristics and normalized cost of 32-bit pruned, mixed and speculative adders synthesized at 3.3 GHz and 1.6 GHz.

paths) with diverse error characteristics. For this reason, the displayed characteristics present steps corresponding to changes of structure sizes. Generally, ISA adders built out of small sub-adders show high errors and high savings (on the left of the figures), whereas ISA with large sub-adders are preferred for low errors and lower savings (on the right of figures).

Fig. 6 clearly show that the two techniques have a different impact on the output quality. The error rate of pruned adders rapidly reaches 100 %, the reason being that in the first steps of the pruning process, some of the least significant outputs are removed. On the other hand, in speculative adders, a small speculation-correction overhead leads to a decrease of the error rate despite lower circuit efficiency.

For both techniques and frequencies, the RE_{RMS} and the RE_{MAX} grow with an exponentially trend versus circuit savings. The ISA adders on the left of figures have a low RE_{MAX}, but this one does not follow the same exponential trend as it is expensive to control. Thus, a gap appears between RE_{MAX} and RE_{RMS} when the constraints on the circuit become too high (at 1 % for 3.3 GHz and 10^{-3} % at 1.6 GHz).

Timing constraint has a significant influence on the result obtained with the two techniques. Fig. 6a and Fig. 6c show that at high frequency of 3.3 GHz, and for a relative PDAP cost of 0.42, the RE_{MAX} and the RE_{RMS} of the pruned adder are equal to 4 % and 0.008 % respectively. In comparison, the speculative adder having a similar PDAP has a RE_{MAX} of 10^{-1} % and a RE_{RMS} of 10^{-4} %. This could lead to the conclusion that the speculation technique can achieve similar energy savings than the pruning technique, at a much higher accuracy level. However, Fig. 6d and Fig. 6f actually depict the opposite trend when using a slightly lower frequency of 1.6 GHz. Hence, a more extensive comparative study might show that the two depicted design techniques might produce uncorrelated errors, and therefore could be combined to get additive savings.

C. Mixing of Pruning and Speculation

All the previous paragraphs were discussing the individual use of the pruning and the speculation techniques on 32 bit adders, once at a frequency of 3.3 GHz, and once at 1.6 GHz. The results clearly show that the pruned and the speculative adders produce different types of errors. It is therefore worth combining the two techniques. Since the pruning methodology is only one additional step in the standard digital flow, the general methodology to combine speculative circuits and pruning is:

- Synthesize speculative circuits at the desired speed constraint starting from the HDL description.
- Apply the pruning methodology to the gate-level netlists of the speculative circuits.

Fig. 6b and Fig. 6e show the normalized costs as well as the error characteristics of *pruned speculative* (mixed) 32-bit adders synthesized at 3.3 GHz and 1.6 GHz respectively. It is very interesting to see that the RE_{RMS} and the RE_{MAX} follow almost the same trend for the mixed adders than for the speculative adders. This is due to the fact that the two techniques produce different types of errors: in speculative adders which are cutted into sub-blocks, errors are rare but can occur as often on

TABLE I: Comparison of the three inexact design techniques.

Specifications		Normalised PDAP		
Frequency	RE_{RMS}	Pruning	Mixed	Speculative
3.3 GHz	3.10^{-6} %	0.76	**0.54**	0.64
	10^{-3} %	0.52	**0.14**	0.3
	2 %	0.2	**0.05**	0.1
1.6 GHz	10^{-5} %	0.64	**0.54**	0.78
	10^{-1} %	0.3	**0.2**	0.43
	1 %	0.2	**0.14**	0.2

LSBs than on MSBs, the latter having a significant impact on the error magnitude. In opposition, the weighted significance attribution of the pruning methodology leads to a large number of errors — and thus an error rate 100 % for most of the pruned adders — but those are limited to the LSBs and have a small impact on the error magnitude. Hence, the assumption can be made that the errors resulting from pruning and speculation are uncorrelated, and thus, equalizing the error levels resulting from each technique enables additive area, power and delay savings. This is verified by simulation, all the *pruned speculative* adders depicted Fig. 6b and 6e have a lower PDAP cost and consume less energy than the pruned or the speculative adders.

Table I summarises the PDAP reduction obtained through the use of the three techniques, namely pruning, speculation and the mixing of both. It is shown that a *pruned speculative* adder with only 2 % relative error magnitude has a normalised PDAP of 0.05, which is a factor 20 improvement. Moreover, energy consumption is reduced by a factor 4 compared to the exact 32 bit adder synthesized at the same speed.

V. Inexact Design in the Sub-Threshold Domain

On the one hand, the demonstration has been made that inexact design techniques applied individually, or even combined, in a UMC 65 nm technology can lead to drastic gains in energy consumption, silicon area and critical path delay. On the other hand, sub-threshold suffer from large variability, low speed of operation and require a large silicon area. In order to verify that sub-threshold circuits can benefit from approximate computing, gate-level pruning has been applied on an 32 bit adder synthesized with a 180 nm sub-threshold library, at a frequency of 22.7 MHz. Sub-threshold pruned adders Fig. 7 show similar savings and error characteristics than the ones synthesized with the UMC 65 nm technology. In this regards, speculative and *pruned speculative* might show similar gains in the sub-threshold domain than in the standard UMC 65 nm library.

VI. Conclusion

This paper reviewed and compared two well established techniques for generating approximate hardware: carry speculation and gate-level pruning. It has been shown that both can achieve up to 85 % PDAP reduction for a RMS relative error of 1 %. Moreover, since the two techniques clearly have

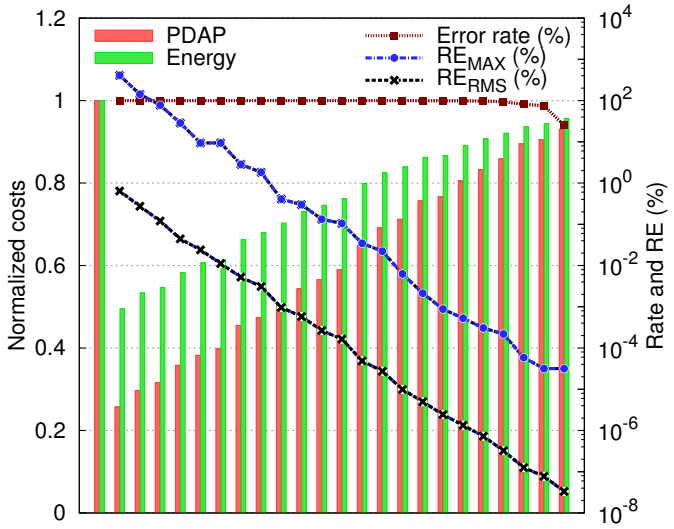


Fig. 7: Pruned 32-bit adder synthesized with a 180 nm sub-threshold library

a different impact on the accuracy of the generated adders, they can be combined to achieve up to a factor 20 saving in EDAP for only 2 % RE_{RMS}. These improvements can clearly be exploited to mitigate the large area and PVT variability as well as the low speed of sub-threshold circuits: pruned sub-threshold 32-bit adders can reduce the PDAP by a factor 4 compared to exact ones and a future work might demonstrate that the use of *pruned speculative* adders in the sub-threshold domain would lead to even higher gains.

References

[1] 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 Exhibition (DATE), 2011*, March 2011, pp. 1–6.

[2] 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.

[3] T. Liu and S.-L. Lu, "Performance Improvement with Circuit-level Speculation," in *Microarchitecture, 2000. MICRO-33. Proceedings. 33rd Annual IEEE/ACM International Symposium on*, 2000, pp. 348–355.

[4] N. Zhu, W.-L. Goh, and K.-S. Yeo, "An Enhanced Low-power High-speed Adder For Error-tolerant Application," in *Integrated Circuits (ISIC), Proc. of the 2009 12th International Symposium on*, Dec 2009, pp. 69–72.

[5] M. Weber, M. Putic, H. Zhang, J. Lach, and J. Huang, "Balancing Adder for Error Tolerant Applications," in *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, May 2013, pp. 3038–3041.

[6] N. Zhu, W.-L. Goh, G. Wang, and K.-S. Yeo, "Enhanced Low-power High-speed Adder for Error-tolerant Application," in *SoC Design Conference (ISOCC), 2010 International*, Nov 2010, pp. 323–327.

[7] Y. Kim, Y. Zhang, and P. Li, "An Energy Efficient Approximate Adder with Carry Skip for Error Resilient Neuromorphic VLSI Systems," in *Computer-Aided Design (ICCAD), 2013 IEEE/ACM International Conference on*, Nov 2013, pp. 130–137.

[8] 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*, May 2015.

[9] J. Hu and W. Qian, "A New Approximate Adder with Low Relative Error and Correct Sign Calculation," in *Design, Automation and Test in Europe (DATE), 2015 IEEE Conference and Exhibition on*, March 2015.