A Low-Leakage Twin-Precision Multiplier Using Reconfigurable Power Gating

Magnus Själander, Mindaugas Draždžiulis, Per Larsson-Edefors, and Henrik Eriksson
VLSI Research Group, Department of Computer Science and Engineering
Chalmers University of Technology, SE-412 96 Göteborg, Sweden

***Abstract*— A twin-precision multiplier that uses reconfigurable power gating is presented. Employing power cut-off techniques in independently controlled power-gating regions, yields significant static leakage reductions when half-precision multiplications are carried out. In comparison to a conventional 8-bit tree multiplier, the power overhead of a 16-bit twin-precision multiplier operating at 8-bit precision has been reduced by 53% when reconfigurable power gating based on the SCCMOS power cut-off technique was applied.**

I. Introduction

Recent development of embedded systems indicates an increased interest in reconfigurable functional units that dynamically can adapt the datapath to varying computational needs. A system may need to switch between, for example, one application that needs speech-encoding functional units operating at 8-bit precision and another application that needs 16-bit functional units to perform audio decoding. Since most embedded systems are associated with a strictly limited power budget, both static and dynamic power are of critical importance. In this context, it is hardly acceptable to have idle functional units which dissipate useless static power while the application is running at low precision.

The twin-precision (TP) multiplier [1] can switch between *N*-bit and *N*/2-bit precision multiplications without significant performance and area overhead. However, in half-precision (*N*/2-bit) mode the TP multiplier is dissipating considerably more power than a conventional fixed-precision *N*/2-bit multiplier, since idle, higher-precision logic gates within the TP multiplier are leaking. In order to efficiently utilize reconfigurable datapath units, leakage reduction techniques need to be incorporated.

In this paper we describe a twin-precision multiplier that uses a power-gating strategy that dynamically adapt to the precision needed and shuts down idle portions of the circuit to save static leakage power.

II. Preliminaries

A. The Twin-Precision Multiplier

The twin-precision multiplier [1] is an *N*-bit tree multiplier that is capable of performing either *i)* single *N*-bit, *ii)* single *N*/2-bit or *iii)* two concurrent and independent *N*/2-bit multiplications. When compared to a conventional fixed-precision *N*-bit tree multiplier, the general gate count overhead of the twin-precision multiplier was very small, and the overhead in delay and active power dissipation for the full-precision mode was shown to be small [1]. The most important features of the twin-precision technique are illustrated in Fig. 1.

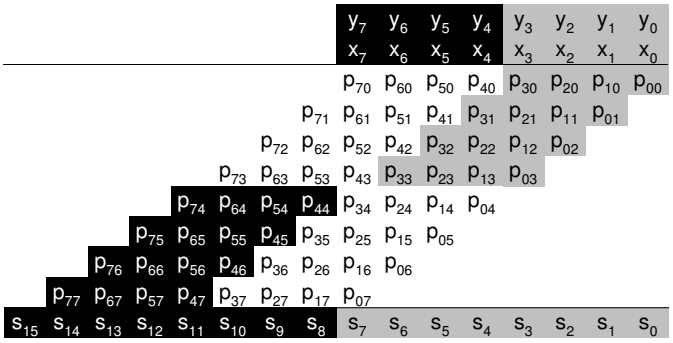

Fig. 1. Partial product representation of an 8-bit multiplication. When performing a 4-bit multiplication in a conventional multiplier only one quarter of the logic gates are performing any useful operations (the grey regions). The twin-precision technique rearranges logic to greatly reduce power dissipation and delay for 4-bit multiplications. Also, this technique allows the logic gates, which are not used in the 4-bit multiplication taking place in the grey region, to perform a second, independent 4-bit multiplication (the black regions).

In order to switch between the three different precision modes it is necessary that the partial products in white and black regions (Fig. 1) can be forced to zero independently of each other. The 2-input AND gates, which in conventional multipliers generate the partial product bits, are replaced with 3-input AND gates, whose extra input is used to mask the output to zero.

In a twin-precision multiplier it is crucial that the partial products used for *N*/2-bit multiplications are moved as close to the final adder as possible. In half-precision mode, this gives the shortest critical path and the largest number of unused logic gates, which can be translated into dramatic dynamic power reductions. The final adder is also important, since its delay profile will determine the delay for the *N*-bit and the two *N*/2-bit multiplications. In conventional adder design, the delay is optimized for full-precision operation. This generally makes such adders relatively slow when used for lower precision. A good tradeoff between the delay for *N*-bit and *N*/2-bit precision is given by the adder presented by Mathew *et al.* [2].

The application of the twin-precision technique to signed *N*-bit and *N*/2-bit multiplications is quite straightforward. An example of an 8-bit signed multiplier using the Baugh-Wooley algorithm [3], capable of performing two concurrent 4-bit multiplications, is shown in Fig. 2. The twin-precision

0-7803-8834-8/05/$20.00 ©2005 IEEE.

technique can also be applied to modified Booth multipliers, but this requires a slightly extended implementation effort.

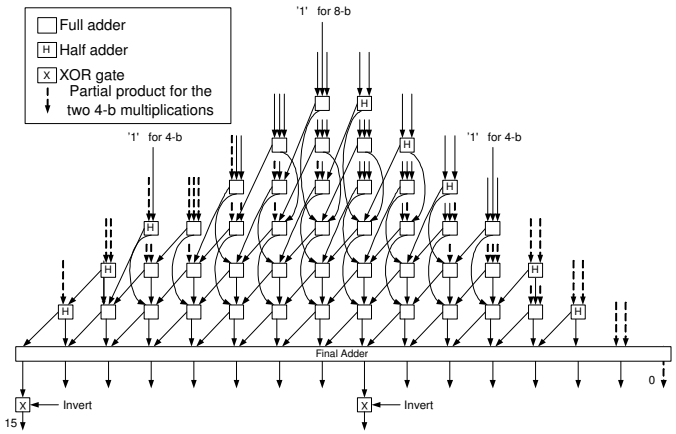

Fig. 2. Signed 8-bit twin-precision Baugh-Wooley multiplier.

B. Circuits for Leakage Reduction

As soon as unused circuit parts of a twin-precision (TP) multiplier stop switching or when the entire multiplier goes into sleep mode, static leakage currents become visible. Transistor stacking, body biasing, multi-V_t technologies and power cut-off techniques are common methods used to suppress static leakage. We have chosen to study only power cut-off techniques in this paper, because these are compatible with conventional CMOS design and require no special process technologies. Power cut-off techniques are using power switches (PMOS or NMOS transistors) that are turned *on* when logic circuits are active and *off* when circuits are in sleep mode. Generally, a power cut-off technique is efficient for circuits that stay in sleep mode for relatively long periods of time.

In this paper we have considered a number of power cut-off techniques [4] that can be applied to a TP multiplier: Super Cut-Off CMOS (SCCMOS), Zigzag Super Cut-Off CMOS (ZSCMOS) and Gate leakage Suppression CMOS (GSCMOS). We found that SCCMOS has long sleep-to-active (wake-up) time, while ZSCCMOS has short wake-up time at the expense of a need for sleep-mode state control (input forcing). The GSCMOS technique, on the other hand, is particularly efficient when gate leakage is significant, but it has complex supply rail routing. All considered power cut-off techniques need on-chip voltage generators to control the power switches. Although power cut-off techniques corrupt logic data during sleep, this is not a serious issue for the TP multiplier, since a fresh multiplication is performed every time the precision mode has been changed.

We chose the SCCMOS technique (Fig. 3), since this fits our needs and constraints (e.g. low supply routing complexity, a wake-up time within a few clock cycles, and absence of gate leakage) in the 0.13-μm technology used. The power switch used here is a low-V_t PMOS transistor, which in sleep mode is overdriven (i.e. above V_{dd}) with ΔV, which roughly is the

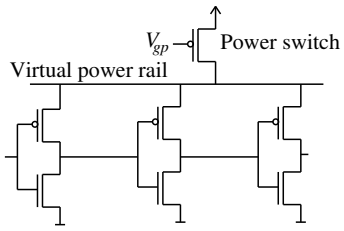

Fig. 3. The SCCMOS technique applied to an inverter chain.

threshold voltage difference between a typical high-V_t and a typical low-V_t device. The overdrive voltage completely turns the power switch *off*, thus the voltage on the virtual power rail drops and leakage currents are reduced.

III. Power Supply Grid and Tree Organization

To suppress static leakage in idle gates of the twin-precision (TP) multiplier, we deploy the SCCMOS technique so that three separate power-gating regions (Fig. 4) are obtained. Independent of other regions, the power switches of one region can be turned *on* or *off*, depending on multiplier precision mode.

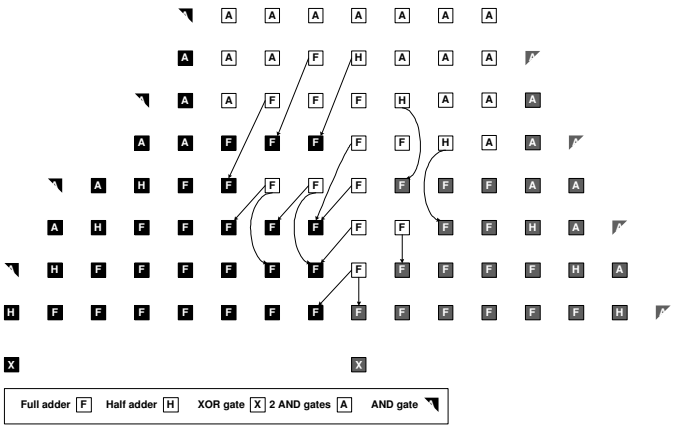

Fig. 4. Power-gating regions of an 8-bit TP multiplier. When one *N*/2-bit multiplication is performed, the power switches in black and white regions are turned *off*. When two concurrent *N*/2-bit multiplications are performed, the power switches in the white region are turned *off*. Finally, when an *N*-bit multiplication is performed, the power switches in all regions are turned *on*.

When the TP multiplier performs an *N*/2-bit multiplication or two concurrent *N*/2-bit multiplications, outputs of some sleeping logic gates are connected to the inputs of active logic gates (Fig. 4). Since those outputs now are floating, the inputs of the active logic gates can not be asserted via the 3-input AND gates that were used in the original TP multiplier. Instead, the inputs of these active logic gates are pulled down by NMOS transistors, which are connected to the respective outputs of the sleeping gates. When the TP multiplier performs an *N*/2-bit multiplication or two concurrent *N*/2-bit multiplications, the pull-down NMOS transistors are turned *on*, thus forcing the outputs of sleeping logic gates (i.e. the inputs of the active logic gates) to ground. The *off*-state power switches eliminate short-circuit currents in sleeping logic gates.

The virtual power rails that are introduced with the SCCMOS technique need special consideration. Within each power-gating region, the virtual power rails can be partitioned into networks of finer granularity than a single network spanning the entire region. In fact, every logic gate could have its own virtual power rail and power switch, leading to all logic gates having individual virtual supply voltages (which are functions of the gate input pattern) and guaranteeing overall optimal leakage suppression in sleep mode. However, this calls for an extra power-switch control wire that needs to be routed through *every* cell in a layout, in turn causing cell footprints and wire lengths to grow. Longer signal wires lead to larger switched capacitance and, thus, both the dynamic power dissipation and the delay of the active TP multiplier regions will increase. As for the other extreme, i.e. leaving the virtual power rails within each power-gating region non-partitioned, this is generally a recipe for larger than minimal leakage currents. This is because all logic gates of a large power-gating region will have one common voltage level on the virtual power rail. In sleep mode this common level rarely leads to minimal leakage currents in individual logic gates.

For the SCCMOS implementation of this paper (Fig. 5) we employ one virtual power rail and one power switch to groups of four full-adder (FA) cells (or for any group of logic cells that have the same total footprint). This partitioning yields a regular power supply grid that is easy to implement in layout. In our implementation, we assume constant cell pitch (cell height) and, as shown in Fig. 5, two AND gates are concatenated into one cell to make cell footprints uniform (two AND gates are now as wide as one FA cell).

In active mode, the power switch resistance will cause supply voltages on virtual power rails to drop below external supply voltage levels and, hence, the circuit delay will increase. The worst (largest) supply voltage drop takes place when the gates on a particular virtual power rail receive the input signals that make the maximum number of PMOS transistors switch simultaneously. We choose to size the power switches so that during switching, for the worst-case conditions described above, the voltage drop on all virtual power rails is equal. Therefore, all virtual power rails have identical width ratios (K), where K is defined for each virtual power supply network as the power switch width divided by the total width of the PMOS transistors that simultaneously switch for the worst-case condition.

Regarding the layout, power switches are inserted where an external power rail crosses a virtual power rail (Fig. 5). If an external power rail is routed between two different power-gating regions then *two* power switches (for two different virtual power rails) are inserted at each crossing. Since the N-wells of the PMOS transistors inside the logic gates are connected to the virtual power rails, the external power rails do not have to be routed horizontally. Hence, in the proposed power supply grid and tree organization the area penalty is limited to the vertically routed external supply rails. Since these rails increase signal wire lengths between cells, we also have a delay and power penalty when circuits are active.

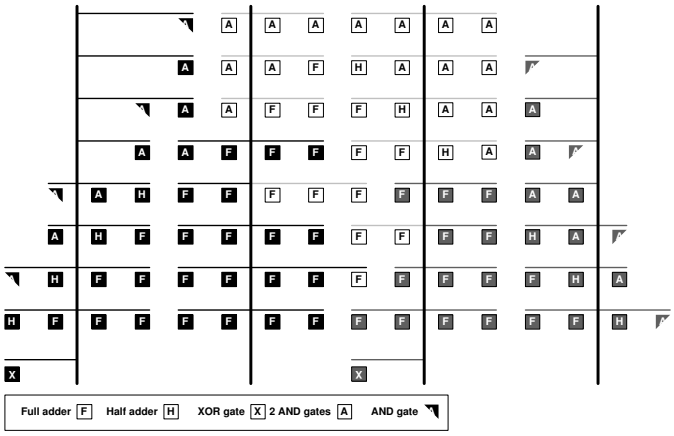

Fig. 5. The power supply grid of the 8-bit TP multiplier. Here, virtual power rails are routed horizontally, whereas external power and ground rails are routed vertically.

Logic gates in the white region, which are "trapped" between the grey and black regions and which do not have external supply rail routed through them, are connected to the black-region virtual power rails; the "white" FA-cell in the third row from the bottom in Fig. 5 is an example of a "trapped" logic gate. This connection simplifies the routing but incurs a small power penalty when two concurrent *N*/2-bit multiplications are performed, since "trapped" gates will be connected to the external power rails via the *on*-state black-region power switches.

Fig. 5 has a final adder (in the second row from the bottom) which for the sake of simplicity in this example is a ripple-carry adder. This adder can be replaced by any kind of tree adder, which subsequently can be partitioned in the same way as a multiplier reduction tree.

IV. Simulation and Results

We will now evaluate power dissipation and delay of a 16-bit twin-precision multiplier *with power-gating regions employing the SCCMOS technique*. As the first reference design we use a conventional twin-precision multiplier, which *does not* use power gating. In the rest of this section, *TP* refers to the conventional twin-precision multiplier in [1], whereas *TP-cutoff* refers to the proposed power-gated twin-precision multiplier. In order to strike a good balance between the delay of 8-bit and 16-bit multiplications, both the TP and TP-cutoff multipliers use the adder presented by Mathew *et al.* [2] as final adder.

As a second set of references we use conventional 8-bit and 16-bit tree multipliers. These references represent fixed-precision multipliers, which have final adders that are optimized for either 8-bit or 16-bit multiplications. Here, we used the Kogge-Stone [5] structures for final adders. Both conventional tree multipliers employ the SCCMOS technique with a power supply network partitioning which is similar to that of the TP-cutoff multiplier.

Simulation data were obtained by running HSpice on a commercially available 0.13-μm technology, at a supply volt-

age of 1.2 V and an operating temperature of 80°C. Circuit netlists were constructed using static CMOS gates and estimated wire capacitances (pre-layout). Power switches were overdriven with 0.2 V. Power figures were obtained by running simulations for 50 random input vectors applied at a 500-MHz frequency. Power dissipation from control signal transitions and overdrive voltage generators was not included.

We could not use PathMill [6] directly to obtain reliable delay figures for the power-gated multipliers. Instead, delay figures for all multipliers had to be obtained by the following steps: First, using PathMill, we found the critical path for a multiplier that did not use the SCCMOS technique. Then we constructed netlists of *only the critical path* for *i)* a multiplier without the SCCMOS technique and *ii)* a multiplier using the SCCMOS technique. These netlists combine, first, the structure of the critical path originally obtained by PathMill and, second, the logic gates used in the HSpice netlists of the respective complete multipliers. Finally, the critical paths were simulated using HSpice. Inputs to the critical paths were asserted such that signal rippling was guaranteed. To define accurate fan-outs, the critical-path netlists also contained all logic gates and respective wire capacitances that the rippling signals were driving. To realistically model capacitance on the virtual power rails, all logic gates connected to the power switches were also included.

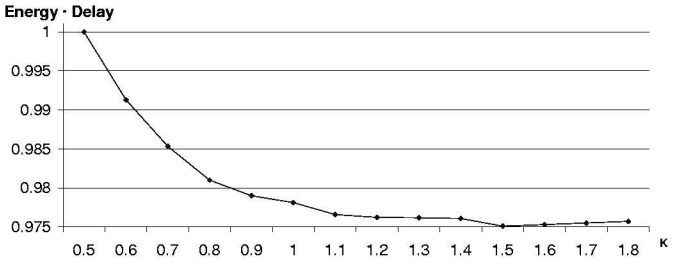

Fig. 6. Normalized energy-delay product for different power switch sizes.

Fig. 6 shows the energy-delay product in a 16-bit TP-cutoff multiplier (operating at 8-bit precision) as a function of power-switch size. When K=1.0, for each virtual supply network the power switch width is equal to the total width of the PMOS transistors that simultaneously switch for the worst-case condition. From Fig. 6 it can be observed that K=1.1 gives a good compromise between expended energy and delay, so K=1.1 is used henceforth. Also, with K=1.1 the wake-up times for the power-gated multipliers were within one clock cycle.

TABLE I

TOTAL POWER DISSIPATION [mW] FOR 8-BIT MULTIPLICATIONS.

Mult.	Mode	Cutoff
8-bit	8-bit	0.9667
16-bit	8-bit	3.473

(a) Conventional

Mult.	Mode	TP	TP-cutoff
16-bit	8-bit	1.216	1.083

(b) Twin-precision

The power dissipation of the 16-bit TP-cutoff multiplier operating at 8-bit precision can be compared to the power of an active conventional 8-bit multiplier, which incorporates power cut-off circuit techniques. In fact, the latter multiplier represents the lower power dissipation bound for any 16-bit multiplier that operates in 8-bit precision mode. As Table I shows, the 16-bit TP-cutoff multiplier only dissipates 12% more power than a conventional 8-bit multiplier. In comparison to a conventional 16-bit multiplier, which operates on 8-bit precision operands that are sign extended to 16 bits, the TP-cutoff multiplier has 3.2 times lower power dissipation. We finally observe that by using precision-dependent power-gating based on the SCCMOS technique, the power overhead (with reference to an active conventional 8-bit multiplier) of the 16-bit TP multiplier operating at 8-bit precision is reduced by as much as 53%.

TABLE II

DELAYS [ns] FOR 8-BIT AND 16-BIT MULTIPLICATIONS.

Mult.	Mode	Cutoff
8-bit	8-bit	1.190
16-bit	16-bit	1.687

(a) Conventional

Mult.	Mode	TP	TP-cutoff
16-bit	8-bit	1.400	1.402
16-bit	16-bit	1.759	1.805

(b) Twin-precision

As shown in Table II, in 16-bit precision mode the TP multiplier is less than 3% faster than the TP-cutoff multiplier. The conventional 16-bit tree multiplier that uses the SCCMOS technique is 7% faster than the TP-cutoff multiplier operating at 16-bit precision, a ratio that is largely due to the use of different final adders—Kogge-Stone in the former and the adder by Mathew *et al.* in the latter.

The difference in delay between the TP multiplier and the TP-cutoff multiplier operating in 8-bit precision mode is hardly noticeable. However, not surprisingly, the conventional 8-bit tree multiplier using the SCCMOS technique outperforms both twin-precision multipliers with 18%. This is due to *i)* use of a fixed 8-bit final adder and *i)* lower logic depth in the 8-bit reduction tree.

V. CONCLUSIONS

We have shown that power cut-off techniques can be deployed in different regions of a twin-precision functional unit, so that static leakage reduction can be effected not only when the entire unit is idle, but also when only parts of the unit are active, i.e. when the unit operates in half-precision mode.

REFERENCES

[1] M. Själander *et al.*, "An Efficient Twin-Precision Multiplier," in *Proc. Intl Conf. on Computer Design (ICCD)*, Oct. 2004, pp. 30–33.
[2] S. Mathew *et al.*, in *ISSCC*, Feb. 2004, pp. 162–3.
[3] C. R. Baugh *et al.*, *IEEE Trans. on Comp.*, pp. 1045–7, Dec. 1973.
[4] M. Dražđžiulis *et al.*, in *ESSCIRC*, Sept. 2004, pp. 171–4.
[5] P. M. Kogge *et al.*, *IEEE Trans. on Comp.*, pp. 786–793, Aug. 1973.
[6] *PathMill user guide, Version, Version U-2003.03-SP1.*