2012 IEEE Computer Society Annual Symposium on VLSI

Data-Width-Driven Power Gating of Integer Arithmetic Circuits

Tung Thanh Hoang and Per Larsson-Edefors
VLSI Research Group, Department of Computer Science and Engineering,
Chalmers University of Technology, SE-412 96 Gothenburg, Sweden
Email: {hoangt, perla}@chalmers.se

***Abstract*— When performing narrow-width computations, power gating of unused arithmetic circuit portions can significantly reduce leakage power. We deploy coarse-grain power gating in 32-bit integer arithmetic circuits that frequently will operate on narrow-width data. Our contributions include a design framework that automatically implements coarse-grain power-gated arithmetic circuits considering a narrow-width input data mode, and an analysis of the impact of circuit architecture on the efficiency of this data-width-driven power gating scheme. As an example, with a performance penalty of 6.7%, coarse-grain power gating of a 45-nm 32-bit multiplier is demonstrated to yield an 11.6x static leakage energy reduction per 8x8-bit operation.**

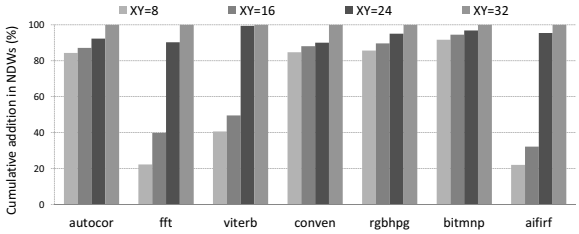


Fig. 1. Addition input operand data widths in EEMBC benchmarks.

I. Introduction

Integer arithmetic units are performance critical and highly active circuits that have a substantial impact on both dynamic and static power dissipation. Various techniques to reduce power have been proposed at software, architecture, circuit and device levels. Operand width adaptation that allows a circuit to efficiently take on the narrower data that are present in many applications is a technique that fits arithmetic circuits very well. Based on architecture simulations on a set of processor benchmarks, it was shown that roughly 35%, 50% and 55% of the integer 64-bit instructions had operand widths less than 8, 16, and 32 bits, respectively [1]. Fig. 1 shows the proportion of data widths for addition instructions of a number of EEMBC benchmarks [2] running on a 32-bit processor. The embedded domain clearly uses a very significant number of narrow data, for example, almost 80% of the additions in autocor and bitmnp are 8 bits or less.

Existing operand width adaptation techniques come in two flavors: In the first one, DSP-dominated applications are statically analyzed to identify the optimal data width for a given error constraint, such as the signal-to-noise ratio (SNR) [3]. As a result, the implemented hardware is minimized to reduce power dissipation. For example, Mallik et al. proposed a methodology that achieved power reductions for several DSP circuits of, on average, 50% for error constraints less than 1% [4]. In a project on image processing applications using the discrete cosine transform (DCT), the power dissipation was reduced by up to 75% at a peak SNR degradation of 5.56 dB by tuning the DCT coefficient widths [5].

The second operand width adaptation scheme entails using versatile circuits that support several data widths, for example, full-data-width (FDW) and narrow-data-width (NDW) computation modes. Under the right circumstances those circuits can be run in NDW mode, leading to reduced power dissipation: By prohibiting unused input data bits to propagate downstream using smart logic solutions [6], [7], [8], [9] and/or gating clock of unused registers [10], unnecessary switching activity is avoided.

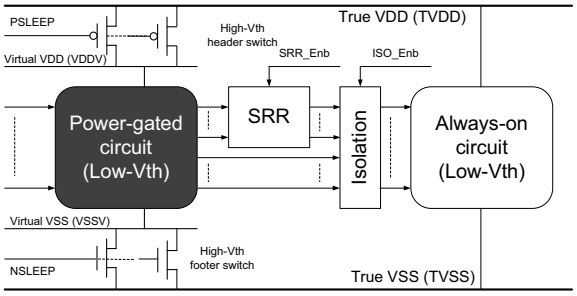
As an unwanted side effect of the relentless process scaling, static power dissipation has come to challenge the implementation of energy-efficient systems and many different techniques to reduce various leakage currents have emerged [11]. Power gating is a commonly used circuit technique that eliminates the static leakage power of a circuit by way of sleep transistors (also known as power switches), that is, transistors inserted to optionally remove the power supply and/or ground from the circuit. The sleep transistors can be clustered, in a coarse-grain scheme, to power down entire circuit blocks or embedded into every standard cell, in a fine-grain scheme. The latter scheme can be applied at the bit level of the input operands [12], however, this requires cell modifications to integrate sleep transistors, which incurs a significant area overhead—up to 30% [13]—if at all compatible with the cell library. An intermediate alternative would be to power down one coherent part of a multiplier, for example, the part that handles the higher significance bits, if the remaining NDW part of the multiplier provides a data precision that is satisfactory for the executed application [14], [15]. Equal operand widths are assumed in the power-gated multipliers above. Combinations of operands with different widths—asymmetric input operand widths—have not been considered despite this is a common scenario in DSP applications.

In this paper, we apply and evaluate coarse-grain power gating for integer arithmetic circuits that can support a range of input operand widths in their NDW mode. Our contributions include:

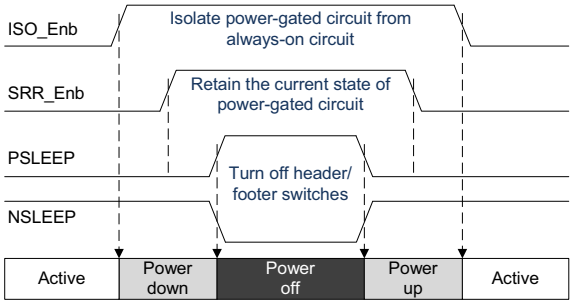
- An automatic framework to deploy coarse-grain power gating in integer arithmetic circuits with arbitrary input operand widths in NDW mode.
- An evaluation of the impact of power gating in terms of energy efficiency, and timing and area overhead for various architectures of arithmetic circuits.
- An evaluation of the impact of symmetric and asymmetric input operand widths on power gating efficiency.

The rest of this paper is organized as follows: Basic power gating is reviewed in Sec. II. Power gating in the context of adders and multipliers that support an NDW mode is discussed in Sec. III. Sec. IV describes the design framework, while Sec. V provides the evaluation methodology, simulation results and an ensuing discussion. The paper is concluded in Sec. VI.

978-0-7695-4767-1/12 $26.00 © 2012 IEEE
DOI 10.1109/ISVLSI.2012.59



(a) Power-gated circuit structure.



(b) Power up/down timing sequence.

Fig. 2. Implementation of power gating.

II. Power Gating

Power gating, or multi-threshold CMOS (MTCMOS), is an effective technique to reduce leakage power dissipation [11] of a circuit that is idle. As shown in Fig. 2, key components of power gating implementations are the high threshold voltage (Vth) sleep transistors (the header and footer power switches) that are inserted between the virtual power/ground (VDDV/VSSV) rails of the circuit and the true power/ground (TVDD/TVSS) rails. When the circuit is idle, the sleep transistors are turned off. Thus, no leakage currents can flow between the global power and ground rails, and consequently the leakage power in the power-gated region is dramatically reduced.

The implementation of power gating is complex since it impacts, for example, the timing of the circuit in active mode [16]. Implementation considerations include issues such as the following:

- The power switches must be carefully sized to limit the performance degradation, which is caused by the voltage drop across the switches, to a given design constraints.
- The energy overhead due to power-up/down transitions must be considered. The higher this overhead, the longer the idle time of the circuit has to be, to reach an overall energy reduction.
- The power and area overhead of introducing interface circuits, which allow the power-gated region to interface with another block that is in always-on state, must be acceptable.

The optional state-retention register (SRR) is used to keep the last state of the power-gated circuit before it is powered down. The output isolation circuit ensures that the floating outputs of the power-gated circuit do not affect the logic functionality of the always-on circuit. We refer to Fig. 2(a) for a schematic of the circuit context. Obviously, the addition of interface circuits degrades the performance of the power-gated circuit, and introduces power and area overheads.

In order to guarantee functional correctness when the power-gated circuit is powered-up/down, the blocks of isolation, SRR and power switches must be enabled/activated with the timing sequence that is shown in Fig. 2(b). For the power-down transition, the isolation circuit must first be asserted, then SRR is enabled to keep the current state of the power-gated circuit, and finally the power switches are turned off. These steps are done in reverse for the power-up transition.

III. Dual FDW/NDW Mode Power-Gated Arithmetic Circuits

Since arithmetic circuits often are timing critical and thus use high-performance devices, a relatively large proportion of their total power dissipation is due to leakage. Hence, it stands to reason that data-width-driven power gating of arithmetic circuits can significantly increase the energy efficiency. Here, power gating can basically be implemented to two different extents: 1) When in NDW mode, logic not necessary for the NDW part of the computation is power gated. 2) When idle, the entire circuit is power gated. The last scenario may not be applicable to a processor with a single ALU, for which there exist almost no idle cycles. However, with an increasing level of computational parallelism, instantiations of many parallel functional units are key to having a system peak performance that can match the need of the most demanding computational phases. Outside the most demanding phases, performance throttling by way of deactivation of arithmetic circuits is important to save energy. This paper addresses circuits in the first scenario, that is, circuits that are never completely idle but always active, either in FDW or in NDW mode.

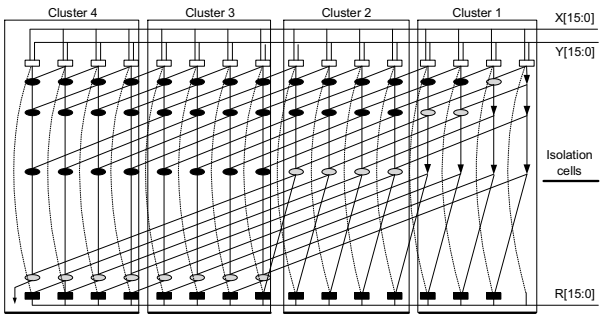
The arithmetic circuits considered here are integer adders and multipliers, since both are very common building blocks, but still represent two different levels of design complexity. As an example, the circuit structure of a 16-bit Kogge-Stone parallel adder is shown in Fig. 3(a). Regardless of the different functions of the logic elements, the implementation of this 16-bit adder requires in total 80 logic elements. In contrast to the multipliers that are treated below, the number of power-gated logic elements depends directly on the number of expected output bits in NDW mode, regardless of the different widths of the input operands.

The adder in Fig. 3(a) can be divided into four 4-bit clusters. When the clusters are sequentially powered down from left to right, we can notice that the fraction of logic elements that can be powered down is significantly increasing: From 30% when only cluster 4 is powered down, up to 85% when only cluster 1 is active. Data-width-driven power gating clearly has a chance to impact power significantly. Furthermore, the implementation of a power-gated adder is straightforward, since this does not require any functional modifications to the original circuit.

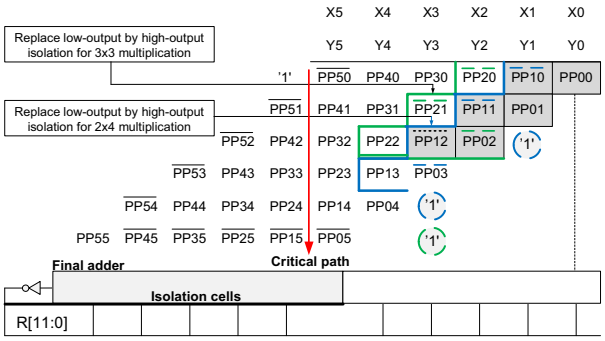
As a second arithmetic circuit in this study, the multiplier is a large block that is not as frequently used as the adder, neither in FDW nor NDW modes [1], [17]. As an example design, we show a signed 6x6-bit multiplier based on the Baugh-Wooley multiplication algorithm, a Wallace reduction tree and an 11-bit Kogge-Stone structure as final adder in Fig. 3(b). Here, PPs refer to partial products which are generated by 2-input AND gates driven by input bits of X and Y operands, for example, PP10 = X1 & Y0.

For symmetric signed FDW multiplications, to begin with, PPs in the leftmost column and the bottom row are negated, except the most significant one, that is, PP55. Then, a '1' is added on the left side of the PP in row 1 [18]. Finally, the most significant output bit is negated [9].

As far as signed NDW multiplications, consider the symmetric signed FDW multiplier above for two simple NDW examples, that



(a) Signed 16-bit Kogge-Stone adder with power-gated 4-bit clusters.



(b) Signed 6x6-bit multiplier for 3x3- and 4x2-bit NDW multiplications.

Fig. 3. Arithmetic circuits that support FDW and NDW operation modes.

is, 3x3-bit symmetric and 2x4-bit asymmetric multiplications[1]. As shown in Fig. 3(b), the PPs needed to be active for 3x3-bit and 2x4-bit multiplications are encircled by green and blue solid lines, respectively. Low-output isolations are inserted at the boundaries due to reasons mentioned in Sec. II. We use the following method to modify the 6x6-bit multiplier to make it support NDW multiplication:

- For the 3x3-bit multiplication, PP20, PP21, PP12 and PP02 (black dot and green dashed bars) are negated. Similarly to the symmetric FDW mode, an extra bit (dashed green circle) needs to be added either at the top or at the bottom of column 3 (fourth from the right). We implement this without any extra logic by replacing the low-output isolation cell connected to PP30 by a corresponding high-output one.
- For the 4x2-bit multiplication, PP10, PP11, PP12, and PP03 (black dot and blue dashed bars) are negated. In contrast to the symmetric FDW mode, two extra bits (dashed blue circles) need to be added at the top or at the bottom of columns 1 (second column from the right) and 3. The '1' added in column 3 can be processed in the same manner as for the 3x3-bit multiplication, while we need extra logic for the bit added in column 1.

The number of active PP cells for 3x3-bit and 4x2-bit multiplications is 9 and 8, respectively. This means that after power gating the asymmetric multiplication may consume less power than the symmetric case, despite both cases have the same output width. This makes asymmetric multiplications, which are common in DSP applications such as filters, interesting from a power gating perspective.

IV. Data-Width-Driven Power-Gating Framework

In this section we present a design framework (Fig. 4) that allows us to deploy data-width-driven coarse-grain power gating in integer arithmetic circuits. The framework automatically modifies the original multiplier circuit netlists using the techniques presented in Sec. III, except the replacement of low-output by high-output isolation cells, which is accomplished in a later step. Since the power gating of adders does not significantly impact the original adder netlist, modifications are not necessary for these blocks.

A. Circuit Clustering for NDW Mode

The arithmetic circuits that operate in both FDW and NDW modes are clustered into two power domains, called active and power-gated domains. First we consider the arithmetic circuit netlist as a graph. Given the two narrow operand widths, a general clustering procedure can be performed to identify what gates need to be active in the NDW mode:

1) Find the first gate cluster by traversing the graph in a forward, breadth-first manner, from the input bits of the NDW operands and mark the visited gates as cluster-1 gates[2].
2) Find the second gate cluster by traversing the graph in a backward, breadth-first manner, from the output bits that produce the NDW results and mark the visited gates as cluster-2 gates.
3) The intersection of the two clusters represents logic gates that should be assigned to the active domain.

The procedure outline above can be applied to hierarchical, pipelined designs supporting multiple NDW modes and, although we do not advocate it, fine-grain power gating.

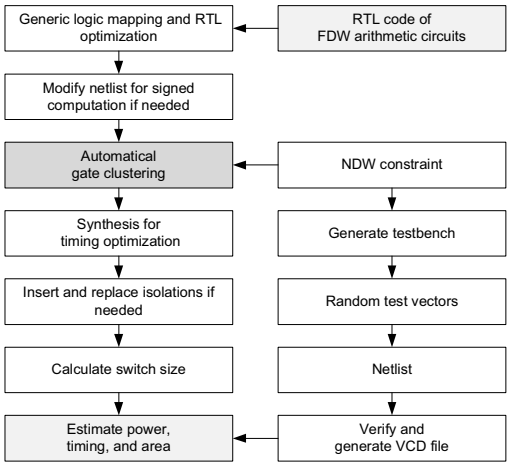


Fig. 4. Design framework for coarse-grain power-gated arithmetic circuits operating in FDW and NDW modes.

B. Power Switch Size Estimation

A power switch designer is faced with several conflicting requirements. First, the power switch must be sufficiently large to limit the voltage drop across the switch, to guarantee a certain level of performance. This voltage drop does not only depend on the switch size, but it also strongly depends on—we assume a footer

[1]Both cases produce the same output width of 5 bits.

[2]This step is not needed in adders.

switch here—the maximum discharge current (MDC). Second, by using a large switch, the wake-up and power-down time during mode transition can be reduced. On the other hand, a larger power switch incurs higher subthreshold leakage currents, increasing the static power of the idle circuit. Also, the in-rush current during wake up increases with the power switch size, increasing power supply variations in the active circuit domains. Thus, finding the optimum power switch size is a difficult problem. In fact, finding an important parameter like MDC is infeasible for large circuits, since the simulation time is growing as 4^n, where n is the number of primary inputs.

One simple and practical method to find an approximative power switch size is the average current method (ACM) method [19]. The main idea is that, given very tight performance degradation constraints, a circuit's MDC is not dependent on the input vectors. As a result, for tight constraints, the average current can be used to replace MDC when identifying power switch sizes. For this study, we have implemented the ACM method by simulating a finite set of input vectors.

The expression used to calculate the power switch size is

$$W_{sw} = \frac{I_{avg} * Ron_{sw}^{nom}}{\Delta V} = \frac{P_{avg} * Ron_{sw}^{nom}}{V_{DD} * \Delta V}$$

where W_{sw} is the total power switch size (in μm) that is required to keep the voltage drop across switches below ΔV. I_{avg} is the average switching current of the logic circuit that needs to be power gated in active mode, that is, the ratio of average switching power P_{avg} to supply voltage V_{DD}. The value of P_{avg} is estimated by gate netlist simulations using random input vectors. The final parameter, Ron_{sw}^{nom}, is the turn-on resistance of a power switch of unit width (in $\Omega\mu$m). Here, Ron_{sw}^{nom} is derived by simulating a single power switch, whose voltage drop is ΔV, extracting its turn-on resistance from the I-V slope in the linear region, and multiplying with the power switch size. For the process technology used in this paper, the high-Vth header and footer switch are used and their R_{sw} are found to be $2.77 \cdot 10^3$ and $9.48 \cdot 10^2$ $\Omega\mu$m, respectively, when V_{DD} is 1.0 V and ΔV is 3% of V_{DD}.

V. Evaluation

A. Architectures and Evaluation Methodology

We consider the following arithmetic circuits:

- Four 32 adders using the well-known Kogge-Stone (KSA-32), Han-Carlson (HCA-32), Carry-Lookahead (CLA-32), and Brent-Kung (BKA-32) architectures.
- One 32-bit signed multiplier using the architecture described in Sec. III.

Initially we generate VHDL descriptions [20] and develop parameterizable testbenches for all evaluated architectures. The testbenches support simulation in active, power-gated and dual modes with different idle times.

The VHDL code of each evaluated 32-bit architecture is first clustered with an NDW granularity of 8 bits, that is, 8, 16, and 24 bits. Subsequently the clustered VHDL code is synthesized using RTL Compiler for timing optimization and a commercial 45-nm low-Vth library, at 1.0 V and a fast-fast corner. Synthesized netlists are verified by using the NCSIM logic simulator. A common power format [13] (CPF) file is developed to specify low and high-output isolations, header power switches and peak voltage drop across the switches. The CPF file is also used for leakage power estimation. When the circuit is powered down, unused bits are set to zero, thus, the state-dependency of leakage power is not utilized. Two power domains are defined by the CPF file, in which the power-on domain includes the gates for the NDW mode, including always-on isolation circuits, and the other gates belong to the power switchable domain. For the physical implementation, the power switches are placed in a column-based style, while the isolation circuits are placed at the output pins of the power switchable domain. As an example, Fig. 5 shows the physical implementation of a power-gated KSA-32 adder which supports both 8-bit NDW and 32-bit FDW modes. Here, header switches and AND-based isolation circuits are used.

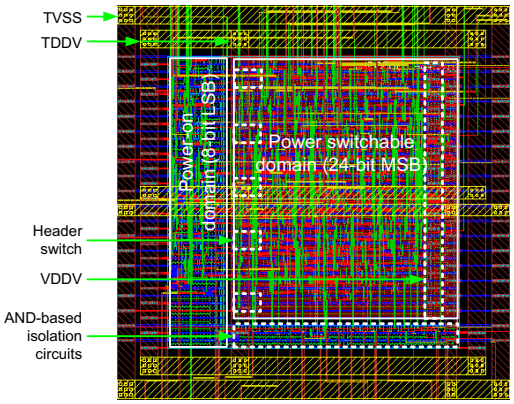


Fig. 5. Layout view of KSA-32 adder that supports an 8-bit NDW mode.

Since the performance of a power-gated circuit depends on both its architecture and the number of gated bits, we use leakage energy per operation to simultaneously capture power and performance for the architectures considered in our evaluation.

B. Evaluation Results

1) Active cell count: Since the tools heuristically perform mapping of RTL code into cell libraries, the number of active gates for each NDW mode is somewhat unpredictable. To cancel out the impact of tool heuristics as much as possible, we perform a generic mapping to find the number of active gates required for a specific NDW mode. As shown in Fig. 6, for all evaluated adders, when both operands are gated in steps of 8 bits, the number of active gates is reduced by 1.5x-15.7x. For the multiplier, shown in Fig. 7, the reduction is 1.2x-14.1x.

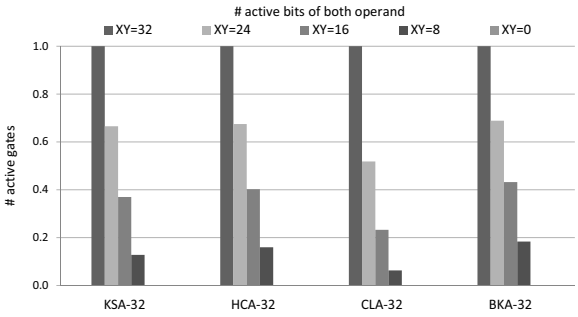


Fig. 6. The number of active gates in adders, normalized to a design without power gating.

2) Leakage energy dissipation: First we consider the leakage energy of the evaluated adders for various input operand widths. The leakage energy dissipation is shown in Fig. 8. We observe that for all adder architectures, the leakage dissipation is reduced by 30-96%

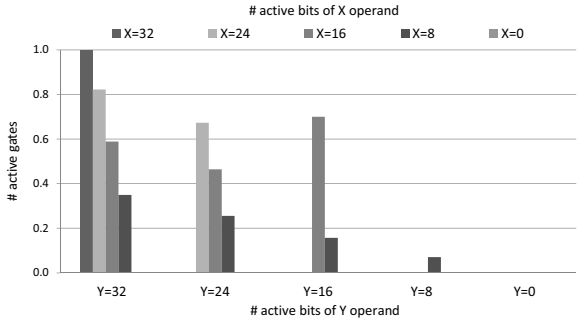


Fig. 7. The number of active gates in the multiplier, normalized to a design without power gating.

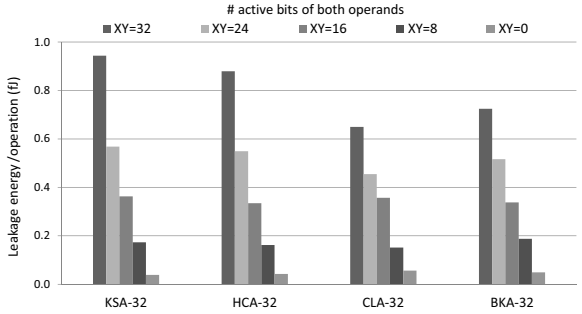


Fig. 8. Leakage energy per operation of 32-bit adders for different NDWs.

when each 8-bit group is gradually powered down. The largest energy reduction, up to 40%, is obtained when the 8 most significant bits of the KSA-32 adder are power gated. This is because the KSA-32 adder has higher logic density than the other adders and the logic gates of the 8 most significant bits mostly belong to the critical paths. If only the 8 least significant bits are used in NDW mode, the energy efficiency is improved in the range of 80-85% compared to circuits without power gating. Note that for the fully power-gated adders, the number of always-on isolation circuits is high and, thus, the leakage energy overhead due to the always-on isolation circuits becomes significant.

For the 32-bit multipliers we identify the leakage energy dissipation for both symmetric and asymmetric operand widths. Evaluation results of the leakage energy of the considered 32-bit multiplier are presented in Fig. 9. For the same active width of input operands, the leakage energy of the multiplier is reduced by 22%, 63% and 91% for the 24, 16, and 8 input bits used in the respective NDW modes. When the multiplier is fully powered down, there is a negligible leakage energy overhead due to the fact that the always-on isolation circuits are relatively few. The use of asymmetric operands widths on X and Y indeed has an impact on the leakage energy. For instance, both the symmetrical case of 16-bit X and 16-bit Y and the asymmetrical case of 24-bit X and 8-bit Y produce the same output width of 31, however, the leakage energy dissipations are significantly different; the asymmetrical case yields a 17% improvement in energy reduction over the symmetrical case.

3) Timing overhead: There is a timing overhead in active mode that is incurred by the reduced voltage on the virtual supply, which is due to the power switch voltage drop, and the inserted isolation circuits. The power switches are sized large enough to limit their

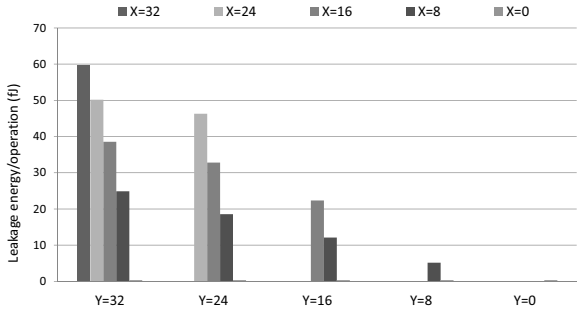


Fig. 9. Leakage energy per operation of 32-bit multiplier for different NDWs.

impact on performance to less than 3%[3].

When an adder operates in NDW mode, one level of isolation is added at the outputs. Depending on the adder netlist, the added isolation delay impacts the performance differently. The original KSA-32 adder has a maximum clock rate of 2.5 GHz, while the power-gated KSA-32 is limited to a clock rate of 2.1 GHz, that is, power gating incurs a 18.3% performance reduction. The corresponding performance degradation for HCA-32, CLA-32, and BKA-32 is 17.2%, 15.6%, and 14.2%, respectively.

In the multiplier, there are as many as three levels of isolation circuits required at the boundary of the active domain: Two are in the reduction tree and one is at the output of the final adder. Depending on the active width of the input operands, a maximum of two levels of isolation circuits are located on the critical paths, introducing a delay overhead. As shown in Fig. 3(b) for a 3x3-bit multiplication, only one level of isolation circuits at the output of the final adder impacts the multiplier performance, since this is located on the critical path (the red line). However, when the widths are increased, as in a 3x6-bit multiplication, two levels of isolation circuits in the reduction tree are on the critical paths.

In addition to the delay in the isolation circuits, an extra delay overhead stems from the circuit modifications that are required to handle signed computations in NDW modes. Fig. 10 shows the total performance degradation of the multiplier with respect to different NDW modes. At best, the performance degradation is merely 3%, at worst, it is 10.6%.

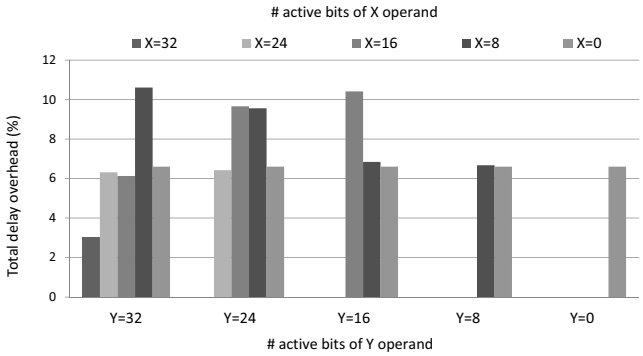


Fig. 10. Delay overhead of the 32-bit multiplier for different NDWs. For X=0, the multiplier is fully powered down.

[3]Techniques to optimize the size of power switches are not in the scope of this work.

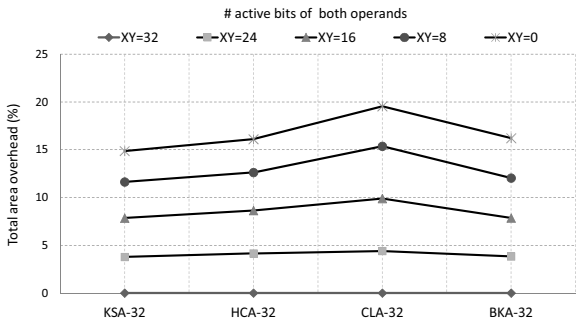


Fig. 11. Total area overhead of 32-bit adders for different NDWs using header switches.

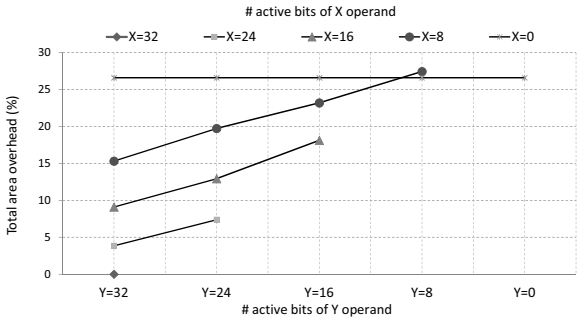


Fig. 12. Total area overhead of the 32-bit multiplier for different NDWs using header switches. For X=0, the multiplier is fully powered down.

4) Area overhead: The power gating area overhead is due to the isolation circuits and the power switches, *including* buffers for sleep signals. Using header switches, which each occupies 9.53 μm^2, Fig. 11 presents the area overhead of all evaluated adders for different NDW modes. For each extra 8 bit group that is idle in the NDW mode, 3-5% more area is required. Regarding the multiplier, Fig. 12 shows the area overhead of header-switch-based designs. When only 8 active bits are used, the area overhead of 27.4% is very significant. Note, however, that if the area of buffers for sleep signals is excluded, the area overhead due to only the sleep transistor insertion is reduced to 5.1%. In order to reduce the area overhead, header switches can be replaced by footer switches, with smaller area (4.23 μm^2) and lower turn-on resistance. A 32-bit fully power-gated multiplier that uses footer switches has a power-gating area overhead of only 4.6%[4].

VI. Conclusion

We present a versatile design flow for the implementation of coarse-grain power gating in integer arithmetic circuits that can support narrow data width (NDW) computations. This flow facilitates the design of power-gated circuits that may adapt to the lower computational precision that abound in applications. The results show that leakage energy, estimated for a range of architectures, can be significantly reduced when narrow data widths are utilized for data-width-driven power gating.

Not only symmetric, such as 8x8 and 16x16-bit operations, but also asymmetric operand widths are explored. Despite the number of output bits are the same, a 32-bit multiplier that uses coarse-grain power gating to support a 24x8-bit NDW mode dissipates 17% less energy than a 16x16-bit NDW mode. An added advantage is that the power gating for the asymmetrical case turns out to have a lower delay overhead than the symmetrical case.

Future work includes the complete physical implementation phase to obtain higher accuracy estimation results.

[4]While having almost the same amount of header switches, the multiplier with an 8x8-bit NDW mode uses more isolation circuits than the fully power-gated multiplier. This results in a slight increase in the total area overhead.

References

[1] D. Brooks and M. Martonosi, "Dynamically exploiting narrow width operands to improve processor power and performance," in *Proc. Int. Symp. High-Performance Computer Architecture*, Jan. 1999, pp. 13–22.

[2] *Embedded Microprocessor Benchmark Consortium*, http://www.eembc.org.

[3] B. Wu, J. Zhu, and F. Najm, "Dynamic-range estimation," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems (CAD)*, vol. 25, no. 9, pp. 1618–1636, 2006.

[4] A. Mallik, D. Sinha, P. Banerjee, and H. Zhou, "Low-power optimization by smart bit-width allocation in a SystemC-based ASIC design environment," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 3, pp. 447–455, Mar. 2007.

[5] J. Park, J. H. Choi, and K. Roy, "Dynamic bit-width adaptation in dct: An approach to trade off image quality and computation energy," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 5, pp. 787–793, May 2010.

[6] M. Munch, B. Wurth, R. Mehra, J. Sproch, and N. Wehn, "Automating RT-level operand isolation to minimize power consumption in datapaths," in *Proc. Design Automation and Test in Europe Conf.*, 2000, pp. 624–631.

[7] Z. Huang and M. Ercegovac, "Two-dimensional signal gating for low-power array multiplier design," in *Proc. IEEE Symp. Circuits and Systems (ISCAS)*, vol. 1, 2002, pp. 489–492.

[8] N. Banerjee, A. Raychowdhury, K. Roy, S. Bhunia, and H. Mahmoodi, "Novel low-overhead operand isolation techniques for low-power datapath synthesis," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 9, pp. 1034–1039, Sep. 2006.

[9] M. Själander and P. Larsson-Edefors, "Multiplication acceleration through twin precision," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 9, pp. 1233–1246, Sep. 2009.

[10] Q. Wu, M. Pedram, and X. Wu, "Clock-gating and its application to low power design of sequential circuits," *IEEE Trans. Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 3, pp. 415–420, Mar. 2000.

[11] 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.

[12] J. Pool, A. Lastra, and M. Singh, "Power-gated arithmetic circuits for energy-precision tradeoffs in mobile graphics processing units," *J. Low Power Electronics*, vol. 7, no. 2, pp. 148–162, 2011.

[13] *A Practical Guide to Low-Power Design, User Experience with CPF*, 1st ed. Powerforward.org, 2008.

[14] M. Själander, M. Drazdziulis, P. Larsson-Edefors, and H. Eriksson, "A low-leakage twin-precision multiplier using reconfigurable power gating," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 2005, pp. 1654–1657.

[15] 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 *Proc. IEEE Int. Conf. IC Design and Technology*, May 2009, pp. 7–10.

[16] Y. Shin, J. Seomun, K.-M. Choi, and T. Sakurai, "Power gating: Circuits, design methodologies, and best practice for standard-cell VLSI designs," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 15, no. 4, pp. 28:1–28:37, Oct. 2010.

[17] L. Wang, S. Paul, and S. Bhunia, "Width-aware fine-grained dynamic supply gating: A design methodology for low-power datapath and memory," in *Proc. Int. Conf. VLSI Design*, Jan. 2012, pp. 340–345.

[18] M. Hatamian and G. L. Cash, "A 70-MHz 8-bit x 8-bit parallel pipelined multiplier in 2.5-μm CMOS," *IEEE J. Solid-State Circuits*, vol. 21, no. 4, pp. 505–513, Aug. 1986.

[19] S. Mutoh, S. Shigematsu, Y. Gotoh, and S. Konaka, "Design method of MTCMOS power switch for low-voltage high-speed LSIs," in *Proc. Asia and South Pacific Design Automation Conf.*, Jan. 1999, pp. 113–116.

[20] Arithmetic module generator based on ARITH. [Online]. Available: http://www.aoki.ecei.tohoku.ac.jp/arith