A Low-power *Carry Cut-Back* Approximate Adder with Fixed-point Implementation and Floating-point Precision

Vincent Camus, Jeremy Schlachter, Christian Enz
Integrated Circuits Laboratory (ICLAB)
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
vincent.camus@epfl.ch

ABSTRACT

This paper introduces an approximate adder architecture based on a digital quasi-feedback technique called Carry Cut-Back in which high-significance stages can cut the carry propagation chain at lower-significance positions. This lightweight approach prevents activation of the critical path, improving energy efficiency while guaranteeing low worst-case relative error. It offers a degree of freedom which allows to dissociate precision and dynamic range in fixed-point implementation. A design methodology is presented along with results and a comparative study. For a worst-case accuracy of 98 %, energy savings up to 44 % and power-delay-area reductions up to 62 % are demonstrated compared to low-power conventional designs.

Categories and Subject Descriptors

B.5.1 [**Register-Transfer-Level Implementation**]: Design—*arithmetic and logic units, data-path design*; G.1.0 [**Numerical Analysis**]: General—*multiple precision arithmetic, error analysis*

Keywords

Approximate adders, error tolerance, approximate computing, approximate circuit design, low-power digital circuits

Acknowledgments

This work has been supported by the NanoTera IcySoC project of the Swiss National Science Foundation.

1. INTRODUCTION

Density, speed and energy efficiency of integrated circuits have been increasing exponentially for the last four decades following Gordon Moore's remarkable prediction. However, power and reliability pose several challenges to the future of technology scaling. Power has definitely emerged as a critical concern due to the poor scaling of V_{dd} and V_{th}, while transistor miniaturization reaching atomic scale has led to tremendous Process-Voltage-Temperature (PVT) variations. Unfortunately, achieving low-power and robustness against variability requires complex and conflicting design constraints. As a result, designers are being pushed to seek new energy-efficient computing techniques to meet the increasing demand of data processing.

Error tolerance, i.e. accepting error in a design to save resources, is a well-known concept existing in many abstraction layers, from physical sensors and analog circuits to software design. Built on these ideas, *approximate computing* [1, 2] has emerged as a promising candidate to improve performance and energy efficiency and sustain technology scaling. Designing *approximate circuits* explores a new trade-off, not only by accepting unreliability, but by intentionally introducing errors to overcome limitations of traditional design. With the exploding amount of data being processed in the cloud and on mobile devices, a wide range of applications can trade a little accuracy without compromising the functionality or the user experience. In image or video processing applications, a small proportion of errors stays imperceptible to humans. The growing demand for statistical algorithms such as data mining, recognition, search or machine learning represents another opportunity to compute in an approximate way as the outcome of those applications is not required to be a single golden result, but an adequate match. Finally, iterative applications like vision and tracking are inherently resilient to errors since those can be compensated in the succeeding frames or steps.

To design approximate circuits, several approaches have been investigated at different levels of hardware design, such as voltage-frequency over-scaling [3] at physical level, Gate-Level Pruning [4] at circuit level, or Significance Driven Computation [5] at algorithmic level. Another way consists in redesigning the architecture of combinational circuits into an approximate version with smaller delay, area or power consumption. This technique is particularly suited for arithmetic operators such as adders. Trading numerical accuracy for circuit efficiency is not a new trend, it exists since the beginning of digital electronics with the discretization of signals into binary data, first in fixed-point arithmetic, and later in Floating-Point Units (FPU). From that time on, those two standards have been at the heart of DSPs and hardware accelerators. Even though FPU features a superior computational ability and greater dynamic range, fixed-point arithmetic continues to be the mainstay in industry by offering a lower hardware complexity and power consumption.

This paper introduces a novel approximate adder architecture optimizing circuit timing together with arithmetic precision thanks to a new technique called Carry Cut-Back (CCB). Using a quasi-feedback between two carry-chain positions, it prevents the critical-path activation, therefore relaxing the timing constraints in the entire design and strongly improving

DAC '16, June 05-09, 2016, Austin, TX, USA

 ISBN 978-1-4503-4236-0/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2897937.2897964

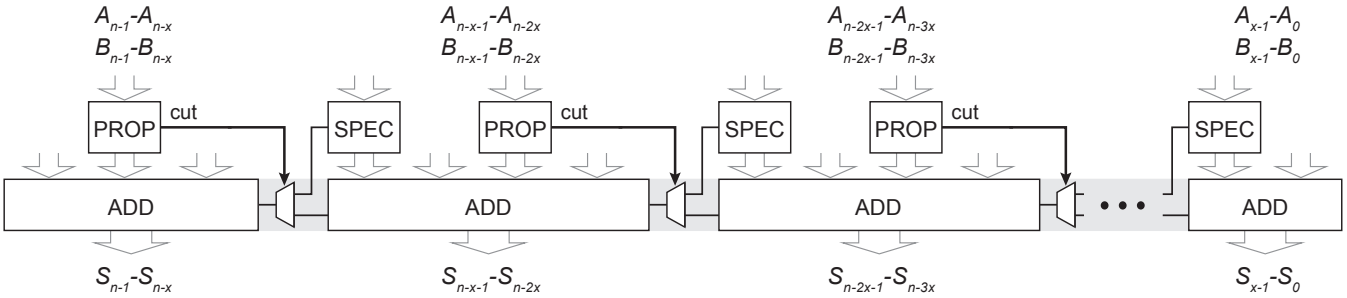

Figure 1: General block diagram of the proposed Carry Cut-Back (CCB) approximate adder.

the circuit efficiency. With this approach, high-significance carry stages are monitored to cut the carry propagation chain at lower-significance positions, guaranteeing low relative errors of floating-point type. A brief design methodology is presented together with results and a comparative analysis for both high-performance and low-power implementations.

2. RELATED WORK

Adders are the most common arithmetic blocks used in DSPs, thus many attempts have been made to build them in an approximate manner. At architectural level, an interesting way to build approximate adders is to use *carry speculation* [6]. This technique exploits the fact that carry propagate sequences in additions are typically short, making it possible to estimate, more or less accurately, an intermediate carry using a limited number of previous stages. Thus, the carry chain, critical path of the circuit, can be split into two or more shorter paths, relaxing the constraints over the entire design and pushing energy, delay and area beyond the limits imposed by traditional design.

A number of speculative adders have been proposed in literature based on the ETAII concept [7]. It consists in slicing the addition into regular sub-adder blocks with input carries speculated in a carry lookahead approach. The ETBA [8], direct descendant of the ETAII, uses an error balancing technique based on multiplexers to mitigate the relative error in case of incorrect carry speculation. The ETAIV [9] and CSA [10] have enhanced the error rate and mean by chaining several blocks for carry speculation, but at the cost of increased complexity and energy. The ISA [11] has generalized and optimized the architecture of speculative compensated adders by shortening the speculation overhead and by introducing a dual-direction compensation mechanism that improves both circuit performances and accuracy. However, the multiplexers required for a good error compensation still represent a substantial area and energy overhead, particularly for low-speed implementations.

With the progressive increase of accuracy and design flexibility, speculative adders have repeatedly been proven to be usable in specific applications [8, 10, 12, 13]. Despite all this progress, they have not yet been adopted by designers, the main reason being that occurrence and impact of errors in real-life applications are not easy to anticipate today. Error metrics are often based on full-range number assumption and uniform distribution that neither fit typical use nor worst possible cases. Moreover, applications and case studies remain singular examples. The goal of this work is to propose a more efficient adder architecture that can ensure that *all* errors remain below an upper bound to be integrated without uncertainty in codecs and hardware accelerators.

3. CARRY CUT-BACK ADDER

3.1 Proposed Architecture

The structural diagram of the proposed adder is depicted in Fig. 1. The CCB adder is based on a conventional fixed-point adder circuit (formed by the chain of ADD blocks) with insertion of several multiplexers that can cut the carry propagation chain to shorten the effective critical path. The decision to cut the chain is taken by the carry propagate block (PROP) by generating the cut signal. The cut occurs upstream in the carry chain, by multiplexing the real carry with a carry speculated from a short chain in an optional carry speculator block (SPEC). Taking place at a lower-significance stage, the carry *cut-back* guarantees a low relative error. The cut-back module appears functionally as a feedback between two carry-chain positions, but is not a recursive loop as it uses the local carry propagate and generate signals directly precomputed from the operand inputs. Hence, it cannot influence the stability of the circuit.

The main advantage of this approach remains in its timing characteristic. Typically, the carry propagation chain of an addition is naturally broken because of short-size operands or by the distribution of the input bits. Spanning over the whole adder length, the critical path is only activated if all the stages are in propagate mode. Even if the adder within the CCB architecture physically contains the entire carry chain through the ADD and multiplexers, *this path can never be activated*. By monitoring one or more stages of the adder, the PROP quickly detects such risk and calls a shorter path to be used instead, ensuring that the design meets tighter timing constraints. The shorter path can either be a carry speculated from the SPEC or a straight cut using a monotonic gate, such as the *OR-cut* of the adder implemented in Fig. 2 (cut = 1 dictates the OR output regardless of its second input).

Fig. 2 shows a case study of the longest propagation chains that can flow through a CCB adder built with two cut-backs. Each cut-back module splits the carry chain with two possibilities:

- *cut = 0:* No deliberate carry-cut in the typical case, i.e. the carry chain is naturally broken somewhere in the PROP. The critical path is limited since it cannot entirely cross over the PROP. The case 1 in Fig. 2 shows two examples of such behavior.
- *cut = 1:* All the stages within the PROP are in propagate mode. The carry chain necessarily propagates through the PROP and there is a risk of long critical-path activation if the other non-monitored stages are also propagates. Therefore, by intentionally cutting the carry chain, its maximum length still remains limited. The case 2 in Fig. 2 shows two examples of such behavior.

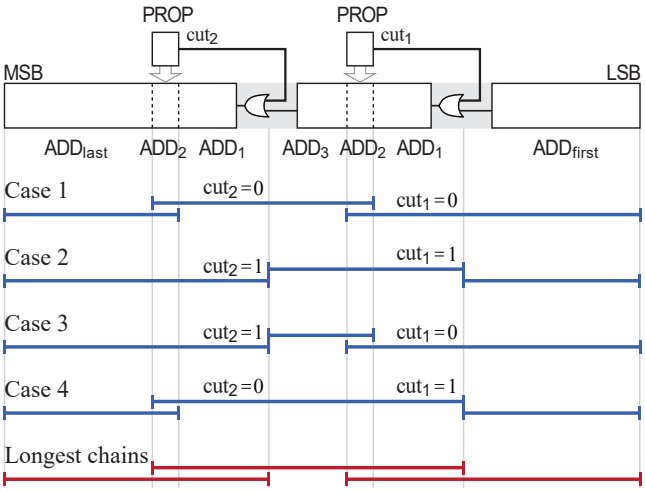

Figure 2: Diagram of the longest carry chains and resulting effective critical paths in the example of an implementation of CCB adder with two OR-cuts.

Cases 3 and 4 both contain one naturally broken chain (cut = 0) and one intentional cut (cut = 1).

Despite the fact that the full carry chain physically exists in the design, no input combination can activate it from the start to the end. It is a *false path* and can therefore be excluded from the timing analysis. The *effective* critical paths in Fig. 2 sum up the longest propagate chains that can occur in the circuit among the different cases. Insertion of more carry cut-back modules, possibly overlapping each others, would lead to shorter effective critical paths.

3.2 Arithmetic and Errors

The CCB addition arithmetic is illustrated in Fig. 3. Errors only occur with the concurrence of three factors:

- Sequence of propagate signals spanning the entire PROP bit-width, triggering the cut.
- Sequence of propagate signals spanning the entire SPEC bit-width, making the exact carry prediction impossible with the SPEC bits only.
- Wrong guess of the carry that inputs the SPEC (Fig. 3a) or that directly substitutes for the real carry (Fig. 3b). This occurs with a 50 % binary probability.

An error occurs in the right-hand path of Fig. 3a because of the simultaneous occurrence of the three aforementioned properties. In the OR-cut implementation of Fig. 3b (without SPEC), the cut signal is also the guessed carry. The first condition of error occurrence is met for the two right-hand paths. The guess unintentionally follows the real carry and leads to a correct sum in the central path, but happens to be wrong and leads to a faulty sum in the right-hand path.

Occurrence of an error implies that one or both operands have non-zero bits at the PROP position. As the error occurs at the carry cut, at a lower-significance position, the expected sum is necessarily much larger than the introduced error. In the computation of Fig. 3a, the absolute error is 16 while the expected sum is 43,265 so the relative error is 0.04 %. In the example of Fig. 3b, the relative error is only 0.006 %. Such low relative errors are typical in speculative adders for calculations involving large value operands. However, it is the worst case that gives the upper-bound relative error and defines the minimum floating-point precision of the adder.

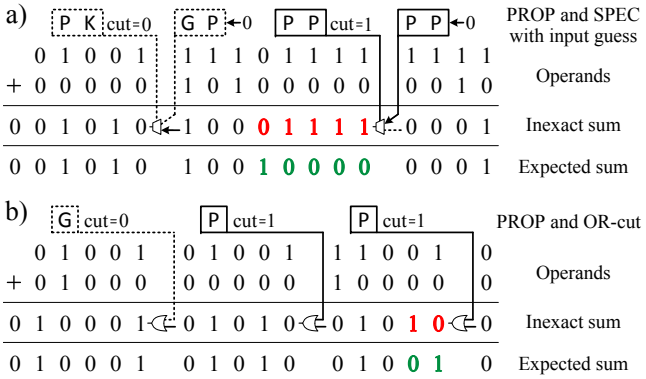

Figure 3: Example of CCB addition arithmetic with (a) 2-bit PROP and SPEC, (b) 1-bit PROP OR-cut.

3.3 Floating-point Precision

Error propagation

It is interesting to note from Fig. 3 that the error caused by the cut can propagate on many bits, but seems to keep the magnitude of the carry cut-back position, to wit, the first wrong bit. This statement that seems straightforward with the example has to be demonstrated carefully. Indeed, a successive series of erroneous sum bits can result in different errors. Let S_i, C_i and P_i denote the sum, carry-in and propagate signals of the i^{th} stage addition, respectively. The sum and carry propagation are defined by:

$$S_i = P_i \oplus C_i \tag{1}$$

$$P_i = 1 \implies C_{i+1} = C_i. \tag{2}$$

Assume a carry error at the i^{th} bit of the adder, with an erroneous carry of value C_{err}. The sum bit and the carry-out depend on the value of P_i. If $P_i = 1$, (1) gives $S_i = \overline{C_{err}}$ and (2) propagates the wrong carry C_{err} to the next stage, where the same formulae apply again. If $P_i = 0$, (1) gives $S_i = C_{err}$ and the wrong carry is not propagated, so the next stage addition is correct. Assuming that the erroneous sum spreads from the m^{th} to the p^{th} stage, the error pattern appears as shown in Fig. 4:

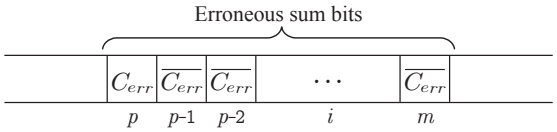

Figure 4: Balanced error pattern

Just as in Fig. 3, where the error patterns are shown in red, the last faulty bit counterbalances the first ones and the absolute error value is reduced to:

$$2^p - 2^{p-1} - 2^{p-2} - \cdots - 2^m = 2^m. \tag{3}$$

This result is valid if the carry propagates normally. But there can be more than one cut-back module, and if all the stages between two cut-backs propagate, it could disrupt the normal propagation driven by (2). Thus, the previous result needs to be recomputed for that case. Assume the same carry error ($C_i = C_{err}$) in a propagating stage ($P_i = 1$, else there would be no carry-chain perturbation). If another cut-back happens to guess the same faulty carry C_{err}, then it transparently follows (2) and the previous result holds. But if the carry-cut is in the opposite direction $\overline{C_{err}}$, as it runs

against (2), it reverses the error: the carry, that was false until now, comes back to the value of the expected addition, so the next stage is correct. But the current sum, determined by (1), is $S_i = \overline{C_{err}}$. The error pattern appears this time as:

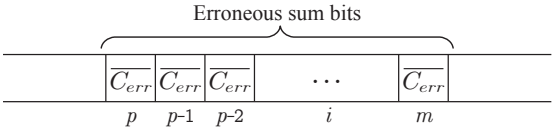

Figure 5: Unbalanced error pattern

All the erroneous bits are in the same direction and the absolute error is simply their sum:

$$2^p + 2^{p-1} + 2^{p-2} + \cdots + 2^m = 2^{p+1} - 2^m. \quad (4)$$

This error is of much higher magnitude than in the first case, but can only occur if several carry-cuts happen in opposite directions. To avoid such dramatic errors, the SPEC guess or the straight carry-cut must be chosen in the same direction for all the CCB modules of the adder.

Worst-case relative error

Having validated the fact that any error has the magnitude of the cut-back bit that caused it, the low impact of the error on the expected sum should be demonstrated.

The worst case happens when the error magnitude is the highest on the lowest expected calculation. Occurrence of an error implies that the three factors mentioned in section 3.2 are realized, this assumes that the PROP and SPEC intercept propagate signals only. All the non-zero operand bits producing those propagates add up to the expected sum:

- Standing at higher-significance positions than the carry error, the PROP non-zero bits significantly contribute to maximizing the expected result and thus to minimizing the worst-case relative error.
- Positioned directly before the carry-cut, the SPEC non-zero bits contribute in a lower extent to increasing the sum by attenuating a portion of the magnitude of the error. Although, they participate equally with the PROP bits in reducing the rate of errors.
- When the SPEC guess or the straight carry-cut is 0, i.e. speculating a low carry, an error replaces a real carry at state 1 coming from a generate stage. Added to the SPEC bits, this stage further increases the sum to 2^m.

Whenever a carry-cut error occurs, while it keeps the magnitude of the cut bit significance, i.e. an arithmetic error of value 2^m, the sum is always expected to be greater than:

$$2^m + \sum_{k\in\text{PROP}} 2^k \quad \text{and} \quad \sum_{k\in\text{SPEC}} 2^k + \sum_{k\in\text{PROP}} 2^k, \quad (5)$$

leading to a relative error lower than:

$$\frac{2^m}{2^m + \sum_{k\in\text{PROP}} 2^k} \quad \text{and} \quad \frac{2^m}{\sum_{k\in\text{SPEC}} 2^k + \sum_{k\in\text{PROP}} 2^k}, \quad (6)$$

in the cases where the carry guess is at 0 and 1, respectively. This result holds if multiple errors occur in different carry-cut modules as the ratio of error over sum is preserved.

A floating-point precision is thus configurable at design time by sizing and positioning PROP and SPEC and selecting the carry guess. It is easy to verify that the worst-case relative error is 7.7 % for the example in Fig. 3a and 12.5 % in Fig. 3b, corresponding to precisions between 4 and 5 bits.

3.4 Design and Implementation Strategy

The CCB technique allows considerable improvements concurrently in circuit performance and error control. This section describes how to exploit its architectural advantages.

Design implementation

Both PROP and SPEC can be implemented in a carry-lookahead approach and should have very short bit-widths to limit overheads. Their areas can fortunately be balanced as the adder segments that they overlay can be cut down to simple sum generators. Moreover, the delay overhead is limited by the slowest between PROP and SPEC since they are executed in parallel.

The CCB adder physically contains the entire adder carry chain but the cut-back scheme prevents from its activation and splits the critical path into multiple shorter paths. However, long-established Static Timing Analysis (STA) used in synthesis tools cannot easily identify those timing exceptions. It is thus necessary to provide the tools with additional timing constraints to manually exclude from the timing analysis all the false paths generated by the CCB modules. This additional information prevents the synthesis tools from unnecessarily trying to meet delay constraints on them.

Delay optimization

Assuming identical cut-back modules inserted in the adder and with the notation of Fig. 2, the effective critical-path delays are:

$$\begin{cases} t_{\text{first}} = t_{\mathsf{ADD}_1} + t_{\mathsf{ADD}_2} + t_{\mathsf{ADD}_{\text{first}}} \\ t_{\text{mid}} = 2\,(t_{\mathsf{ADD}_1} + t_{\mathsf{ADD}_2}) + t_{\mathsf{ADD}_3} + t_{\text{mux}} \\ \qquad\quad + \mathit{max}(t_{\mathsf{SPEC}}, t_{\mathsf{PROP}}) \\ t_{\text{last}} = t_{\mathsf{ADD}_1} + t_{\mathsf{ADD}_2} + t_{\mathsf{ADD}_{\text{last}}} + t_{\text{mux}} \\ \qquad\quad + \mathit{max}(t_{\mathsf{SPEC}}, t_{\mathsf{PROP}}) \end{cases} \quad (7)$$

where t_{first} and t_{last} are the two boundary-path delays, and t_{mid} is the delay of the all the intermediate paths. The multiplexer delay t_{mux} can eventually be replaced by the delay of the straight-cut gate.

In order to optimize the timing budget, it is possible to equalize those paths by sizing $\text{ADD}_{\text{first}}$ and ADD_{last}, and to equalize the SPEC and PROP bit-widths. Note that the PROP and ADD_2 blocks are proportionate as they cover the same adder segment, the first as carry-lookahead and the latter as sum generator.

Error optimization

The CCB adder enables to dissociate the precision from the dynamic range of the adder, which is fixed by the total adder bit-width. It offers a large design space to minimize the application quality loss and maximize the savings by trading off mean, maximum and rate of errors, configurable by choosing positions and bit-widths of the CCB modules.

The error rate depends on the number of cut-back modules and of the PROP and SPEC bit-widths. The maximum error can be adjusted mainly by sizing the PROP bit-width and positioning the carry-cut (i.e. sizing ADD_1), and to a lesser extent by modifying the SPEC bit-width and input guess. Optimum trade-offs to adjust Signal-to-Noise Ratio (SNR), Root Mean Square (RMS) error or any other accuracy metric can be achieved using the same models than those built for speculative adders [12, 13].

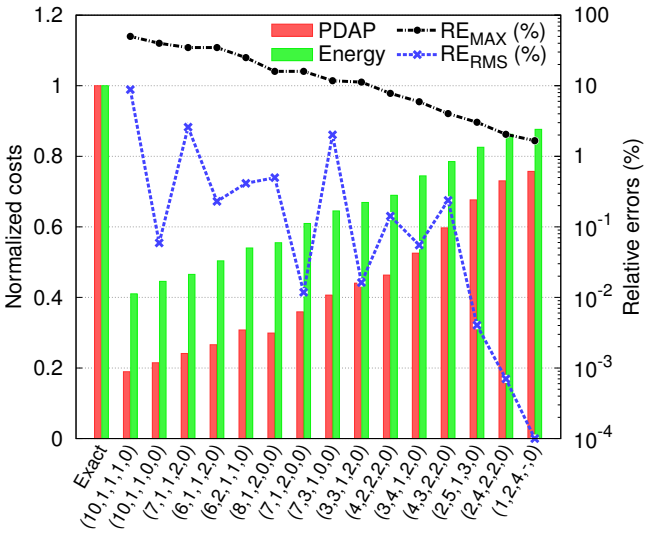

Figure 6: Relative errors and normalized costs of various 32-bit CCB adders (on the horizontal axis) synthesized at 3.3 GHz in a 65 nm technology.

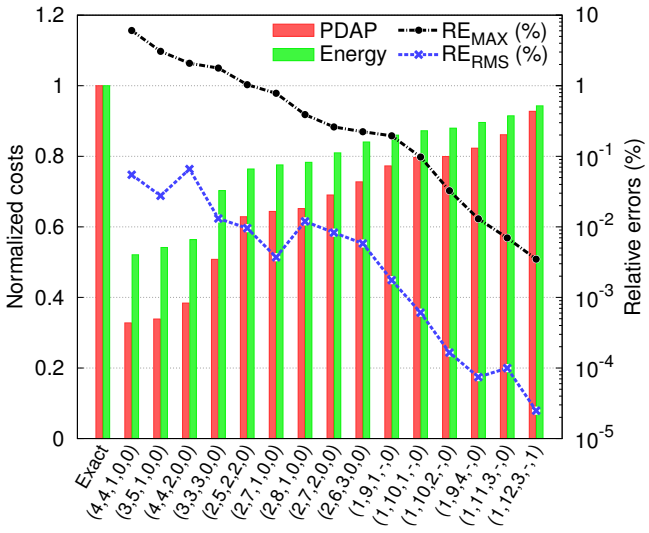

Figure 7: Relative errors and normalized costs of various 32-bit CCB adders (on the horizontal axis) synthesized at 0.8 GHz in a 65 nm technology.

4. RESULTS AND COMPARATIVE STUDY

4.1 General Considerations

Metrics

The metrics used to characterize approximate adders in this work are based on the relative error (RE), which has the advantage of being independent of the size of the adder. It is defined as:

$$\mathrm{RE} = \left| \frac{S_{\mathrm{approx}} - S_{\mathrm{exact}}}{S_{\mathrm{exact}}} \right| \tag{8}$$

where S_{approx} and S_{exact} are the approximate and correct sums of an addition, respectively.

The main metric considered is the maximum of the relative error ($\mathrm{RE_{MAX}}$) that delimits the minimum precision of the circuit. The RMS of the relative error ($\mathrm{RE_{RMS}}$) is also taken into account as it is proportional to the SNR and interesting for many applications, particularly in multimedia processing.

Methodology

Approximate adders are commonly characterized and validated through the simulation of random sets of inputs. As a matter of fact, the presented results are statistical estimations depending on the random sample distribution (occurrence of specific patterns initiates errors in specific adders). In this work, adders are characterized using two samples of five million unsigned random inputs. First, a logarithmically uniform distribution exhibiting a very large dynamic range is used to detect the worst-case error $\mathrm{RE_{MAX}}$. Then, a uniform distribution is used to estimate $\mathrm{RE_{RMS}}$.

In this work, several 32-bit approximate adders have been synthesized for low-power (0.8 GHz) and high-performance (3.3 GHz) in an industrial 65 nm technology. Over 5000 implementations with diverse error characteristics have been investigated by varying design parameters. All circuits have been generated with regular block structures from high-level descriptions in order to benefit from the compiler's optimization libraries and most favorable architecture choices to fit each timing constraint. Delay, area and power have been estimated using Synopsys Design Compiler.

4.2 CCB Adder Results

Error characteristics and normalized costs in terms of energy and Power-Delay-Area Product (PDAP) are shown for a selection of CCB adders at 3.3 GHz in Fig. 6 and at 0.8 GHz in Fig. 7. Costs are normalized to the exact adder represented on the left of the figures. CCB adders are denoted by quintuples of bit-widths: (*number of cut-backs*, $\mathrm{ADD_1}$, PROP, $\mathrm{ADD_3}$, SPEC) assuming a regular block structure and the optimizations described in 3.4. These figures highlight the large design space and error engineering possibilities enabled by the proposed adder. The CCB design parameters allow to tune the precision on more than three orders of magnitude of errors with optimal circuit efficiency.

Timing constraints have a significant influence on the results. At equivalent precision, low-speed implementations show better savings than high-performance ones compared to the exact adders. At 2 % $\mathrm{RE_{MAX}}$, CCB adders at 3.3 GHz achieve 14 % energy savings and 27 % PDAP reductions against 44 % and 62 % for adders at 0.8 GHz. This is due to the fact that high-speed circuits require more CCB modules to split the carry chain into smaller pieces, but at the cost of additional hardware overhead.

Fig. 7 presents a small but sharp drop in circuit efficiency at 1.7 % $\mathrm{RE_{MAX}}$. This corresponds to the precision from which the design becomes delay constrained. Indeed, higher precision demands wider PROP, SPEC and $\mathrm{ADD_1}$ which all lie in the effective critical path. This does not appear for 3.3 GHz adders which are always tightly constrained.

Note that $\mathrm{RE_{RMS}}$ and $\mathrm{RE_{MAX}}$ follow the same trend, but with a larger variability for high-speed and low-precision adders. Those generally contain several cut-back modules, so a small change in their structure repeated over many of them strongly impacts the overall error rate and mean.

4.3 Comparative Study

Tables 1 and 2 compare the costs and PDAP of the CCB adder with other approximate adders at 3.3 GHz and 0.8 GHz. Only ETBA [8] and ISA [11] are shown for comparison as they exhibit good savings for a bit-width of 32 bits. A modified ETBA has been built with fixed carry guess

Table 1: Comparison of 32-bit adders at 3.3 GHz

Architecture	RE_{MAX} (%)	Energy (fJ)	Area (μm^2)	PDAP
Exact	0	47	910	424
ETBA modified (2)	35	24	497	117
ISA $(2, 0, 1, 2)$		13	317	41
CCB adder $(7, 1, 1, 2, 0)$		22	472	102
ETBA modified (4)	6	38	738	281
ISA $(4, 2, 2, 2)$		37	689	252
CCB adder $(3, 4, 1, 2, 0)$		35	643	223
ISA $(16, 0, 5, 4)$	3	46	826	378
CCB adder $(2, 5, 1, 3, 0)$		38	749	287

Table 2: Comparison of 32-bit adders at 0.8 GHz

Architecture	RE_{MAX} (%)	Energy (fJ)	Area (μm^2)	PDAP
Exact	0	79	401	316
ETBA modified (4)	6	50	380	187
ISA $(8, 4, 0, 4)$		49	309	150
CCB adder $(4, 4, 1, 0, 0)$		41	253	103
ETBA modified (8)	0.4	82	451	370
ISA $(8, 4, 4, 4)$		69	379	263
CCB adder $(2, 8, 1, 0, 0)$		62	334	216
ISA $(16, 3, 7, 2)$	0.2	78	401	313
CCB adder $(1, 10, 1, -, 0)$	0.1	68	362	246

as the original variable guess was weakening its efficiency. For given RE_{MAX}, the best implementation of each architecture has been selected. All structures are regular and denoted by n-tuples of bit-widths: ($block\,size$) for ETBA, ($block\ size$, SPEC, $correction$, $reduction$) for ISA and as already stated for CCB adders.

Among high-performance adders (Table 1), CCB and ETBA architectures are completely overtaken by the ISA for very low precisions (35 % RE_{MAX}). In this case, the minimal architecture of the ISA optimally fits the difficult delay constraint without loss of circuit efficiency. The situation reverses at higher accuracy, for which the need of wider speculation and compensation hardware in the critical path reduces the efficiency of ETBA and ISA. At 6 % RE_{MAX}, the CCB adder performs 11 % better than the ISA and 21 % better than the ETBA in term of PDAP. Increasing the precision to 3 % RE_{MAX} widens the gap with the CCB adder performing 24 % better than the ISA.

For low-power implementations (Table 2), the CCB adder always outperforms the state-of-the-art. Indeed, low speed allows smaller and more energy-efficient architectures to be used in the addition sub-blocks. The speculation and compensation blocks of ISA and ETBA thus become a large area and energy overhead. Thanks to its lightweight cut-back mechanism, CCB architectures exhibit 18-30 % PDAP reductions compared to ISA and 40-45 % compared to ETBA while maintaining equal or greater precision.

Moreover, while circuit savings of ISA and ETBA progressively disappear at higher accuracy compared to the exact adder, the CCB architecture still offers significant savings. Up to 14 % energy savings and 22 % PDAP reductions are demonstrated for 0.1 % RE_{MAX}, corresponding to 11-bit precision, i.e. the mantissa precision of a 16-bit FPU.

5. CONCLUSION

This paper has introduced a novel architecture of approximate adder optimizing circuit timing together with arithmetic precision. By performing a quasi-feedback in the carry chain, the Carry-Cut Back (CCB) technique prevents the critical-path activation, therefore relaxing timing constraints and strongly improving circuit efficiency. In this approach, high-significance carry stages are monitored to cut the carry chain at lower-significance positions to guarantee a precision of floating-point type with a marginal overhead.

For a worst-case relative error of 2 %, the results for 32-bit adders show energy savings up to 44 % and PDAP reductions of up to 62 % compared to low-power conventional circuits. Besides, the proposed adder surpasses the state-of-the-art approximate adders, performing up to 30 % better than the ISA and 45 % better than the ETBA in term of PDAP.

Thanks to the instinctive floating-point precision which ensures that all errors remain below an upper bound, this approximate adder could help designing low-power and highly-efficient hardware accelerators with a more acceptable and predictable impact on their accuracy.

6. REFERENCES

[1] C. M. Kirsch and H. Payer. Incorrect Systems: It's not the Problem, It's the Solution. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 913–917, June 2012.

[2] K. Palem and A. Lingamneni. Ten Years of Building Broken Chips: The Physics and Engineering of Inexact Computing. In *ACM Transactions on Embedded Computing Systems*, volume 12, pages 87:1–87:23, May 2013.

[3] S. Ghosh, S. Bhunia, and K. Roy. CRISTA: A New Paradigm for Low-power, Variation-tolerant, and Adaptive Circuit Synthesis Using Critical Path Isolation. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, volume 26, pages 1947–1956, Nov 2007.

[4] 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*, pages 173–176, May 2015.

[5] D. Mohapatra, G. Karakonstantis, and K. Roy. Significance Driven Computation: A Voltage-scalable, Variation-aware, Quality-tuning Motion Estimator. In *Low Power Electronics and Design (ISLPED), 2009 ACM/IEEE International Symposium on*, pages 195–200, Aug 2009.

[6] T. Liu and S.-L. Lu. Performance Improvement with Circuit-level Speculation. In *Microarchitecture (MICRO-33), 2000 33rd Annual IEEE/ACM International Symposium on*, pages 348–355, Dec 2000.

[7] N. Zhu, W.-L. Goh, and K.-S. Yeo. An Enhanced Low-power High-speed Adder For Error-tolerant Application. In *Integrated Circuits (ISIC), 12th IEEE International Symposium on*, pages 69–72, Dec 2009.

[8] 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*, pages 3038–3041, May 2013.

[9] 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 IEEE International*, pages 323–327, Nov 2010.

[10] 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*, pages 130–137, Nov 2013.

[11] 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*, pages 45–48, May 2015.

[12] C. Liu, J. Han, and F. Lombardi. An Analytical Framework for Evaluating the Error Characteristics of Approximate Adders. In *IEEE Transactions on Computers*, volume 64, pages 1268–1281, May 2015.

[13] J. Miao, K. He, A. Gerstlauer, and M. Orshansky. Modeling and Synthesis of Quality-energy Optimal Approximate Adders. In *Computer-Aided Design (ICCAD), 2012 IEEE/ACM International Conference on*, Nov 2012.