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 6-T SRAM cell with high-V_{th} transistors, the sleepy stack SRAM cell with 2xV_{th} at 110^oC achieves more than 2.77X leakage power reduction at a cost of 16% delay increase and 113% area increase. Alternatively, by widening `wordline` *transistors and transistors in the pull-down network, the sleepy stack SRAM cell can achieves 2.26X leakage reduction without increasing delay at a cost of a 125% area penalty.*

1 Introduction

Power consumption is one of the top concerns of Very Large Scale Integration (VLSI) circuit design, for which Complementary Metal Oxide Semiconductor (CMOS) is the primary technology. Today's focus on low power is not only because of the recent growing demands of mobile applications. Even before the mobile era, power consumption has been a fundamental problem. Power consumption of CMOS consists of dynamic and static components. Although dynamic power accounted for 90% or more of the total chip power previously, as the feature size shrinks, e.g., to 0.09μ and 0.065μ, static power has become a great challenge for current and future technologies. Based on the International Technology Roadmap for Semiconductors (ITRS) [1], Kim et al. report that subthreshold leakage power dissipation of a chip may exceed dynamic power dissipation at the 65nm feature size [2].

One of the main reasons causing the leakage power increase is increase of subthreshold leakage power. When technology feature size scales down, supply voltage and threshold voltage also scale down. Subthreshold leakage power increases exponentially as threshold voltage decreases. Furthermore, the structure of the short channel device lowers the threshold voltage even lower. Another contributor to leakage power is gate-oxide leakage power due to the tunneling current through the gate-oxide insulator. Although gate-oxide leakage power may be comparable to subthreshold leakage power in nanoscale technology, we assume other techniques will address gate-oxide leakage; for example, high-k dielectric gate insulators may provide a solution to reduce gate-leakage [2]. Therefore, this paper focuses on reducing subthreshold leakage power consumption.

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 [3], 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 [4] achieves greatly reduced leakage power while maintaining precise logic state in sleep mode, which may be crucial for a product spending the majority of its time in sleep or 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

In this section, we discuss state-of-the-art low-power memory techniques, especially SRAM and cache techniques on which our research focuses.

One easy way to reduce leakage power consumption is by adopting high-V_{th} transistors for all SRAM cell transistors. This solution is simple but incurs delay increase.

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 (ACC), which partially applies high-V_{th} transistors in an SRAM cell to save leakage power if the SRAM cell is in the zero state [5]. However, the ACC leakage power savings are quite limited in case of a benchmark which fills SRAM with mostly non-

zero values.

Nii et al. propose Auto-Backgate-Controlled Multi-Threshold CMOS (ABC-MTCMOS), which uses Reverse-Body Bias (RBB) to reduce leakage power consumption [6]. RBB increases threshold voltage without losing logic state. This increased threshold voltage reduces leakage power consumption during sleep mode. However, since the ABC-MTCMOS technique needs to charge large wells, ABC-MTCMOS requires significant transition time and power consumption.

The forced stack technique achieves leakage power reduction by forcing a stack structure [7]. This technique breaks down existing transistors into two transistors 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 a memory element such as a register [8] or an SRAM cell [9]. 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 [9].

Sleep transistors can be used for SRAM cell design. Using sleep transistors, the gated-V_{dd} SRAM cell blocks pull-up networks from the V_{dd} rail (pMOS gated-V_{dd}) and/or blocks pull-down networks from the Gnd rail (nMOS gated-V_{dd}) [10]. The gated-V_{dd} SRAM cell achieves low leakage power consumption from both the stack effect and high-V_{th} sleep transistors. However, the gated-V_{dd} SRAM cell [10] loses state when the sleep transistors are turned off.

Flautner et al. propose the "drowsy cache" technique that switches V_{dd} dynamically [11]. For short-channel devices such as 0.07μ channel length devices, leakage power increases due to Drain Induced Barrier Lowering (DIBL), thereby increasing subthreshold leakage current. The drowsy cache lowers the supply voltage during drowsy mode and suppresses leakage current using DIBL. The drowsy cache technique can retain stored data at a leakage power reduction of up to 86% [11].

Our sleepy stack SRAM cell can achieve more power savings than a high-V_{th}, an ACC or a drowsy cache SRAM cell. Furthermore, the sleepy stack SRAM does not require large transition time and transition power consumption unlike ABC-MTCMOS.

3 Approach

We first introduce our recently proposed low-leakage structure named "sleepy stack" in Section 3.1. Then, we explain our newly proposed "sleepy stack SRAM" in Section 3.2.

3.1 Sleepy stack leakage reduction

The sleepy stack technique has a structure merging the forced stack technique and the sleep transistor technique. Figure 1 shows a sleepy stack inverter. 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_0 (i.e., $W_1 = W_0/2$),

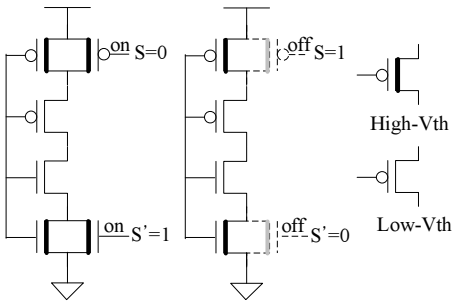

Figure 1: (a) Sleepy stack inverter active mode (left) and (b) sleep mode (right)

thus maintaining equivalent input capacitance. The sleepy stack inverter in Figure 1(a) uses $W/L = 3$ for the pull-up transistors and $W/L = 1.5$ for the pull-down transistors, while a conventional inverter with the same input capacitance would use $W/L = 6$ for the pull-up transistor and $W/L = 3$ for the pull-down transistor (assuming $\mu_n = 2\mu_p$). Then sleep transistors are added in parallel to one of the transistors in each set of two stacked transistors. We use half size transistor width of the original transistor (i.e., we use $W_0/2$) for the sleep transistor width of the sleepy stack.

During active mode, S=0 and S'=1 are asserted, and thus all sleep transistors are turned on. This structure potentially reduces circuit delay (compared to not adding sleep transistors) because (i) added sleep transistors are always on during active mode and thus at each sleep transistor drain, the voltage value connected to a sleep transistor is always ready during active mode and (ii) there is a reduced resistance due to the two parallel transistors. Therefore, we can introduce high-V_{th} transistors to the sleep transistors and transistors in parallel with the sleep transistor without incurring large (e.g., 2X or more) delay overhead. During sleep mode, S=1 and S'=0 are asserted, and so both of the sleep transistors are turned off. The high-V_{th} transistors and the stacked transistors in the sleepy stack approach suppress leakage current. In short, using high-V_{th} transistors, the sleepy stack technique potentially achieves 200X leakage reduction over the forced stack technique. Furthermore, unlike the sleep transistor technique [10], the sleepy stack technique can retain exact logic state while achieving similar leakage reduction.

3.2 Sleepy stack SRAM cell

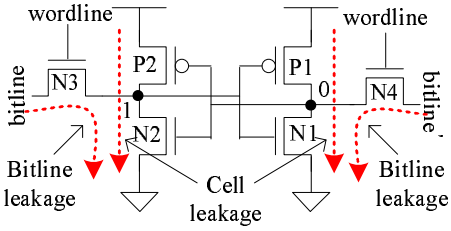

Figure 2: SRAM cell leakage paths

We design an SRAM cell based on the sleepy stack technique. The conventional 6-T SRAM cell consists of two coupled inverters and two `wordline` pass transistors as shown in Figure 2. 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 to do so potentially increases the risk of soft errors [9]), a symmetric design approach is used.

Table 1: Sleepy stack applied to an 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

There are two main types of subthreshold leakage currents in a 6-T SRAM cell: cell leakage and bitline leakage (see Figure 2). It is very important when applying the sleepy stack technique to consider the various leakage paths in the SRAM cell. 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 (PD) sleepy stack" means that the sleepy stack technique is only applied to the pull-down transistors of an SRAM cell as indicated in the bottom dashed box in Figure 3. "Pull-Down (PD), `wordline` (WL) sleepy stack" means that the sleepy stack technique is applied to the pull-down transistors as well as `wordline` transistors. Similarly, "Pull-Up (PU), Pull-Down (PD) sleepy stack" means that the sleepy stack technique is applied to the pull-up transistors and the pull-down transistors (but **not** to the `wordline` transistors) of an SRAM cell. Finally, "Pull-Up (PU), Pull-Down (PD), `wordline` (WL) sleepy stack" means that the sleepy stack technique is applied to all the transistors in an SRAM cell.

The PD sleepy stack can suppress some part of the cell leakage. Meanwhile, the PU, 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, the PU, PD and WL sleepy stack approach needs to be considered as shown in Figure 3.

The sleepy stack SRAM cell design results in area increase because of the increase in the number of transistors. However, we halve the transistor widths in a conventional SRAM cell to make the area increase of the sleepy stack SRAM cell not necessarily directly proportional to the number of transistors. Halving a transistor width is possible when the original transistor width is at least 2X larger than the minimum transistor width (which is typically the case in modern high performance SRAM cell design). Unlike the conventional 6-T SRAM cell, the sleepy stack SRAM cell requires the routing of one or two extra wires for the sleep control signal(s).

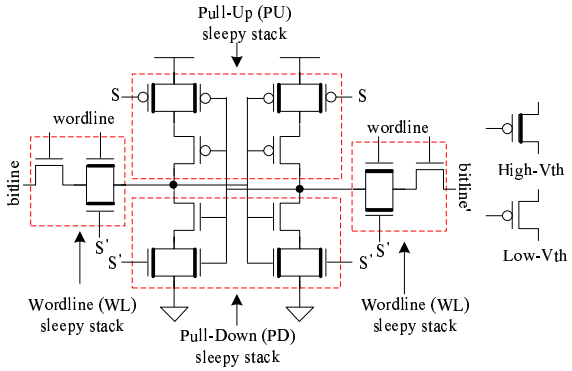

Figure 3: Sleepy stack SRAM cell

4 Experimental methodology

To evaluate the sleepy stack SRAM cell, we compare our technique to (i) using high-V_{th} transistors as direct replacements for low-V_{th} transistors (thus maintaining only 6 transistors in an SRAM cell) and (ii) the forced stack technique [7]; we choose these techniques because these two techniques are state saving techniques without high risk of soft error [9]. Although Asymmetric-Cell SRAM explained in Section 2 is also a state-saving SRAM cell design, we do not consider Asymmetric-Cell SRAM because we assume that our SRAM cells are filled equally with '1s' and '0s.' This is not the condition that ACC prefers, and under this condition the leakage power savings of ACC are smaller than the high-V_{th} SRAM cell, which uses high-V_{th} for all six transistors.

We first layout SRAM cells of each technique. Instead of starting from scratch, we use the CACTI model for the SRAM structure and transistor sizing [12]. We use NCSU Cadence design kit targeting TSMC 0.18μ technology [13]. By scaling down the 0.18μ layout, we obtain 0.07μ technology transistor level HSPICE schematics [4], and we design a 64x64bit SRAM cell array.

We estimate area directly from our custom layout using TSMC 0.18μ technology and scale to 0.07μ using the following formula: 0.07μ area = 0.18μ area $\times$ $(0.07\mu)^2$ / $(0.18\mu)^2$ $\times$ 1.1 (non-linear overhead) [4]. We are aware this is not exact, hence the word "estimate." We also assume the area of the SRAM cell with high-V_{th} transistors is the same as with low-V_{th} transistors. This assumption is reasonable because high-V_{th} can be implemented by changing gate oxide thickness, and this almost does not affect area at all. We estimate dynamic power, static power and read time of each of the various SRAM cell designs using HSPICE simulation with Berkeley Predictive Technology Model (BPTM) targeting 0.07μ technology [14]. The read time is measured from the time when an enabled `wordline` reaches 10% of the V_{dd} voltage to the time when either `bitline` or `bitline'` drops from 100% of the precharged voltage to 90% of the precharged voltage value while the other remains high. Therefore, one of the `bitline` signal remains at V_{dd}, and the other is 0.9xV_{dd}. This 10% voltage difference between `bitline` and `bitline'` is typically enough for a sense amplifier to detect the stored cell value [15]. Dynamic power of

the SRAM array is measured during the read operation with cycle time of 4ns. 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 6-T SRAM cell, high-V_{th} 6-T SRAM cell and forced stack SRAM cell. For the "high-V_{th}" technique and the forced stack technique, we consider the same technique combinations we applied to the sleepy stack SRAM cell – see Table 1.

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

5.1 Area

Table 2: Layout area

	Technique	Height(u)	Width(u)	Area(u^2) $0.18u^2$	Area(u^2) $0.07u^2$	Normalized area
Case1	Low-Vth Std	3.825	4.500	17.213	2.864	1.00
Case2	PD high-Vth	3.825	4.500	17.213	2.864	1.00
Case3	PD, WL high-Vth	3.825	4.500	17.213	2.864	1.00
Case4	PU, PD high-Vth	3.825	4.500	17.213	2.864	1.00
Case5	PU, PD, WL high-Vth	3.825	4.500	17.213	2.864	1.00
Case6	PD stack	3.465	4.680	16.216	2.698	0.94
Case7	PD, WL stack	3.465	5.760	19.958	3.320	1.16
Case8	PU, PD stack	3.285	4.680	15.374	2.558	0.89
Case9	PU, PD, WL stack	3.465	5.760	19.958	3.320	1.16
Case10	PD sleepy stack	4.545	5.040	22.907	3.811	1.33
Case11	PD, WL sleepy stack	4.455	6.705	29.871	4.969	1.74
Case12	PU, PD sleepy stack	5.760	5.040	29.030	4.829	1.69
Case13	PU, PD, WL sleepy stack	5.535	6.615	36.614	6.091	2.13

Table 2 shows the area of each technique. Please note that SRAM cell area can be reduced further by using minimum size transistors, but reducing transistor size increases cell read time. Some SRAM cells with the forced stack technique show smaller area even compared to the base case. The reason is that divided transistors can enable a particularly squeezed design [4]. 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 (an overhead we do not consider in Table 2) makes the design more complicated.

5.2 Cell read time

Table 3: Normalized cell read time

	Technique	25°C			110°C		
		1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth
Case1	Low-Vth Std	1.000	N/A		1.000	N/A	
Case2	PD high-Vth	N/A	1.022	1.043	N/A	1.020	1.061
Case3	PD, WL high-Vth		1.111	1.280		1.117	1.262
Case4	PU, PD high-Vth		1.022	1.055		1.020	1.048
Case5	PU, PD, WL high-Vth		1.111	1.277		1.110	1.259
Case6	PD stack	1.368	N/A		1.345	N/A	
Case7	PD, WL stack	1.647			1.682		
Case8	PU, PD stack	1.348			1.341		
Case9	PU, PD, WL stack	1.704			1.678		
Case10	PD sleepy stack	N/A	1.276	1.307	N/A	1.263	1.254
Case11	PD, WL sleepy stack		1.458	1.551		1.435	1.546
Case12	PU, PD sleepy stack		1.275	1.306		1.287	1.319
Case13	PU, PD, WL sleepy stack		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 (with all low-V_{th} transistors) is between 35% and 70% compared to the standard 6-T SRAM cell. This is one of the primary reasons that the forced stack technique cannot use high-V_{th} transistors without incurring dramatic delay increase (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 cell read time. The PU, PD, WL high-V_{th} with 2xV_{th} is 16% faster than the PU, PD, WL sleepy stack with 2xV_{th} at 110^o. 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 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 forced stack technique (Cases 6, 7, 8 and 9) are not affected by changing V_{th} because these use only low-V_{th}. (Please note the absolute numbers are available in [4].)

Table 4: Normalized leakage power

	Technique	Normalized leakage power					
		25°C			110°C		
		1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth
Case1	Low-Vth Std	1.0000	N/A		1.0000	N/A	
Case2	PD high-Vth	N/A	0.5466	0.5274	N/A	0.5711	0.5305
Case3	PD, WL high-Vth		0.2071	0.1736		0.2555	0.1860
Case4	PU, PD high-Vth		0.3785	0.3552		0.4022	0.3522
Case5	PU, PD, WL high-Vth		0.0391	0.0014		0.0857	0.0065
Case6	PD stack	0.5541	N/A		0.5641	N/A	
Case7	PD, WL stack	0.2213			0.2554		
Case8	PU, PD stack	0.3862			0.3950		
Case9	PU, PD, WL stack	0.0555			0.0832		
Case10	PD sleepy stack	N/A	0.5331	0.5315	N/A	0.5282	0.5192
Case11	PD, WL sleepy stack		0.1852	0.1827		0.1955	0.1820
Case12	PU, PD sleepy stack		0.3646	0.3630		0.3534	0.3439
Case13	PU, PD, WL sleepy stack		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, respectively, with 1.5xV_{th}.

When the low-leakage techniques are applied only to the pull-up and pull-down transistors, leakage power reduction is at most 65% (2xV_{th}, 110^oC) because `bitline` leakage cannot be suppressed. The remaining 35% of leakage power can be suppressed by applying low-leakage techniques to `wordline` transistors. This implies that `bitline` leakage power addresses around 35% of SRAM cell leakage power consumption. This trend is observed for all three techniques considered, i.e., high-V_{th}, forced stack and sleepy stack.

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 cell read time, we consider four more cases for sleepy stack SRAM in addition to the cases already considered in Table 4; we increase the widths of all `wordline` and pull-down transistors (including sleep transistors). Specifically, for the sleepy stack technique, we find new transistor widths of `wordline` transistors and pull-down transistors such that the result is delay approximately equal to the delay of the 6-T high-V_{th} case, i.e., Case5. The new cases are marked with '*' (Cases 10*, 11*, 12*, 13*). The results are shown in Table 5. To enhance readability of tradeoffs, each table is sorted by leakage power. Although we compared four different simulation conditions, we take the condition with 2xV_{th} at 110^oC and 2xV_{th} at 110^oC as important representative technology points 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."

Table 5: Tradeoffs (2xV_{th}, 110^oC)

	Technique	Normalized leakage	Normalized delay	Normalized area
Case1	Low-Vth Std	1.000	1.000	1.000
Case6	PD stack	0.564	1.345	0.942
Case2	PD high-Vth	0.530	1.061	1.000
Case10	PD sleepy stack	0.519	1.254	1.331
Case10*	PD sleepy stack*	0.519	1.254	1.331
Case8	PU, PD stack	0.395	1.341	0.893
Case4	PU, PD high-Vth	0.352	1.048	1.000
Case12*	PU, PD sleepy stack*	0.344	1.270	1.713
Case12	PU, PD sleepy stack	0.344	1.319	1.687
Case7	PD, WL stack	0.255	1.682	1.159
Case3	PD, WL high-Vth	0.186	1.262	1.000
Case11*	PD, WL sleepy stack*	0.183	1.239	1.876
Case11	PD, WL sleepy stack	0.182	1.546	1.735
Case9	PU, PD, WL stack	0.083	1.678	1.159
Case5	PU, PD, WL high-Vth	0.007	1.259	1.000
Case13*	PU, PD, WL sleepy stack*	0.003	1.265	2.253
Case13	PU, PD, WL sleepy stack	0.002	1.504	2.127

In Table 5, we observe six Pareto points, respectively, which are in shaded rows, considering three variables of leakage, delay, and area. Case13 shows the lowest possible leakage, 2.7X smaller than the leakage of any of the prior approaches considered; however, there is a corresponding delay and area penalty. Alternatively, Case13* shows the same delay (within 0.2%) as Case5 and 2.26X leakage reduction over Case5; however, Case13* uses 125% more area than Case5. In short, this paper presents new, previously unknown Pareto points at the low-leakage end of the spectrum (for a definition of a "Pareto point," please see [16]).

5.5 Active power

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

Table 6: Normalized active power

	Technique	25°C			110°C		
		1xVth	1.5xVth	2xVth	1xVth	1.5xVth	2xVth
Case1	Low-Vth Std	1.000	N/A		1.000	N/A	
Case2	PD high-Vth	N/A	0.936	0.913	N/A	0.724	0.691
Case3	PD, WL high-Vth		0.858	0.829		0.618	0.478
Case4	PU, PD high-Vth		0.928	0.893		0.572	0.582
Case5	PU, PD, WL high-Vth		0.838	0.842		0.432	0.368
Case6	PD stack	0.926	N/A		0.669	N/A	
Case7	PD, WL stack	0.665			0.398		
Case8	PU, PD stack	0.905			0.596		
Case9	PU, PD, WL stack	0.637			0.293		
Case10	PD sleepy stack	N/A	0.981	0.981	N/A	0.807	0.811
Case11	PD, WL sleepy stack		0.773	0.717		0.586	0.600
Case12	PU, PD sleepy stack		0.961	1.005		0.786	0.797
Case13	PU, PD, WL sleepy stack		0.719	0.708		0.588	0.546

to BPTM [14]. 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 6-T SRAM cell. 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. The SNM of the sleepy stack SRAM cell in active mode is $0.299V$ and almost exactly the same as the SNM of a conventional SRAM cell; the SNM of a conventional SRAM cell is $0.299V$. Although we do 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 and future work

In this paper we have presented and evaluated our newly proposed "sleepy stack SRAM." Our sleepy stack SRAM provides the largest leakage savings 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 424X as compared to Case1; unfortunately, this 424X 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 an important 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 non-trivial area penalty (e.g., up to 125% 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 explore how process variations affect leakage power reduction using sleepy stack SRAM.

7 References

[1] International Technology Roadmap for Semiconductors by Semiconductor Industry Association, 2002. [Online]. Available http://public.itrs.net.

[2] 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.

[3] 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.

[4] J. Park, "Sleepy Stack: a New Approach to Low Power VLSI and Memory," Ph.D. dissertation, School of Electrical and Computer Engineering, Georgia Institute of Technology, 2005. [Online]. Available http://etd.gatech.edu/theses/available/etd-07132005-131806/.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[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] 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.

[12] 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.

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

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

[15] 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.

[16] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*. USA: McGraw-Hill Inc., 1994.