# Truncated Multipliers through Power-Gating for Degrading Precision Arithmetic

Pietro Albicocco, Gian Carlo Cardarilli, Alberto Nannarelli[(1)], Massimo Petricca[(2)] and Marco Re
Department of Electrical Engineering, University of Rome "Tor Vergata", Rome, Italy
[(1)] DTU Compute, Technical University, Denmark
[(2)] Politecnico di Torino, Turin, Italy

***Abstract*—When reducing the power dissipation of resource constrained electronic systems is a priority, some precision can be traded-off for lower power consumption. In signal processing, it is possible to have an acceptable quality of the signal even introducing some errors. In this work, we apply power-gating to multipliers to obtain a programmable truncated multiplier. The method consists in disabling the least-significant columns of the multiplier by power-gating logic in the partial products generation and accumulation array.**

## I. Introduction

In Digital Signal Processing (DSP) applications, it might be desirable, in some cases, to decrease the precision of the operands/operations to save power/energy when a given level of quality is sufficient for the application [1], [2], [3], [4].

In [5], we introduced the concept of Degrading Precision Arithmetic, or DPA, that is the implementation of a system in which the precision is tunable depending on some operating conditions. When the system is working at reduced precision power is saved. In [5], the precision is degraded by two methods:

I) **DPA-I**: By reducing the precision in the $k$-LSBs (Least Significant Bits) of the datapath. This is obtained by freezing the values of the $k$-LSBs in registers by clock-gating them.

II) **DPA-II**: By reducing the power supply voltage. The increased paths delays introduce errors in the datapath.

Although DPA-I, can significantly reduce the power dissipation, the technique is not effective in large combinational blocks which are not directly connected to the clock-gated registers.

In contrast, by implementing power gating any logic can be disabled by disconnecting the power supply from the gates we want to disable in the datapath.

In this work, we introduce power-gating as a method to obtain fine-grained configurability of datapaths. In continuity with our previous work, we label the method **DPA-III**.

As a case study, we present a fixed-point parallel multiplier in which we disable the LSBs by power-gating and obtain a truncated multiplier with a variable number of truncated bits. We chose a multiplier, because it is a very common unit in DSP, and it is quite power hungry. We apply power-gating with different granularity, i.e., we disable logic gates in clusters of different sizes, and we analyze the tradeoffs with respect to error, delay, area, and power dissipation.

In Sec. II, we briefly recap on the power-gating method to disable logic gates. In Sec. III, after recollecting the features of truncated multipliers, we explain our design and present several implementation alternatives. In Sec. IV, we present the results of synthesis of the DPA-III multipliers and compare their characteristics with those of truncated multipliers. In Sec. V, we apply DPA-III multipliers to a FIR filter and we show their impact on the filter layout. Finally, in Sec. V, we draw the conclusions and present ideas for future work.

## II. Power-Gating

Power gating can be implemented by inserting a high-threshold PMOS transistor between the source terminal of the cell/block and the power $V_{DD}$ rail, as shown in Fig. 1, or by inserting a NMOS transistor between the source terminal and the ground rail. This operation is called Sleep Transistor Insertion (STI). The insertion point of the Sleep Transistor (ST) is called Virtual $V_{DD}$ line (VVDD), or Virtual Ground line (VGND).

Normally, STI is performed on groups, or clusters, of cells (Clustered STI) to minimize the area overhead and have a lower congestion for the routing of the sleep signals [6].

Node $Z$ in Fig. 1 is at the interface between the power-gated cells and those that are always active. When the logic is disabled, node $Z$ can drift to some voltage between $V_{DD}$ and $0\ V$ and bias some of the transistors in the active logic to be always 'on' causing a static power dissipation. For this reason, *isolation cells* (AND or OR gates) are inserted in node $Z$ to prevent this static power dissipation. For fine-grained power-gating, isolation cells might create a large overhead and must be limited to a minimum.

Suitable control signals must be provided to ensure correct operations. The sequence of operations to power-off, and power-on again, the cluster of gates is the following:

1) Isolate node $Z$ by setting the isolation control signal so that node $Z$ is set either to 0 or 1.
2) Power-off the gated logic block by setting the SLEEP control signal to 1 (ST is off).
3) Wake up the gated logic block by setting the SLEEP control signal to 0 (ST is on).
4) Reconnect the logic block to node $Z$ by setting the isolation gate to transparent mode.

In addition to the isolation and SLEEP signals, other signals must be added if we want to preserve the state of the system

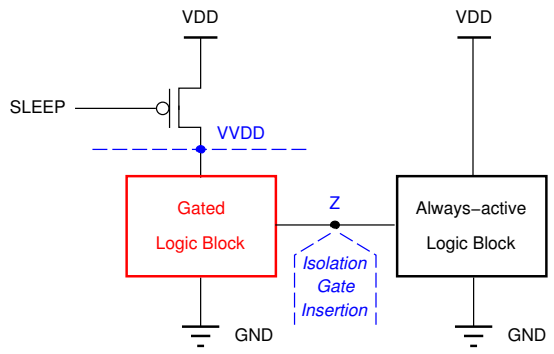


Fig. 1. Power gating by sleep transistor insertion.

when registers are powered-off. We do not consider this latter case in this work since we apply power-gating only to combinational blocks.

## III. Programmable Truncated Multiplier

Truncated multipliers have been extensively studied in the past [7]. When area on chips was a major concern, truncated multipliers were a suitable solution for DSPs.

By truncating the multiplier's array, we reduce the area and introduce an error in the product. The maximum error $\epsilon_{max}$, computed in closed form, is:

$$\epsilon_{max} = \sum_{j=0}^{k} (2^{k-j} - 1) \cdot 2^{j} \qquad (1)$$

while the average error $|\overline{\epsilon}|$ is determined by simulating the truncated multiplier with input patterns resembling the target application.

Nowadays, power dissipation is the major concern and area is less important. Therefore, we use power-gating to obtain configurable truncated multipliers to reduce power dissipation.

Power-gating has already been applied to twin-precision (e.g 32 and 16 bit operands) multipliers [8], [9].

Our idea is to have a full-precision parallel multiplier and to reduce its precision, on demand, by disabling the LSBs of the multiplier by power-gating. To limit the number of isolation cells, the multiplier architecture which looks more promising is the scheme in which the Partial Products (PPs) are added by columns [10]. The dot-array reduction scheme for a 8×8 multiplier, derived from [11], is shown in Fig. 2 for operands in two's complement and it includes sign extension correction.

Clustered STI is applied to cells belonging to the same column in the PPs generation (AND gates) and in the reduction array (Fig. 2). In Fig. 2 two power-gating domains for truncation of $k = 4$ (magenta dashed vertical line) and $k = 8$ (red dashed vertical line) bits are shown. Thicker vertical short lines indicate the position of isolation gates for the $k = 4$ (magenta) and $k = 8$ (red) configurations.

The multiplier of Fig. 2 can be truncated at granularity $g = 4$ (clusters of 4 bits) by setting the control signals in the two power domains PG $k = 4$ and PG $k = 8$. Fig. 3 shows the

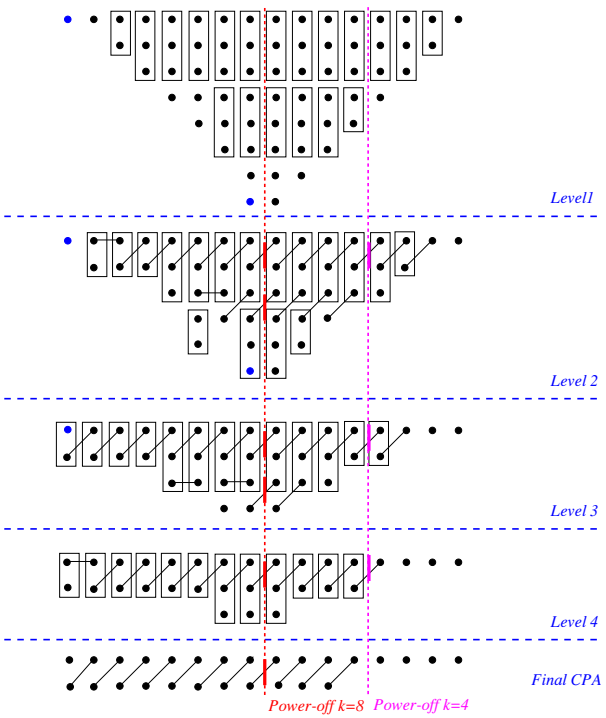


Fig. 2. Reduction tree for 8×8 two's complement multiplier. Blue dots are extra bits for sign extension correction.

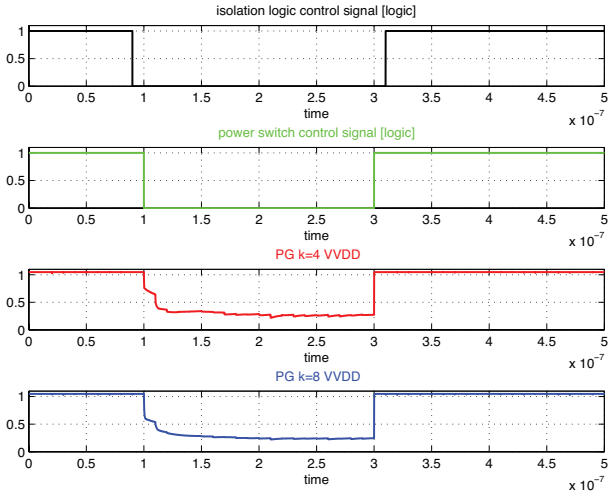


Fig. 3. Power-gating signals timing diagram.

time diagram, obained by Spice simulations, of the isolation and sleep signals and the voltage at nodes VVDD for the two power domains PG $k = 4$ and PG $k = 8$.

The latency overhead when switching power mode is two clock cycles to power-off the gated-blocks: one cycle for isolation and one cycle for switching-off STs, and two clock cycles to wake up the block: one cycle for power-on and one cycle for de-isolation.

The granularity of the power domains can be increased to

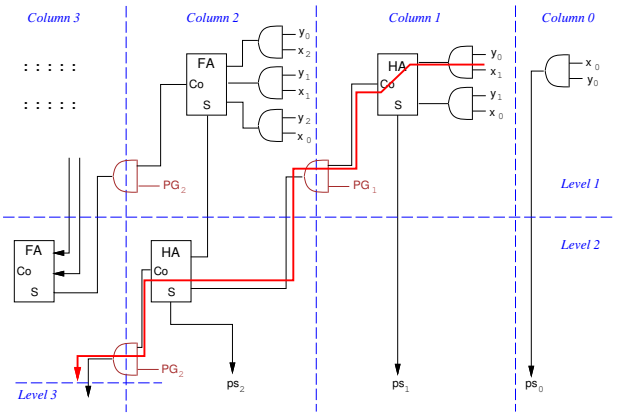


Fig. 4. Detail of isolation gates overhead on the critical path (red) for $g = 1$.

| $k$ | $P_{ave}$ $[\mu W]$ | savings [%] | Error $\lvert\overline{\epsilon}\rvert$ | $\lvert\epsilon_{max}\rvert$ |
|---|---|---|---|---|
| 0 | 154 | – | 0.0 | 0 |
| 1 | 158 | −2.6 | 0.2 | 1 |
| 2 | 156 | −1.3 | 1.2 | 5 |
| 3 | 147 | 4.5 | 4.2 | 17 |
| 4 | 142 | 7.8 | 11.2 | 49 |
| 5 | 125 | 18.8 | 28.3 | 129 |
| 6 | 112 | 27.3 | 73.4 | 321 |
| 7 | 96 | 37.7 | 180.1 | 769 |
| 8 | 81 | 47.4 | 561.6 | 1793 |

$P_{ave}$ measured at 100 MHz.

TABLE I
POWER-GATING RESULTS FOR $8 \times 8$-BIT MULTIPLIER ($g = 1$).

$g = 1$. That is, each multiplier column (up to a given minimum precision) has its power domain. In this way. we can have fine tunable control on the precision of the multiplier. However, the number of isolation gates necessary to implement the scheme is large, and their delays have a negative impact on the critical path, as illustrated in Fig. 4 and discussed in the next section.

## IV. Implementation of DPA-III Multipliers

In this section, we provide more detail on the design of the programmable truncated multiplier, or DPA-III multiplier, and report the experimental results of the physical implementation of the unit.

As test-case unit, we choose a programmable 8×8-bit truncated multiplier. The unit is implemented in the Synopsys EDK 32 nm library of standard cells [12].

### A. Design Flow

The design flow is divided into two main phases: the first phase is from the HDL description of the unit to logic synthesis, and the second phase is from logic synthesis to post-layout simulation.

*1)* Design Flow (Part I): The first step in the design flow is the generation of the HDL description of the unit. We use a Matlab script to automatically generate a VHDL description where we associate to each logic gate an instance name carrying the information of the bit-weight the gate output is connected to.

The design is then synthesized at its maximum operating speed by Synopsys Design Compiler. The synthesized netlist is then converted in Spice format and the virtual power nets (VVDD) are manually added to the netlist. We add a sufficient number of power-switches (STs) to guarantee an IR-drop less than 10% the nominal supply voltage.

The power consumption is estimated by simulating the Spice netlist in Synopsys NanoSim simulator. To compute the power dissipation, we simulated the multiplier by providing a set of random input patterns.

*2)* Design Flow (Part II): The information extracted in the first part of the design flow are used to set the constraints to guide Synopsys IC Compiler in the floorplanning and in the power network synthesis. The design constraints are iteratively updated, based on the results of the power network synthesis and the IR-drop analysis, to obtain design closure (i.e., the circuit is working properly).

Then, the layout is completed, area and static timing reports are generated, and parasitics annotated.

Finally a post-layout simulation is performed by Synopsys NanoSim simulator to estimate the power consumption in the physical implementation of the unit.

### B. Multiplier Implementation

We first implement a DPA-III multiplier with granularity one bit/column ($g = 1$) from column 0 (LSB) to column 7. Table I reports the power dissipation and the errors, for the implementation with power-gating at granularity $g = 1$, when the $k$ least-significant columns of the array are disconnected (e.g., truncated). The table reports the corresponding maximum error $\epsilon_{max}$, from (1), and the average error $|\overline{\epsilon}|$ determined by simulating the truncated multiplier with random input vectors. Both errors are given with reference to the *unit-in-last-position*, or *ulp*[1].

By comparing the DPA-III unit with $g = 1$ to a plain 8×8-bit multiplier (same structure), the overhead introduced by the isolation gates in the DPA-III multiplier is quite significant.
The delay overhead in the critical path is about 20%: $t_{plain} = 1.59\ ns$ vs. $t_{g=1} = 1.86\ ns$.
For the area, the overhead is about 33%:
$A_{plain} = 927\ \mu m^2$ vs. $A_{g=1} = 1220\ \mu m^2$.

This overhead clearly shows that granularity $g = 1$ is too fine: one isolation gate per power-gated column contributes to the delay of critical path when the multiplier works in full precision (refer to Fig. 4).

From Table I, we notice a significant gap between $k = 4$ and $k = 5$. The bits affected by the average error are 4 $ulp$ ($11.2 < 2^4$) for $k = 4$, while the average error is 5 $ulp$ ($28.3 < 2^5$) for $k = 5$. However, the power savings are almost

[1] $ulp = 1$ for integers.

| Unit | Area $[\mu m^2]$ | $t_{MAX}$ $[ns]$ |
|---|---|---|
| Regular | 927 | 1.59 |
| Trunc-4 | 856 | 1.45 |
| Trunc-8 | 457 | 1.22 |
| DPA-III | 973 | 1.67 |

| Configuration | $P_{ave}$ $[\mu W]$ | savings [%] | Error $\|\overline{\epsilon}\|$ | $\|\epsilon_{max}\|$ |
|---|---|---|---|---|
| Regular | 153 | – | 0.0 | 0 |
| DPA-III ($k = 0$) | 150 | 2.0 | | |
| Trunc-4 | 133 | 13.1 | 11.2 | 49 |
| DPA-III ($k = 4$) | 139 | 9.1 | | |
| Trunc-8 | 78 | 49.0 | 561.6 | 1793 |
| DPA-III ($k = 8$) | 70 | 54.2 | | |

$P_{ave}$ measured at 100 MHz.

TABLE II
COMPARISON OF SYNTHESIS RESULTS FOR $8 \times 8$-BIT MULTIPLIER: DPA-III ($g = 4$) VS. TRUNCATED MULTIPLIER.

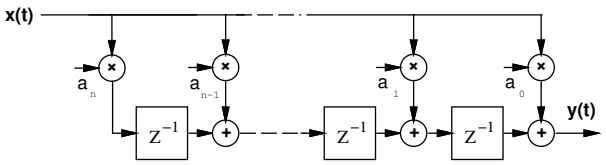


Fig. 5. FIR filter in transposed form.

| $k$ | $P_{ave}$ $[\mu W]$ | savings [%] | Error $\|\overline{\epsilon}\|$ |
|---|---|---|---|
| 0 | 1293 | – | 0.0 |
| 4 | 1127 | 13 | 211 |
| 8 | 728 | 44 | 1210 |

$P_{ave}$ measured at 100 MHz.

TABLE III
POWER-GATING RESULTS FOR 16-TAPS FIR FILTER.

three-fold for $k = 5$ over $k = 4$: 18.8% vs. 7.8%. That is, with one extra bit more inaccurate results, we can save significantly more power.

By increasing the granularity to $g = 2$, or $g = 4$, the overhead is significantly reduced. For example, by having granularity $g = 4$, as shown in Fig. 2, the delay overhead is reduced to 5%: $t_{g=4} = 1.67\ ns$, and the area overhead to 5%: $A_{g=4} = 973\ \mu m^2$.

We also compare the delay/area/power tradeoffs for hardware truncated multipliers with truncation in the 4 and 8 LSBs. The results are summarized in Table II for the four units:

1) *Regular*: plain 8×8-bit multiplier;
2) *Trunc-4*: 8×8-bit multiplier truncated in its 4 LSBs;
3) *Trunc-8*: 8×8-bit multiplier truncated in its 8 LSBs;
4) *DPA-III*: 8×8-bit multiplier truncated by power-gating with $g = 4$.

The results show that at expenses of a 5% overhead in delay and area when working in full precision, the DPA-III multiplier can save up to half the power when the 8 LSBs are power-gated ($k = 8$). Moreover, the power dissipation overhead with respect to a hardware truncated multiplier is negligible.

## V. CASE STUDY: FIR FILTER

As a case study, we implement a programmable 16-tap FIR filter in transposed form (Fig. 5). The dynamic range of both the input ($x$) and the coefficients $a_k$ is 8-bit, while the output of the filter is 16 bits when working in full precision. Power-gated ($g = 4$) 8×8-bit multipliers, similar to the one in Table II, are used in the filter. The registers holding the coefficients $a_k$, once loaded, are clock-gated to reduce the power dissipation to a minimum.

The results for power consumption and average error, reported in Table III, show that by working with truncated $k = 8$ multipliers more than 40% of the power can be saved. The truncated bits in the multipliers ($k = 0,\ 4,\ 8$) can be programmed by controlling the power-gating on-the-fly, although the transition 'on-off' (and 'off-on') requires two clock cycles as explained in Sec. III.

The screenshots of the layout of the filter are shown in Fig. 6. In the figure, the three different pictures show in dark color the gates which are active (not powered-off) when the DPA-III multipliers composing the filter are operated with power-gating $k = 0$ (full precision), $k = 4$ and $k = 8$, from left to right.

Fig. 6 shows that DPA-III implements the concept of *"dark silicon"*[2]. That is, by powering down parts of the chip area, we can reduce the power budget of the system, but at the same time we can still have full functionality, if required.

## VI. CONCLUSIONS AND FUTURE WORK

In this work, we introduce a programmable truncated multiplier in which the bits to be truncated are set by fine-grained power-gating (DPA-III).

The experimental results show that for truncation at the middle of the dynamic range (truncation that is tolerable in DSP), the power savings are about 50%. These power dissipation figures are similar to those obtained by hardware truncated multipliers of the same precision. However, differently from hardware truncated multipliers, DPA-III multipliers can work in full precision with a modest overhead in delay and power dissipation. If the delay overhead can be tolerated, power-gating can be applied with granularity as fine as 1 bit.

Several improvements can be applied to the DPA-III multiplier. For example, we can introduce error compensation when the precision is reduced by modifying the isolation gates to force the correction constants in some of the nodes at the on/off interface.

Moreover, we plan to combine power-gating (DPA-III) to clock-gating (DPA-I) to characterize the power savings vs. error trade-offs, and to combine DPA-III to voltage scaling when the circuit is working at reduced precision by exploiting

[2]Normally, the parts considered *"dark"* are the ones powered-off, but in the case of Fig. 6, intended for paper printing, the coloring is reversed.

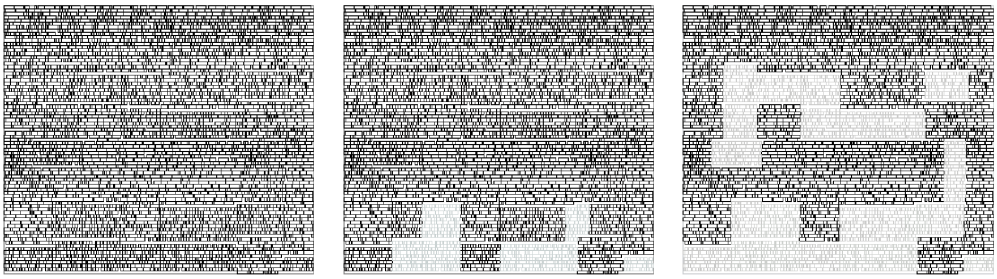


Fig. 6. Screenshots of the FIR filter layout under different power-gating modes. Dark areas are cells which are active.

the shorter delay paths due to the powering-off of portions of the datapath.

## REFERENCES


[1] K. He, A. Gerstlauer, and M. Orshansky, "Controlled Timing-Error Acceptance for Low Energy IDCT Design," *Proc. of 2011 Design, Automation and Test in Europe Conference (DATE)*, Mar. 2011.

[2] A. Lingamneni, J.-L. N. C. Enz, K. Palem, and C. Piguet, "Energy Parsimonious Circuit Design through Probabilistic Pruning," *Proc. of 2011 Design, Automation and Test in Europe Conference (DATE)*, Mar. 2011.

[3] D. Mohapatra, V. Chippa, A. Raghunathan, and K. Roy, "Design of Voltage-Scalable Meta Functions for Approximate Computing," *Proc. of 2011 Design, Automation and Test in Europe Conference (DATE)*, Mar. 2011.

[4] P. Albicocco, G. C. Cardarilli, A. Nannarelli, M. Petricca, and M. Re, "Imprecise Arithmetic for Low Power Image Processing," in *Proc. of 46th Asilomar Conference on Signals, Systems, and Computers*, Nov. 2012, pp. 983–987.

[5] M. Petricca, G. C. Cardarilli, A. Nannarelli, M. Re, and P. Albicocco, "Degrading Precision Arithmetic for Low Power Signal Processing," in *Proc. of 44th Asilomar Conference on Signals, Systems, and Computers*, Nov. 2010, pp. 1163–1167.

[6] A. Sathanur, A. Calimera, A. Pullini, L. Benini, A. Macii, E. Macii, and M. Poncino, "On Quantifying the Figures of Merit of Power-Gating for Leakage Power Minimization in Nanometer CMOS Circuits," *Proc. of IEEE International Symposium on Circuits and Systems (ISCAS 2008)*, pp. 2761–2764, May 2008.

[7] M. J. Schulte and E. E. Swartzlander, "Truncated multiplication with correction constant [for DSP]," in *Workshop on VLSI Signal Processing*, 1993, pp. 388–396.

[8] M. Sjalander, M. Drazdziulis, P. Larsson-Edefors, and H. Eriksson, "A low-leakage twin-precision multiplier using reconfigurable power gating," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2005, pp. 1654–1657.

[9] K. Usami, M. Nakata, T. Shirai, S. Takeda, N. Seki, H. Amano, and H. Nakamura, "Implementation and evaluation of fine-grain run-time power gating for a multiplier," in *IEEE International Conference on IC Design and Technology (ICICDT)*, 2009, pp. 7–10.

[10] L. Dadda, "On parallel multipliers," *Alta Frequenza*, vol. 45, pp. 574–580, 1976.

[11] M. Ercegovac and T. Lang, *Digital Arithmetic*. Morgan Kaufmann Publishers, 2004.

[12] Synopsys Inc. Synopsys 32/28nm Generic Library. [Online]. Available: http://www.synopsys.com/ Community/UniversityProgram/Pages/32-28nm-generic-library.aspx