Design Methodology for Fine-Grained Leakage Control in MTCMOS

Benton H. Calhoun, Frank A. Honore, and Anantha Chandrakasan
Massachusetts Institute of Technology, Cambridge, MA, 02139
{bcalhoun,honore,anantha}@mtl.mit.edu

Abstract

Multi-threshold CMOS is a popular technique for reducing standby leakage power with low delay overhead. MTCMOS designs typically use large sleep devices to reduce standby leakage at the block level. We provide a formal examination of sneak leakage paths and a design methodology that enables gate-level insertion of sleep devices for sequential and combinational circuits. A fabricated 0.13μm, dual V_T testchip employs this methodology to implement a low-power FPGA core with gate-level sleep FETs and over 8X measured standby current reduction. The methodology allows local sleep regions that reduce leakage in active CLBs by up to 2.2X (measured) for some CLB configurations.

Categories and Subject Descriptors

B.7.1 Integrated Circuits - *types and design styles*

General Terms

Performance, Design, Reliability

Keywords

MTCMOS, low power, circuit design, leakage, fine-grain sleep regions, sleep mode, sneak leakage, design methodology

1. Introduction

Circuit designers have recognized MTCMOS as a valuable design approach for limiting standby leakage. The majority of MTCMOS designs gate the power to sizable blocks of logic using large sleep transistors. The use of sleep devices at the gate level has remained largely unexplored even though it could potentially have several advantages that we will describe. Locally placed sleep devices are only feasible if sneak leakage currents are prevented. This paper presents two contributions to leakage reduction. First, a formal definition of sneak leakage paths provides designers with insight required for gate-level insertion of sleep devices. A set of four design rules offers a simplified approach to preventing sneak leakage that covers most cases. Secondly, the use of local sleep regions reduces leakage in unused circuit components at a local level even while the surrounding circuit remains active. We describe the implementation and benefits of local sleep regions in our design, and we examine the interfacing issues necessary for this technique to work.

ISLPED'03, August 25-27, 2003, Seoul, Korea.

Process scaling to deep submicron levels catapulted the issue of device leakage to the front tier of problems for circuit designers. Since switching power is proportional to V_{DD}^2 new processes are tailored for lower supply voltages, and the corresponding drop in device threshold voltage, V_T, maintains performance but causes exponential increase of subthreshold leakage. Other components of leakage including gate leakage, GIDL, and reverse biased diode leakage have also increased[1]. At the 0.13μm node, however, subthreshold leakage dominates the other components. If unaddressed, leakage power will continue to increase in both active and standby modes [2].

Some subthreshold leakage reduction techniques take advantage of the *stack effect* caused by stacking transistors in series. The stack effect causes V_{GS} of the upper device to become negative and V_T to increase due to the body effect. Stacks of transistors occur naturally in many CMOS gates, and algorithms also exist for introducing them artificially [3][4]. Other algorithms assign input vectors for standby mode that use stack effect to reduce leakage current [5][6].

Another set of techniques uses body bias to tune device V_T for performance or reduced leakage[7][8]. This approach allows active leakage reduction when timing slack exists[9]. Body biasing techniques generally require a triple well process for isolating body contacts.

Dual-threshold CMOS techniques use all low V_T FETs on the critical path and high V_T FETs off of the critical path. The placement of these devices can become complicated since the high V_T devices can slow down parts of the circuit enough to create new critical paths. Nevertheless, several algorithms exist for that purpose (e.g. [10]). One advantage of dual-threshold CMOS is that the leakage savings occur in both active and standby modes. Disadvantages include more complicated design time and the inability to stop critical path leakage during standby mode.

Multi-Threshold CMOS (MTCMOS) refers to a circuit technique that uses a high V_T footer and/or header FET to sever a circuit from the power rails during standby[11]. Similar approaches have been applied to caches[12][13] and to DRAMs[14]. The Boosted Gate MOS (BGMOS) approach [15] and Super Cut-off CMOS (SCCMOS)[16] both enhance performance by respectively overdriving the sleep device in active mode and underdriving in standby mode.

Sequential MTCMOS circuits require more thought since they must hold state in standby mode. For example, high V_T devices placed in parallel with low V_T devices can hold the state of storage nodes during sleep[11]. A high V_T "balloon" circuit consisting of back to back inverters can hold state at a given node[17]. Another approach uses the input of a flip-flop to control the sleep state of the device conditionally[18]. This paper uses the Leakage Feedback Flip-flop (LFBFF) proposed in [19]. The LFBFF uses feedback from the output to cut-off one power rail conditionally during standby.

2. Trade-offs of Local vs. Global Sleep Devices

Most MTCMOS designs insert high V_T sleep devices that cut-off the leakage current to large blocks of logic. The use of sleep devices at a local, gate level remains largely unexamined. One reason for choosing large sleep FETs is the design complexity associated with placing them locally. Segmenting the sleep transistor into many devices at the gate level creates a greater possibility for sneak leakage paths. Generally, sneak leakage paths are high leakage paths between power and ground that remain during sleep mode. Analysis of sneak leakage paths at the local level can provide insight that eliminates much of this complexity. Dispersing the sleep devices among the gates of a design will also limit sharing sleep devices. The total sleep device width for the distributed approach probably will be higher than for the single large FET. On the other hand, locally placed sleep devices offer several advantages over the large sleep FET approach.

First, inserting local sleep devices makes it easier to guarantee circuit functionality at high speed. This guarantee is possible because determining the worst case speed for a sleep device placed at the gate level is very feasible. Exhaustive gate-level simulations can easily show that a gate will always meet a given timing specification. The same guarantee becomes virtually impossible to offer for large blocks with only one sleep FET. For a large block, the worst case speed occurs when the worst case current flows through the sleep device. Identifying this worst case can be quite difficult without comprehensive simulation [20]. Sizing the device based on average current draw or even on the worst case current from a subset of possible input vectors does not guarantee that the absolute worst case is met.

Gate-level sleep devices also could enhance design speed and even reduce design time by increasing the flexibility of the design. For example, MTCMOS flip-flops that require local sleep FETs outperform designs that can share a large, block-level sleep device [19]. Including local sleep devices in standard cells could eliminate the need for extensive, full-chip simulations for every new design. Instead, a place-and-route approach with these standard cells would provide short design time with all the benefits of leakage reduction from a custom design. Gate-level sleep devices also improve circuit noise margins [21].

3. Sneak Leakage Prevention

In general, a *sneak leakage path* is any current path from V_{DD} to ground that continues to draw high current relative to a cut-off path during sleep mode. Formally, a sneak leakage path is a current path that flows from V_{DD} to ground through a set of "on" devices, A, and through a set of "off" devices, B. Set B contains only low V_T devices, while Set A may contain either low or high V_T devices. Since the off devices are low V_T, the sneak leakage current is at least an order of magnitude higher than other currents in the circuit. Even if Set B contains devices in series (stacks of low V_T devices), the leakage current remains higher than a current path through a high V_T device for the 0.13 μ m technology node.

Ignoring even a small number of sneak leakage paths carries a significant cost. Since sneak leakage currents tend to be about an order of magnitude higher than other current paths, a few sneak paths can dominate the savings achieved by cutting off many other paths. It may seem unlikely that such paths would exist in a carefully designed circuit, but the term sneak leakage implies that these current paths can be quite subtle. For example, Figure 1 shows how sneak leakage paths can exist through many circuit blocks and over several layers of hierarchy. Such a distributed path is difficult to discover by inspection. In the figure, MTCMOS logic on the critical path is XORed with high V_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. In this example, Path B is only gated by parallel low V_T FETs, and Path A is gated by a stack of low V_T devices. The example also shows that a sneak leakage path may exist in a data dependent fashion. In other words, simulation may not reveal the presence of the sneak path if the appropriate data vectors are never used. If the high V_T CMOS inverter in Figure 1 had a 0 at its input, neither leakage path would appear in simulation. The subtle nature of sneak leakage paths makes a careful analysis of their causes very important.

An MTCMOS circuit uses a high V_T sleep device between a low V_T circuit and one rail, usually the ground rail. Whether the sleep device is one large device or many small ones, the basic structure of MTCMOS circuits suggests that sneak leakage paths must occur only where the sleep device(s) can be bypassed. Thus, the focal point for preventing sneak leakage paths will be at the interface between MTCMOS and CMOS circuits. Previous analysis of MTCMOS sequential circuits makes this same observation [19]. Their treatment of leakage paths in MTCMOS flip-flops highlights several specific configurations that lead to sneak leakage paths, and they propose flip-flop designs to eliminate those paths. This work expands on the analysis of sneak leakage at MTCMOS interfaces, and we propose general design guidelines that prevent most instances of sneak leakage.

An ultra-conservative approach to sneak leakage prevention could use both polarities of sleep device for each MTCMOS gate without sharing any sleep devices. At the cost of area, this type of design would prevent all sneak paths by ensuring that Set B always contains a high V_T device. In practice, most MTCMOS gates only require one polarity of sleep device.

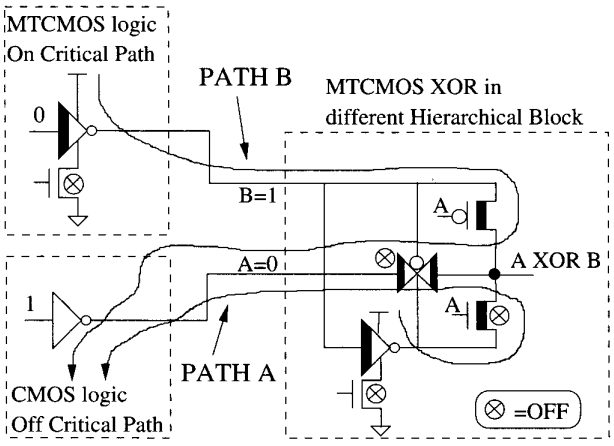

Figure 1. Example of Sneak Leakage Paths at CMOS, MTCMOS interface. Paths occur over several hierarchical levels and through several blocks. Paths are also data-dependent (In this example, both paths disappear for A=B=1).

Even in a gate-level sleep device regime, local clusters of gates can still share sleep devices for reduced area and improved leakage reduction. Since low impedence paths can exist from MTCMOS outputs to power rails in this type of design, sneak leakage paths are possible whenever an MTCMOS output node is connected electrically to another node with low impedence to a power rail. This electrical connection most likely will not occur directly, so it most often appears as a connection through low V_T passgates or transmission gates.

An awareness of the sources of sneak leakage paths can allow designers to prevent them at the gate level with the minimum necessary area overhead. Table 1 provides four guidelines for designing MTCMOS circuits that prevent sneak leakage paths. While these rules may not cover every case, they prevent the most common occurrences of sneak leakage paths. These guidelines provide useful rules of thumb for designers, and they are not intended to be comprehensive. A CAD tool for identifying sneak leakage paths would include more rules to account for all cases. In accordance with the sneak leakage definition we described before, shared outputs between CMOS and MTCMOS gates include outputs shared through low V_T transmission gates.

Table 1. Design Rules

1.	Any MTCMOS gate that shares an output with a CMOS gate or power rail needs to use both polarity sleep devices.
2.	An MTCMOS gate that shares outputs with other MTCMOS gates must use the same polarity sleep device (or devices) as the other gates.
3.	Any MTCMOS gate that shares a sleep device with a gate that uses both polarity sleep devices must also have (or share) both polarity sleep devices.
4.	Do not share sleep devices if the shared line creates a connection between outputs of multiple CMOS gates.

Figure 2 shows examples of the four design rules. The arrow in each case shows the leakage path being prevented. The examples indicate how most sneak leakage paths occur through low V_T transmission gates since that is the most common configuration where MTCMOS gates and CMOS gates share output nodes. Collectively, the rules show that sneak leakage paths occur at the interface of MTCMOS gates.

In practice, these rules simplified the design process for the MTCMOS testchip by focusing the designers' attention on problematic interfaces. As the next section will show, the testchip architecture uses transmission gates throughout the design to improve speed on the critical path. The abundance of transmission gates along with the dual V_T design made the rules in Table 1 invaluable for eliminating potential leakage paths from the design. Figure 3 shows two examples of sneak leakage paths in the testchip circuit that were prevented by applying the design rules. In the first case, a transmission gate multiplexor connects the outputs of an MTCMOS gate and a CMOS inverter to create a data dependent leakage path. Applying RULE 1 fixes this problem. In the second case, a shared sleep device allows leakage paths to flow into a flip-flop. This leakage path occurred over several hierarchical levels. The figure shows two options for removing this problem according to the rules. We applied RULE 4 to eliminate this potential leakage path.

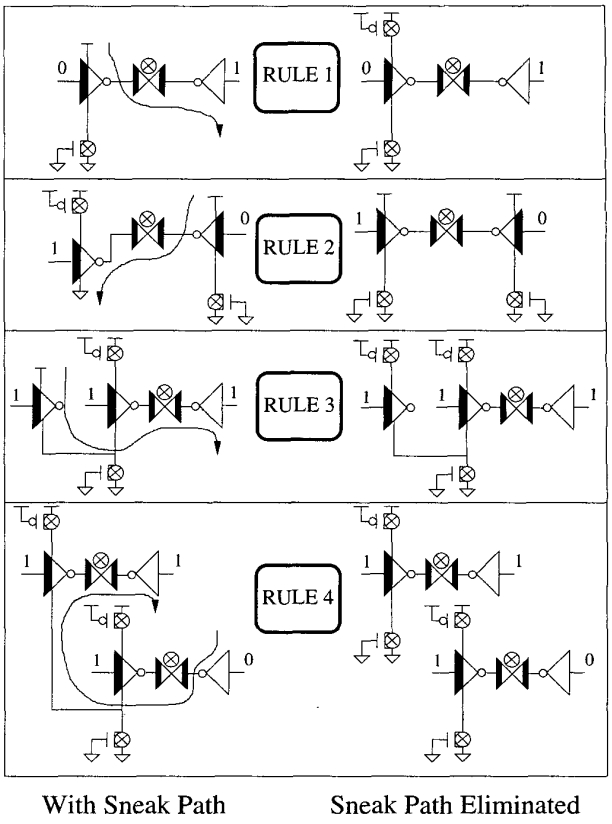

Figure 2. Illustration of Design Rules. Arrows show the prevented leakage paths.

4. Sleep Regions

The analysis of sneak leakage presented above facilitated design of a 0.13μm, dual V_T testchip using sleep devices at the gate level. The two threshold voltages are about 100mV apart. The sleep devices are sized so that each gate sees less than 10% delay penalty, and the entire circuit thus has less than a 10% delay penalty for V_{DD} = 1.0V. The total area penalty of the sleep devices on the testchip is less than 5%. The testchip implements a low power FPGA architecture with 12 Configurable Logic Blocks (CLBs) in 3 slices. The FPGA core houses circuits common to more generic architectures.

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. 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.

Some of the configurations available for this CLB do not use all of these components. Figure 4 shows with dashed boxes that the CLB is divided into four sleep regions. Three of the sleep regions each contain one of the main elements in the CLB, and the fourth region holds the remaining control circuits. The sleep regions allow any unused elements of the

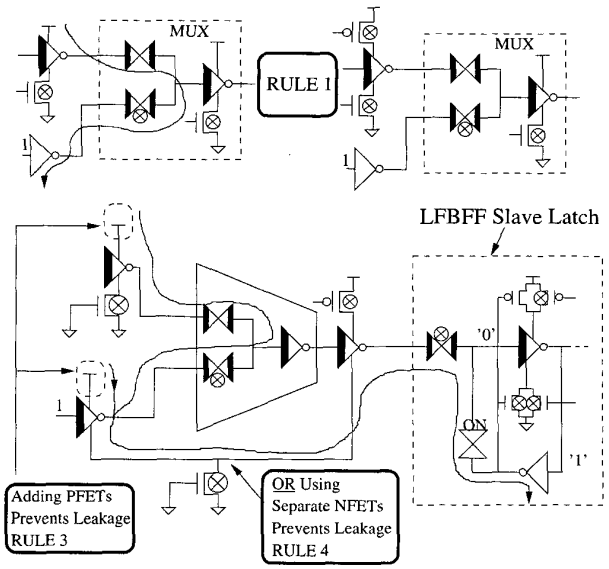

Figure 3. Leakage Paths Prevented on the Testchip.

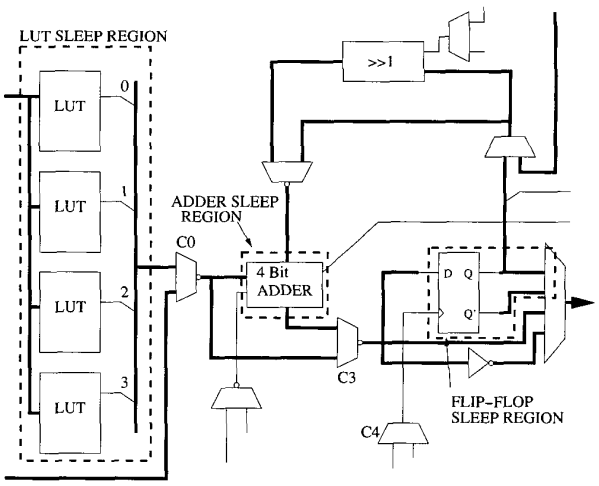

Figure 4. CLB Architecture with Local Sleep Regions (in hashed boxes).

CLB to enter sleep mode while the other components remain active. The fine-grained placement of sleep devices makes this approach possible.

Since the testchip implements programmable logic, configuration bits tell each CLB how to organize its internal parts at program time. These configuration bits can also act as control signals for the sleep regions. For example, the configuration bit that causes the adder to be bypassed (C3 in Figure 4) also asserts the sleep signal to the adder sleep region. All sleep regions respond in unison to the global sleep signal along with the multiplexors and gates in the rest of the CLB. Minimal control logic is required for deciding when to assert the sleep signal for each local sleep region. The overhead control circuits for incorporating sleep regions in our design comprised only 1% area overhead for the CLB, while the total area overhead including all of the sleep devices was less than 5%. Thus, distributing sleep devices at the gate level allows a fine-grained partitioning of the CLB with negligible overhead.

As with sneak leakage analysis, the interfaces of the sleep regions require special attention. Specifically, a block in cutoff might drive CMOS gates with a floating node, thereby opening a static current path from power to ground. Careful design of the CLB prevents such a situation from occurring. Take the LUT block as an example. The output of the 16-bit LUTs drives one input of a 2:1 multiplexor that uses transmission gates to select between its input signals. When the LUT region is asleep, its outputs drive the "off" transmission gate in the 2:1 mux. Therefore, any floating nodes inside the sleep region remain isolated from the gates of other devices. A large subthreshold leakage current may flow through the transmission gate because it is low V_T. However, this current is insignificant relative to the savings obtained by placing the entire LUT region into sleep.

The outputs of the flip-flops also drive transmission gates, but the situation is different than for the LUT sleep region. In the previous case, the transmission gate driven by the floating inputs always was off whenever the LUT region was asleep. In the case of the flip-flop sleep region, the interface requires careful partitioning to prevent static current. Let us first assume that the multiplexor at the output of the FF sleep region is not included inside the FF sleep region. With this partitioning decision, Figure 5 shows that one of the flip-flop outputs, either Q or $\overline{Q}$, will always drive the input to an inverter because one of the transmission gates controlled by S is on. If node Q or $\overline{Q}$ either floats or takes a long time to charge up to V_{DD}, a large static current flows in the output inverter of *MUX A* in Figure 5 because MS1 is still on. Careful partitioning can eliminate this problem. Including *MUX A* from the 4:1 mux in the flip-flop sleep region eliminates the static current problem. This better partitioning solution ensures that the output of *MUX A* drives a transmission gate that is always off whenever the flip-flop region is asleep, thus preventing static current.

The FPGA architecture inherently avoids most interfacing problems for sleep regions by using transmission gate multiplexors. Thoughtful partitioning makes use of these muxes for interfacing without static currents. Partitioning a design with gate-level sleep transistors into sleep regions only requires that the correct local sleep signals be routed to the proper FETs. The properly interfaced sleep regions in out design provide automatic power savings for any CLB that does not use one or more of its major elements.

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 can find use in more generic architectures that require small local blocks to transition into and out of sleep with some rapidity. Simulations show that the voltages at every node inside the CLB sleep regions settle to the correct, stable values within 4ns 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 in generic architectures. Additionally, fine-

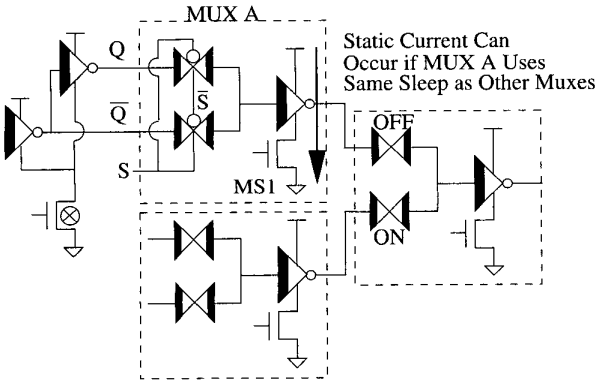

Figure 5. Sleep Region Interface: Partition to Avoid Static Currents.

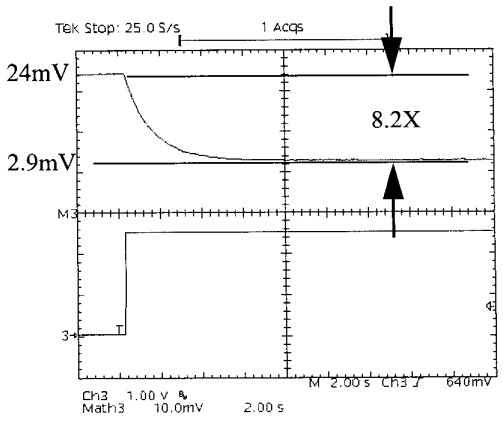

Figure 6. Oscilloscope plot of voltage drop over current-sense resistor when entering sleep. Initial and final voltage values (adjusted for probe offset of 10mV) show an 8.2X savings. The test setup introduces a long time constant.

grained sleep regions can be used in cooperation with other techniques for decreasing the wake-up time that are also based on the MTCMOS approach [22].

5. Measured Results

The testchip 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 7.0X to 8.6X. These savings coincide with expected savings based on the 100mV V_T difference between the threshold voltages in the process we used. The range in savings comes from different stored state during sleep. As previously mentioned, the LUTs use high V_T storage elements to maintain their values in sleep. Likewise, the CLB uses LFBFFs that preserve their state while reducing leakage. The state consisting of these stored values impacts the total leakage current during sleep mode. Figure 6 presents an oscilloscope waveform that shows the voltage drop over a current-sense resistor when the entire chip enters sleep mode. Accounting for a 10mV offset in the oscilloscope probes, the waveform shows an 8.2X reduction corresponding to current savings. The measurement setup (including the series current-sense resistor) introduces a long time constant that dominates the chip response, but the initial and final values match well with independent results from an ammeter. The measurements show chip-wide leakage savings during sleep mode that would not be possible should even a handful of sneak leakage currents exist in the circuit. The results imply that the testchip design successfully avoids sneak leakage currents and gives standby leakage savings.

5.1 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 takes care of the rest. All HSPICE simulations and chip measurements are for V_{DD} at 1.0V.

Table 2 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 FF). In all cases, the high V_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 6.0X reduction in leakage is less than the 8X acheived by the whole chip largely because of the known sneak leakage path through the mux 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 area penalty of the sleep devices is under 5%, and the area penalty for sleep region control logic is 1%. The results show that the sleep regions give leakage savings of from 10% to 2.2X for an active CLB in different configurations.

Table 2. 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**

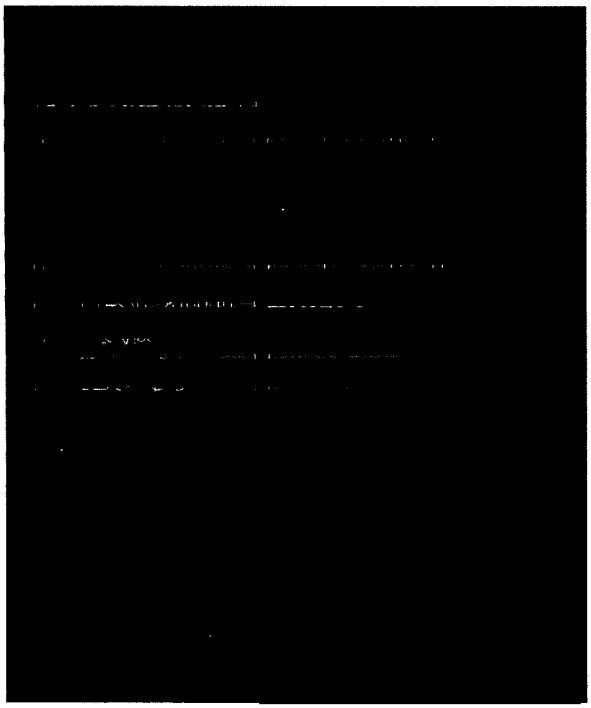

Figure 7. Annotated Die Photo.

6. Conclusions

We have proposed a design approach using gate-level sleep devices and provided a thorough analysis of sneak leakage paths to facilitate the approach. Insight that leakage paths occur at MTCMOS interfaces provides a set of design guidelines for preventing the most common sneak leakage paths. These guidelines allow designers to insert sleep devices locally without having leakage savings degraded by sneak leakage. A 0.13μm testchip 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 7.0X to 8.6X for different stored states. Figure 7 shows the annotated die photo of the testchip. The three slices (containing 12 CLBs) cover 300μm by 740μm.

We also propose fine-grained sleep regions and implement them on the testchip. The chip measurements match closely with simulation, and they show the benefit of the sleep region technique. The use of gate-level sleep devices allows inactive circuit regions to enter sleep at a fine grain. The 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 2.2X for different configurations as shown in Table 2.

7. Acknowledgements

This research is funded by Texas Instruments; it is partly sponsored by DARPA and the Air Force Research Laboratory, under agreement number F33615-02-2-4005. Ben is supported with an Infineon Fellowship. We thank Cypress Semiconductor for chip fabrication and Alan Refalo for his assistance.

8. References

[1] A. Keshavarzi et al, "Intrinsic Leakage in Low Power Deep Submicron CMOS ICs", *Int'l Test Conf.*, 1997.

[2] S. Borkar, "Design Challenges of Technology Scaling", *IEEE Micro*, vol. 19, no. 4, pp. 23-29, July-Aug. 1999.

[3] M. Johnson et al, "Leakage Control with Efficient Use of Transistor Stacks in Single Threshold CMOS", *IEEE Trans. on VLSI Sys.*, 2002.

[4] S. Narendra et al, "Scaling of Stack Effect and its Application for Leakage Reduction", *ISLPED*, pp 195-200, Aug 2001.

[5] J. P. Halter and F. N.Najm, "A Gate-Level Leakage Power Reduction Method for Ultra-Low-Power CMOS Circuits", *CICC*, 1997.

[6] S. Naidu and E. Jacobs, "Minimizing stand-by leakage power in static CMOS circuits", *Proc. DATE, 2001*, pp. 370-376, March 2001.

[7] Xiaomei Liu and S. Mourad, "Performance of submicron CMOS devices and gates with substrate biasing", *ISCAS*, pp. 9-12, 2000.

[8] A. Keshavarzi et al, "Tech. scaling behavior of optimum reverse body bias for standby leakage power reduction in CMOS IC's", *ISLPED*, 1999.

[9] C. H. Kim and K. Roy, "Dynamic V_{TH} Scaling Scheme for Active Leakage Power Reduction", *DATE, 2002*. pp. 163 -167, 2002.

[10] V. Sundararajan and K.Parhi, "Low Power Synthesis of Dual Threshold Voltage CMOS VLSI Circuits", *ISLPED*, 1999.

[11] S. Mutoh et al, "1-V Power Supply High-Speed Digital Circuit Technology with Multithreshold-Voltage CMOS", *JSSC*, vol.30, no.8, Aug 1995.

[12] M. Powell et al, "Gated-V_{dd}: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories", *ISLPED*, 2000.

[13] M. Powell et al, "Reducing Leakage in a High-Performance Deep-Submicron Instruction Cache", *Trans. on VLSI*, v.9, no.1, Feb. 2001.

[14] T. Sakata et al, "Subthreshold-Current Reduction Circuits for Multi-Gigabit DRAM's", *JSSC*, vol. 29, no. 7, 1994.

[15] T. Inukai et al, "Boosted Gate MOS (BGMOS): Device/Circuit Cooperation Scheme to Achieve Leakage-Free Giga-Scale Integration", *CICC*, 2000.

[16] H. Kawaguchi et al, "A CMOS Scheme for 0.5V Supply Voltage with Pico-Ampere Standby Current", *ISSCC*, 1998.

[17] S. Shigematsu et al, "A 1-V High-Speed MTCMOS Circuit Scheme for Power-Down Applications", *Symp. on VLSI Circuits*, 1995.

[18] P. van der Meer et al, "Ultra-low Standby Currents for deep sub-micron VLSI CMOS Circuits: Smart Series Switch", *Intl Symp. on Circuits and Systems*, 2000.

[19] J. Kao and A. Chandrakasan, "MTCMOS Sequential Circuits", *Proc. ESSCIRC 2001*, 2001.

[20] J. Kao et al, "MTCMOS hierarchical sizing based on mutual exclusive discharge patterns", *Proc. DAC*, pp. 495 -500, 1998.

[21] M. Stan, "Low Threshold CMOS Circuits with Low Standby Current", *ISLPED*, 1998.

[22] K. 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", *ISSCC*, 2003.