A Leakage Reduction Methodology for Distributed MTCMOS

Benton H. Calhoun, Frank A. Honoré, and Anantha P. Chandrakasan, *Fellow, IEEE*

***Abstract*—Multithreshold CMOS (MTCMOS) circuits reduce standby leakage power with low delay overhead. Most MTCMOS designs cut off the power to large blocks of logic using large sleep transistors. Locally distributing sleep devices has remained less popular even though it has several advantages described in this paper. However, locally placed sleep devices are only feasible if sneak leakage currents are prevented. This paper makes two contributions to leakage reduction. First, we examine the causes of sneak leakage paths and propose a design methodology that enables local insertion of sleep devices for sequential and combinational circuits. A set of design rules allows designers to prevent most sneak leakage paths. A fabricated 0.13-μm, dual $V_{\rm T}$ test chip employs our methodology to implement a low-power FPGA architecture with gate-level sleep FETs and over 8$\times$ measured standby current reduction. Second, we describe the implementation and benefits of local sleep regions in our design and examine the interfacing issues for this technique. Local sleep regions reduce leakage in unused circuit components at a local level while the surrounding circuits remain active. Measured results show that local sleep regions reduce leakage in active configurable logic blocks (CLBs) on our chip by up to 2.2$\times$ (measured) based on configuration.**

***Index Terms*—Active leakage, fine-grain leakage reduction, leakage reduction, MTCMOS, sleep regions, sneak leakage.**

I. Introduction

LEAKAGE power continues to increase in both active and standby modes as technologies scale [1]. Gate leakage, gate-induced drain leakage (GIDL), and reverse-biased diode leakage all contribute to the higher leakage power [2], but subthreshold leakage dominates for 0.13–μm devices. Multithreshold CMOS (MTCMOS) [3] circuits use a high $V_{\rm T}$ transistor to cut off a low $V_{\rm T}$ circuit from the power rails during sleep mode as shown in Fig. 1, but the cutoff device adds a resistance to ground during active mode. The boosted-gate MOS (BGMOS) approach [4] and super-cutoff CMOS (SCCMOS) [5] both enhance performance by respectively overdriving the sleep device's gate voltage in active mode and underdriving in sleep mode. Sequential MTCMOS circuits require special attention since they must hold state in sleep mode. MTCMOS flip-flop designs include using high $V_{\rm T}$ devices placed in parallel with low $V_{\rm T}$ devices [3], high $V_{\rm T}$ balloon latches [6], and input-referred conditional cutoff [7]. This paper uses a leakage feedback flip-flop (LFBFF) [8] that cuts off one power rail conditionally during standby based on feedback from the output. Specific sneak leakage paths have been addressed in sequential logic in [3] and [8], but this work examines them for the general case. MTCMOS standard cells have recently been developed, which show the usefulness of local sleep devices for a synthesis design approach [9].

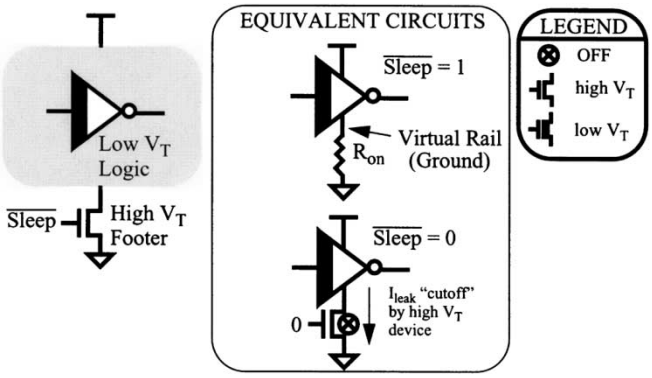

Fig. 1. Example of MTCMOS circuit style. High $V_{\rm T}$ power switches may be NMOS (shown) and/or PMOS.

II. Tradeoffs of Local versus Global Sleep Devices

This section compares local and global sleep devices considering overhead, sizing, noise margins, and leakage reduction. Table I provides a summary of the comparison. The local, or distributed, sleep methodology inserts sleep devices at the local block or gate level. We define a local block as a section of the circuit that can be independently idle, such as a multiplier or a buffer chain. The global sleep methodology uses a single sleep transistor for a large circuit block containing multiple local blocks, such as an entire chip or a full data path.

Both approaches require routing new traces to every cell in the circuit: the sleep signal for the local sleep devices, and the virtual node for global devices. For a global sleep device, routing the virtual rail is equivalent to routing $V_{\rm DD}$ or ground, so wide wires are required to avoid resistive voltage drops. True ground might be routed with smaller traces, but it still needs to access most cells for substrate contacts. The sleep control signal can be minimum width for the local approach, so its routing overhead is always less than that of the global case. However, the sleep control circuitry for local sleep devices depends on the circuit, while global designs only enter sleep when the entire circuit is idle. Although some local sleep designs will not require complicated control, random logic could demand complicated circuitry for determining when certain blocks are

Manuscript received July 15, 2003; revised January 5, 2004. This work was supported in part by Texas Instruments Incorporated and in part by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory under agreement number F33615-02-2-4005. The work of B. Calhoun was supported by an Infineon Fellowship.

The authors are with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: bcalhoun@mtl.mit.edu; honore@mtl.mit.edu; anantha@mtl.mit.edu).

Digital Object Identifier 10.1109/JSSC.2004.826335

TABLE I
GLOBAL VERSUS LOCAL SLEEP DEVICES SUMMARY

Comparison Point	Favored Approach	Reason
Routing Overhead	Local	sleep control signal routing less critical and smaller than virtual rail
Control Logic Area Overhead	Global	local sleep regions require control for when to turn on/off
Less Total Optimum Sleep Width (Greater Savings in Full Sleep)	Global	more opportunity for sharing width across mutally exclusive discharge regions
Ease of Sizing Sleep Devices	Local	Delay depends only on critical path and not on offpath discharge patterns
Noise Margins	Local	global has larger, data-dependent virtual rail bounce
Shutdown Flexibility (Active Leakage Savings)	Local	global can only turn off when entire circuit is idle

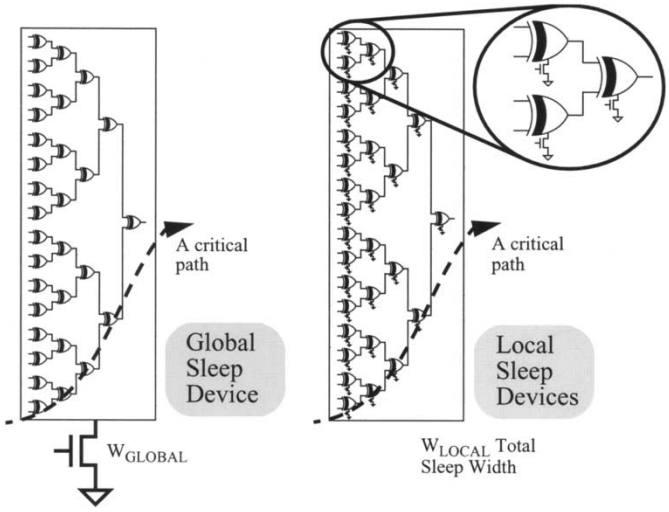

Fig. 2. Example of a 32-bit parity checker using global (left) and local (right) sleep methodologies. In practice, global blocks are often much bigger.

in sleep. This overhead makes the local approach impractical for some circuits. However, most low-power designs already employ clock-gating methods for reducing active power [10]. Clock-gating already requires signals that indicate when certain blocks are idle, so the presence of such signals could reduce the additional overhead for local sleep regions.

Let us define the optimum sleep size as the minimum sleep device width for which a circuit never exceeds a given delay. Suppose a circuit has optimum sleep devices distributed locally. Summing these widths gives a suboptimum global sleep device because, by superposition, the summed width is guaranteed to meet or exceed the performance of the local version even if all the gates discharge at once. Reducing the global sleep width is possible based on mutually exclusive discharge patterns of different gates or blocks [11]. Thus, for the same delay penalty, the optimum global sleep device is always smaller than the sum of optimum local sleep widths. The optimum global approach consequently reduces leakage current more than the local approach when the *entire chip* is asleep.

Fig. 3. Critical path from the 32-bit parity checker. Simulations of the entire parity checker in Fig. 4 and Fig. 5 measure delay through this path. Although unrelated nodes may switch, the 0 nodes shown above remain at 0 in all simulations.

Optimum global sleep devices are smaller, but their optimum size is difficult to determine. *Virtual rail bounce* is the primary cause of the sizing problem. Active mode current creates an IR drop across the on resistance ($R_{\rm on}$ in Fig. 1) of the NMOS sleep device that raises the virtual ground voltage by an amount $V_{\rm x}$. An analogous result occurs for a PMOS sleep device and a virtual power rail. This rail bounce slows the output high-to-low transition by reducing $V_{\rm GS}$ of the discharging, low $V_{\rm T}$ NMOS from $V_{\rm DD}$ to $V_{\rm DD} - V_{\rm x}$, and by raising its $V_{\rm T}$ through the body effect since $V_{\rm SB} = V_{\rm x}$. In deep-submicron devices, the lower $V_{\rm DS}$ will further reduce the discharge current because of drain-induced barrier lowering (DIBL).

For local sleep devices, exhaustive simulations can easily show that a gate or local block will always meet a given timing specification. The same guarantee becomes difficult to offer for large blocks without comprehensive simulation because the total circuit delay depends on the current drawn by the entire chip. The 32-bit parity checker in Fig. 2 provides a simple example to illustrate the effects of virtual rail bounce on local-style (right) and global-style (left) sleep devices. The parity checker is really a local block by our definition, but it behaves like a global block relative to the case with sleep devices in each gate. The local-style sleep devices are sized so that the slowest transition of each XOR gate is slowed by less than 20%, so the total delay penalty (average of all transitions) is close to 10%.

Fig. 3 shows a critical path through the parity checker. For the following simulations, the critical path acts as a buffer chain because the second inputs to the XOR gates are always 0 as shown. Fig. 4 shows a simulation of the global-style parity circuit in which two vectors are applied to the inputs. Vector 1 ($0 \times 00\,000\,001$) activates only the critical path, and Vector 2 exercises the critical path and switches some of the off-path gates.

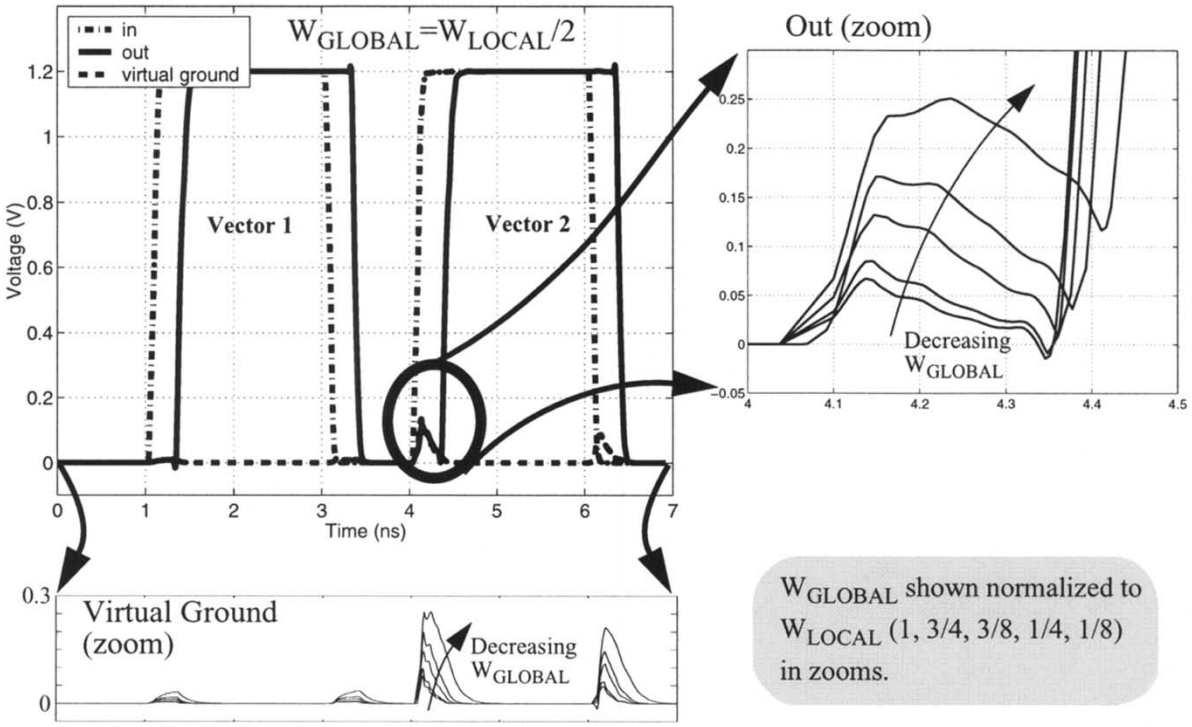

Fig. 4. Simulation of the 32-bit parity checker showing delay along the critical path in Fig. 3. The amount of virtual ground bounce for global-style sleep depends on the sleep device size ($\mathrm{W_{GLOBAL}}$) and on off-path data (Vector 1, Vector 2). Virtual bounce also decreases noise margins.

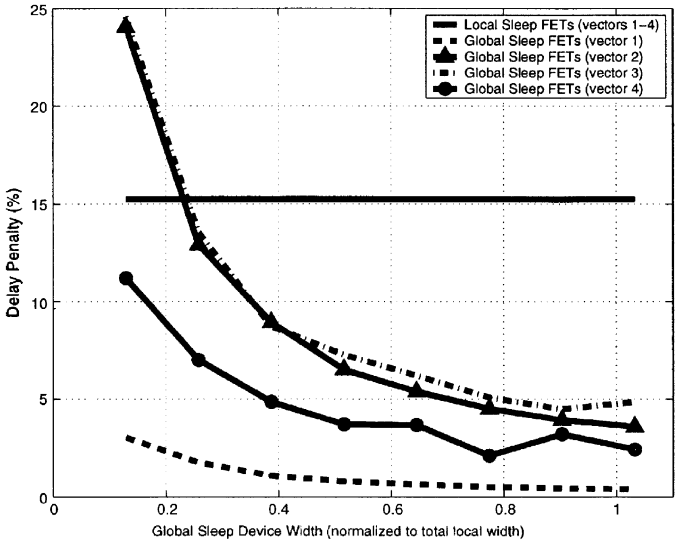

Fig. 5. Simulation results for the 32-bit parity checker showing delay penalty (average from two pulse edges) on the critical path from Fig. 3. For global-style sleep, delay depends on width and off-path data, so comprehensive simulation may be necessary to find worst case delay. For Vector 1, no other nodes are switching. Off-path nodes switch for vectors 2–4.

Clearly, the off-path switching causes a larger ground bounce. The large capacitance at the shared virtual rail filters the rail bounce so that its peak value is lower than might be expected. Although it decreases the delay, this filtering can have a negative effect since the resulting slow tailoff of the virtual bounce can affect future transitions. For circuits with a clock period close to the worst case delay, the tailoff can make delay depend on the previous transition. Fig. 5 shows that the critical path delay (average delay for the two edges of the pulse) depends strongly on the off-path data, whereas the local-style delay depends only on the critical path. The selected vectors do not necessarily show the worst case. Consequently, the worst case delay for global-style circuits is difficult to predict without accounting for all discharge patterns through comprehensive simulation.

Fig. 4 also illustrates how global sleep devices degrade noise margins in the closeup of the out signal. The voltage bounce that appears in the out signal of the simulation occurs at every node driven to 0, since the shared virtual ground node affects all devices. Clearly, this erosion of noise margins could lead to errors in sensitive circuits. Local sleep devices have better, more predictable noise margins [12] because their ground bounce depends on fewer discharging devices.

Local sleep devices offer a significant opportunity for leakage savings by using local sleep regions to turn off local blocks when they are idle. Thus, the circuit can remain active at a global level, but unused blocks will draw reduced leakage current. In general, the local approach is preferred for ease of design (standard cells [9]) or when local sleep regions can give active leakage savings that reduce total system leakage. The rest of this paper describes how to implement these local blocks without developing sneak leakage paths.

III. Sneak Leakage Prevention

A *sneak leakage path* is any current path from V_{DD} to ground that continues to draw high current relative to a cutoff path during sleep mode. Sneak leakage paths can occur whenever an MTCMOS output node is connected electrically to another node with low impedance to a power rail. This electrical connection most often occurs through low V_{T} transmission gates, but sneak

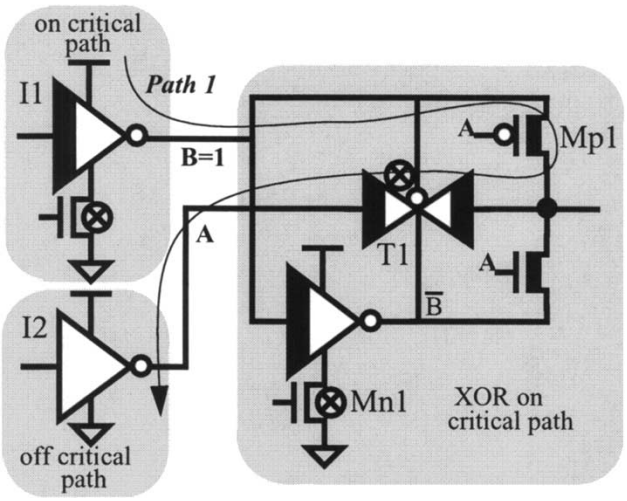

Fig. 6. Example of sneak leakage path at CMOS–MTCMOS interface. Path 1 occurs over several hierarchical levels and through several blocks. Path 1 is also data-dependent (in this example, the path disappears for $A = B = 1$).

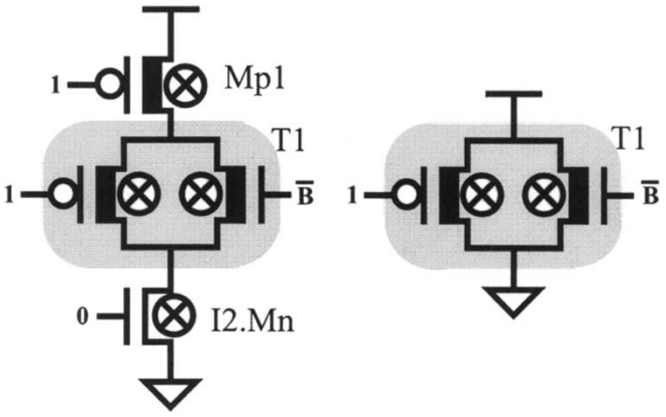

Fig. 7. Equivalent circuits for Path 1 in Fig. 6 with $A = 1$ (left) and $A = 0$ (right).

paths may also occur through structures such as clock-gated inverters and tri-state buffers. Formally, a sneak leakage path is a current path that flows from $V_{\rm DD}$ to ground through a set of "on" devices $\mathbf{A}$ and through a set of "off" devices $\mathbf{B}$. Set $\mathbf{B}$ contains only low $V_{\rm T}$ devices, while Set $\mathbf{A}$ contains low and/or high $V_{\rm T}$ devices. Since the off devices are low $V_{\rm T}$, the sneak leakage current is roughly an order of magnitude higher than other currents in the circuit, and floating nodes can drive them much higher. Even if Set $\mathbf{B}$ contains devices in series (stacks of low $V_{\rm T}$ devices), the leakage current remains higher than a current path through a high $V_{\rm T}$ device for 0.13-μm CMOS. The higher leakage current lets a few sneak paths erode the savings achieved by cutting off many other paths, so sneak leakage paths are a prohibitive problem that must be prevented to make local sleep devices feasible.

It may seem unlikely that sneak paths would exist in a carefully designed circuit, but the term *sneak leakage* implies that they can be quite subtle. For example, Fig. 6 shows how sneak leakage paths can exist through many circuit blocks and over multiple layers of hierarchy. In the figure, MTCMOS logic on the critical path is XORed with high $V_{\rm T}$ logic off of the critical path. Since the circuits fall in three blocks, the designer might tie them together at the top level without considering the circuit-level problem. The example also shows that a sneak leakage path may exist in a data-dependent fashion, so simulation will not reveal the presence of the sneak path if the appropriate data vectors are never used. Fig. 7 shows the equivalent circuit for Path 1 from Fig. 6 for two values of A. When $A = 1$, the NMOS in I2 (I2.Mn) is off, so it limits the Path 1 current. When $A = 0$, I2.Mn is on, so the current path is gated only by parallel low $V_{\rm T}$ devices in T1.

This example illustrates an additional problem that can make sneak paths even more costly. So far, we have assumed that the node $\overline{B}$ in Fig. 6 remains at 0 during sleep mode, but the impedance of Mn1 actually causes the voltage at $\overline{B}$ to rise above 0 V during sleep. The floating node voltage increases current in T1 (T1.Mn), and may even turn that device on. Fig. 8 shows the normalized current on Path 1 from a simulation of the circuit in Fig. 6. The first column shows the case where $A = 1$, so the sneak path is eliminated. The sneak path appears in the second column as $\sim$10$\times$ higher when $\overline{B}$ is forced to 0 V for illustration. The third column shows the current for the unmodified circuit with $A = 0$. In this extreme example, $\overline{B}$ settles to near $V_{\rm DD}$ and actually turns T1 on, increasing current by five orders of magnitude over the case where $A = 1$.

The basic structure of MTCMOS circuits suggests that sneak leakage paths can occur only where the sleep device(s) can be bypassed, at the interface between MTCMOS and CMOS type circuits [8]. A conservative approach to ensure that Set $\mathbf{B}$ always contains a high $V_{\rm T}$ device could use both polarities of the sleep devices for each MTCMOS gate without sharing any sleep devices. This approach incurs a large area penalty over the optimum sleep device size by using many unnecessary sleep devices. Proper design techniques can permit designers to approach the optimum sleep device size without allowing sneak leakage even in circuits using transmission gates for speed.

Fig. 9 provides four design rules for building MTCMOS circuits that prevent sneak leakage paths [13]. While these rules are not comprehensive, they prevent the most common occurrences of sneak leakage paths. The circuits in the figure are simplified to illustrate each rule clearly, but the text describes examples of real cases. Rule 1 states that a shared output (through low $V_{\rm T}$ transmission gate) between a high $V_{\rm T}$ gate and an MTCMOS gate can be prevented by using both polarity sleep devices. This type of sneak leakage path might occur at the input of a flip-flop that has a high $V_{\rm T}$ feedback inverter. It also appears when MTCMOS logic on the critical path interfaces with high $V_{\rm T}$ logic off of the critical path through a transmission gate multiplexer. Rule 2 requires MTCMOS gates with shared outputs to have the same polarity sleep device(s). This type of sneak path can appear when a design is optimized for minimum leakage using both polarity sleep devices. For example, if a known input is applied to a circuit during sleep mode, then the outputs at every node are determined prior to asserting sleep. Local sleep devices can be selectively placed to force stacks of off devices at each logic stage for extra leakage reduction. Such a design approach could create the leakage path in Rule 2. Rule 3 states that a gate with a shared sleep device must have the same polarity sleep device(s) as the other gate. As previously mentioned, local sleep devices for gates with

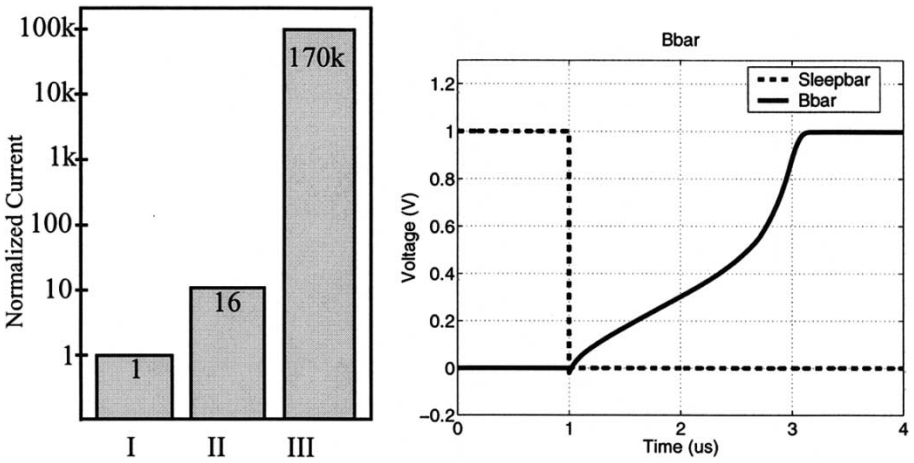

Fig. 8. Simulated current for Path 1 in Fig. 6, normalized. (I) $A = 1$. (II) $A = 0$, $\overline{B}$ forced to 0. (III) $A = 0$, $\overline{B}$ allowed to settle to its steady state value ($\sim V_{\rm DD}$). In this extreme case, floating $\overline{B}$ turns on the transmission gate. Right: simulation of floating node $\overline{B}$.

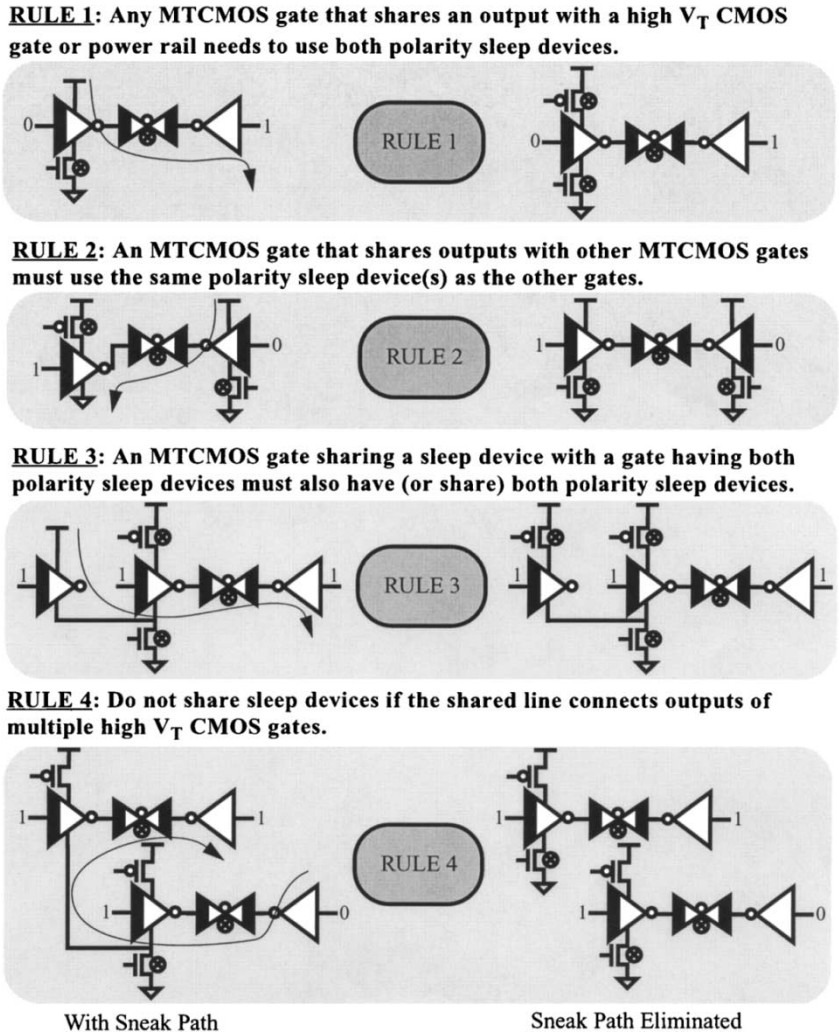

Fig. 9. Sneak leakage prevention rules illustrated with simplified examples. Arrows show the prevented leakage paths.

mutually exclusive discharge patterns can be shared to reduce area. If sleep device widths are optimized in this way and then Rule 1 violations are fixed, this type of sneak leakage path can occur. Rule 4 illustrates a sneak path for a circuit that complies with Rules 1–3. In this case, a sleep device is shared between two MTCMOS gates that in turn share outputs with high $V_{\rm T}$ gates. This path could arise if a designer tries to reduce area by sharing sleep devices for the input buffers to several flip-flops.

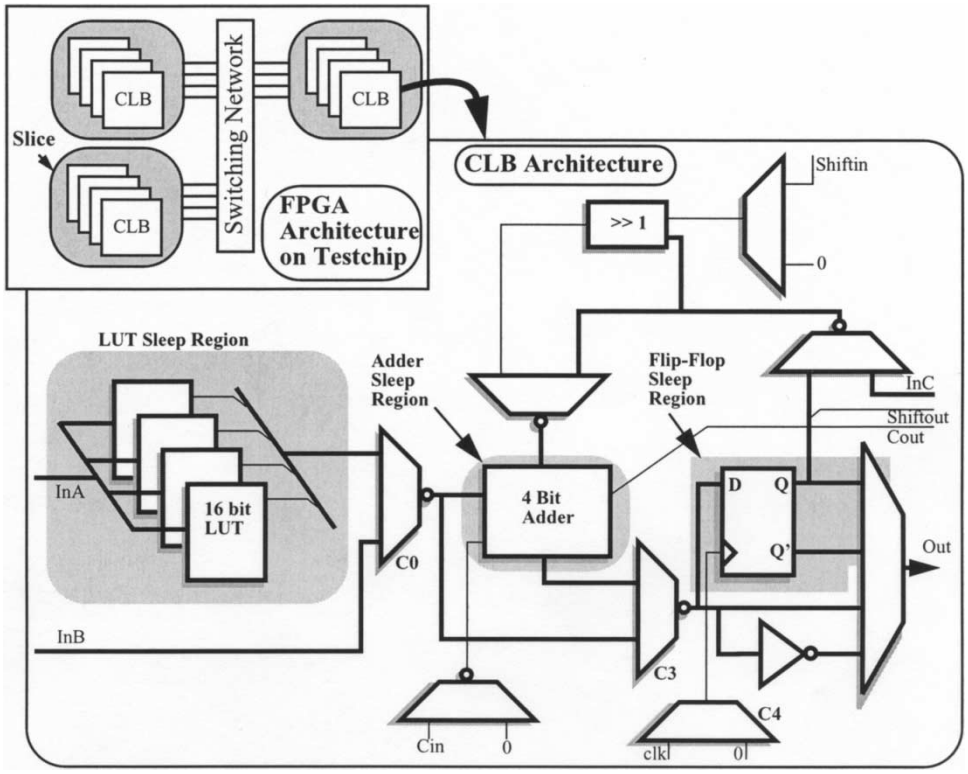

Fig. 10. FPGA test chip architecture and CLB architecture with local sleep regions (shaded).

Sleep devices should not be shared when they connect multiple high V_{T} outputs.

IV. Test Chip

The analysis of sneak leakage presented above facilitated design of a 0.13-μm, dual V_{T} test chip using sleep devices at the gate level. The two threshold voltages are about 100 mV apart for both NMOS and PMOS. The chip uses local sleep devices sized so that each local block sees less than 10% delay penalty for $V_{\mathrm{DD}} = 1.0$ V. The area penalty for the sleep devices on the test chip is less than 5% of total area. The test chip implements a low-power field programmable gate array (FPGA) architecture with 12 configurable logic blocks (CLBs) in three slices (Fig. 10). The chip has 31 K transistors in the FPGA and covers 300 μm by 740 μm. Its idle power is 29 μW and sleep power is 3.6 μW at 1.2 V.

In accordance with good dual V_{T} design, the memories that hold look-up table (LUT) values and configuration bits use high V_{T} devices, and the CLBs use MTCMOS circuits for the critical path. Fig. 10 shows that each CLB in the architecture has three primary components: four 16-bit LUTs, a 4-bit adder, and a 4-bit register. The register consists of LFBFFs that hold their state during sleep mode.

Using the sneak leakage prevention rules simplified the design process for the test chip by focusing the designers' attention on problematic interfaces. The abundance of transmission gate multiplexers along with the dual V_{T} design made sneak leakage paths a strong possibility. For example, we describe two specific sneak leakage paths that were eliminated from the CLB by applying the design rules. In the first case, a transmission gate multiplexer connecting the outputs of an MTCMOS gate on the critical path and a CMOS inverter off of the critical path created a data-dependent leakage path. Applying Rule 1 prevents this path. Second, the input driver to a flip-flop originally shared a sleep device with an inverter having a mutually exclusive discharge pattern. The shared node created a leakage path from the power rail of the adjacent circuit into the flip-flop. This leakage path occurred over several hierarchical levels, and it was eliminated using Rule 4.

V. Sleep Regions

Some of the configurations available for this CLB do not use all of its major components. Fig. 10 shows with shaded regions that the CLB is divided into three sleep regions encompassing the LUT, the adder, and the flip-flops. Partitioning a design with gate-level sleep transistors into sleep regions requires that the correct local sleep signals be routed only to the proper FETs. This local division of the circuit using local sleep devices allows any unused regions to enter sleep mode while other components remain active. This technique fits very well with FPGAs, but it is also extensible to any circuit containing local blocks that may become idle.

On the test chip, the configuration bits that program the FPGA also act as control signals for the sleep regions. For example, the configuration bit that bypasses the adder (C3 in Fig. 10) also asserts the sleep signal to the adder sleep region, so minimal control logic is required. All sleep regions respond in unison to the global sleep signal along with the multiplexers and gates in the rest of the CLB. The minimum width sleep signals were routed over existing blocks in our design, creating no extra area overhead. The sleep control circuits comprise only 1% area overhead for the CLB, while the total area overhead including all of the

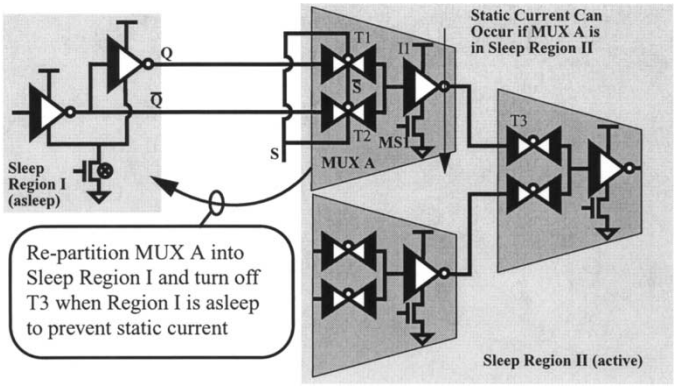

Fig. 11. Sleep region interface: partition to avoid static currents.

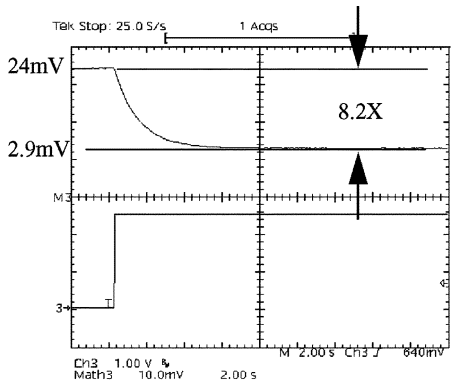

Fig. 12. Oscilloscope plot of voltage drop over a current-sense resistor when entering sleep. Initial and final voltage values (adjusted for probe offset of 10 mV) show an 8.2× savings in current drawn by the test chip. The test setup introduces a long time constant.

sleep devices and sleep control was less than 5% chip-wide. As mentioned previously, more random logic could require more complicated overhead. However, the existing enable signals or clock-gating signals could double as sleep signals in the same way that configuration bits do in the test chip.

As with sneak leakage analysis, the interfaces of the sleep regions require special attention. Specifically, a floating output from a sleeping circuit may drive the input of an active gate, thereby causing a static current path from power to ground. The FPGA architecture inherently avoids interfacing problems for sleep regions by using transmission gate multiplexers and thoughtful partitioning without requiring special circuits to latch the output of the sleeping region. Fig. 11 shows how the interface works for the flip-flop sleep region. Let us first assume that Mux A is included inside Region II, and Region I (flip-flop region) is asleep. The figure shows that either Qor $\overline{Q}$ will always drive the input to an active inverter, I1, because either T1 or T2 is on. If node Q or $\overline{Q}$ either floats or charges up to $V_{\rm DD}$ slowly, a large static current flows in I1 since MS1 is still on. Moving Mux A from Region II into Region I and turning off T3 when Region I is asleep ensures that the output of Mux A drives an off transmission gate and eliminates the static current problem.

The FPGA architecture is especially amenable to local sleep regions since the configuration bits only change during device programming. Thus, the unused components in a CLB never need to switch on during active operation. Local sleep regions also can work in architectures that require local blocks to transition into and out of sleep rapidly. Simulations show that the voltages at every node inside the CLB sleep regions settle to the correct, stable values within 4 ns during wake-up. This settling time is reasonable for one or two cycles in a low-power system. Thus, the use of an enable signal to turn sleep regions on and off at a local level seems feasible for generic architectures. Additionally, fine-grained sleep regions can be used in conjunction with other MTCMOS techniques for decreasing the wake-up time [14].

VI. Measured Results

The test chip confirms that gate-level sleep devices can provide standby leakage savings. Placing the entire chip in sleep mode provides a measured reduction in leakage current by from 7.0× to 8.6×. These savings coincide with the expected order of magnitude based on the 100-mV $V_{\rm T}$ difference between the threshold voltages. The range in savings comes from different stored states during sleep. Fig. 12 presents an oscilloscope waveform that shows the voltage drop over a current-sense resistor when the entire chip enters sleep mode. The waveform shows an 8.2× reduction corresponding to current savings. The measurement setup introduces a long time constant that dominates the chip response time. The measurements show chip-wide leakage savings during sleep mode that would not be possible if even a handful of sneak leakage currents exist in the circuit. In fact, sneak leakage paths in this architecture would cause dramatic increases in the total current during sleep mode because of floating nodes, described previously. The results show that the test chip design successfully avoids sneak leakage currents and gives standby leakage savings.

TABLE II
Simulated and Measured Sleep Region Leakage Savings for an Active CLB

Sleep Region in sleep	Local Supply Savings (Simulated / **Measured**)	Total CLB Savings (Simulated / **Measured**)
Flip-Flop	6.67X / **6.0X**	1.15X / **1.11X**
Adder	1.08X / **1.11X**	1.07X / **1.10X**
LUT	2.26X / **2.51X**	1.97X / **2.19X**

A. Sleep Region Leakage Savings

The CLBs use three power supplies to facilitate measurement: flip-flop power supply, sleep power supply, and core supply. The sleep power supplies the sleep signal network, the FF supply powers the flip-flops, and the core supply powers the rest. All HSPICE simulations and measurements are for $V_{\rm DD}$ at 1.0 V.

Table II shows the simulated and measured leakage current reduction using sleep regions. The first column shows the factor reduction in current for the power supply that should change for the given sleep region (core supply for adder and LUT; FF supply for flip-flop). In all cases, the high $V_{\rm T}$ sleep signal network consumed negligible power relative to the rest of the circuits. Since the flip-flop sleep region has a separate power supply, we can measure exactly how the FF region reduces leakage during standby mode while retaining state. The measured 6.0× reduction in leakage is less than the 8× achieved

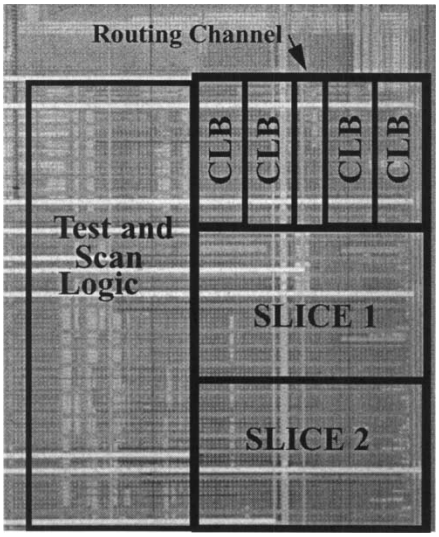

Fig. 13. Annotated die photo.

by the whole chip largely because of the known sneak leakage path through the multiplexer at the sleep region interface. As previously mentioned, the total savings for the sleep region are reduced by this path, but they are still significant enough to justify placing the sleep region into sleep. The second column shows the factor reduction in steady-state current for the entire CLB when each sleep region is asleep. The LUT region provides the greatest savings since that circuitry comprises roughly 60% of the CLB area and uses large drivers for speed. The flip-flop region and adder region comprise 10% and 6% of the CLB area, respectively. As previously mentioned, the total chip-wide area penalty of the sleep devices is under 5%. The results show that the sleep regions give leakage savings of from 10% to 2.2× for an active CLB in different configurations.

VII. Conclusion

We have proposed a design approach using local sleep devices and provided an analysis of sneak leakage paths to facilitate the approach. We presented a set of design guidelines for preventing common sneak leakage paths. A 0.13-μm MTCMOS test chip confirms that gate-level sleep devices can prevent sneak leakage and provide total standby leakage savings measured from 7.0× to 8.6× for different stored states. Fig. 13 shows the annotated die photo of the test chip. The three slices (containing 12 CLBs) cover 300 μm by 740 μm.

We also propose fine-grained sleep regions based on local sleep devices and implement them on the test chip. The sleep regions allow idle blocks to enter sleep mode while other circuit components remain active and with unaffected performance. The total steady-state power (clock-gated) for an active CLB reduces by from 10% to 2.2× for different configurations as shown in Table II.

Acknowledgment

The authors would like to thank Cypress Semiconductor for chip fabrication.

References

[1] S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, vol. 19, pp. 23–29, July–Aug. 1999.
[2] A. Keshavarzi *et al.*, "Intrinsic leakage in low power deep submicron CMOS ICs," in *Proc. Int. Test Conf.*, 1997, p. 146.
[3] S. Mutoh *et al.*, "1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *J. Solid-State Circuits*, vol. 30, pp. 847–854, Aug. 1995.
[4] T. Inukai *et al.*, "Boosted Gate MOS (BGMOS): device/circuit cooperation scheme to achieve leakage-free giga-scale integration," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2000, pp. 409–412.
[5] H. Kawaguchi *et al.*, "A CMOS scheme for 0.5 V supply voltage with pico-ampere standby current," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 1998, pp. 192–193, 436.
[6] S. Shigematsu *et al.*, "A 1-V high-speed MTCMOS circuit scheme for power-down applications," in *Symp. VLSI Circuits Dig. Tech. Papers*, 1995, pp. 125–126.
[7] P. van der Meer *et al.*, "Ultra-low standby-currents for deep sub-micron VLSI CMOS circuits: smart series switch," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS)*, vol. 4, 2000, pp. 1–4.
[8] J. Kao and A. Chandrakasan, "MTCMOS sequential circuits," in *Proc. ESSCIRC*, 2001, pp. 332–339.
[9] K. Usami *et al.*, "Automated selective multi-threshold design for ultra-low standby applications," in *Proc. Int. Symp. Low-Power Electronics and Design (ISLPED)*, 2002, pp. 202–206.
[10] C. Bittlestone *et al.*, "Architecting ASIC libraries and flows in nanometer era," in *Proc. Design Automation Conf.*, 2003, pp. 776–781.
[11] J. Kao *et al.*, "MTCMOS hierarchical sizing based on mutual exclusive discharge patterns," in *Proc. Design Automation Conf.*, 1998, pp. 495–500.
[12] M. Stan, "Low threshold CMOS circuits with low standby current," in *Proc. Int. Symp. Low-Power Electronics and Design (ISLPED)*, 1998, pp. 97–99.
[13] B. H. Calhoun, F. Honore, and A. Chandrakasan, "Design methodology for fine-grained leakage control in MTCMOS," in *Proc. Int. Symp. Low-Power Electronics and Design (ISLPED)*, 2003, pp. 104–109.
[14] K.-S. Min *et al.*, "Zigzag super cut-off CMOS (ZSCCMOS) block activation with self-adaptive voltage level controller: an alternative to clock-gating scheme in leakage dominant era," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2003, pp. 400–502.

Benton H. Calhoun received the B.S. degree in electrical engineering with a concentration in computer science from the University of Virginia, Charlottesville, in 2000. He received the M.S. degree from the Massachusetts Institute of Technology (MIT), Cambridge, in 2002, where he is currently working toward the Ph.D. degree.

His research interests include leakage reduction, sensor networks, and energy-efficient circuits.

Frank A. Honoré received the S.B. and S.M. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, in 1991 and 1994, respectively. Since fall 2000, he has been pursuing the Ph.D. degree at MIT.

He has held summer internships with Hewlett-Packard Stanford Park Division working on circuits for multipath fading and high-precision frequency synthesis test equipment. From 1997 to 2000, and during summer 2002, he was with Texas Instruments Incorporated, Dallas, TX, as a Member of Technical Staff in the DSP Solutions Research and Development group. He has contributed to the development of high-speed switch fabrics, multichannel ADSL systems, 3G CDMA base station architectures, and MIMO antenna digital baseband systems. His research interests include energy-aware reconfigurable circuits and CAD for signal processing applications. He holds two U.S. patents and has several pending. He has authored or coauthored papers on scheduled communication in high-performance systems and implementation of wide-band CDMA, and a chapter in *The Application of Programmable DSPs in Mobile Communications* (Chichester, U.K.: Wiley, 2002).

Anantha P. Chandrakasan (M'95–SM'01–F'04) received the B.S, M.S., and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 1989, 1990, and 1994, respectively.

Since September 1994, he has been with the Massachusetts Institute of Technology, Cambridge, where he is currently a Professor of electrical engineering and computer science. His research interests include low-power digital integrated circuit design, distributed wireless microsensors, ultra wideband radios, and emerging technologies. He is a coauthor of *Low Power Digital CMOS Design* (Norwell, MA: Kluwer, 1995) and *Digital Integrated Circuits* (Upper Saddle River, NJ: Pearson Prentice Hall, 2002, 2nd ed.). He is also a coeditor of *Low Power CMOS Design* (Piscataway, NJ: IEEE Press, 1997) and *Design of High-Performance Microprocessor Circuits* (Piscataway, NJ: IEEE Press, 2000).

Dr. Chandrakasan has received several Best Paper Awards, including the 1993 IEEE Communications Society's Best Tutorial Paper Award, the IEEE Electron Devices Society's 1997 Paul Rappaport Award for the Best Paper in an EDS publication during 1997, and the 1999 Design Automation Conference Design Contest Award. He has served as a technical program co-chair for the 1997 International Symposium on Low-Power Electronics and Design (ISLPED), VLSI Design '98, and the 1998 IEEE Workshop on Signal Processing Systems. He was the Signal Processing Subcommittee Chair for ISSCC 1999–2001, the Program Vice-Chair for ISSCC 2002, the Program Chair for ISSCC 2003, and the Technology Directions Chair for ISSCC 2004. He was an Associate Editor for the IEEE JOURNAL OF SOLID-STATE CIRCUITS from 1998 to 2001. He serves on the SSCS AdCom. He is the Technology Directions Chair for ISSCC 2005.