Journal of
Electrical & Electronic Systems

Vishnoi and Noll, J Electr Electron Syst 2017, 6:3
DOI: 10.4172/2332-0796.1000238

Review Article OMICS International

Quantitative Optimization of Highly Efficient Dedicated CORDIC Macros for SoCs in Deep-Submicron CMOS Technologies

Upasna Vishnoi* and Tobias G Noll

Electrical Engineering and Computer Systems, RWTH Aachen University, Germany

Abstract

The CORDIC algorithm is frequently used in many applications and dedicated CORDIC macros are attractive components for SoCs e.g., as hardware accelerators to Application-Specific Instruction-set Processors. Therefore, there is a wide variety of CORDIC specifications. In order to achieve high area and energy efficiency for a given specification, a systematic exploration of the design space is required. Due to the features of today's deep-submicron CMOS technologies, there are strong interactions between the design levels which makes a quantitative optimization challenging. In this contribution, an attractive approach based on algebraic cost models is described, which allows for exploring the design space with adequate accuracy and in reasonable time. The benefits from this approach are demonstrated by an evaluation for different exemplary CORDIC specifications.

Keywords: Dedicated CORDIC macros; Design space exploration; Quantitative efficiency optimization; Deep-submicron CMOS

Introduction

The CORDIC (Coordinate Rotation Digital Computer) algorithm is a well-known versatile approach, which can be used to implement a large variety of arithmetic functions. Consequently, it is widely applied in Digital Signal Processing (DSP) on today's SoCs. Typical tasks comprise Numerically Controlled Oscillator (NCO), phase rotation, equalization, modulation and demodulation, DCT [1], to name a few. In future, further applications for CORDIC building blocks can be expected, for example Singular Value Decomposition (SVD) in sophisticated algorithms for spatial filtering and direction of arrival estimation in communication and navigation systems [2,3].

Dedicated CORDIC blocks can be implemented in today's deep-submicron CMOS technologies at very low area and energy costs [4], which make them attractive as hardware accelerators for Application Specific Instruction Set Processors (ASIPs) [3]. Due to that, even complex tasks like SVD with challenging requirements become feasible as sub-blocks of future SoCs.

Many strategies and options have been elaborated to optimize dedicated CORDIC implementations [2]. Considering the manifold current and potential future applications of dedicated CORDIC macros it seems to be worth the effort to revisit and to re-evaluate the design space under the specific conditions and constraints of today's deep-submicron CMOS technologies: Due to fact that there are strong interactions between the choices on architectural, arithmetic as well as on circuit level and the features on device level of ultra-deep-submicron CMOS technologies (especially leakage and variability), the exploration of the design space becomes rather complex.

Thus, there is a strong demand to come up with a systematic and quantitative approach which allows for exploring the design space with adequate accuracy and in reasonable time. Based on CORDIC architecture templates such a quantitative approach is described in this paper along with selected evaluation results for exemplary design requirements.

This paper is divided into four sections. In the next section, the underlying CORDIC architecture templates are described. Section III refers to the building blocks required for implementations and the characterization of basic components followed by description of the CORDIC cost model. Cost model evaluation and Pareto optimization are described in detail in Section IV.

Template Architectures

The CORDIC algorithm allows for a hardware-efficient implementation of arithmetic and trigonometric functions by rotating a vector in a two-dimensional plane using only shift and addition/subtraction operations [5]. Thereby, a given input vector (x_{in}, y_{in}) is rotated in an iterative approach by a series of predetermined angles. The direction of each elementary rotation is determined similarly to non-restoring division by the result of the previous iteration. After an initialization as $x_{1=}x_{in}$, $y_{1=}y_{in}$ and $z_1=x_{in}$ in each iteration only the basic CORDIC equations

$$x_{i+1}=x_i-y_i.\ d_i.\ 2^{-i},\ y_{i+1}=y_i+x_i.\ d_i.\ 2^{-i},\ z_{i+1}=z_i-d_i.\ e_i \quad (1)$$

have to be evaluated. Where, $e_i=\tan^{-1}(2^{-i})$ is the i-th element of the predefined step angle sequence and the direction d_i is chosen based on the result of the previous iteration. In the so-called rotate mode d_i is determined by the sign of the remaining angle z_i by

$$d_i=\mathrm{sgn}(z_i) \quad (2)$$

As eqn. (1) actually performs so-called pseudo-rotations only, at the very end after M iterations, x- and y-components have to be corrected by a constant factor K: $x_{out}=K\times x_M$, $y_{out}=K\times y_M$.

This algorithm features very attractive characteristics for mapping to a micro-architecture: Being iterative but non-recursive, it can be easily unrolled and pipelined in order to increase throughput and area efficiency. As sets of input operands can be processed independently, it is inherently well suited for multiplexing. Multiplexing in time (time

***Corresponding author:** Upasna Vishnoi, Electrical Engineering and Computer Systems, RWTH Aachen University, Germany, Tel: +49241801; E-mail: vishnoiupasna@gmail.com

Received August 29, 2017; **Accepted** September 12, 2017; **Published** September 22, 2017

Citation: Vishnoi U, Noll TG (2017) Quantitative Optimization of Highly Efficient Dedicated CORDIC Macros for SoCs in Deep-Submicron CMOS Technologies. J Electr Electron Syst 6: 238. doi: 10.4172/2332-0796.1000238

sharing) or multiplexing in space (parallelization) can be applied in order to trade area for throughput.

At arithmetic level, the CORDIC algorithm can be implemented in conventional non-redundant (e.g., two's complement) as well as in redundant (e.g., carry save) number representation promising higher throughput and shorter latencies [5].

Typically, the required throughput rate, total iteration count M, word lengths and potentially the maximum latency are specified according to the application. Together with the architectural and arithmetic decisions the choice of the individual building blocks (e.g., adder type), the choice of the supply and back-bias voltage span a high-dimensional design space. In this design space an implementation has to be found which fulfills the specification and allows for highest area and energy efficiencies.

Two parameterized template architectures are considered in the following: The first one (TPA1), for applying two's complement arithmetic and the second one (TPA2), for carry-save arithmetic. In the following, because of limited space we restrict ourselves to the rotate mode, but all the considerations and evaluations can be easily extended to the other CORDIC mode, too. Figure 1a shows TPA1 for a partially unrolled implementation of the algorithm according to eqn. (1) for an unrolling parameter of *m=2* This unrolling parameter is equal to the number of iterations being implemented in parallel. The extreme cases are *m=1*, i.e., a fully iterative architecture, and *m=M*, i.e., a fully unrolled architecture.

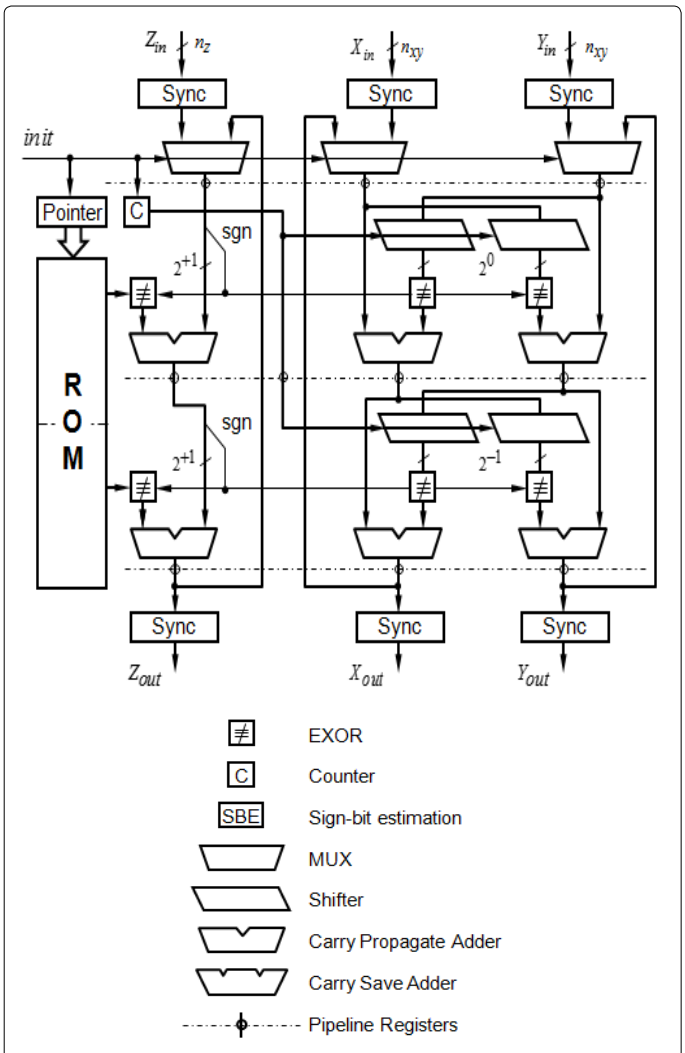

Figure 1a: Two's complement CORDIC template architecture (m=2, p=1): TPA1.

The step angles e_i, required in the z-path are stored in a ROM and the access to it is controlled by a pointer, which is synchronized to the counter for the current iteration state *i*. The latter one is used to control the shifters in the x- and y-paths. In the fully unrolled case (*m=M*), neither shifters nor step angle ROM are required as the shift operations and the step angle values can be realized by dedicated interconnect. Additions are implemented by carry-propagate adders and two's complement subtraction is implemented by using EXOR stages and proper LSB correction. For a complete CORDIC operation with M iterations in total, the whole block is used iteratively for *M/m* times. For that purpose the adder outputs at the bottom are fed back to the input multiplexers on top.

The degree of pipelining is described by the pipelining parameter p, being the number of iterations performed per clock period. In Figure 1a *p=1* iterations per pipeline stage are assumed. As the feedback interconnects potentially get long, an extra pipeline stage is assumed for the buffers driving its RC delay and the input multiplexers. Proper synchronization of input and output data with the pipeline is ensured by the synchronization stages ("Sync") on top and bottom. The final multiplications by constant K which can be implemented by simple shift and add operations are not shown in Figure 1a.

Figure 1b shows the block diagram of TPA2 using redundant carry-save arithmetic. The main problem in applying redundant number representation is that the exact sign of the remaining angle, according to eqn. (2) is not determinable without re-conversion (i.e., vector merging addition) to a non-redundant number representation. Many strategies have been elaborated in the past on how to cope with that issue [5]. In

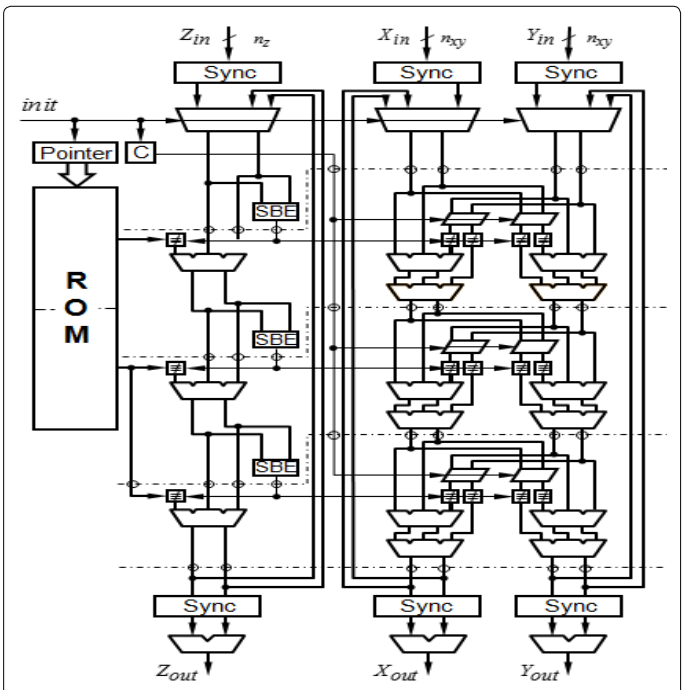

Figure 1b: Carry-save CORDIC template architecture (m=2, $\tilde{m}=3$, p=1): TPA2.

TPA2, sign bit estimation (SBE) from the four most significant z-digits (MSDs) and compensation for resulting misdetections by repeating every second iteration is applied.

Obviously, the potential speed improvement due to breaking up the carry propagation is additionally paid for by the separate sum and carry vectors. As can be seen from Figure 1b, all basic components in the core of the x- and y-paths of TPA2 (shifters, EXORs, adder stages etc.) have to be duplicated. SBE from the four MSDs of the remaining angle z_i, can be implemented by using a simple 4-bit carry-ripple adder. In order to avoid prolongation of the time critical path, in rotate mode it can be pre-calculated one pipeline stage before. Figure 1b shows the case for *m=2* net iterations, requiring $\tilde{m} = 3$ gross iterations, as every second net iteration has to be repeated. Pipelining is shown for *p=1* again, which means that one gross iteration is performed per clock period.

Cost Model Framework

The parameterized template architectures were mapped to a parameterized signal flow graph description from which the macro layout can be generated automatically for any set of parameters, by using an in-house data path generator tool. Obviously, an exhaustive search in the design space for the most efficient implementation is practically not reasonable. An attractive approach is to perform a quantitative design space exploration on a higher abstraction level and based on highly accurate algebraic cost models for timing, energy and area. Input parameters to such a cost model are the specified total iteration count *M* and word lengths *n* as well as the unrolling parameter *m*, the degree of pipelining *p*, type of adders (i.e., template architectures), supply- and back-bias voltages.

Sole accumulation of coarse cost contribution figures from all individual basic building blocks like adders, shifters etc. would not result in a sufficiently accurate cost model. In this work, the elaboration of adequately accurate cost-models will be described taking into account the impact of interconnects and interactions concerning circuit level details.

In the following, a heterogeneous design style is assumed where timing, energy and area critical basic cells are designed as physically optimized macro specific cells and non-critical cells are taken from a standard cell library. But, in general the same methodology could also be used for a synthesized-"standard"-cell based flow.

Basic building blocks and their cost characterization

All basic building blocks were built up using the data path generator tool mentioned above. A widely automatized simulation environment was used for sweep experiments to characterize delay, output slope, dynamic energy and static leakage power by circuit simulations. Sweep-parameters were supply and back-bias voltage, input slope and output load capacitances. Building block specific parameters were for example word lengths, shift amounts and number of buffer stages.

The delay values and output slopes were derived in SS-technology and slow-application corner while the energy and power values were derived in FF-technology and fast-application corner. Even though no fabricated silicon die would feature this combination of cost figures, it is still the adequate worst case approach to ensure meeting specified figures.

The back-biasing experiments were conducted assuming a reverse back-biasing voltage of $V_{BS=}$-0.5 V in order to reduce leakage power in the FF corner and assuming a forward back-biasing voltage of $V_{BS=}$+0.5 Volts in order to reduce critical path delay in the SS corner. The simulation results were appropriately stored in a cost figure database used by the cost model described in the next chapter.

For TPA1, two types of carry-propagate adders were considered: An energy-efficient carry-ripple adder (CRA) and a speed-optimized carry-select adder (CSCA). For TPA2, an area-energy efficient 4:2-compressor built up from cascaded mirror full adders is considered. Figure 2 shows the adder delay times vs. word lengths for a supply voltage of 0.8 V and for the two different cases of back-biasing in 40-nm CMOS technology.

For the shifters, a logarithmic shifter structure is assumed as it is more efficient for large shift amounts. In the case of a partially unrolled micro-architecture with $m \neq 1$ and with m being a power of two, in each iteration state the same effort-reduced shifter can be applied as the remaining shift amounts can be realized by interconnects (Figure 1a and 1b). By that, also in the partially iterative case the shifter costs can be more and more reduced with increased unrolling. However, m to be a power of two is not really necessary, but it is just beneficial for energy- and area-efficiency. The individual shifter cells, as well as the multiplexer and EXOR cells are based on symmetrical EXOR gates. Figure 3 shows the different shifter delays versus word lengths for a

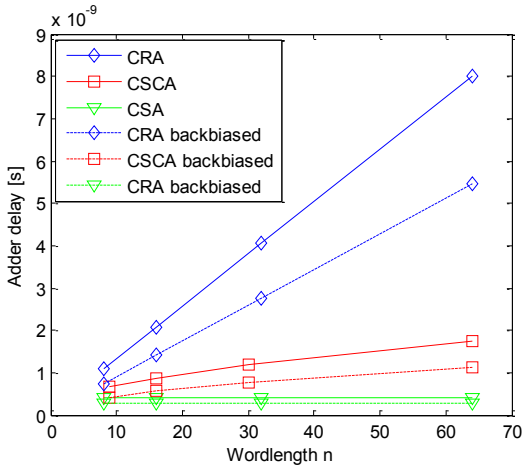

Figure 2: Adder delay vs. wordlength (V_{DD}=0.8 V).

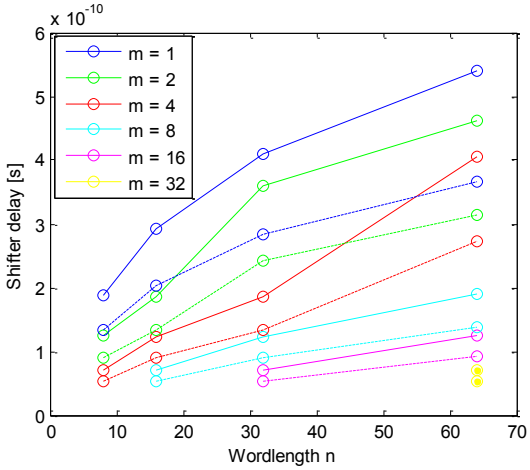

Figure 3: Shifter delay vs. wordlengths (V_{DD}=0.8 V) (solid line: without back-biasing, dashed line: with back-biasing).

supply voltage of 0.8 V and for two different cases of back-biasing in 40 nm CMOS.

CORDIC cost model

Algebraic cost models were elaborated for area, timing and energy. In the area model, the basic building block areas were accumulated according to the parameters of the TPAs. Timing figures were derived from the database according to the TPA parameters as well as supply and back-bias voltage. Polynomial-fit interpolation in-between the database entries, was applied to model the impact of output load capacitances to the timing figures of the basic building blocks. Interpolation in the database is also possible for supply voltage values. In that case, quite accurate results could be achieved by applying an exponential polynomial-fit model. As a next step, also device variability effects, becoming crucial in case of high degrees of pipelining will be considered in the timing-model.

In the energy model, the contributions of the individual building blocks to total power dissipation is calculated by $P_{tot}=E_{dyn} / (\sigma \times f) + P_{stat}$. E_{dyn} and P_{stat} again are derived according to the TPA parameters, supply and back-bias voltage as well as output load capacitance from the database entries. The actual clock frequency *f* can be taken as the maximum clock frequency resulting from the timing model described before or from the specifications. The dynamic part of the energy model can be refined by importing switching activity figures σ derived from bit-true, cycle accurate, MATLAB based system-level-simulations for realistic application data. In future, even the increase of switching activity due to glitching will be taken into account by including adequate glitching information in the simulation result database.

Beside the individual building blocks, the interconnects, especially between the shifter and adder stages in the x- and y-paths, for the feedback loop and in the clock network are significant cost contributors to total area, timing and energy. Therefore, special care has been taken to come up with an accurate modeling of this interconnect impact. From three different floorplans (depending from the type of adder being used) wiring models were derived. The resulting wiring channel dimensions were used in the CORDIC's area model in addition to the accumulated building block areas. In the timing model, maximum wire lengths were used to derive proper output load capacitances for the driving stages. Also, the selection of the proper number of buffer stages is based on this wiring capacitance models. In the energy model, accumulated total wire lengths were used to derive the contributions to dynamic power dissipation from interconnects.

The cost models as well as the evaluation procedures being described in the next chapter were implemented in MATLAB. Figure 4 shows cost contribution breakdowns for an exemplary implementation generated this way.

Cost Model Evaluation

For the cost model evaluation, initially it is assumed that the required throughput is specified in CORDIC operations per second along with the other specification parameters mentioned before. The design space has to be explored in order to fulfill these specifications and to find the implementation featuring highest energy- and area-efficiency. The reciprocal of energy efficiency is defined as energy per CORDIC operation being equivalent to power dissipation per CORDIC operations per second. The reciprocal of area efficiency is defined as silicon area per CORDIC operations per second. These metrics are invariant against changes of throughput rate by parallelization (multiplexing in space) as long as the costs of the demultiplexers and

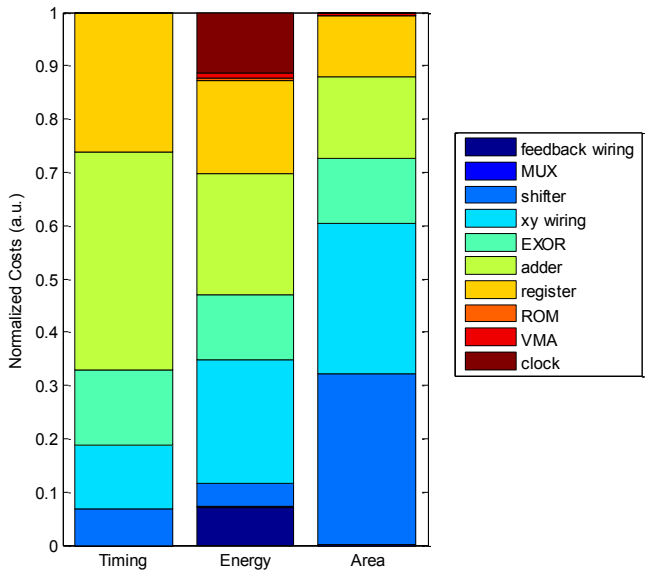

Figure 4: Total Cost breakdown for CSA implementation (M=64, m=32, p=1, V_{DD}=0.8V, no V_{BS}).

multiplexers can be neglected in comparison to the block hardware, which is a reasonable assumption in the fully unrolled case n=m. It is worth mentioning that the product of these two metrics is equivalent to ATE- (area- time- energy-) complexity.

Figure 5a shows the result of a model evaluation (for 5(a)-5(d) total iteration count M and word lengths n is equal to 64) in that design space. The selection of these particular features was done as, in future applications high accuracy requirements are to be expected and differences between the TPAs should become more significant. To restrict complexity, only fully pipelined (p=1) designs without back-biasing are discussed in this work. The unrolling parameter is varied according to m=1, 2, 4, 8, 16, 32, 64. Data points for the individual implementations belonging to one of the three supply voltages (0.6V, 0.8V and 1.0V) are connected by straight lines. Adder types are differentiated by solid (CRA in TPA1), dashed (CSCA in TPA1) and dashed-dotted (CSA in TPA2) lines. Each implementation is assumed to be operated at its individual maximum clock frequency.

Obviously, the fully unrolled implementations (m=M) using the CRA with 1 V and 0.8 V supply voltage, Pareto-dominate all other implementations. 1.0 V gives the best area-efficiency and 0.8 V gives the best energy-efficiency. In case the supply voltage is further decreased to 0.6 V, energy increases significantly due to reduced frequency and therefore higher leakage energy. As can be seen from Figure 5b, back-biasing keeps this increase of leakage under control. Without back-biasing the minimum energy operation point is near to 0.8 V. It is worth noting the different characteristics of the three adder types and that even at such a large word length the CSA-based design does not feature an advantage. The latter is due to the fact that the delay becomes dominated by the other components like shifters and interconnects.

Up to now only throughput rate has been considered as timing-specification, but depending on the application, latency can become another critical specification parameter, too. The maximum clock frequency of the fully unrolled CRA implementation is about 500 MHz at 1.0 V compared to about 1 GHz for the CSA implementation. Taking into account the fact that in the CSA implementation the number of

iterations has to be increased by a factor of 1.5 the saving in latency is limited to about 25%. At the same time, the energy increases by a factor of about two and area by a factor of about 2.5. So, as long as there are no tight latency requirements, the most efficient implementations are based on CRAs.

Figure 5b shows the design space for the case of using back-biasing as explained above. Using back-biasing does not change the relationships discussed so far, which also holds for lowering the degree of pipelining. In the design space discussed above, the absolute throughput rate is not considered: In case the maximum clock frequency of the fully unrolled implementations does not fulfill the throughput specification of a given application, parallelization has to be applied. In fact, all implementations can be parallelized to meet higher throughput specification resulting in an individual factor of required parallelization. In the opposite case of specified throughput rates that are lower than the respective maximum rates, even partially unrolled implementations may meet the specification without parallelization. This way area can be expected to be saved at the cost of energy per CORDIC operation. Actually, the situation is more complicated due to the fact that the parallelization degree is quantized to integer numbers and as on a typical SoC only certain clock frequencies are available. Due to this reason, the ratio of actual to maximum operation frequency of each parallelized implementation varies resulting in different impacts of leakage energy.

Figure 6a shows the situation of a specified throughput rate of 0.1 billions of CORDIC operations per second which for higher unrolling

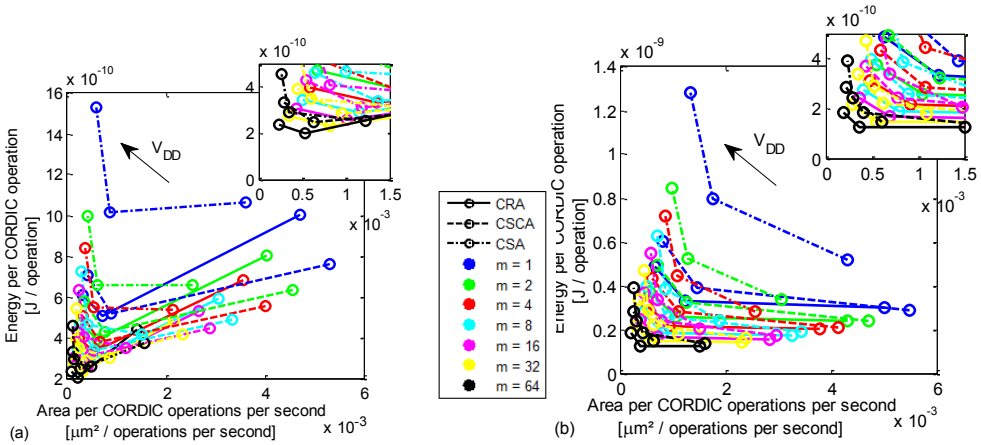

Figure 5: Design space (M=64, p=1, V_{DD}=0.6, 0.8, 1V) (a) without, (b) with back-biasing.

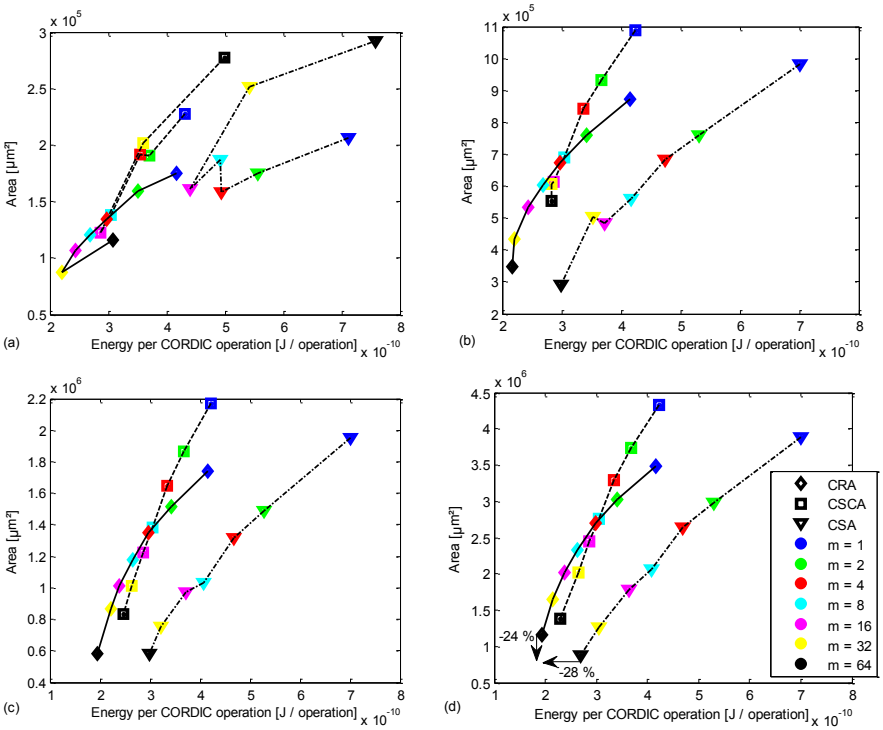

Figure 6: Area-energy complexities for different specified throughput rates: (M=64, n=64, p=1, V_{DD}=0.8V, no V_{BS}) (a) 0.1, (b) 0.5, (c) 1 and (d) 2 billions of CORDIC operations.

factors is already met without parallelization. In addition, due to the fact that the unrolled implementations are operated at relatively low clock frequency leakage energy increases significantly leading to the horseshoe shape of the characteristics. Figures 6b-6d are shown for specified throughputs of 0.5, 1 and 2 billions of CORDIC operations per second, respectively. Again, the quantization and leakage effects result in changing relationships between the costs of the different implementations. Contrary to the discussion at the beginning of this section the Pareto-optimal implementation strongly depends on the specified throughput rate. These observations prove the need for this kind of systematic quantitative exploration. A comparison with an existing design [4] proves that the cost-model described above yields quite accurate figures. Beside the possibility of design-optimization, it allows to conduct many other what-if experiments in seconds of computing time. The approach itself is not limited to CORDIC designs at all, but can be applied to other frequently used SoC hardware building blocks like FFTs etc.

Conclusion

An attractive approach for design space exploration for dedicated CORDIC macros in deep-submicron CMOS technologies has been presented. Based on algebraic cost-models at a high abstraction level, the approach ensures adequate accuracies in very short computation time. To the best of our knowledge, for the first time, a quantitatively detailed comparison between the available arithmetic options has been derived. In future, this approach will be refined for considering device variability effects in the timing part of the model and glitching effects in the energy part of the model.

Acknowledgment

This research work was done at the Chair of Electrical Engineering and Computer Systems, RWTH Aachen University, Germany under the able guidance of Professor Dr Ing. Tobias G Noll.

References

1. Teichmann P, Vollmer M, Fischer J, Heyne B, Götze J, et al. (2009) Saving potentials of adiabatic logic on system level: A CORDIC-based adiabatic DCT. Proceedings of the International Symposium on Integrated Circuits, pp: 105-108.
2. http://cordic-bibliography.blogspot.de/p/cordic-bibliography-other-publications.html
3. Kappen G, Kurz L, Priebe O, Noll TG (2008) Design Space Exploration for an ASIP/Co-Processor Architecture used in GNSS Receivers. Journal of Signal Processing Systems 58: 41-51.
4. Vishnoi U, Noll TG (2012) Area- and energy-efficient CORDIC accelerators in deep sub-micron CMOS technologies. Advances in Radio Science 10: 207-213.
5. Ercegovac MD, Lang T (2004) Digital Arithmetic. Elsevier, pp: 626-635.

Citation: Vishnoi U, Noll TG (2017) Quantitative Optimization of Highly Efficient Dedicated CORDIC Macros for SoCs in Deep-Submicron CMOS Technologies. J Electr Electron Syst 6: 238. doi: 10.4172/2332-0796.1000238