Pareto Points in SRAM Design Using the Sleepy Stack Approach

Jun Cheol Park and Vincent J. Mooney III
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332
{jcpark, mooney}@ece.gatech.edu

Abstract

Leakage power consumption of current CMOS technology is already a great challenge. ITRS projects that leakage power consumption may come to dominate total chip power consumption as the technology feature size shrinks. Leakage is a serious problem particularly for SRAM which occupies large transistor count in most state-of-the-art chip designs. We propose a novel ultra-low leakage SRAM design which we call "sleepy stack SRAM." Unlike many other previous approaches, sleepy stack SRAM can retain logic state during sleep mode, which is crucial for a memory element. Compared to the best alternative we could find, a 6T SRAM cell with high-V_{th} transistors, the sleepy stack SRAM cell with 1.5xV_{th} at 110^oC achieves more than 5X leakage power reduction at a cost of 31% delay increase and 113% area increase. Alternatively, by widening `wordline` *pass transistors, the sleepy stack SRAM cell can match the delay of the high-V_{th} 6T SRAM and still achieve 2.5X leakage power reduction at a cost of a 139% area penalty.*

1 Introduction

Today, power consumption is one of the top concerns of Complementary Metal Oxide Semiconductor (CMOS) circuit design. This is not only because of the recent growing demands of mobile applications. Even before the mobile era, power consumption has been a fundamental problem. To solve the power dissipation problem, many researchers have proposed different ideas including a plethora at the device level and the architectural level. However, due to the significant trade-offs possible in power, delay and area, designers are required to choose appropriate techniques that satisfy application and product needs.

Although dynamic power is dominant for technologies at 0.18μ and above, leakage (static) power consumption starts to become a nearly equal partner for technologies below 0.18μ. One of the main contributor to leakage power consumption of a CMOS circuit is subthreshold leakage current, i.e., the source to drain current when the gate voltage is smaller than the transistor threshold voltage. Since subthreshold current increases exponentially as the threshold voltage decreases, deep sub-micron technologies with scaled down threshold voltages will severely suffer from subthreshold leakage power consumption. In addition to subthreshold leakage, another contributor to leakage power is gate-oxide leakage power due to the tunneling current through the gate-oxide insulator. Since gate-oxide thickness will be reduced as the technology decreases, in deep sub-micron technology, gate-oxide leakage power may be comparable to subthreshold leakage power if not handled properly. In this paper, we focus on subthreshold leakage power because we assume other techniques will address gate-oxide leakage; for example, high-k dielectric gate insulators may provide a solution to reduce gate-leakage [1]. Nonetheless, please note that our experimental results measure power consumption of the non-active period which includes both subthreshold and gate-oxide

leakage power.

Although leakage power consumption is a problem for all CMOS circuits, in this paper we focus on SRAM because SRAM typically occupies large area and transistor count in a System-on-a-Chip (SoC). Furthermore, considering an embedded processor example, SRAM accounts for 60% of area and 90% of the transistor count in Intel XScale [2], and thus may potentially consume large leakage power.

In this paper, we propose the sleepy stack SRAM cell design, which is a mixture of changing the circuit structure as well as using high-V_{th}. The sleepy stack technique [3] achieves ultra-large leakage power while maintaining precise logic state in sleep mode, which may be crucial for a product spending the majority of its time in stand-by mode. Based on the sleepy stack technique, the sleepy stack SRAM cell design takes advantage of ultra-low leakage and state saving.

This paper is organized as follows. In Section 2, prior work in low-leakage SRAM design is discussed. In Section 3, our sleepy stack SRAM cell design approach is proposed. In Section 4 and 5, experimental methodology and the results are presented. In Section 6, conclusions are given.

2 Previous work

Many ideas have been proposed to reduce leakage power consumption of SRAM because SRAM occupies large area and accounts for large leakage power consumption.

One way to reduce leakage power consumption is to exclusively use high-V_{th} transistors in the SRAM. This solution is simple but induces high performance degradation. Azizi et al. observe that in normal programs, most of the bits in a cache are zeros. Therefore, Azizi et al. propose an Asymmetric-Cell Cache; they use high-V_{th} in a subset of transistors in each SRAM cell to save leakage power if the SRAM cell is in the zero state [4]. Although this technique can reduce leakage power at a cost of increased delay overheads, if a benchmark uses mostly non-zero values, leakage power saving is limited.

The forced stack technique achieves leakage power reduction by forcing a stack structure [5]. This technique breaks down existing transistors into two transistor and takes an advantage of the stack effect, which reduces leakage power consumption by connecting two or more turned off transistors serially. The forced stack technique can be applied to memory elements such as a register [6] and/or an SRAM cell [7]. However, delay increase may occur due to increased resistance, and the largest leakage savings reported under specific conditions is 90% compared to conventional SRAM in 0.07μ technology [7].

Nii et al. propose Auto-Backgate-Controlled Multi-Threshold CMOS (ABC-MTCMOS) based on the conventional MTCMOS technique, which cuts off logic circuits using high-V_{th} sleep transistors [8]. By using reverse source-body bias during sleep mode, ABC-MTCMOS technique can save leakage power while retaining original state. However, the ABC-MTCMOS technique requires an additional supply voltage throughout the whole SRAM cell array. Further, large electric fields across gates may affect reliability [9].

Similar to MTCMOS, the gated-V_{dd} technique separates a logic block from V_{dd} and Gnd rails using sleep transistors [10]. The gated-V_{dd} technique achieves low-leakage power consumption mainly from the stack effect. However, unlike ABC-MTCMOS, which saves state, the conventional gated-V_{dd} technique loses state in sleep mode (i.e., when the sleep transistors are turned off). To overcome this problem, Powell et al. propose an architectural technique which attempts to put cache lines to sleep which do not currently hold valid data [10].

Although the conventional gated-V_{dd} technique loses the state data when placed in low-power mode, a carefully sized gate transistor may retain the original data. Agarwal et al. study various retaining conditions including temperature, V_{th}, and gate size, and propose Data Retention Gated-Ground Cache (DRG-Cache) [11]. However, since the virtual-Gnd node does not hold value "0" firmly, the DRG-Cache design is vulnerable, and even a small

induced charge may change the stored value [7].

Flautner et al. propose the "drowsy cache" technique that scales down the supply voltage during drowsy mode [12]. This technique can save leakage power by reducing Drain Induced Barrier Lowering (DIBL) of the short channel devices. The leakage power saving of this technique can be up to 70% [12].

3 Approach

Although previous approaches are effective in some ways, no ideal solution for reducing leakage power consumption is yet known. Therefore, designers choose techniques based upon technology, and associated tradeoffs. In Section 3.1, we introduce our recently proposed low-leakage technique named "sleepy stack." In Section 3.2, we then explain our newly proposed "sleepy stack SRAM."

3.1 Sleepy stack leakage reduction

The sleepy stack technique provides the possibility of choosing a new pareto point considering leakage power and delay. The sleepy stack technique can achieve 1000X leakage power reduction compared to the forced stack technique with some delay and area penalty while saving logic state [3].

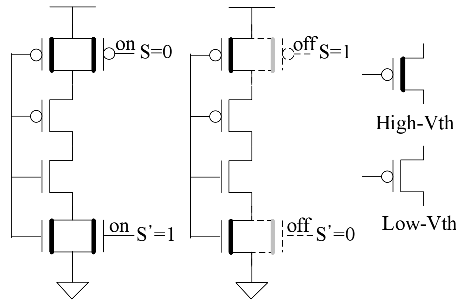

Figure 1: Sleepy stack inverter active mode (left) and sleep mode (right)

The sleepy stack technique has a combined structure of the forced stack technique and the sleep transistor technique. This combined structure may achieve smaller delay overhead than the forced stack technique while saving state (unlike other sleep transistor techniques [10, 13], which lose state when in sleep mode). The structure of the sleepy stack approach is shown in Figure 1. The sleepy stack technique divides existing transistors into two transistors each typically with the same width W_1 half the size of the original single transistor's width W_2 (i.e., $W_1 = W_2/2$). Then sleep transistors are added in parallel to one of the transistors in each set of two stacked transistors. The divided transistors reduce leakage power using the stack effect while retaining state. The added sleep transistors operate similar to the sleep transistors used in the sleep technique, in which sleep transistors are turned on during active mode and turned off during sleep mode. During active mode, S=0 and S'=1 are asserted, and thus all sleep transistors are turned on. Due to the added sleep transistor, the resistance through the activated (i.e., "on") path decreases, and the propagation delay decreases (compared to not adding sleep transistors while leaving the rest of the circuitry the same, i.e., with stacked transistors). During the sleep mode, S=1 and S'=0 are asserted, and so both of the sleep transistors are turned off. The stacked transistors in the sleepy stack approach suppress leakage current.

One huge advantage of the sleepy stack technique over the forced stack technique is that the sleepy stack technique can use high-V_{th} for both the sleep transistors as well as the transistors in parallel with the sleep transistors [3]. Figure 2 shows results from a chain of 4 inverters, which follows the experimental methodology used

in [3] while using $V_{dd} = 0.8V$ and varying V_{th}. The results are measured at 25^oC and 110^oC. The delay of the sleepy stack technique with V_{th}=0.4V and V_{th}=0.42V matches the delay of the forced stack technique with original threshold voltage (V_{th}=0.2) at 25^oC and 110^oC, respectively. The sleepy stack technique achieves 215X and 103X leakage reduction at 25^oC and 110^oC, respectively. The sleepy stack technique achieved roughly the same leakage as the sleep transistor technique but with state being saved [3].

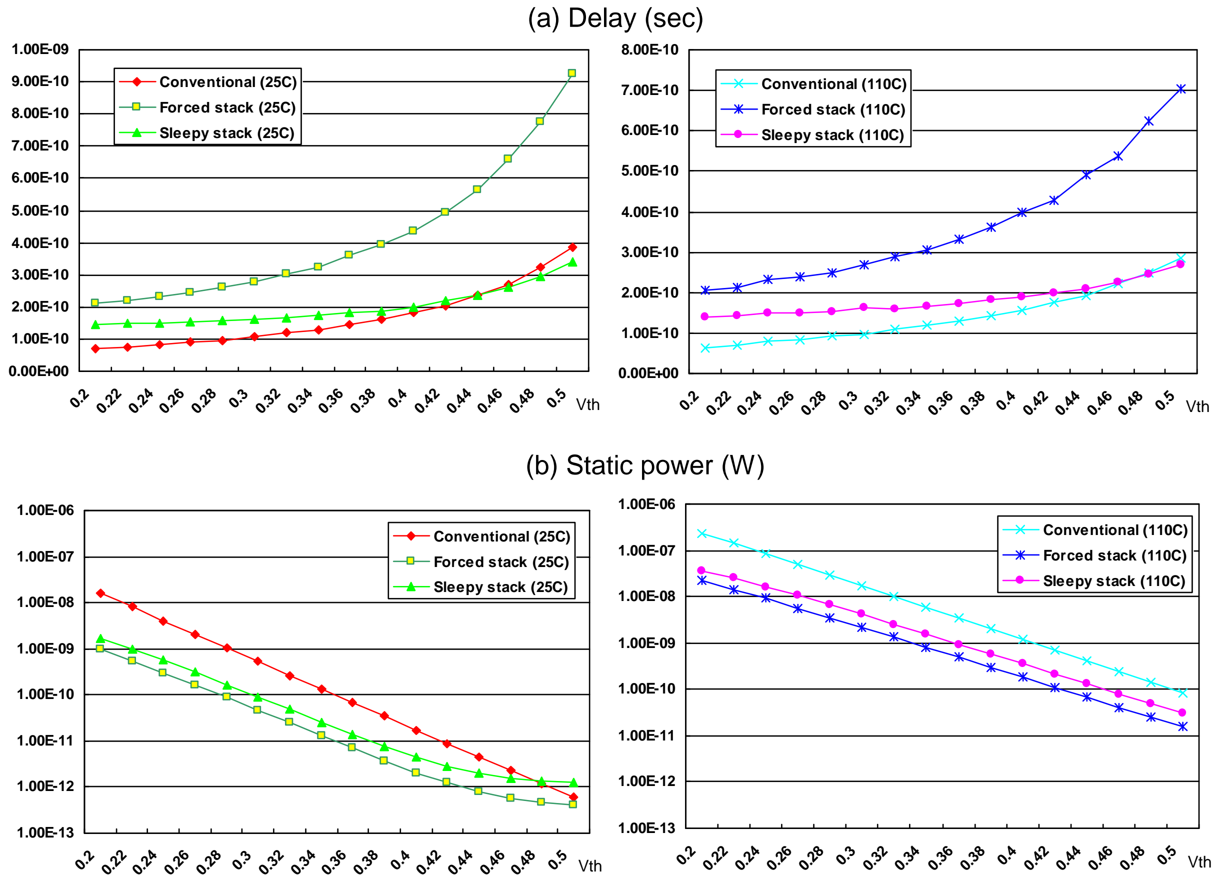

Figure 2: Results from a chain of 4 inverters while varying V_{th}

3.2 Sleepy stack SRAM cell

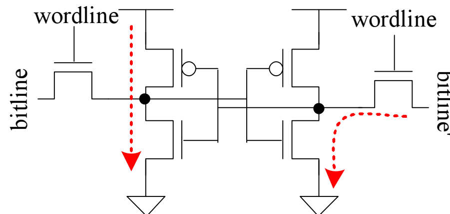

Figure 3: SRAM cell leakage paths

We design an SRAM cell based on the sleepy stack technique. The conventional 6T SRAM cell consists of two coupled inverters and two `wordline` pass transistors as shown in Figure 3. Since the sleepy stack technique can be applied to each transistor separately, the six transistors can be changed individually. However, to balance current flow (failure of this potentially increases the risk of soft errors [7]), a symmetric design approach is used.

Table 1: Sleepy stack technique on a SRAM cell

Combinations	cell leakage reduction	bitline leakage reduction
Pull-down (PD) sleepy stack	medium	low
Pull-down (PD), wordline (WL) sleepy stack	medium	high
Pull-up (PU), pull-down (PD) sleepy stack	high	low
Pull-up (PU), pull-down (PD), wordline (WL) sleepy stack	high	high

It is very important when applying the sleepy stack technique to consider the various leakage paths in the SRAM cell. The subthreshold leakage current in an SRAM cell is typically categorized into two kinds as shown in Figure 3: (i) cell leakage current that flows from V_{dd} to Gnd internal to the cell and (ii) `bitline` leakage current that flows from `bitline` (or `bitline'`) to Gnd. The `bitline` leakage occurs due to precharging of `bitline` and `bitline'`, and the `bitline` current accounts for 20% of SRAM cell leakage power according to our experiments. Although an SRAM cell is symmetric, the `bitline` current and `bitline'` current are different according to the stored value. The `wordline` pass transistor connected to the inverter that holds '1' suppresses leakage current thanks to the stack effect between the `wordline` pass transistor and the turned off pull-down transistor. However, the `wordline` pass transistor connected to an SRAM inverter that holds '0' incurs large leakage current.

To address the effect of the sleepy stack technique properly, we consider four combinations of the sleepy stack SRAM cell as shown in Table 1. In Table 1, "Pull-down sleepy stack" means that the sleepy stack technique is only applied to the pull-down transistors of an SRAM cell as indicated in Figure 4. "Pull-down, `wordline` sleepy stack" means that the sleepy stack technique is applied to the pull-down transistors as well as `wordline` transistors. Similarly, "Pull-up, pull-down sleepy stack" means that the sleepy stack technique is applied to the pull-up transistors and the pull-down transistors of an SRAM cell, and "Pull-up, pull-down, `wordline` sleepy stack" means that the sleepy stack technique is applied to all the transistors in an SRAM cell.

The pull-down (PD) sleepy stack can suppress some part of the cell leakage. Meanwhile, pull-up (PU) and pull-down (PD) sleepy stack can suppress the majority of the cell leakage. However, without applying the sleepy stack technique to the `wordline` (WL) transistors, `bitline` leakage cannot be significantly suppressed. Although lying in the `bitline` leakage path, the pull-down sleepy stack is not effective to suppress both `bitline` leakage paths because one of the pull-down sleepy stacks is always on. Therefore, to suppress subthreshold leakage current in a SRAM cell fully, PU, PD and WL sleepy stack need to be considered as shown in Figure 4.

The sleepy stack SRAM cell design results in area increase because of the increase of the number of transistors. However, we halve existing transistors and thus the area increase is not directly proportional to the number of transistors. Unlike the conventional 6T SRAM cell, the sleepy stack SRAM cell requires routing of one or two extra wires for the sleep control signal. Figure 5 shows a possible layout of the PU, PD, WL sleepy stack SRAM cell. We only use metal 1 and metal 2 layers for routing because we assume metal layers above metal 2 are reserved for the global routing. Further, the sleepy stack SRAM cell is designed to abut easily.

4 Experimental methodology

To evaluate the sleepy stack SRAM cell, we mainly use a simulation based methodology utilizing HSPICE. We compare our technique to (i) using high-V_{th} transistor as direct replacements for low-V_{th} transistors (thus maintaining only 6 transistors in an SRAM cell) and (ii) the forced stack technique [5] because these two techniques are state saving techniques without high risk of soft error [7]. We do not consider Asymmetric-Cell SRAM because

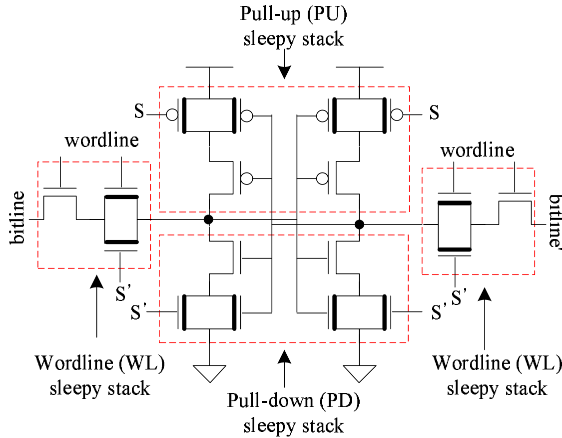

Figure 4: Sleepy stack SRAM cell

leakage power savings are limited compared to high-v_{th} transistor technique.

We first layout SRAM cells of each technique including the conventional 6T SRAM cell. Instead of starting from scratch, we use the CACTI model for the SRAM structure and transistor sizing [14]. We use NCSU Cadence design kit targeting TSMC 0.18μ technology [15]. By scaling down the 0.18μ layout, we obtain 0.07μ technology transistor level HSPICE schematics [16], and we design a 64x64bit SRAM cell array.

We estimate area directly from a custom layout using TSMC 0.18μ technology, and we assume the ratios are maintained after scaling the technology down to 0.07μ technology (we are aware this is not exact, hence the word "estimate"). We also assume the area of the SRAM cell with high-V_{th} technique is the same as low-V_{th}. This assumption is reasonable because high-V_{th} can be implemented by changing gate oxide thickness, and this almost does not affect area. We estimate dynamic power, static power and read time of the SRAM cell using HSPICE simulation with Berkeley Predictive Technology Model targeting 0.07μ technology [17]. The read time is measured from the time when an enabled `wordline` reaches 10% of the V_{dd} to the time when either `bitline` or `bitline'` drops to 90% of the precharged voltage value while the other remains high. This 10% voltage difference between `bitline` and `bitline'` is typically enough for a sense amplifier to detect the stored cell value [4]. Dynamic power of the SRAM array is measured during the read operation with 2ns of cycle time. Static power of the SRAM cell is measured by turning off sleep transistors if applicable. To avoid leakage power measurement biased by a majority of '1' versus '0' (or vice-versa) values, half of the cells are randomly set to '0,' with the remaining half of the cells set to '1.'

5 Results

We compare the sleepy stack SRAM cell to the conventional 6T SRAM cell, high-V_{th} 6T SRAM cell and the forced stack SRAM cell. For the high-V_{th} technique and the forced stack technique, we consider the same technique combinations applied to the sleepy stack SRAM cell as shown in Table 1 on the previous page.

To properly observe the techniques, we compare 13 different cases as shown in Table 2. Case1 is the conventional 6T SRAM cell, which is our base case. Cases 2, 3, 4 and 5 are 6T SRAM cells with the high-V_{th} technique. PD high-V_{th} is the high-V_{th} technique applied only to the pull-down transistors. PD, WL high-V_{th} is the high-V_{th} technique applied to the pull-down transistors as well as the `wordline` transistors. PU, PD high-V_{th} is the high-V_{th} technique applied to the pull-up and pull-down transistors. PU, PD, WL high-V_{th} is the high-V_{th} technique applied to all the SRAM transistors. Cases 6, 7, 8 and 9 are 6T SRAM cells with the forced stack technique [5]. PD

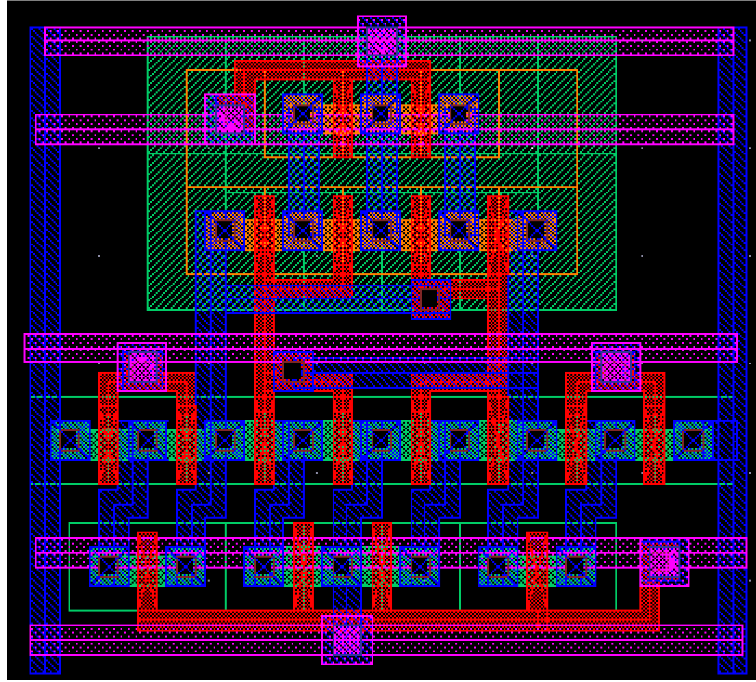

Figure 5: Sleepy stack SRAM cell layout

stack is the stack technique applied only to the pull-down transistors. PD, WL stack is the stack technique applied to the pull-down transistors as well as the `wordline` transistors. PU, PD stack is the stack technique applied to the pull-up and pull-down transistors. PU, PD, WL stack is the stack technique applied to all the SRAM transistors. Please note that we do not apply high-V_{th} to the forced stack technique because the forced stack SRAM with high-V_{th} incurs more than 2X delay increase. Cases 10, 11, 12 and 13 are the four sleepy stack SRAM cell approaches as listed in Table 1. For the sleepy stack, high-V_{th} is applied only to the sleep transistors and the transistors parallel to the sleep transistors as shown in Figure 4.

5.1 Area

Table 2: Layout area

	Technique	Height(u)	Width(u)	Area(u^2)	Normalized area
Case1	Low-Vth Std	3.825	4.500	17.213	1.00
Case2	PD high-Vth	3.825	4.500	17.213	1.00
Case3	PD, WL high-Vth	3.825	4.500	17.213	1.00
Case4	PU, PD high-Vth	3.825	4.500	17.213	1.00
Case5	PU, PD, WL high-Vth	3.825	4.500	17.213	1.00
Case6	PD stack	3.465	4.680	16.216	0.94
Case7	PD, WL stack	3.465	5.760	19.958	1.16
Case8	PU, PD stack	3.285	4.680	15.374	0.89
Case9	PU, PD, WL stack	3.465	5.760	19.958	1.16
Case10	PD sleepy stack	4.545	5.040	22.907	1.33
Case11	PD, WL sleepy stack	4.455	6.705	29.871	1.74
Case12	PU, PD sleepy stack	5.760	5.040	29.030	1.69
Case13	PU, PD, WL sleepy stack	5.535	6.615	36.614	2.13

Table 2 shows the area of each technique. Please note that the SRAM cells can be reduced further by using

minimum size transistors, but reducting transistor size increases cell read time. Also note, some SRAM cell design, e.g., [18] has $8.2\mu m^2$ cell size using 0.20μ technology, may have smaller size than our design. However, [18] achieves 80% of area reduction over the conventional SRAM cell using a non-conventional CMOS process (while we use a conventioal CMOS process). Furthermore, [18] does not consider area occupied by routing wires while we consider routing wire area to measure more accurate area as shown in Figure 6.

Some SRAM cells with the stack technique show smaller area even compared to the base case. For example, the layout of Case8 shown in Figure 7 has smaller area than Case1. Although Case8 has larger width than Case1, the smaller height of Case8 due to reduced transistor width eventually achieves smaller area than Case1. The sleepy stack technique increases area by between 33% and 113%. The added sleep transistors are a bottleneck to reduce the size of the sleepy stack SRAM cells. Further, wiring the sleep control signals makes the design more complicated as shown inn Figure 5.

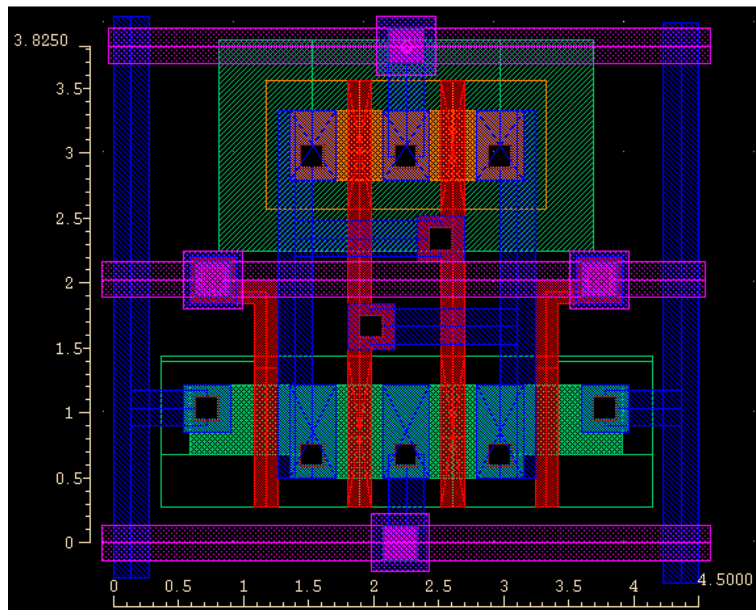

Figure 6: Conventional 6-T SRAM cell layout

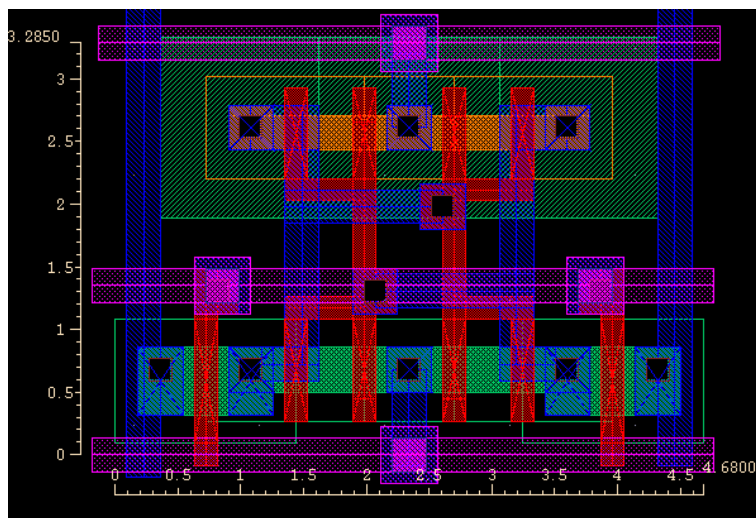

Figure 7: Forced stack SRAM cell layout

5.2 Cell read time

Table 3: Cell read time

Technique	Delay (sec)						Normalized delay					
	25°C			110°C			25°C			110°C		
	1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth
Low-Vth Std	1.04E-10	N/A		1.05E-10	N/A		1.000	N/A		1.000	N/A	
PD high-Vth	N/A	1.06E-10	1.08E-10	N/A	1.07E-10	1.11E-10	N/A	1.022	1.043	N/A	1.020	1.061
PD, WL high-Vth		1.16E-10	1.33E-10		1.17E-10	1.32E-10		1.111	1.280		1.117	1.262
PU, PD high-Vth		1.06E-10	1.10E-10		1.07E-10	1.10E-10		1.022	1.055		1.020	1.048
PU, PD, WL high-Vth		1.15E-10	1.33E-10		1.16E-10	1.32E-10		1.111	1.277		1.110	1.259
PD stack	1.42E-10	N/A		1.41E-10	N/A		1.368	N/A		1.345	N/A	
PD, WL stack	1.71E-10			1.76E-10			1.647			1.682		
PU, PD stack	1.40E-10			1.40E-10			1.348			1.341		
PU, PD, WL stack	1.77E-10			1.75E-10			1.704			1.678		
PD sleepy stack	N/A	1.33E-10	1.36E-10	N/A	1.32E-10	1.31E-10	N/A	1.276	1.307	N/A	1.263	1.254
PD, WL sleepy stack		1.52E-10	1.61E-10		1.50E-10	1.62E-10		1.458	1.551		1.435	1.546
PU, PD sleepy stack		1.33E-10	1.36E-10		1.35E-10	1.38E-10		1.275	1.306		1.287	1.319
PU, PD, WL sleepy stack		1.51E-10	1.67E-10		1.52E-10	1.57E-10		1.456	1.605		1.450	1.504

Although SRAM cell read time changes slightly as temperature changes, the impact of temperature on the cell read time is quite small. However, the impact of threshold voltage is large. We apply 1.5xV_{th} and 2xV_{th} for the high-V_{th} technique and the sleepy stack technique. As shown in Table 3, the delay penalty of the forced stack technique is between 35% and 70% compared to the standard 6T cell. This is one of the primary reasons that the stack technique cannot use high-V_{th} without incurring dramatic delay increases (e.g., 2X or more delay penalty is observed using either 1.5xV_{th} or 2xV_{th}). Among the three low-leakage techniques, the sleepy stack technique is the second best in terms of delay. Since we are aware that area and delay are critical factors when designing SRAM, we will explore area and delay impact using tradeoffs in Section 5.4. However, let us first discuss leakage reduction (i.e., without yet focusing on tradeoffs, which will be the focus of Section 5.4).

5.3 Leakage power

We measure leakage power while changing threshold voltage and temperature because the impact of threshold voltage and temperature on leakage power is significant. Table 4 shows normalized leakage power consumption with two high-V_{th} values, 1.5xV_{th} and 2xV_{th}, and two temperatures, 25^oC and 110^oC, where Case1 and the cases using the stack technique (Cases 6, 7, 8 and 9) are not affected by changing V_{th} because these use only low-V_{th}.

Table 4: Leakage power

	Technique	Leakage power (W)						Normalized leakage power					
		25°C			110°C			25°C			110°C		
		1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth
Case1	Low-Vth Std	9.71E-05	N/A		1.25E-03	N/A		1.0000	N/A		1.0000	N/A	
Case2	PD high-Vth	N/A	5.31E-05	5.12E-05	N/A	7.16E-04	6.65E-04	N/A	0.5466	0.5274	N/A	0.5711	0.5305
Case3	PD, WL high-Vth		2.01E-05	1.69E-05		3.20E-04	2.33E-04		0.2071	0.1736		0.2555	0.1860
Case4	PU, PD high-Vth		3.68E-05	3.45E-05		5.04E-04	4.42E-04		0.3785	0.3552		0.4022	0.3522
Case5	PU, PD, WL high-Vth		3.79E-06	1.38E-07		1.07E-04	8.19E-06		0.0391	0.0014		0.0857	0.0065
Case6	PD stack	5.38E-05	N/A		7.07E-04	N/A		0.5541	N/A		0.5641	N/A	
Case7	PD, WL stack	2.15E-05			3.20E-04			0.2213			0.2554		
Case8	PU, PD stack	3.75E-05			4.95E-04			0.3862			0.3950		
Case9	PU, PD, WL stack	5.39E-06			1.04E-04			0.0555			0.0832		
Case10	PD sleepy stack	N/A	5.18E-05	5.16E-05	N/A	6.62E-04	6.51E-04	N/A	0.5331	0.5315	N/A	0.5282	0.5192
Case11	PD, WL sleepy stack		1.80E-05	1.77E-05		2.45E-04	2.28E-04		0.1852	0.1827		0.1955	0.1820
Case12	PU, PD sleepy stack		3.54E-05	3.52E-05		4.43E-04	4.31E-04		0.3646	0.3630		0.3534	0.3439
Case13	PU, PD, WL sleepy stack		1.62E-06	3.24E-07		2.09E-05	2.95E-06		0.0167	0.0033		0.0167	0.0024

5.3.1 Results at 25^oC

Our results at 25^oC show that Case5 is the best with 2xV_{th} and Case13 is the best with 1.5xV_{th}. Specially, at 1.5xV_{th}, Case5 and Case13 achieve 25X and 60X leakage reduction over Case1, respectively. However, the leakage reduction comes with delay increase. The delay penalty is 11% and 45%, respectively, compared to Case1.

5.3.2 Results at 110^oC

Absolute power consumption numbers at 110^oC show more than 10X increase of leakage power consumption compared to the results at 25^oC. This could be a serious problem for SRAM because SRAM often resides next to a microprocessor whose temperature is high.

At 110^oC, the sleepy stack technique shows the best result in both 1.5xV_{th} and 2xV_{th} even compared to the high-V_{th} technique. The leakage performance degradation under high temperature is very noticeable with the high-V_{th} technique and the forced stack technique. For example, at 25^oC the high-V_{th} technique with 1.5xV_{th} (Case5) and the forced stack technique (Case9) show around 96% leakage reduction. However, at 110^oC the same techniques show around 91% of leakage power reduction compared to Case1. Only the sleepy stack technique achieves superior leakage power reduction; after increasing temperature, the sleepy stack SRAM shows 5.1X and 4.8X reductions compared to Case5 and Case9, respectivley, with 1.5xV_{th}.

When the low-leakage techniques are applied only to the pull-down transistors, leakage power reduction is at most 47% (1.5xV_{th} 110^oC) because `bitline` leakage cannot be suppressed. However, if the techniques are applied to the `wordline`, additional leakage reduction is achieved. The results are similar in case of techniques only applied to pull-up and pull-down. Without handling `bitline` leakage properly, 65% or more leakage power reduction cannot be achieved in our simulation result. Although `bitline` leakage is smaller than cell leakage, without `bitline` leakage reduction, the overall leakage power reduction is limited.

5.4 Tradeoffs in low-leakage techniques

Although the sleepy stack technique shows superior results in terms of leakage power, we need to explore area, delay and power together because the sleepy stack technique comes with non-negligible area and delay penalties. To be compared with the high-V_{th} technique at the same delay, the `wordline` and pull-down transistors of the sleepy stack are increased until the sleepy stack delay is approximately equal to the delay of the 6T high-V_{th} case. The results are shown in Table 5 in which "*" means a technique with adjusted transistor width. To enhance readability of tradeoffs, the table is sorted by leakage power. Although we compared four different simulation conditions, we take the condition with 1.5xV_{th} at 110^oC as an important representative technology point at which to compare the trade-offs between techniques. We choose 110^oC because generally SRAM operates at a high temperature and also because high temperature is the "worst case." Furthermore, 1.5xV_{th} is chosen because the delay with 1.5xV_{th} is 10% less than with 2xV_{th} (the relative areas are approximately equal).

In Table 5, we observe six pareto points, which are in shaded rows, considering three variables of leakage, delay, and area. Case13 shows the lowest possible leakage, at least 5X smaller than the leakage of any of the prior approaches considered; however, there is a corresponding delay and area penalty. Case13* shows the same delay (within 0.2%) as Case5 and is approximately 25% faster than Case13; furthermore, Case13* shows a 2.5X leakage reduction over Case5. In addition, Case13* is only 11.2% slower than the fastest pareto point, Case1, yet has 29X less power consumption than Case1. In short, this paper presents new, previously unknown pareto points at the low-leakage end of the spectrum.

Table 5: Tradeoffs (1.5xV_{th}, 110^oC)

	Technique	Static (W)	Delay (sec)	Area (u^2)	Normalized leakage	Normalized delay	Normalized area
Case1	Low-Vth Std	1.25E-03	1.05E-10	17.21	1.000	1.000	1.000
Case2	PD high-Vth	7.16E-04	1.07E-10	17.21	0.571	1.020	1.000
Case3	PD, WL high-Vth	3.20E-04	1.17E-10	17.21	0.256	1.117	1.000
Case4	PU, PD high-Vth	5.04E-04	1.07E-10	17.21	0.402	1.020	1.000
Case5	PU, PD, WL high-Vth	1.07E-04	1.16E-10	17.21	0.086	1.110	1.000
Case6	PD stack	7.07E-04	1.41E-10	16.22	0.564	1.345	0.942
Case7	PD, WL stack	3.20E-04	1.76E-10	19.96	0.255	1.682	1.159
Case8	PU, PD stack	4.95E-04	1.40E-10	15.37	0.395	1.341	0.893
Case9	PU, PD, WL stack	1.04E-04	1.75E-10	19.96	0.083	1.678	1.159
Case10	PD sleepy stack	6.62E-04	1.32E-10	22.91	0.528	1.263	1.331
Case11	PD, WL sleepy stack	2.45E-04	1.50E-10	29.87	0.196	1.435	1.735
Case12	PU, PD sleepy stack	4.43E-04	1.35E-10	29.03	0.353	1.287	1.687
Case13	PU, PD, WL sleepy stack	2.09E-05	1.52E-10	36.61	0.017	1.450	2.127
Case10*	PD sleepy stack*	6.74E-04	1.15E-10	25.17	0.538	1.102	1.463
Case11*	PD, WL sleepy stack*	2.72E-04	1.16E-10	34.40	0.217	1.111	1.998
Case12*	PU, PD sleepy stack*	4.53E-04	1.15E-10	31.30	0.362	1.103	1.818
Case13*	PU, PD, WL sleepy stack*	4.31E-05	1.16E-10	41.12	0.034	1.112	2.389

5.5 Active power

Table 6: Active power

<table>
<tr><th rowspan="3"></th><th rowspan="3">Technique</th><th colspan="6">Active power (W)</th><th colspan="6">Normalized active power</th></tr>
<tr><th colspan="3">25°C</th><th colspan="3">110°C</th><th colspan="3">25°C</th><th colspan="3">110°C</th></tr>
<tr><th>1xVth</th><th>1.5xVth</th><th>2xVth</th><th>1xVth</th><th>1.5xVth</th><th>2xVth</th><th>1xVth</th><th>1.5xVth</th><th>2xVth</th><th>1xVth</th><th>1.5xVth</th><th>2xVth</th></tr>
<tr><td>Case1</td><td>Low-Vth Std</td><td>8.19E-04</td><td colspan="2">N/A</td><td>2.04E-03</td><td colspan="2">N/A</td><td>1.000</td><td colspan="2">N/A</td><td>1.000</td><td colspan="2">N/A</td></tr>
<tr><td>Case2</td><td>PD high-Vth</td><td rowspan="4">N/A</td><td>7.67E-04</td><td>7.48E-04</td><td rowspan="4">N/A</td><td>1.48E-03</td><td>1.41E-03</td><td rowspan="4">N/A</td><td>0.936</td><td>0.913</td><td rowspan="4">N/A</td><td>0.724</td><td>0.691</td></tr>
<tr><td>Case3</td><td>PD, WL high-Vth</td><td>7.02E-04</td><td>6.78E-04</td><td>1.26E-03</td><td>9.75E-04</td><td>0.858</td><td>0.829</td><td>0.618</td><td>0.478</td></tr>
<tr><td>Case4</td><td>PU, PD high-Vth</td><td>7.60E-04</td><td>7.31E-04</td><td>1.17E-03</td><td>1.19E-03</td><td>0.928</td><td>0.893</td><td>0.572</td><td>0.582</td></tr>
<tr><td>Case5</td><td>PU, PD, WL high-Vth</td><td>6.86E-04</td><td>6.89E-04</td><td>8.82E-04</td><td>7.50E-04</td><td>0.838</td><td>0.842</td><td>0.432</td><td>0.368</td></tr>
<tr><td>Case6</td><td>PD stack</td><td>7.58E-04</td><td rowspan="4" colspan="2">N/A</td><td>1.37E-03</td><td rowspan="4" colspan="2">N/A</td><td>0.926</td><td rowspan="4" colspan="2">N/A</td><td>0.669</td><td rowspan="4" colspan="2">N/A</td></tr>
<tr><td>Case7</td><td>PD, WL stack</td><td>5.45E-04</td><td>8.12E-04</td><td>0.665</td><td>0.398</td></tr>
<tr><td>Case8</td><td>PU, PD stack</td><td>7.41E-04</td><td>1.22E-03</td><td>0.905</td><td>0.596</td></tr>
<tr><td>Case9</td><td>PU, PD, WL stack</td><td>5.22E-04</td><td>5.97E-04</td><td>0.637</td><td>0.293</td></tr>
<tr><td>Case10</td><td>PD sleepy stack</td><td rowspan="4">N/A</td><td>8.03E-04</td><td>8.03E-04</td><td rowspan="4">N/A</td><td>1.65E-03</td><td>1.66E-03</td><td rowspan="4">N/A</td><td>0.981</td><td>0.981</td><td rowspan="4">N/A</td><td>0.807</td><td>0.811</td></tr>
<tr><td>Case11</td><td>PD, WL sleepy stack</td><td>6.32E-04</td><td>5.87E-04</td><td>1.20E-03</td><td>1.22E-03</td><td>0.773</td><td>0.717</td><td>0.586</td><td>0.600</td></tr>
<tr><td>Case12</td><td>PU, PD sleepy stack</td><td>7.87E-04</td><td>8.23E-04</td><td>1.60E-03</td><td>1.63E-03</td><td>0.961</td><td>1.005</td><td>0.786</td><td>0.797</td></tr>
<tr><td>Case13</td><td>PU, PD, WL sleepy stack</td><td>5.89E-04</td><td>5.80E-04</td><td>1.20E-03</td><td>1.11E-03</td><td>0.719</td><td>0.708</td><td>0.588</td><td>0.546</td></tr>
</table>

Table 6 shows power consumption during read operations. The active power consumption includes dynamic power used to charge and discharge SRAM cells plus leakage power consumption. At 25^oC leakage power is less than 20% of the active power in case of the standard low-V_{th} SRAM cell in 0.07μ technology according to the modeling we use from [17]. However, leakage power increases 10X as the temperature changes to 110^oC although active power increases 3X. At 110^oC, leakage power is more than half of the active power from our simulation results. Therefore, without an effective leakage power reduction technique, total power consumption – even in active mode – is affected significantly.

5.6 Static noise margin

Changing the SRAM cell structure may change the static noise immunity of the SRAM cell. Thus, we measure the static noise margin (SNM) of the sleepy stack SRAM cell and the conventional 6T SRAM cell using the butterfly plots in Figure 8. The SNM is defined by the size of the maximum nested square in a butterfly plot. The SNM of the sleepy stack SRAM cell is measured twice in active mode and sleep mode, and results are shown in Table 7. The

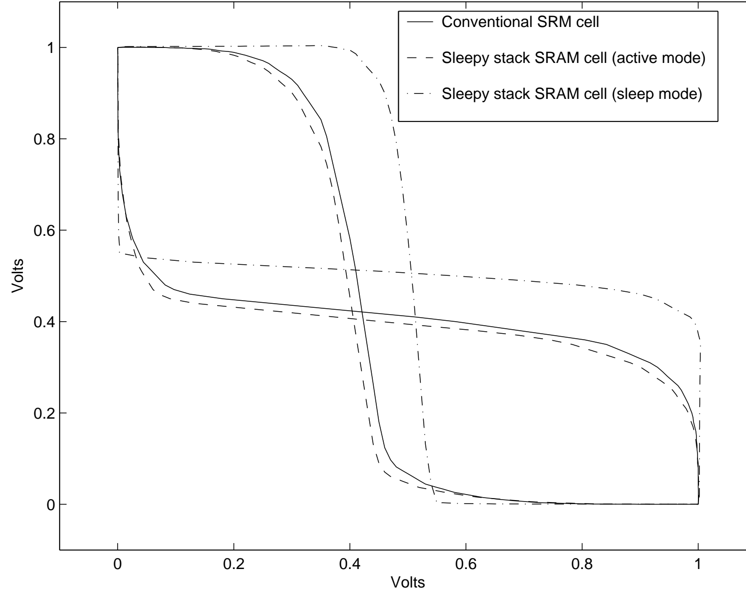

Figure 8: Static noise margin analysis

Table 7: Staci noise margin

	Technique	Static noise margin (V)	
		Active mode	Sleep mode
Case1	Low-Vth Std	0.299	N/A
Case10	PD sleepy stack	3.167	0.362
Case11	PD, WL sleepy stack	0.324	0.363
Case12	PU, PD sleepy stack	0.299	0.384
Case13	PU, PD, WL sleepy stack	0.299	0.384

SNM of the sleepy stack SRAM cell in active mode is $0.30V$ and almost similar to the SNM of the conventional SRAM cell. In sleep mode, the SNM of the sleepy stack SRAM cell is improved to $0.38V$. Therefore, the sleepy stack SRAM cell appears to be similar to a conventional SRAM cell in static noise immunity. Although we did not perform a process variation analysis, we expect that the high SNM of the sleepy stack SRAM cell makes the technique as immune to process variations as a conventional SRAM cell.

6 Conclusions

In this paper we have presented and evaluated our newly proposed "sleepy stack SRAM." For example, the sleepy stack SRAM provides the largest leakage savings – 416X – among all alternatives considered. Specifically, compared to a standard SRAM cell – Case1 – Table 4 shows that at 110^oC and 2xV_{th}, Case13 reduces leakage by 416X as compared to Case1; unfortunately, this 416X reduction comes as a cost of a delay increase of 50.4% and an area penalty of 113%. Resizing the sleepy stack SRAM can reduce delay significantly at a cost of less leakage savings; specifically, Case13* is an interesting pareto point as discussed in Section 5.4.

We believe that this paper presents a dramatic development because our sleepy stack SRAM seems to provide, in general, the lowest leakage pareto points of any VLSI design style known to the authors. Given the nontrivial area penalty (e.g., up to 138.9% for Case13* in Table 5), perhaps sleepy stack SRAM would be most appropriate for a small SRAM intended to store minimal standby data for an embedded system spending significant time in standby mode; for such a small SRAM (e.g., 16KB), the area penalty may be acceptable given system-level standby power

requirements. If absolute minimum leakage power is extremely critical, then perhaps specific target embedded systems could use sleepy stack SRAM more widely.

For future work, we will model the dynamic and leakage power consumption and delay of the rest of SRAM (address decoder, sense amplifier, precharging logic and column MUX) and evaluate techniques for architectural level SRAM power reduction.

References

[1] N. S. Kim, T. Austin, D. Baauw, T. Mudge, K. Flautner, J. Hu, M. Irwin, M. Kandemir, and V. Narayanan, "Leakage Current: Moore's Law Meets Static Power," *IEEE Computer*, vol. 36, pp. 68–75, December 2003.

[2] L. Clark, E. Hoffman, J. Miller, M. Biyani, L. Luyun, S. Strazdus, M. Morrow, K. Velarde, and M. Yarch, "An Embedded 32-b Microprocessor Core for Low-Power and High-Performance Applications," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 11, pp. 1599–1608, November 2001.

[3] J. Park, V. J. Mooney, and P. Pfeiffenberger, "Sleepy Stack Reduction in Leakage Power," *Proceedings of the International Workshop on Power and Timing Modeling, Optimization and Simulation*, pp. 148–158, September 2004.

[4] N. Azizi, A. Moshovos, and F. Najm, "Low-Leakage Asymmetric-Cell SRAM," *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 48–51, August 2002.

[5] S. Narendra, V. D. S. Borkar, D. Antoniadis, and A. Chandrakasan, "Scaling of Stack Effect and its Application for Leakage Reduction," *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 195–200, August 2001.

[6] S. Tang, S. Hsu, Y. Ye, J. Tschanz, D. Somasekhar, S. Narendra, S.-L. Lu, R. Krishnamurthy, and V. De, "Scaling of Stack Effect and its Application for Leakage Reduction," *Symposium on VLSI Circuits Digest of Technical Papers*, pp. 320–321, June 2002.

[7] V. Degalahal, N. Vijaykrishnan, and M. Irwin, "Analyzing soft errors in leakage optimized SRAM design," *IEEE International Conference on VLSI Design*, pp. 227–233, January 2003.

[8] K. Nii, H. Makino, Y. Tujihashi, C. Morishima, Y. Hayakawa, H. Nunogami, T. Arakawa, and H. Hamano, "A Low Power SRAM Using Auto-Backgate-Controlled MT-CMOS," *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 293–298, August 1998.

[9] H. Hanson, M. S. Hrishikesh, V. Agarwal, S. W. Keckler, and D. Burger, "Static energy reduction techniques for microprocessor caches," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 3, pp. 303–313, June 2003.

[10] M. Powell, S.-H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar, "Gated-Vdd: A Circuit Technique to Reduce Leakage in Deep-submicron Cache Memories," *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 90–95, July 2000.

[11] A. Agarwal, L. Hai, and K. Roy, "DRG-Cache: A Data Retention Gated-Ground Cache for Low Power," *Proceedings of the Design Automation Conference*, pp. 473 – 478, June 2002.

[12] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy Caches: Simple Techniques for Reducing Leakage Power," *Proceedings of the International Symposium on Computer Architecture*, pp. 148–157, May 2002.

[13] K.-S. Min, H. Kawaguchi, and T. Sakurai, "Zigzag Super Cut-off CMOS (ZSCCMOS) Block Activation with Self-Adaptive Voltage Level Controller: An Alternative to Clock-gating Scheme in Leakage Dominant Era," *IEEE International Solid-State Circuits Conference*, vol. 1, pp. 400–401, February 2003.

[14] S. Wilton and N. Jouppi, An Enhanced Access and Cycle Time Model for On-Chip Caches. [Online]. Available http://www.research.compaq.com/wrl/people/jouppi/CACTI.html.

[15] NC State University Cadence Tool Information. [Online]. Available http://www.cadence.ncsu.edu.

[16] P. Pfeiffenberger, J. Park, and V. Mooney, “Some Layouts Using the Sleepy Stack Approach,” Georgia Institute of Technology, Tech. Rep. GIT-CC-04-05, June 2004. [Online]. Available: http://www.cc.gatech.edu/research/pubs.html

[17] Berkeley Predictive Technology Model (BPTM). [Online]. Available http://www-device.eecs.berkeley.edu/˜ptm/.

[18] F. Ootsuka, M. Nakamura, T. Miyake, S. Iwahashi, Y. Ohira, T. Tamaru, K. Kikushima, and K. Yamaguchi, “A Novel 0.20um Full CMOS SRAM cell Using Stacked Cross Couple with Enhanced Soft Error Immunity,” *IEEE International Electron Devices Meeting*, pp. 205–208, December 1998.