Fine-grain Back Biasing for the Design of Energy-Quality Scalable Operators

Fine-grain Back Biasing for the Design of Energy-Quality Scalable Operators

Daniele Jahier Pagliari, *Student Member, IEEE,* Yves Durand, *Member, IEEE,* David Coriat, *Member, IEEE,* Edith Beigne, *Member, IEEE,* Enrico Macii, *Fellow, IEEE,* and Massimo Poncino, *Fellow, IEEE*

***Abstract*—Energy-quality scalable systems are a promising solution to cope with the small energy budgets and high processing demands of mobile and IoT applications. These systems leverage the error resilience of applications to obtain high energy efficiency, at the expense of tolerable reductions in the output quality. Hardware datapath operators able to reconfigure their precision and power consumption at runtime are key components of such systems. However, most implementations of these operators require manual, architecture-specific modifications and tend to have large power overheads compared to standard designs, when working at maximum precision. One promising design-independent alternative is Dynamic Voltage and Accuracy Scaling, whose adoption, however, is hindered by incompatibilities with standard design flows.**

In this paper, we propose a new methodology for the design of energy-quality scalable operators; our solution leverages runtime tuning of transistors threshold voltages to obtain a fine-grain control of the speed and power consumption of standard-cells within an operator. Thanks to the additional flexibility provided by this fine-grain knob, our method overcomes the main limitations of previous solutions, at the cost of a small area overhead. We demonstrate our approach on a 28nm FDSOI technology; by exploiting the strong effect of back-gate biasing on threshold voltage, we achieve a power consumption reduction of more than 40% compared to the state-of-the-art, for the same precision.

***Index Terms*—Low-power design, energy-quality tradeoff**

I. Introduction

Energy consumption is known to be a major design bottleneck in modern digital systems for mobile and Internet of Things (IoT) applications [1]. In spite of their low energy budgets as battery-powered systems, their performance requirements are constantly increasing. Fortunately, many of these applications are *error resilient*, i.e. they can tolerate some reduction in the quality of computations without a significant degradation of the final results [2], [3]. This error tolerance stems from a number of factors. Devices that process inputs from analog sensors can tolerate quality degradations within the margin of error caused by external noise. In applications targeted for human interaction (e.g. multimedia), a certain amount of quality degradation is allowed by the limited perceptive capabilities of our sense organs. Finally, some tasks tend to be robust against small or occasional errors due to the nature of the involved algorithms, e.g., iterative [4] and statistical applications [5].

The amount and magnitude of tolerable errors are constrained by their impact on final output results, which depends on the target application, as well as on input data and environmental conditions [1], [2]. For a given application, acceptable quality can be quantified by imposing a threshold on relevant output metrics (e.g. a lower-bound on PSNR for multimedia, or on classification accuracy for machine learning tasks) [2]. In that respect, computational quality can be seen as *a new design dimension*, in addition to the traditional ones (performance, area and power/energy) [1], [2]. Recent research has investigated ways to explore the tradeoffs offered by this new dimension, with particular focus on its usage for energy efficiency, at all abstraction levels [6]–[13]. Although the idea of tailoring quality to the task at hand is as old as computing itself, the objective of this new research branch is to investigate general design patterns and platforms to support quality-dependent computation [1], [14]. From the hardware point of view, *datapath operators* (e.g. adders, multipliers, etc.) able to perform computations at reduced precision are fundamental elements of an energy-quality scalable system. One way to realize such operators is through *approximate architectures*, which implement a relaxed version of the intended output function, thus reducing area, delay and power, at the expense of controlled errors [6]–[10], [15]. The main limitation of these approaches is that *the functional relaxation is fixed at design time*. However, error tolerance in real systems most often varies over time, either because an application has variable quality requirements depending on environmental conditions (e.g. input noise, battery level), or because multiple applications are executed on the same hardware [2], [14].

For this reason, researchers have gradually shifted their interest from *approximate hardware* to *precision-scalable hardware*, i.e. operators whose amount of error is reconfigurable at runtime. One possible approach to implement such operators consists of modifying their architecture in order to support multiple precision "modes" [16]–[21]. However, these solutions are based on manual redesign and are difficult to generalize or automate. Moreover, they often have large overheads in terms of latency, area and power at maximum precision [11], [22].

Dynamic Voltage and Accuracy Scaling (DVAS) is an alternative that relies mostly on technological knobs, with minimal architectural modifications [11]–[13]. In DVAS, power savings are obtained simply scaling the supply voltage. The input dynamic range (i.e., bitwidth) is reduced to cope with the delay

D. Jahier Pagliari, E. Macii and M. Poncino are with the Dipartimento di Automatica e Informatica, Politecnico di Torino, Torino, IT e-mail: {daniele.jahier,enrico.macii,massimo.poncino}@polito.it

Y. Durand, D. Coriat and E. Beigne are with the CEA LETI, Minatec Campus, Grenoble, FR {yves.durand,david.coriat,edith.beigne}@cea.fr.

Manuscript received January 1, 1970; revised January 1, 1970.

increase caused by voltage scaling, at the expense of some precision loss due to quantization errors [11]. This simple idea has proven very effective, but its usability is limited by the fact that, in realistic implementations of hardware operators, a high percentage of timing paths have a delay close to the critical one (the so-called *wall of slack* phenomenon) [23]. Moreover, in DVAS, each independent operator must be placed in a separate supply voltage domain, unless the design is extensively (and manually) modified, through the insertion of additional logic to support critical path balancing [11].

In this paper, we extend the work of [24] and propose a novel technique for designing precision-scalable operators. We focus specifically on the *implementation of single operators* that support multiple precision modes. The decision on the optimal precision to use is outside the scope of this work and should be addressed at application level [1], [4], [5], [22].

Our method combines DVAS with dynamic threshold voltage (V_{th}) scaling. Modifying the V_{th} of standard cells allows to selectively speed up the timing-critical regions of the operator, thus increasing the usable dynamic under scaled voltage. In principle, our solution can work with any technological knob for dynamic V_{th} tuning. However, in this work, we demonstrate it on state-of-the-art FDSOI technology using dynamic back-gate biasing [25], [26]. Thanks to the much finer spatial granularity at which the tuning of V_{th} can be applied, compared to dynamic voltage scaling, our method allows to overcome the main limitations of DVAS. On one hand, it contrasts the wall-of-slack, allowing larger bitwidth operation at iso-voltage. On the other hand, it allows one to configure the precision and power consumption of different operators within the same voltage domain independently, without the need of level shifters. Although dynamic V_{th} assignment has been already used in previous research to fine-tune speed and power [28], [29], to the best of our knowledge this is the first time that this knob is used for quality-scalable computing.

The proposed technique is fully automated and integrated with state-of-the-art tools and yields a power reduction of about 40% with respect to standard DVAS.

The rest of the paper is organized as follows. Section II summarizes the required background on precision-scalable operators. Section III provides the motivation for the problem we are addressing, which is then detailed in Section IV. Section V describes the proposed automated design flow and focuses in particular on the identification of the optimal configuration of technological knobs (supply and threshold voltages) for a given precision level. Experimental results are reported in Section VI and Section VII concludes the paper.

II. Background and Related Work

A. Architectural Solutions for Energy/Quality Scalable Hardware Operators

At the architectural level, the traditional paradigm to design energy/quality scalable operators consists of two main concepts. First, the hardware complexity of the *active part* of the operator is reduced when it operates in low-precision mode, by either disabling or replacing some of its logic. This translates in a reduction of dynamic power and delay. Second, the extra slack made available by the simplification of the logic is then exploited for further power reduction by means of supply voltage scaling. Most efforts in the design of precision-scalable operators focus on adders and multipliers, due to their ubiquitous presence in digital circuits.

A popular class of adders is based on enhancing an approximate unit (i.e. a modified version of the adder with a predefined, fixed error) with *error recovery* circuitry, selectively activated when precise results are needed [16], [18]. Since error recovery happens in the next clock cycle with respect to the approximate output computation, these designs exhibit a significant latency penalty. Moreover, the error recovery circuitry also introduces a non-negligible power overhead.

An alternative solution for adders is reconfigurable carry-chain segmentation [20], in which the adder is split in smaller sub-adders and the carry signal from each sub-adder to the next can be selectively replaced with the output of a simpler *carry prediction* circuit.

Concerning multipliers, the architecture of [19] relies on disabling columns of the partial product matrix by means of signal gating. In [21], alternatively, partial product accumulation is implemented by means of special adders, able to produce an approximate result and a *correction output* that, depending on the selected precision mode, is used to reduce the accumulation error.

A general approach for the design of precision-scalable operators is proposed in [17]. This paper applies some of the previously described techniques (e.g. carry-chain segmentation) to different operators such as Multiply And Accumulate (MAC), L1 Norm, etc.

Using architectural modifications to implement precision-scalable hardware has several drawbacks. First, most of the proposed techniques apply only to specific operators (e.g,, the works of [19] and [21] are only for array multipliers). Second, the additional circuitry required to implement multiple precision modes often results in significant overheads. Consequently, these operators typically incur in large power (and sometimes also delay/latency) overheads when working at maximum precision [16], [18], [22]. Last, most of these solutions only support a small number of modes (very often only one approximate and one precise mode).

B. Dynamic Voltage and Accuracy Scaling

Dynamic Voltage and Accuracy Scaling (DVAS), first proposed in [11] addresses some of these issues. As in previous solutions, DVAS leverages voltage scaling to reduce power consumption in the operators. However, rather than modifying the architecture, the authors propose to amortize the increase in delay due to voltage scaling through the reduction of the *input dynamics*, specifically by *gating some of the LSBs of each input operand.* This simple solution works well in theory, since in most datapath operators the longest timing paths are those connecting input LSBs to output MSBs. The results of [11] and [22] show that, despite its simplicity, input dynamics reduction, as in DVAS, is almost always superior to approximate (or precision-scalable) architectures. Conceptually, this happens because the latter, although using simpler

logic with respect to a standard design, still invest a significant amount of power to compute erroneous, and therefore useless, information. Conversely, DVAS directly *avoids* computing the outputs related to LSBs. Moreover, in principle, DVAS has almost zero overheads in terms of area and power/delay at maximum precision, and can be applied to *any* operator in an automatic way, as long as the mentioned relation between input/output bits and path delays holds.

Dynamic Voltage Accuracy and Frequency Scaling (DVAFS) is a further improvement of DVAS, proposed in [12]. The idea is to reuse the disabled part of an operator to perform other computations. For example, when a 32-bit multiplier is working at a reduced precision of 16-bit, the LSB-part of the circuit can be used to perform *another* 16-bit multiplication in parallel, by means of some additional gating logic to decouple the two halves. The critical delay of the operator is still reduced thanks to the separation of the two operations (at least in principle, see Section III), allowing to scale the supply voltage as in DVAS. Moreover, thanks to the introduced parallelism, the clock frequency can be reduced of a factor of 2 while maintaining the original throughput, for additional power savings.

Although very effective, DVAFS is a less flexible approach compared to DVAS, as it requires some manual architectural modifications, and can only be applied to circuits whose netlist can be modified to support subword-parallel operation (e.g. adders, multipliers), but not to more complex accelerators. For this reason, in the following, we mostly consider the combination and comparison of our technique with DVAS.

C. Fine-Grain Speed/Power Control Using Back-Biasing

Using supply voltage (V_{DD}) scaling as a knob to regulate power (and indirectly precision) has some technological downsides. First, V_{DD} scaling has a relatively coarse granularity and cannot be applied to arbitrarily small regions, due to the power and area overheads of level shifters: practical voltage islands have sizes in the order of hundreds of standard cell rows [30]–[32]. Second, the supply voltage cannot be tuned in arbitrarily small increments; indeed, to contain generator costs, V_{DD} is normally selected among a pool of few (≤ 10) pre-generated voltages (V_{DD} hopping) [32].

A finer-grain knob that can be exploited to tune the power/precision operating point is the regulation of the threshold voltage V_{th}. At runtime, this can be achieved with *body biasing* (BB), i.e. controlling the voltage applied to the body contact of a MOSFET transistor. Detailed analyses of the relations between BB and the delay and power consumption of a MOSFETs can be found in [27], [28].

In classical bulk CMOS, BB is not a very powerful knob, since applicable voltage ranges are limited ($\approx \pm 300mV$) and consequently the V_{th} is only moderately tunable. On the contrary, runtime V_{th} tuning is much more effective in advanced technologies like FDSOI. For example, in the *Ultra-Thin Body and Box* (UTBB) FDSOI technology [25] used in this work, the BB voltage range can be as high as $\pm 2V$ and the *body factor* (i.e., the sensitivity of V_{th} to BB) reaches $85mV/V$ [26]. This is due to the presence of the Buried Oxide (BOX) layer, which removes body-source P/N junctions and acts as a *back-gate* (hence BB is referred to as *back-gate biasing* or just *back-biasing* in FDSOI technology).

Such a large BB voltage range can be leveraged for dynamic tuning of the performance/power tradeoff. Specifically, in the aforementioned technology, Forward BB (FBB) lowers V_{th} and reduces the delay of a transistor, at the expense of an increase in sub-threshold leakage currents, whereas Reverse BB (RBB) has the opposite effect.

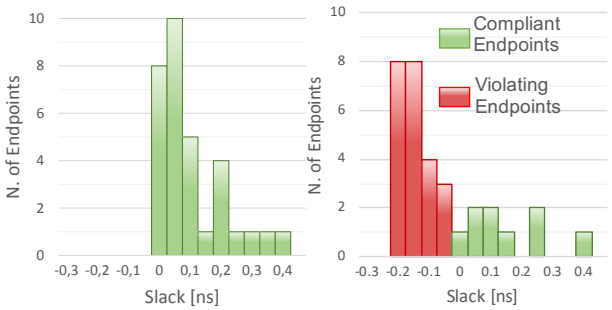
With respect to V_{DD} scaling, BB can be applied at *much finer granularity*, since no level-shifters are required. However, there are still technological constraints that prevent its application to single devices. As a matter of fact, domains with different BB voltages must be separated by *guardbands* that have a size comparable to that of a standard cell [26]. Further technological details about the specific technology used in our work will be given in Section IV-D.

III. Motivation

As discussed in Section II, DVAS can be regarded as the state-of-the-art technique for implementing energy-quality scalable operators. However, DVAS incurs in a major issue when integrated in a standard EDA flow.

As a matter of fact, synthesis and place and route (P&R) tools normally optimize the most timing-critical paths of a circuit for performance, whereas less critical paths are exploited for area and power optimization. Therefore, gates belonging to short paths are mapped to standard cells with lower output currents (e.g. smaller), with the objective of reducing area/power at the expense of an increase in delay [33]. Consequently, the delay of non-critical paths tends to increase, to the point where their slack is comparable to that of critical ones, causing a phenomenon known as *wall-of-slack* [23].

An example is shown in Figure 1a, which reports the slack histogram of the endpoints of a 16x16-bit multiplier. The histogram is obtained after P&R, at the nominal implementation supply voltage ($1V$).



(a) $V_{DD} = 1.0V$ (b) $V_{DD} = 0.8V$

Fig. 1. Endpoint slack histogram for a 16x16-bit multiplier.

The effect of the wall-of-slack on DVAS is that the input dynamic usable without incurring in timing violations decreases rapidly when V_{DD} is scaled. As an example, Fig. 1b shows the multiplier endpoint slack histogram after the supply voltage has been downscaled to $0.8V$. Red bars correspond to endpoints violating timing constraints. Such a large amount

of violating paths indicates that, in order to restore timing compliance, DVAS should reduce the input dynamics drastically, even for such a limited V_{DD} reduction. Reversing the perspective, in order to work at high-precisions with DVAS, V_{DD} cannot be scaled more than few tens of mV.

One solution to cope with this issue could be to modify the synthesis flow, so to avoid the creation of the wall-of-slack [11]. However, this implies preventing the synthesis tools from performing standard power and area optimizations on short paths, which would result in circuits that consume more power than what they could at maximum precision. Moreover, *to what extent the slack histogram should be modified* strongly depends on which precision configurations are used most often. Intuitively, a substantial modification of the path delays may allow lower voltage operation at low precision, but will clearly incur larger overheads at high precision.

Finally, another limitation of DVAS occurs when multiple operators within the same design must work with independently tunable precisions. In such case, each operator must receive its own supply voltage, and therefore must be placed in a dedicated *voltage domain*. In MOS technology, voltage domains are separated inserting level shifters, which introduce significant power overheads, as mentioned in Section II-C.

In this work we propose a new technique for the implementation of precision-scalable hardware, that overcomes the main limitations of DVAS. Specifically, we maintain the underlying runtime precision-setting method used by DVAS, that is, *zero-gating some input LSBs* depending on the required precision [11]. The novelty of our method is in the way in which reduced bit-width operation is exploited to reduce power; more specifically, the use of a fine-grain power/speed control mechanism such as BB to achieve a better tuning of the power/precision tradeoff.

IV. Fine-Grain Back Biasing for Quality-Scalable Operators

A. The Need of Fine-Grain Delay Control

Unless some countermeasure is employed to contrast the wall-of-slack, the lowest V_{DD} usable by an operator that relies on input LSB-gating to provide multiple precision modes will often be very close to the nominal one, as detailed in Section III. In order to overcome this limitation without modifying the synthesis and P&R algorithms, a *fine-grain control* of the speed and power of different parts of the operator is needed. To motivate this statement, we first show that additional power savings could be obtained using *multiple* $V_{DD}s$ within the same operator. We then go on to explain that the same benefits can be achieved by fine-grain BB, but with much smaller overheads.

In a reduced-precision operator, many timing paths are disabled as a consequence of zeroing the input LSBs, and therefore they are irrelevant for timing. These, plus the few paths that are not critical despite the wall-of-slack, could theoretically receive a smaller V_{DD}, while the rest of the circuit is supplied with a "conservative" (higher) V_{DD} to prevent timing violations (see Figure 2). However, such a partitioning of the circuit in multiple V_{DD} *domains* in unfeasible at this granularity, due to the large power overheads associated with level shifters [30]. Consequently, the "conservative" value of V_{DD} must be applied to the *entire circuit*, resulting in less power savings than what could potentially be achieved.

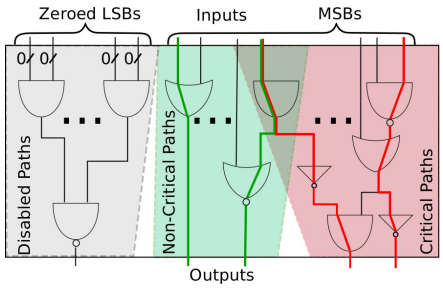


Fig. 2. Conceptual subdivision of the timing paths within an operator working at reduced bit-width. Some of the paths in each group are highlighted.

To solve this issue, a different method to *selectively* alter the speed of specific paths of the circuit (and consequently the power consumption of the cells belonging to them) is required. This will allow the designer, for example, to use a lower value of V_{DD} in the entire circuit, and recover the timing violations by speeding up *only the critical paths*.

B. Dynamic V_{th} Tuning Through Back-Bias Domains

A technological solution to obtain the required fine-grain tuning of power/performance is provided by dynamic threshold voltage tuning by means of back-biasing. As explained in Section II-C, BB domains only require a separation guardband but no level shifters and can therefore be applied at a fine granularity, thanks to a much smaller power overhead compared to the use of multiple V_{DD} domains.

Topologically close cells must still be grouped into domains with a common back-bias in order to contain *area* overheads. However, considering that separation guardbands have a thickness comparable to the height of a cell, the minimum size of a BB domain is in the order of few tens of rows.

A conceptual depiction of our methodology is shown in Figure 3. We propose to split an operator in multiple BB domains (i.e. V_{th} domains), each with an independent back-bias control, whereas the entire circuit shares a single V_{DD}. The additional fine-grain back-bias knob is leveraged to selectively speedup the (precision dependent) timing critical paths of the circuit, by applying low V_{th} to a subset of domains, shown as shaded red areas in the figure.

Thanks to back-biasing, the operator will be able to use higher precision compared to DVAS, for the same value of V_{DD} and consequently achieve higher power efficiency. To better highlight this advantage, Figure 3 also reports the Pareto frontiers obtainable with DVAS (i.e. without fine-tuning) and with our method, i.e. the curves connecting the minimum power points for each precision, considering all possible combinations of knobs. For example, in the figure, DVAS allows to lower the supply voltage ($V_{DD,1} \rightarrow V_{DD,2}$) only for a 7-bit precision, whereas the selective speed-up offered by BB allows the design to run at $V_{DD,2}$ already at 14-bit.

In this work, we assume that each domain can be assigned one out of two possible V_{th} values: Standard V_{th} (SVT), which

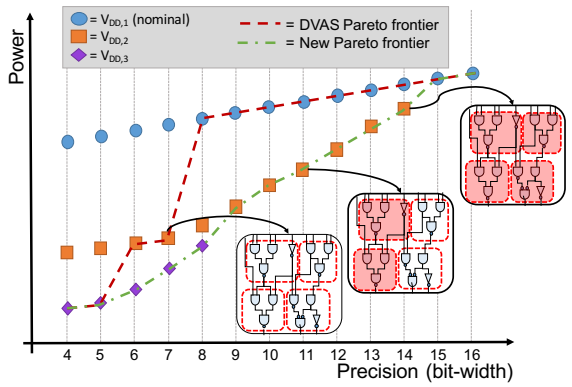


Fig. 3. Conceptual scheme of the proposed methodology.

is considered the nominal condition and Low V_{th} (LVT), which is the "boosting" (faster and more leaky) condition. In 28nm FDSOI with flip-well devices [26], we map SVT to No Back-Bias (NoBB) and LVT to Forward Back-Bias (FBB).

Limiting the choice to two V_{th} values has several advantages with respect to more complex assignments. First, it simplifies the search of the optimal V_{th} assignment to domains for a given precision. Second, it makes the generation of back-bias voltages easier, as we will show in Section IV-D. Nonetheless, our methodology can also be applied to more than two V_{th} values, with increased flexibility at the expense of higher complexity. Similarly, it can also be implemented using other forms of V_{th} tuning (e.g., body biasing in bulk CMOS).

It is worth re-emphasizing that our approach is not alternative but rather *complementary* to DVAS, since we still apply voltage scaling to the entire design and then use BB to fine-tune the consumption and speed of different parts of the circuit. The same is true also for DVAFS, which benefits from a reduction of the wall-of-slack as well, and can be implemented *on top* of our technique. Of course, the effect of DVAFS on timing paths slightly differs from the one discussed in Section IV-A. In particular, disabled paths are not simply those driven by a zeroed input bit, but rather those that are never activated due to the "split" of the operator into two sub-circuits [12].

C. The Circuit Partitioning Problem

The use of back-biasing for V_{th} tuning brings about new design issues. First of all, we need to determine the maximum number N of V_{th} domains that can be inserted in a given operator. This number depends on a user-defined acceptable area overhead, due to separation guardbands. Even once N is defined, however, there are still many degrees of freedom available regarding the possible shapes of the V_{th} domains. The definition of these regions is a fundamental issue in the proposed methodology.

The basic problem of partitioning a circuit into multiple regions for selective application of a knob (V_{th} tuning, in our case) to control some design tradeoffs (power/precision, in our case) has been already studied the literature for various knobs and tradeoffs [34]. However, this particular instance, i.e. the application to precision-scalable designs, introduces several new elements of complexity.

The ideal goal of the partitioning process can be formulated as follows: *build a partitioning that ensures timing compliance with minimum power consumption, considering all relevant precision configurations (i.e. bit-widths) and V_{DD} values*. Here, a relevant configuration is one that will be actually used in applications, e.g. 1-bit precision might not be interesting.

This goal is achieved if the partitioning can isolate exactly the cells that define the critical paths (the red area in Figure 2). This must be true for all relevant precisions at their corresponding optimal V_{DD}, i.e. the supply voltage that, after boosting critical cells with FBB, allows to reach timing compliance with minimum total power for that bit-width.

However, finding this optimal solution is in general impossible. In fact, the physical partitioning of the circuit in domains must be done at design time, and is *single* for a given operator. In contrast, each precision mode implies a different set of critical paths, and a solution that is optimal, (i.e., that allows to speed up *only* the critical cells) for a given precision, might not be optimal for others. The only solution that would always guarantee perfect isolation of critical cells for all precisions is the independent back-biasing of each cell, which is technologically unfeasible.

Moreover, the partitioning process also impacts the other figures of merit of the circuit (timing, dynamic power consumption, etc.), due to the fact that V_{th} domains are physically isolated regions in the die. In the partitioned circuit, cells should be kept as close as possible to their optimal locations (i.e., those determined by a standard placement algorithm) in order to minimize this impact [30], [34].

The fundamental criterion we use to drive the circuit partitioning in BB domains is to keep the figures of merit of the precision-scalable operator as close as possible to those of the original design, when the former is used in maximum-precision mode. To this end, we only consider partitioning solutions in which cells are *minimally displaced* with respect to a standard P&R.

The default choice used in this work consists of the simplest possible partitioning, i.e. a *regular tiling* of the operator into N BB domains. In this solution, each domain is assigned an identical area of rectangular shape, equal to a fraction $1/N$ of the total operator surface. Cells of a standard P&R are mapped to the *closest* domain in the newly created grid of tiles, as shown by the arrows of Figure 4 for the case $N = 4$. This solution has the desirable property of a regular structure, which eases the physical implementation [35]. Moreover, it is easy to automate and embed in a standard EDA flow.

The drawback of regular tiling is that it does not consider the spatial distribution of timing paths. Therefore, for a given precision and V_{DD} combination, each BB domain will generally include a mix of critical cells (highlighted in red in the figure for one particular combination) and non-critical/disabled cells. In order to meet timing constraints, the *entire* BB domain must be assigned to LVT/FBB, resulting in a power overhead due to the unnecessary speed-up of non-critical and disabled cells. However, thanks to the relatively small granularity of BB domains, this overhead is normally smaller with respect

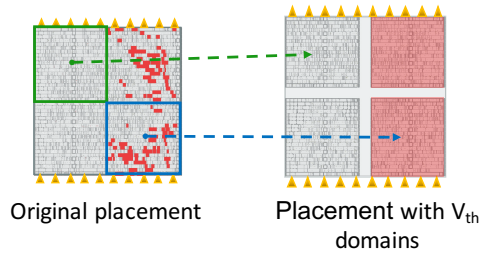


Fig. 4. Regular tiling partition of an operator into multiple V_{th} domains, and example of critical cells and boosted domains, for one particular V_{DD} and precision combination.

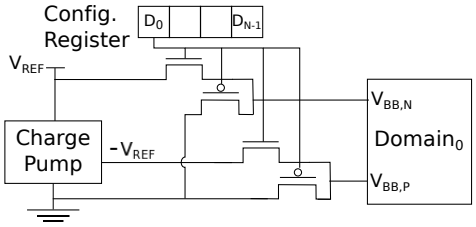


Fig. 5. Block diagram of the back-bias generation circuit. FBB biasing voltages: $V_{BB,N} = V_{REF}$, $V_{BB,P} = -V_{REF}$.

to that of raising V_{DD}. The domains that must be boosted to "cover" all the critical cells in the example of Figure 4 are shown with a red-shaded background.

One benefit of a regular tiling is flexibility: since the partitioning is determined a priori, without a specific target precision or V_{DD}, it tends to perform reasonably well *on average* for all precisions. Conversely, an irregular tiling that tries to better match the spatial distribution of the critical cells will be very effective for the specific precision/V_{DD} these cells refer to, but it will simply be inappropriate for other precision/V_{DD} points. Moreover, since the set of critical cells also depends on clock frequency, process, temperature, etc, an irregular solution will also not adapt to changes in the operating conditions.

Nevertheless, in our results, we also evaluate the performance of an irregular approach. Specifically, we obtain this partitioning using a regular grid as a starting point, and *merging* some of the BB domains according to their criticality in different precision configurations. The description of this *criticality-driven partitioning* is postponed to Section V-C, as it requires some of the concepts introduced in Section V.

D. Enabling Technologies to Support Multiple BB Domains and Associated Overheads

As already briefly discussed in Section II, the practical implementation of multiple BB domains requires two main enabling technologies, i.e. separation guardbands and back-bias generators. Each of these elements introduces some overheads compared to a standard design, as discussed below.

Guardbands: Areas with independent BB voltage require a separation (guardband), in order to insert deep well trenches. In UTBB 28nm, the minimum BB domain guardband width is $2\mu m$, but in our work we chose a more conservative width for better quality of results. Specifically, we chose a width equal to three times the height of a standard cell row, i.e. $3.6\mu m$. More information on the technological details of our target process node can be found in literature [25], [26].

Back-Bias Generators: Figure 5 shows a block diagram of the circuit used to assign the back-biasing voltage to each BB domain at runtime, in the particular case of a dual assignment (NoBB or FBB). Each domain must receive two biasing voltages, $V_{BB,N}$ for NMOS transistors and $V_{BB,P}$ for PMOS. Since $V_{BB,N}$ and $V_{BB,P}$ can only assume two values, back-biasing voltages can be pre-generated, without the need of a controllable voltage generator, such as [37], and can be shared by multiple operators. In particular, charge pumps can be used to generate the negative voltage required for $V_{BB,P}$, as shown in the figure. Two power switches allow the connection of either NoBB or FBB voltages to the bias pins in each domain (shown only for one domain in the figure). Bias voltages are then routed to N-well and P-well through *well tap* cells placed at regular intervals on the die. Power switches are controlled by a configuration register, whose width is equal to the number of domains N. The register is memory-mapped, and is written when the precision of the operator must be modified. Thus, the area overheads for generating back-bias voltages are limited to a few switches and flip-flops, and are negligible compared to those caused by the insertion of separation guardbands. Notice that we do not consider V_{DD} generation, as the latter is already present in most modern systems.

Runtime assignment of back-bias voltages also introduces a *time* overhead, since the switching between two different assignments is not instantaneous. This overhead is due to the loading of well capacitances by the circuit of Figure 5. As reported in Section VI, thanks to the small size of the BB domains considered in this work, the transition time is in the order of tens of nanoseconds. This overhead is acceptable, considering that the precision requirements are expected to vary for different applications (or portions thereof) and not for single instructions [2], [14].

V. Design Flow

In this section, we describe a fully automated EDA flow to implement precision-scalable operators based on our proposed V_{th} tuning approach. A high-level block diagram is shown in Figure 6; the entire flow is implemented leveraging commercial EDA tools for P&R and timing analysis.

The main input to the flow is a gate-level netlist describing the original (maximum precision) operator. The user needs to specify four additional parameters:

1) the V_{DD} values that can be applied to the entire operator (a single value or more than one, if our method is combined with DVAS).
2) the FBB voltages for N-well and P-well.
3) the relevant precision configurations to be considered throughout the flow (b_i).
4) the number of V_{th} domains N, which determines the acceptable area overhead.

The sequence of operations can be split in two main parts. The first *implementation phase* (green background, on the left of the figure), consists of the physical design of the

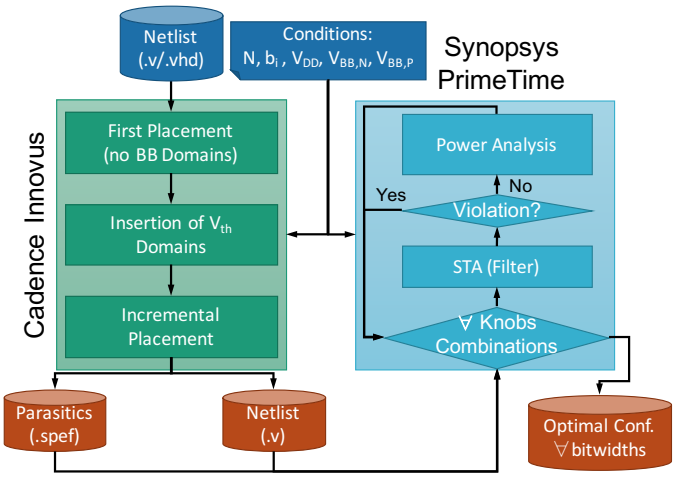


Fig. 6. Proposed design flow.

operator, with the insertion of V_{th} domains. The result of the implementation then undergoes a second *analysis phase* (blue background, right), in which the best assignment of technological knobs for each bit-width is determined. In the next two sections, we describe these two phases in detail.

A. Implementation Phase

First placement: The implementation phase begins with a first placement of the operator at nominal voltage, *without* V_{th} domains. This operation is performed by the P&R tool using its standard algorithms. Hence, cells are placed according to the usual constraints of timing, area and power. The information gathered in this step is used during V_{th} domains creation to assign the cells to the closest domain, as shown in Figure 4.

Insertion of $\mathbf{V_{th}}$ domains: With the placement available, V_{th} domains can be created. The die area reserved to the operator is also enlarged to insert separation guardbands. The details of this phase depend on the selected circuit partitioning technique (Section IV-C). When using regular tiling, domain creation is straight-forward and simply consists of the insertion of guardbands at equally spaced intervals. The details for an irregular partitioning are provided in Section V-C.

Incremental placement and routing: Finally, an incremental placement is executed on the modified circuit. In this step, the tool takes into account the newly inserted V_{th} domains and their possible operating modes (NoBB or FBB), and consequently modifies gate sizing, position, etc., in order to meet all timing (setup and hold) and DRC constraints. The tool is also instructed to insert well taps for the connection of back-bias voltage rails to the different domains. Finally, the implementation is completed with routing.

The outputs of the implementation phase are the placed and routed netlist of the operator, including V_{th} domains, and the corresponding parasitics to be used in the following analysis.

B. Analysis Phase

We propose two algorithms to identify the combination of knobs (i.e. back-bias voltage assigned to each domain, and global V_{DD}) that minimizes power for each bit-width b_i. An optimal method based on exhaustive exploration, which has exponential complexity in the number of domains N, and a greedy heuristic with linear complexity. The two methods are detailed in Sections V-B1 and V-B2. The flow of the analysis phase is the same regardless of the search algorithm, and is composed of two main steps.

Static Timing Analysis (STA) Filter: For a given combination of knobs, the operator is analyzed using static timing analysis, checking for setup and hold timing compliance. In case a violation is found, that combination is discarded. For example, a configuration in which NoBB (high V_{th}) is assigned to all domains and V_{DD} is set to a low value (e.g. $0.6V$), will probably incur in violations at maximum precision (e.g. $b_i = 32$ bit). STA is quite fast (in the order of 0.1s for the scale of our operators), and even faster for low precision netlists, thanks to the fact that many paths are disabled. On average, we have observed that about 75% of the configurations are filtered by STA, hence this step allows to greatly reduce the total analysis time.

Power Analysis: Configurations that pass the STA filter are then analyzed for total power. In this phase, switching activity data from simulations can be optionally loaded to increase the estimation accuracy. If the configuration consumes less power than the previous best for its precision, it is saved in an internal database. When all configurations have been analyzed, the tool produces in output a list of configurations (one for each relevant precision), with the corresponding knobs settings.

Note that this process (specifically the STA filter) is dependent on the target clock frequency f_{clk}. If the operator is to be used at multiple frequencies, the analysis must be repeated.

1) Exhaustive Search: The easiest method to identify the optimal knobs configuration is to analyze all possible combinations of domains V_{th} and global V_{DD} exhaustively. This requires to evaluate all 2^N combinations of NoBB/FBB assignment to the N domains, for each V_{DD} and b_i. An example is shown in Figure 7, where red squares represent FBB domains.

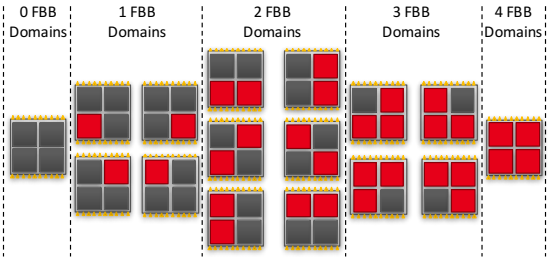


Fig. 7. Example of the back-biasing configurations considered in exhaustive search, with $N = 4$, for a given (b_i, V_{DD}) point. Red squares correspond to boosted domains.

Therefore, the overall analysis complexity is $O(2^N \cdot B \cdot N_{V_{DD}})$, where B is the number of considered bit-widths, and $N_{V_{DD}}$ is the number of supply voltages to be explored.

Since these values are all relatively small, and since STA filters out most of the combinations, exhaustive exploration is feasible. Indeed, N is in the order of a few tens (more domains would generate unacceptable area overheads). B in the worst case coincides with all possible bit-widths (e.g. from 1 bit to 32 bit) but is typically smaller. $N_{V_{DD}}$ depends on the step

and range of variation of V_{DD}: assuming a 100mV step and a range between 0.6V and 1.0V, $N_{V_{DD}} = 5$. With these values, the number of points to consider is in the order of thousands. Power estimation including importing of VCD traces takes about 1s on our Intel Xeon E5-2630@2.4GHz with 128GB DDR3, hence the entire analysis can be carried out in tens of minutes or few hours. This time is acceptable for a final stage of design, and it ensures that the *optimal* combination of knobs is detected. However, during exploratory phases (e.g. when deciding the best value of N for a given operator), a faster heuristic solution described in the next section is preferable.

2) Dual-V_{th} Guided Search: Dual-V_{th} assignment (multi-V_{th} in general) is a design-time solution for leakage power optimization, applicable when the target standard-cell library includes cells designed to have different V_{th} values [36], [38]. It consists of mapping gates in critical timing paths to low-V_{th} devices for performance maximization and the remaining gates to high-V_{th} devices for leakage recovery. Differently from dynamic V_{th} assignment (e.g. back-biasing), it can be applied at the single cell granularity.

We exploit the algorithms normally used for static dual-V_{th} as a *guidance* for our analysis. To do so, we provide the P&R tool with library characterizations in NoBB and FBB conditions, mapping them to high-V_{th} and low-V_{th} respectively. With these libraries available, we instruct the tool to implement static dual-V_{th} assignment, exactly as if the different V_{th} values were obtained technologically (rather than through biasing).

After running the algorithm, the circuit contains a mix of NoBB and FBB cells, with the latter mostly located in critical paths. We then use *concentration of FBB cells* as an indication of the domains that are most "important" for a particular bit-width and V_{DD}; this allows the designer to drastically reduce the number of configurations to test. The overall algorithm can be summarized as follows. For a given b_i and V_{dd}:

1) We disable the appropriate timing paths for b_i, and have the synthesis tool to execute a dual-V_{th} assignment on the operator.
2) If dual-V_{th} did not insert *any* FBB cell, we just analyze the configuration in which all domains are in NoBB.
3) Otherwise, we start boosting the domain which contains the highest number of FBB cells.
4) We analyze this configuration as described in Section V-B.
5) If timing compliance is reached, we stop. Otherwise, we apply FBB to the next domain with the highest number of FBB cells, and repeat from step 4, until one configuration is compliant, or all domains have been boosted.

An example of this procedure is shown in Figure 8, where blue squares represent individual FBB cells identified by dual-V_{th}. With this method, the number of analyses performed in the worst case for a given b_i and V_{DD} is linear in the number of domains. Thus, the overall complexity is $O(N \cdot B \cdot N_{V_{DD}})$, and with the same setup described in Section V-B1, the number of configurations to test reduces to few hundreds. The time to execute single-cell dual-V_{th} with our P&R tool is less than 10s, and this step is only executed *once* for every b_i and V_{DD}.

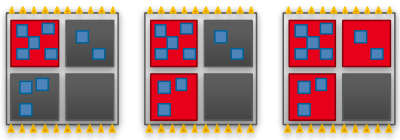


Fig. 8. Example of the back-biasing configurations considered in guided search, for a given (b_i, V_{DD}) point. Red squares correspond to boosted domains, while blue squares correspond to individual boosted cells according to a dual-V_{th} assignment algorithm.

It is worth emphasizing that we do not boost all domains that contain at least one FBB cell *in a single step*. This is because when we apply FBB to an entire domain, more cells than those strictly needed (blue squares) will be sped-up. In practice, we observed that these additional cells often "compensate" for the fact that some of those identified by dual-V_{th} in other domains are not boosted. Thus, timing compliance could be reached with a smaller number of domains.

This procedure does not necessarily identify the optimal configuration of knobs. Firstly, because the single-cell dual-V_{th} algorithms embedded in EDA tools are heuristic [36]. Secondly, because the concentration of FBB cells might not be sufficient to identify the optimal domain to boost. For example, if the concentration of FBB cells in two domains is similar and small, boosting either one might be sufficient for timing compliance, due to the "compensation" effect described above. However, the power increase caused by forward back-biasing will generally be different among domains. Therefore, the least leaky domain should be selected in this case, rather than the one containing more FBB cells. Nonetheless, we show in Section VI that this heuristic identifies the optimal design point in most of the cases and is therefore interesting for early design stages, or for cases in which N is too large for exhaustive exploration.

C. Irregular Partitioning

Besides being used to guide the selection of back-biasing configurations as described above, dual-V_{th} assignment can also be used during implementation, as a way to construct irregular V_{th} domains.

Specifically, we propose to run dual-V_{th} *before* the insertion of back-bias domains (after the first placement of the operator), in all precision and V_{DD} conditions. This allows to identify the *minimum set of cells* that should ideally be boosted to reach timing compliance in that condition (i.e. the critical cells described in Section IV-A). This information is then used as basis to determine domain boundaries.

As explained in Section IV-C, the key issue is merging the results of different dual-V_{th} runs into a single physical partitioning. As a first heuristic simplification, when multiple V_{DD} values are considered, we select, for each precision b_i, the supply voltage that produces the minimum power consumption according to the dual-V_{th} output. In this way, we are left with B different sets of FBB cells to be considered for the construction of domains.

Classic clustering approaches (k-means, dbscan, etc.) cannot be applied straight-forwardly to this multi-objective problem. We propose instead a simple aggregative algorithm to generate the domains, named *criticality-driven* partitioning hereinafter. The approach is depicted in Figure 9 for a simplified example in which $B = 2$. We first subdivide the placement of the

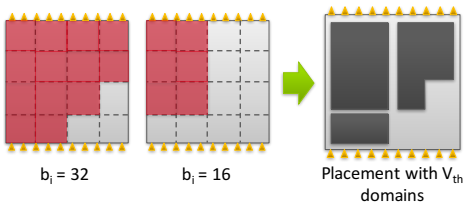


Fig. 9. Example of the proposed criticality-driven partitioning to build irregular V_{th} domains.

operator in a virtual grid of identical regions, each of which represents a *potential* V_{th} domain. As for regular tiling, the maximum number of regions is determined by the acceptable area overhead. Then, we apply the analysis of Section V-B2 on the virtual grid, i.e. we identify the regions that should be boosted for each considered bit-width, using the dual-V_{th} output as guidance. A possible result is shown in the two leftmost drawings of Figure 9. Finally, we generate V_{th} domains merging regions that, for all bit-widths, always receive the same BB voltage (rightmost drawing in the figure). The merging phase only deals with N configurations in total, therefore its complexity is negligible compared to the previous guided search.

The advantage of this solution with respect to regular tiling is a possible reduction of the number of domains, for similar results in terms of power consumption. This translates into smaller area overheads and simpler control. However, irregular partitioning also has a disadvantage in terms of flexibility, as anticipated in Section IV-C. First, merging domains as shown in Figure 9 is only possible when few bit-width configurations are considered; with too many precision modes, no group of domains *always* receives the same BB voltage. Second, all the previous analyses depend on the target operating conditions, and in particular on the clock frequency f_{clk}. Hence, the partitioning determined with the aggregative method will be optimal only for a single clock frequency; modifying f_{clk} would change the way in which virtual regions should be merged into physical domains. Regular tiling, in contrast, uses the minimum acceptable size for each domain, and leaves more freedom to reconfigure the V_{th} assignment depending on operating conditions at runtime.

VI. Experimental Results

A. Experimental Setup

In this section, we show the benefits of applying our method on a set of representative operators, implemented on a 28nm flip-well UTBB FDSOI standard-cell technology library from ST Microelectronics. We used operators extracted from *OpenCores* [39], specified in behavioral VHDL or Verilog. We performed synthesis using Synopsys Design Compiler L-2016.06 and place & route using Cadence Innovus 16.1. Post-implementation netlists and parasitics were loaded in Synopsys PrimeTime L-2016.06 for timing and power analyses.

Throughout our analyses, we considered a total of five V_{DD} conditions, from $1.0V$ to $0.6V$, in steps of $0.1V$. As forward back-biasing (FBB) voltages, we used $V_{BB,N} = 1.1V$ and $V_{BB,P} = -1.1V$ for NMOS and PMOS devices respectively. Unless specified differently, analyses are performed in a slow-slow process corner, at 125C. During the first P&R of the operators (without BB domains), we considered as nominal operating condition $V_{DD} = 1V$, with FBB applied to *all* cells of the circuit.

We considered the following four representative operators:

- A 16x16-bit multiplier, with radix-4 Booth encoder and Wallace tree.
- A 64-bit Kogge-Stone adder.
- A 16-bit, 30-tap FIR filter in Direct Form.
- A 16-bit *butterfly* unit (the main datapath component of a FFT accelerator)

The adder and the multiplier were considered because they are the basic building blocks of most datapath circuits. The remaining two designs were chosen as representative of datapath accelerators for signal processing applications, a typical error-tolerant domain [2]. These two benchmarks prove that our method yields good results also when applied directly on bigger and more complex circuits.

DVAS [11] was already shown to provide better results than traditional architectural solutions for precision-scalable hardware operators design ([19], [21], etc.). Therefore, in this work we consider DVAS as the reference for comparison.

B. Comparison with Dynamic Accuracy Scaling

Our method can provide power benefits using *only* dynamic V_{th} tuning, independently from the availability of a global supply voltage scaling knob. To show these benefits, we implemented a precision-scalable version of the Booth multiplier composed of 4 BB domains, using a 2x2 regular tiling grid.

Figure 10 shows the power versus bit-width curves obtained by this multiplier at a fixed $V_{DD} = 0.90V$. The graph shows the Pareto frontier, i.e. the *minimum* total power for each bit-width, obtained testing all possible back-biasing assignments to the 4 domains, with the exhaustive analysis method described in Section V-B1. Power consumption includes both leakage and dynamic components. The clock frequency is set to the maximum value that does not incur timing violations at maximum precision, with FBB applied to all 4 domains (i.e. $f_{clk} = 1.25GHz$). Only configurations that *fully* meet timing constraints, taking into account disabled paths due to reduced dynamic, are reported in the figure. We neglect bit-widths smaller than 4-bit as they are of very limited interest, even in error tolerant applications.

For comparison, Figure 10 also reports the results of simple input bit-width reduction (with no modification of the operating conditions), on the original multiplier without BB domains, i.e. the so-called Dynamic Accuracy Scaling (DAS) [11]. Two set of DAS results are shown: one when no back-bias is used,

the other when FBB is applied "monolithically" to the entire operator. In both cases, the clock frequency is set to the same value used for our method.

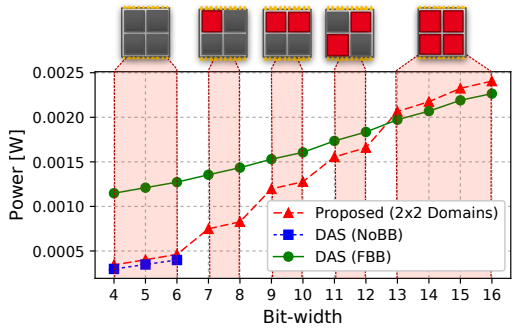


Fig. 10. Power versus bit-width curves for the Booth multiplier, at fixed $V_{DD} = 0.90V$.

At the considered f_{clk}, the multiplier without BB cannot operate at more than 6-bit precision, without incurring timing violations, whereas our operator, thanks to FBB, can reach maximum precision (16-bit). At the same time, our fine-grain solution is (on average) significantly more efficient than using FBB on the entire design. For example, at 7 and 8-bit precision, the power savings with respect to monolithic FBB are 44.8% and 42.3% respectively.

These benefits are paid with a small power overhead with respect to the operator without domains at very low and very high precision (6.2% worst case). This is mostly due to cells resizing performed in the incremental placement phase, to account for the insertion of separation guardbands. Secondarily, the additional area also causes an increase in the length of metal routes, and consequently in the dynamic power consumption. The area overhead is approximately 14% compared to the operator without domains.

On the top of Figure 10, we also reported graphically the back-bias configurations used for each bit-width, where red squares represent domains that receive FBB. Notice that, as expected, a smaller bit-width requires a smaller or equal number of boosted domains, and that all domains are boosted in at least one precision mode.

C. Comparison and combination with DVAS

When possible, fine-grain V_{th} tuning can also be combined with voltage scaling (i.e. DVAS). We implemented both standard DVAS and our method on all four benchmark circuits, using regular grids of BB domains, with the configurations shown in Table I. The table also reports the corresponding area overheads and the clock frequencies used for analysis.

Figures 11–14 show the results of this experiment on a bit-width versus power consumption plane. The curves are generated as in Section VI-B, and report the *minimum power* for each bit-width, considering all possible combinations of V_{DD} and (for our method) BB assignment. For a fair comparison, for each benchmark we report both the results of DVAS with NoBB and DVAS with "monolithic" FBB.

The shapes of the curves are similar to the case of simple DAS. Thanks to fine-grain power and speed tuning, our

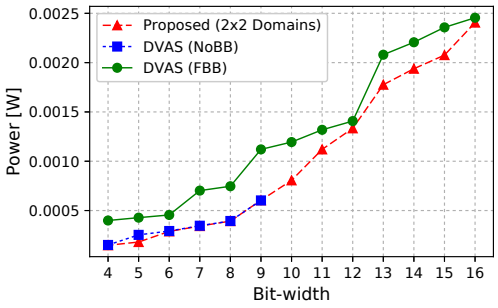


Fig. 11. Booth Multiplier, proposed method and DVAS.

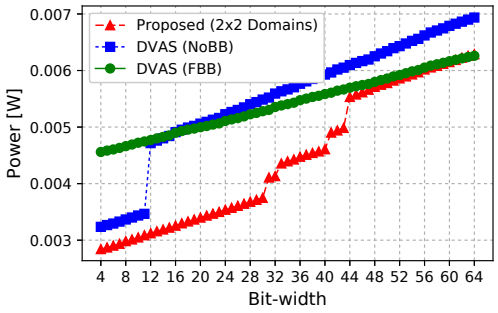


Fig. 12. Kogge-Stone Adder, proposed method and DVAS.

method is significantly more efficient than DVAS with FBB at the same frequency, whereas DVAS with NoBB can seldom reach maximum precision without violations. Even when the latter is timing compliant (e.g. in the case of the adder) our method can provide better power efficiency, by working at a lower V_{DD} and *only* speeding up (with FBB) the critical parts of the operator, at the cost of a small (thanks to the fine granularity of domains) leakage overhead.

Notice that, for adder, multiplier and FIR, the effects of the wall-of-slack on DVAS are particularly evident looking at the step-wise shape of the Pareto frontier. Each step corresponds to a change of V_{DD}, and occurs because scaling the supply voltage drastically reduces the maximum bit-width that can be used without violations. Therefore, the supply voltage must be kept constant for large ranges of bit-widths, obtaining only a linear power decrease due to the reduction in circuit activity. In contrast, our method can leverage fine-grain back-bias to obtain numerous "intermediate steps", and therefore produces a more graceful power versus precision dependency. In the case of the butterfly unit, the more linear curves of DVAS show that this unit is less affected by the wall-of-slack (probably due to a very relaxed implementation frequency) and consequently the improvement thanks to back-biasing is less marked.

The maximum power savings and overheads obtained by our method with respect to DVAS are reported in Table I, together with the corresponding bit-widths. For all designs, the maximum saving is higher than 25%, and reaches 43% for the FIR filter. When combined with DVAS, the power overheads of our method are generally very small (less than 1%), except for the FFT butterfly. This exception is motivated by the fact that the number of domains used in the FFT experiments is large compared to the size of the circuit. Thus, the compensation of

TABLE I
IMPLEMENTATION DETAILS AND RESULTS SUMMARY OF THE COMPARISON BETWEEN THE PROPOSED METHOD AND DVAS.

Circuit	Domains	f_{clk} [GHz]	Area Overhead [%]	Max. Power Saving [%]	Bits	Max. Power Overhead [%]	Bits
Multiplier	2x2	1.25	14.5	32.67	10	0.26	9
Adder	2x2	2.00	11.3	33.78	12	0.46	64
Butterfly	3x3	1.00	17.0	26.45	5	17.15	4
FIR	3x3	0.75	15.1	43.06	16	0	-

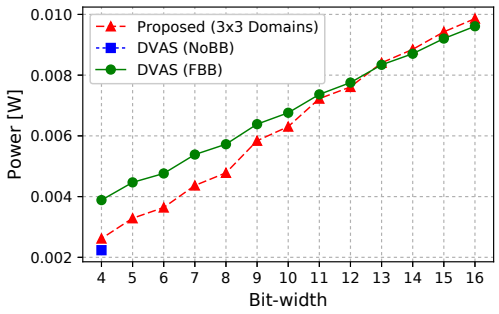


Fig. 13. FFT Butterfly, proposed method and DVAS.

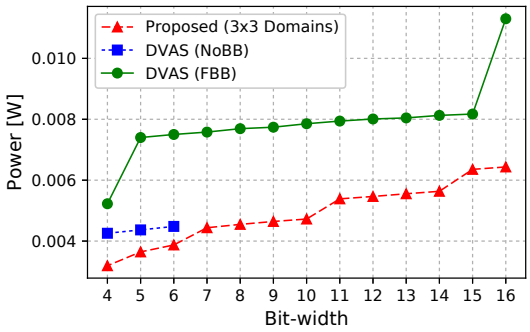


Fig. 14. FIR Filter, proposed method and DVAS.

area overheads described in Section VI-B is more marked. Butterfly implementations with less BB domains were not chosen as they poorly isolated the critical paths of the circuit, and provided very limited power savings compared to DVAS, as shown in Table II.

D. Impact of the number of V_{th} domains

The number of V_{th} domains has a strong impact on all figures of merit in our method. This is shown in Figures 15 and 16 using the multiplier as an example. Specifically, Figure 15 shows the maximum savings obtained at different bit-widths for different configurations of BB domains. For ease of visualization, the graph only reports precision modes between 8 and 16 bits. These results still refer to the regular tiling solution, and the only difference among configurations is the number and position of the tiles.

As expected, increasing the number of domains produces a general reduction in power consumption, especially at high bit-widths. Indeed, the presence of more domains allows a finer-grain control of the parts of the operator to boost. In few particular cases, this trend is not respected and increasing the number of domains has a negative effect of power (e.g. 2x1 versus 3x1 domains at 10-bits precision). This is due to the addition of guardbands between groups, which may cause displacements of critical cells that were previously placed close to each other by the tool. As a consequence, the incremental placement must perform substantial modifications to restore timing compliance, causing the trend inversion. Table II shows the maximum and mean power savings with respect to DVAS obtained by the four benchmark circuits for a subset of groups configurations, in the same conditions of Section VI-C. As for the multiplier, finer granularity generally allows larger power reductions.

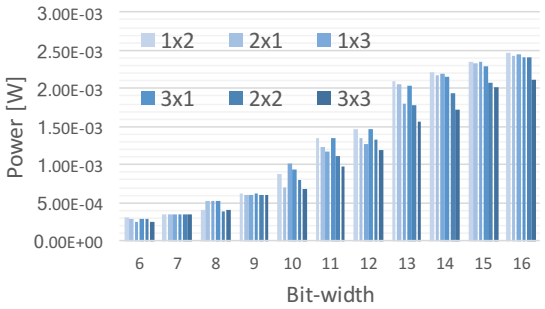


Fig. 15. Minimum power versus bit-width in the Booth multiplier, for different configurations of regular BB tiles.

Figure 16 reports the area and settling time overheads for the groups configurations analyzed in the multiplier. The settling times refer to a SPICE simulation performed using an implementation of the circuit in Figure 5 on the target technology, and the values reported are *worst case*, i.e. they refer to the domain with the largest well capacitance. Expectedly, the area overhead increases with the number of groups. On the contrary, the settling time for single domains is smaller for a larger number of groups, due to the smaller well area.

In general, the selection of the number of BB domains strongly depends on system-specific constraints; the power reduction in precision modes of interest must be balanced with the area budget. However, as demonstrated in the next section, the design space of possible implementations can be explored quickly, at least for a small number of domains (≤ 10). Notice that area and timing overheads are mostly dependent on the shape of BB domains, and are only slightly influenced by benchmarks architectures. Thus, for space reasons, we omit results for the other three benchmarks.

E. Exhaustive and heuristic search algorithms

Figures 17 and 18 show the comparison between exhaustive search and the dual-V_{th} guided heuristic for the identification of the optimal configuration of V_{DD} and BB. The curves

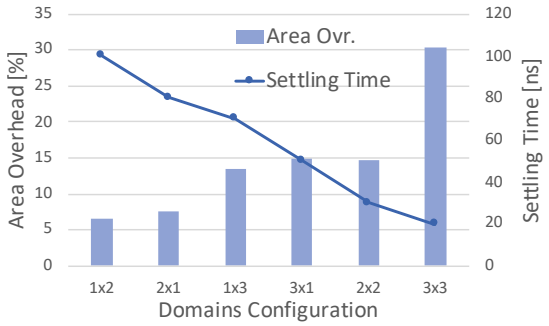


Fig. 16. Area and settling time overheads versus bit-width in the Booth multiplier, for different configurations of regular BB tiles.

TABLE II
MAXIMUM (MEAN) POWER SAVING [%] WITH RESPECT TO DVAS, FOR DIFFERENT CONFIGURATIONS OF REGULAR BB TILES.

Circuit	1x2	1x3	2x2	3x3
Multiplier	25.90(3.61)	26.82(3.87)	32.67(9.75)	42.56(14.10)
Adder	24.20(14.53)	22.61(15.98)	33.78(19.56)	33.90(22.11)
Butterfly	16.33(2.86)	19.70(2.76)	19.39(3.74)	26.45(9.14)
FIR	23.65(16.89)	34.60(15.31)	41.59(22.74)	43.06(31.40)

referring to exhaustive search are the same as in Figures 11 and 14, whereas the ones referring to guided search are obtained with the heuristic method described in Section V-B2.

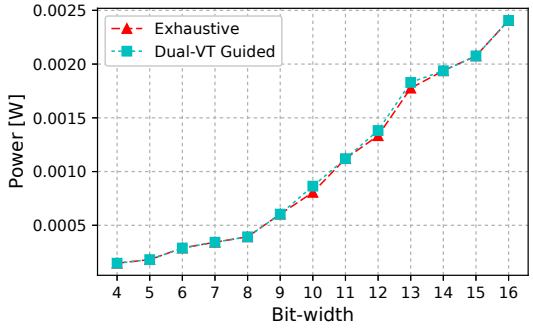


Fig. 17. Comparison between exhaustive and dual-V_{th} guided search in the Booth multiplier.

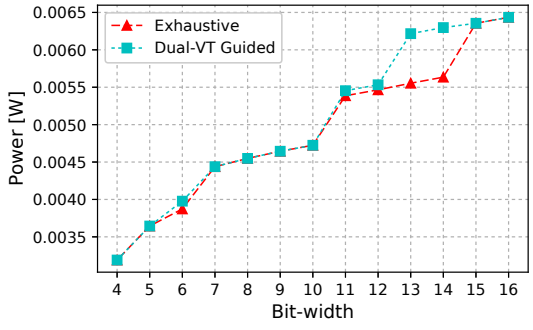


Fig. 18. Comparison between exhaustive and dual-V_{th} guided search in the FIR filter.

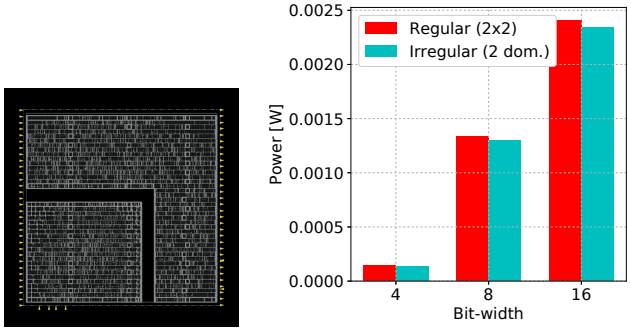
For both circuits, the proposed heuristic is able to identify the optimal configuration for the majority of precisions. Even when it fails to identify the correct combination of knobs, the algorithm still outputs a configuration that is close to the optimal. The maximum (average) power "error" of the heuristic for all four benchmarks is reported in Table III. While these errors might be unacceptable for a final design, the small average deviations prove that the heuristic method can be used to quickly explore the design space (e.g. to identify the optimal number and configuration of BB tiles, or to compare regular with irregular partitioning), providing results that closely resemble the optimal ones, despite the complexity reduction from exponential to linear. On our testing platform, whose technical specifications have been introduced in Section V, the exhaustive search applied to the multiplier requires about 1 hour and 20 minutes, whereas the guided search only takes about 12 minutes.

TABLE III
MAXIMUM (MEAN) POWER "ERROR" [%] OF THE HEURISTIC SEARCH METHOD, COMPARED TO THE EXHAUSTIVE ONE.

Multiplier	Adder	Butterfly	FIR
7.2(1.0)	9.3(1.4)	6.6(1.5)	11.9(2.2)

F. Irregular Partitioning

In this experiment, we compare regular and irregular partitioning on an example design, namely the Booth multiplier. We compare the regular tiling approach with a 2x2 grid with an irregular implementation built using the approach described in Section V-C. To generate the irregular domains, we started from a virtual 3x3 grid, and assumed that the target application only requires three precision modes: 4, 8 and 16 bit. The clock frequency was set to the same value used for the regular design (see Table I). After the merging phase, the resulting irregular circuit only contains 2 domains; a placement snapshot obtained from the tool is shown in Figure 19a.



(a) Placement snapshot (b) Power versus bit-width

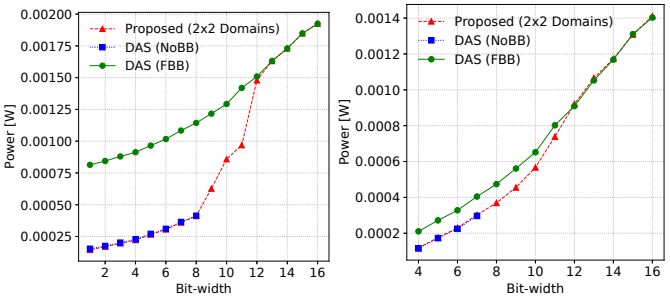
Fig. 19. Regular versus irregular BB domains for the Booth multiplier.

Figure 19b shows the power consumption of the two versions of the circuit in the three considered bit-width conditions. The irregular version is slightly more efficient than the regular one due to the smaller number of domains, which eases the placement process. Most importantly, reducing the number of domains simplifies the control logic of the modified circuit (e.g. the size of the configuration register in

Figure 5) and reduces separation guardbands overheads, while maintaining comparable power. However, these advantages must be balanced with the reduction in flexibility discussed in Section V-C. In fact, the irregular version of the operator is tailored for three precision modes only, and for a *single* clock frequency.

G. Effect of operating conditions

As a final experiment, we discuss how the results of our methodology depend on operating conditions. Intuitively, the larger the share of leakage in the total power, the higher the benefits of our approach. In other words, larger savings are expected at high temperature, low voltage, and low frequency [25], i.e. the corners that cause circuits to be most leaky. As a matter of fact, when leakage is negligible, applying FBB to the *entire* circuit has almost no impact on total power, and the difference between DAS/DVAS with FBB applied to the entire circuit and our solution becomes less evident. Conversely, the fine-grain approach is more effective when the leakage increase due to FBB has a strong impact on power, since only critical portions of the circuit are boosted.



(a) $f_{clk} = 1GHz$ (b) $T = 25C$

Fig. 20. Effects of operating conditions on the Booth multiplier power versus bit-width Pareto curve, for a fixed $V_{DD} = 0.90V$.

Two examples are shown in Figure 20. Both plots refer to the same conditions of Figure 10. However, in Figure 20a, the clock frequency has been reduced from 1.25 to 1 GHz, leaving the temperature at the default 125C. Conversely, Figure 20b was obtained leaving f_{clk} at 1.25 GHz, and reducing T from 125C to 25C. As expected, the savings of our method compared to global back-biasing increase at lower frequency: the maximum saving at 1 GHz is 48.5% at 9 bit precision. Conversely, they reduce at low temperatures, although the maximum power saving at 25C is still 22.1% (at 8 bit), a value that can justify the area overhead of our method, especially considering that the typical junction temperature is higher than 25C [1]. The impact of leakage on total power in response to variations of operating conditions follows similar trends for all benchmarks, thus we omit similar plots for all circuits.

In general, low power IoT devices normally operate in conditions were leakage is comparable to dynamic power, e.g. ultra-low voltage and low frequency [1], which are exactly the operating points where our method performs best.

[1]Intermediate temperatures between 25C and 125C were not tested due to lack of library characterizations.

VII. Conclusion

In this paper, we have described a new method for the implementation of energy-quality scalable hardware operators. Our solution is based on the precision-dependent tuning of speed and power in selected critical portions of the circuit, at runtime. We have implemented our method in FDSOI technology, leveraging the strong effect of back-biasing on transistor threshold voltage, and the small overheads associated with back-bias domains. Thanks to the flexibility offered by the fine-grain back-biasing knob, our method outperforms DVAS, a state-of-the-art solution based on voltage scaling of the entire operator, at the cost of a small increase in the die area occupation of the operator.

Future developments of this work include the study of a method for the runtime adaptation of V_{DD} and back-biasing in response to changes in the operating conditions (temperature, aging, etc.) detected by on-chip sensors.

References

[1] M. Alioto, "Energy-quality scalable adaptive vlsi circuits and systems beyond approximate computing," in *DATE* 2017, pp. 127–132.
[2] V. Chippa et al, "Analysis and characterization of inherent application resilience for approximate computing," in *DAC* 2013, pp. 1–9.
[3] D. Jahier Pagliari, M. Poncino, and E. Macii, *Energy-Efficient Digital Processing via Approximate Computing*, In: N. Bombieri, M. Poncino, G. Pravadelli (eds), *Smart Systems Integration and Simulation*. Springer, Cham, 2016.
[4] Q. Zhang et al, "Approxit: An approximate computing framework for iterative methods," in *DAC* 2014, pp. 1–6.
[5] Q. Zhang et al, "Approxann: An approximate computing framework for artificial neural network," in *DATE* 2015, pp. 701–706.
[6] H. Jiang, J. Han, and F. Lombardi, "A comparative review and evaluation of approximate adders," in *ACM GLSVLSI* 2015, pp. 343–348.
[7] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, "Low-power high-speed multiplier for error-tolerant application," in *EDSSC* 2010, pp. 1–4.
[8] J. Huang, J. Lach, and G. Robins, "A methodology for energy-quality tradeoff using imprecise hardware," in *DAC* 2012, pp. 504–509.
[9] F. Farshchi, M. S. Abrishami, and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing," in *CSI CADS* 2013, pp. 25–30.
[10] A. Lingamneni et al, "Synthesizing parsimonious inexact circuits through probabilistic design techniques," *ACM Trans. Embed. Comput. Syst.*, vol. 12, no. 2s, pp. 93:1–93:26, May 2013.
[11] B. Moons and M. Verhelst, "Dvas: Dynamic voltage accuracy scaling for increased energy-efficiency in approximate computing," in *ISLPED* 2015, pp. 237–242.
[12] B. Moons et al, "Dvafs: Trading computational accuracy for energy through dynamic-voltage-accuracy-frequency-scaling," in *DATE* 2017, pp. 488–493.
[13] B. Moons et al, "14.5 envision: A 0.26-to-10tops/w subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm fdsoi," in *ISSCC* 2017, pp. 246–247.
[14] Q. Xu, N. S. Kim, and T. Mytkowicz, "Approximate computing: A survey," *IEEE Design Test*, vol. 33, no. 1, pp. 8–22, Feb 2016.
[15] A. Ranjan et al, "Aslan: Synthesis of approximate sequential circuits," in *DATE* 2014, pp. 1–6.
[16] A. Verma et al, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in *DATE*, 2008, pp. 1250–1255.
[17] D. Mohapatra et al, "Design of voltage-scalable meta-functions for approximate computing," in *DATE* 2011, pp. 1–6.
[18] A. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *DAC*, 2012, pp. 820–825.
[19] M. de la Guia Solaz, W. Han, and R. Conway, "A flexible low power dsp with a programmable truncated multiplier," *IEEE TCAS I, Reg. Papers*, vol. 59, no. 11, pp. 2555–2568, Nov 2012.
[20] R. Ye et al, "On reconfiguration-oriented approximate adder design and its application," in *IEEE/ACM ICCAD* 2013, pp. 48–54.
[21] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in *DATE* 2014, pp. 95:1–95:4.

[22] B. Barrois et al, "The hidden cost of functional approximation against careful data sizing - a case study," in *DATE* 2017, pp. 181–186.

[23] A. Kahng et al, "Slack redistribution for graceful degradation under voltage overscaling," in *ASP-DAC* 2010, pp. 825–831.

[24] D. Jahier Pagliari et al, "A methodology for the design of dynamic accuracy operators by runtime back bias," in *DATE* 2017, pp. 1165–1170.

[25] P.-E. Gaillardon et al, "A survey on low-power techniques with emerging technologies: From devices to systems," *ACM JETC*, vol. 12, no. 2, pp. 12:1–12:26, Sep. 2015.

[26] E. Beigné et al, "A 460 mhz at 397 mv, 2.6 ghz at 1.3 v, 32 bits vliw dsp embedding f max tracking," *IEEE JSSC*, vol.50, no.1, pp.125–136, Jan 2015.

[27] S. M. Martin et al, "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads," *ICCAD* 2002, pp. 721-725.

[28] M. Cochet et al, "Experimental model of adaptive body biasing for energy efficiency in 28nm utbb fd-soi," in *S3S* 2014, pp. 1–2.

[29] D. Puschini et al, "Body bias usage in utbb fdsoi designs: A parametric exploration approach," *Solid-State Electron.*, vol.117, pp.138–145, 2016

[30] J. Hu et al, "Architecting voltage islands in core-based system-on-a-chip designs," in *ISLPED* 2004, pp. 180–185.

[31] E. Ebrahimi, R. T. Possignolo and J. Renau, "Level shifter design for voltage stacking," *ISCAS* 2017, pp. 1-4.

[32] S. Dighe et al., "Within-Die Variation-Aware Dynamic-Voltage-Frequency-Scaling With Optimal Core Allocation and Thread Hopping for the 80-Core TeraFLOPS Processor," in *IEEE JSSC*, vol. 46, no. 1, pp. 184-193, Jan. 2011.

[33] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*, 1st ed, McGraw-Hill Higher Education, 1994.

[34] K. Usami and M. Horowitz, "Clustered voltage scaling technique for low-power design," in *ISLPED* 1995, pp. 3–8.

[35] T. Kutzschebauch and L. Stok, "Regularity driven logic synthesis," in *IEEE/ACM ICCAD* 2000, pp. 439–446.

[36] Q. Wang and S. B. K. Vrudhula, "Algorithms for minimizing standby power in deep submicrometer, dual-vt cmos circuits," *IEEE Trans. on CAD*, vol. 21, no. 3, pp. 306–318, Mar 2002.

[37] M. Blagojevi et al, "A fast, flexible, positive and negative adaptive body-bias generator in 28nm FDSOI," *IEEE VLSI-Circuits*, 2016, pp. 1-2.

[38] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer cmos circuits," *Proc. of the IEEE*, vol. 91, no. 2, pp. 305–327, Feb 2003.

[39] [Online] opencores.org

Daniele Jahier Pagliari (M'15) received the M.Sc. degree in computer engineering from the Politecnico di Torino, Italy, in 2014, where he is currently pursuing the Ph.D. degree with the Department of Control and Computer Engineering. His main research interests focus on computer-aided design of digital systems, with particular emphasis on low-power optimization and approximate computing.

Yves Durand (M03) received his engineering degree in 1983 and a PhD in computer science in 1988. He worked with ST Microelectronics as a research engineer, then moved to Hewlett Packard in 1993 and led R&D projects related to networking interfaces and smart communicating objects. He then joined the Laboratoire d'Electronique et de Technologie de l'Information (CEA-LETI), Grenoble, in 2003. His background is in performance modeling and his current research is in architecture for floating-point arithmetics and variable precision computing.

David Coriat received his M.Sc. degree for the University of Science of Montpellier, France, in 2012, and subsequently joined the CEA-LETI Minatec. He has worked on dynamic management of power and variability in MP-SoC architectures as well as power estimation techniques in large MP-SoC architectures. His research interests now lie in low power architectures and transprecision computing at hardware level.

Edith Beigne is graduated from Grenoble INPG electrical engineering school in 1998. She joined CEA-LETI, Grenoble, France, in 1998 and obtained her HDR diploma in 2014. She is the Research Director of Design and Embedded Software division at CEA LETI since 2017. Since 2009, she has been a senior scientist in the digital and mixed-signal design lab where she researches low power and adaptive circuit techniques, exploiting asynchronous design and advanced technology nodes like FDSOI, 3D technologies, for many different applications from high performance MPSoC to ultra-low power IoT applications, including low power neuro-inspired techniques. She is part of ISSCC TPC since 2014 and part of VLSIsymposium since 2015, SSCS Distinguished Lecturer in 2016-2017 and elected at the IEEE SSCS Adcom.

Enrico Macii (SM02, F07) is a Full Professor of Computer Engineering at Politecnico di Torino, Italy. Prior to that, he was an Associate Professor (1998-2001) and an Assistant Professor (1993-1998) at the same institution. From 1991 to 1997 he was also an Adjunct Faculty at the University of Colorado at Boulder. He holds a Laurea Degree in Electrical Engineering from Politecnico di Torino (1990), a Laurea Degree in Computer Science from Universit di Torino (1991) and a PhD degree in Computer Engineering from Politecnico di Torino (1995). Since 2007, he is the Vice Rector for Research at Politecnico di Torino; he was also the Rector's Delegate for Technology Transfer (2009-2015) and for International Affairs (2012-2015). His research interests are in the design of electronic circuits and systems, with particular emphasis on low-power consumption, optimization, testing, and formal verification. In the last few years, he has been growingly involved in projects focusing on the development of new technologies and methodologies for smart cities and bioinformatics. In the fields above he has authored over 450 scientific publications. Enrico Macii is a Fellow of the IEEE.

Massimo Poncino (SM12, F18) received the Ph.D. degree in Computer Engineering and the Dr. Eng. degree in Electrical Engineering from the Politecnico di Torino, Italy. He is currently a Full Professor of Computer Engineering at Politecnico di Torino. His research interests include several aspects of design automation of digital systems, with particular emphasis on the modeling and optimization of low-power systems. He is the author or coauthor of more than 350 journal and conference papers. He is an Associate Editor of the ACM Transactions on Design Automation of Electronic Systems and of IEEE Design & Test. Prior to that, he was an Associate Editor of IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2006-2012). He was the Technical Program Co-Chair (in 2011) and the General Chair (in 2012) of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED). He serves on the Technical Program Committee of several IEEE and ACM technical conferences, including DAC, ICCAD, DATE, ISLPED, ASP-DAC, CODES-ISSS, and GLSVLSI. Massimo Poncino is a Fellow of the IEEE.