Adaptive SRAM Memory for Low Power and High Yield

Baker Mohammad, Stephen Bijansky, Adnan Aziz, and Jacob Abraham
Department of Electrical and Computer Engineering
University of Texas at Austin
bijansky@ece.utexas.edu

***Abstract*— SRAMs typically represent half of the area and more than half of the transistors on a chip today. Variability increases as feature size decreases, and the impact of variability is especially pronounced on SRAMs since they make extensive use of minimum sized devices. Variability leads to a large amount of guard banding in the design phase in order to meet frequency and yield targets. We develop an SRAM architecture that eliminates guard banding. Specifically, our SRAM uses multiple supply voltages that are assigned post-manufacturing. We compensate for variation by powering up manufactured devices that are slower than designed. Specifically, we assign supply voltages to 6T cells on a per-column basis; this gives us sufficiently fine-grained control over devices without excessive area overhead. We show that post-manufacturing voltage assignment results in a 28% reduction in bitline energy compared to a fixed voltage design for the same yield using data from a real-world 45 nm process.**

I. Introduction

SRAM is a hugely important component of modern chips—it is used in caches, register files, FIFOs, etc. SRAMs impact area, power, timing, yield, and schedule. Since DRAM's primary emphasis is density rather than speed, the gap between design frequencies and DRAM access times has continued to increase, which has resulted in more on-die SRAM in order to meet performance targets. Consequently, SRAM constitutes more than half of chip area and more than half of the number of devices in modern designs [1].

SRAMs use the smallest transistors, which are particularly sensitive to process variations. Balancing the tradeoffs between small area, low power, fast reads/writes, and sensitivity to process variations are an essential part of any SRAM design optimization.

For sub 90nm process technology, pmos *negative bias temperature instability* (NBTI) [2] has resulted in a shifting pmos threshold voltage. The change in pmos threshold voltage reduces the *static noise margin* (SNM) of the SRAM. In order to compensate, the minimum allowed supply voltage V_{ddmin} is increased. SRAM stability and yield are often the limiting factor in determining the minimum supply voltage for a design. Thus increasing SRAM stability and yield would directly lead to higher chip yield and lower supply voltage.

In this work, we assign supply voltages post-manufacturing to columns of SRAMs cells. This way, the SRAM supply voltage is based on actual silicon instead of relying on 5σ estimated values. Furthermore, we are able to customize supply voltage to a small group of SRAMs cells instead of requiring a fixing supply voltage for the entire SRAM array. We use realistic process data to study the effects of our post-manufacturing voltage assignment. Our results show that post-manufacturing voltage assignment results in a 28% reduction in bitline energy compared to a fixed voltage design.

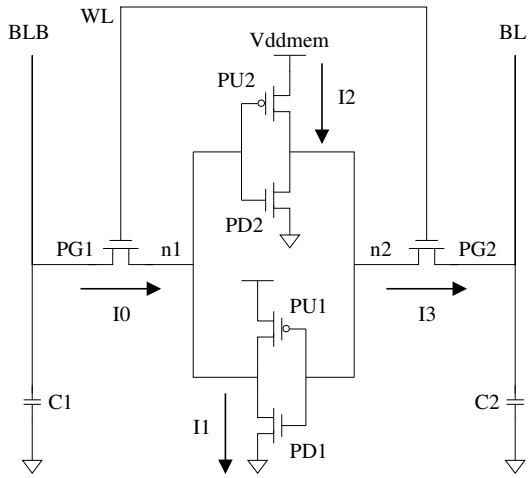

Fig. 1. 6T SRAM schematic.

The remaining sections of this paper are organized as follows: Section II gives background on SRAM cell design. We describe our approach to reducing SRAM power in Section III. Section IV contains experimental results. Section V presents related work in SRAM design. We conclude in Section VI.

II. Background: SRAM Cell Design

An SRAM consists of a matrix of 6T SRAM cells, which is surrounded by logic for row and column decode, and control. The 6T SRAM cells typically constitute 80–90% of the SRAM area.

The main functionality of the 6T SRAM cell, shown in Fig. 1, is to store data. SRAM cell design involves balancing a number of requirements, some of which are:

- Minimizing cell area
- Ensuring read and write stability
- Minimizing supply voltage to reduce power
- High cell read current to minimize access time
- Minimizing leakage current
- Reducing bitline swing to reduce power
- Good soft error immunity

Many of these criteria are conflicting in nature. For example, good cell stability, fast access time, and good soft

978-1-4244-2658-4/08/$25.00 ©2008 IEEE

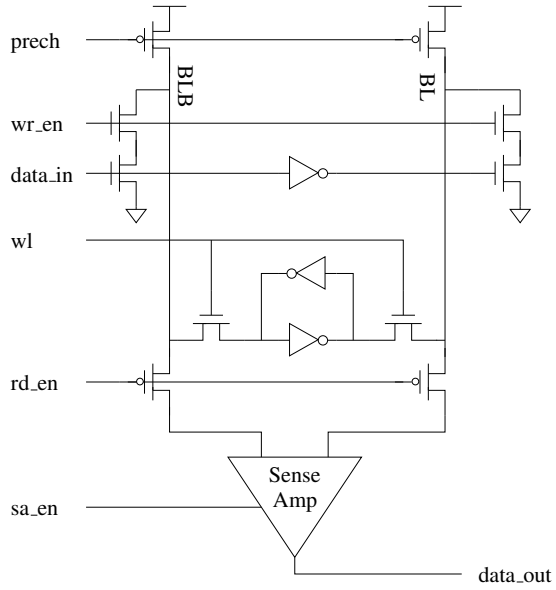

Fig. 2. SRAM column schematic.

error immunity would benefit from using larger transistors, but larger transistors result in larger area and increased leakage. Another example is that increasing the *cell ratio* (CR), defined in (1), through the use of a smaller pass-gate transistor (*PG*) improves the SNM, but the smaller *PG* transistor decreases the write margin.

$$\textit{Cell Ratio (CR)} \quad \triangleq \quad \frac{W_{pd}/L_{pd}}{W_{pg}/L_{pg}} \tag{1}$$

For the SRAM to function properly during read access at all process, voltage, and temperature corners (PVT), the current through *PD1* (I1) has to be greater than or equal to the current through *PG1* (I0), i.e.,

$$\mu_n C_{ox}\left(\frac{W_{pd}}{L_{pd}}\right)\left(V_{ddmem}-V_{tn}-\frac{V_{n1}}{2}\right)V_{n1} \quad \geq \quad \frac{\mu_n C_{ox}}{2}\left(\frac{W_{pg}}{L_{pg}}\right)(V_{ddwl}-V_{n1}-V_t)^{\alpha} \tag{2}$$

During write, the current through *PG2* (I3) has to be greater than or equal to the current through *PU2* (I2), i.e.,

$$\mu_n C_{ox}\left(\frac{W_{pg}}{L_{pg}}\right)\left(V_{ddwl}-V_{bit}-V_{tn}-\frac{V_{n2}}{2}\right)V_{n2} \quad \geq \quad \frac{\mu_p C_{ox}}{2}\left(\frac{W_{pu}}{L_{pu}}\right)(0-V_{ddmem}-V_{tp})^{\alpha} \tag{3}$$

The SRAM cell transistor sizes must satisfy (2) and (3). The SRAM cell must also produce the required read access current, I_{read}, which is determined by the *PG* and *PD* transistors. The current I_{read} is especially important because it is usually the determining factor for overall cache speed. It is often necessary to obtain silicon data in order to best tune the SRAM cell sizes and layout for robust design.

III. Our Adaptive SRAM

Current memory design uses a large number of identically designed SRAM cells. With process variation, though, the on-silicon SRAM cells are far from identical. The current design methodology means that all the SRAM cells have to be designed to work with the worst process variation. While general datapath and control logic might be designed for 3σ worth of process variation, because of the larger number of cells, SRAMs typically have 5σ of guard banding in order to achieve acceptable yields.

In the next section we show how a guard banded SRAM design leads to a large amount of wasted power.

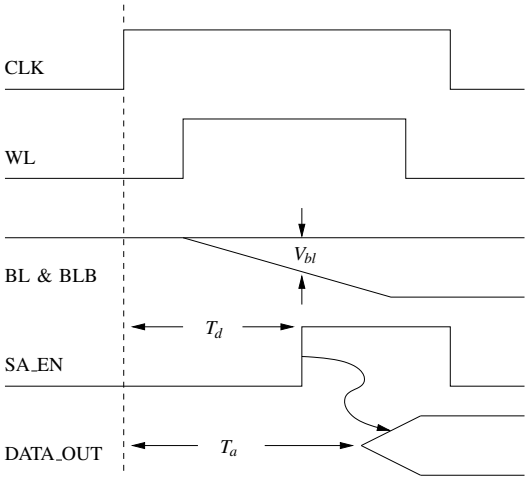

Fig. 3. Bitline timing diagram. V_{bl} is the amount of seperation between BL and BLB when the SA_EN is asserted.

A. Memory Operation

The bitline is a large contributor to memory bank power [3], so bitline power savings have a direct effect on overall SRAM power. Fig. 2 shows the schematic for one column of an SRAM array, and Fig. 3 shows the associated timing diagram for reading a single bit from that column.

The read operation starts with both bitline (BL) and bitline-bar (BLB) being pre-charged high. When the word line (WL) is asserted, the SRAM cell starts to pull down either BL or BLB via the nmos devices *PD1* or *PD2* (cf. Fig. 1). After a delay of T_d, the sense amplifier enable signal (*sa_en*) initiates the sense amplifier reading of the bitlines.

The amount of separation between BL and BLB when *sa_en* is asserted is referred to as the *bitline development* (V_{bl}). The sense amplifier is designed to guarantee correct read operation from the SRAM cell when V_{bl} has a certain minimum value.

If C is the total capacitance on a bitline, the total energy expended in pre-charging and evaluating the bitlines is given by the following equation (cf. Appendix):

$$\text{Energy}_{bitline} = \text{C}\ V_{bl}\ V_{dd} \tag{4}$$

Equation (4) does not have a $\frac{1}{2}$ term because the bitline is both pre-charged and evaluated in the same cycle.

From (4) it is clear that the larger bitline development is, the more energy is consumed. Bitline development is controlled by the difference in time between asserting the wordline and asserting the sense amplifier enable. The longer

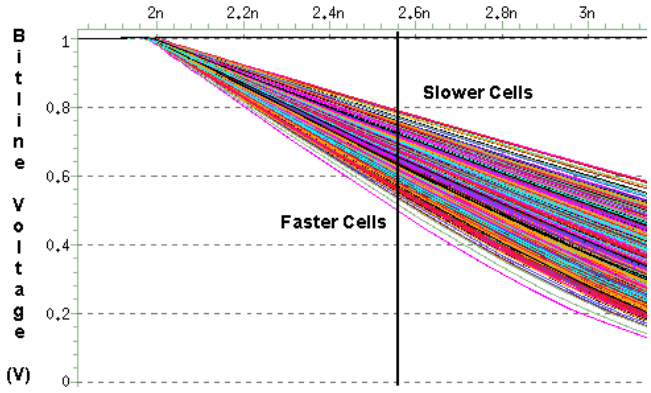

Fig. 4. 1000 SRAM bitline voltages. The bitline is sampled by the sense amp 600 ps after the word line is asserted. The sampling time is represented by the solid line. The SRAM supply voltage is 1.0 V.

the wordline pulse stays high, the lower the bitline voltage falls.

Fig. 4 shows the amount of bitline development for 1,000 SRAM instances considering process variation (the detailed setup is given in Sec. IV). For this design, the sense amp required 200 mV of bitline development for proper operation. In order to for all cells to reach 200 mV of bitline development, *sa_en* has to be asserted 600 ps after the wordline is asserted. As shown in Fig. 4, because the slow cells determine the minimum time to assert *sa_en*, most of the SRAM cells produce larger than required amounts of bitline development. This is a clear waste of energy.

B. Adaptive Voltage Supply

Fig. 5 shows a 4-bit by 4-bit SRAM array that illustrates our adaptive design. Each column of SRAM cells can be assigned to one of four different voltage supplies by enabling one of the power configuration transistors at the top of each column. Note that we are assigning voltage supplies only to the 6T cells, and not to the pre-charge or the word line drivers. By enabling the power configuration transistor that best matches the needs of all of the SRAMs in a column, we are able to tailor the voltage supply to the particular needs of that column.

C. Area Overhead

When designing the power configuration transistor, it is important to consider the current requirements for the column voltage supply. Fig. 2 shows that the pre-charge transistor *prech* pulls up the bitlines. Therefore, the SRAM cell only has to pull down the bitline. From Fig. 1, we can see that V_{ddmem} is used to drive the nmos gate for *PD1* and *PD2*. In the SRAM array, the wordline ensures that only one SRAM cell in a column is active at one time as well. At any given time, the SRAM column supply voltage never drives more than a single nmos gate; it never charges the bitline. Therefore, the power configuration transistor only needs to supply a very small amount of current.

Since the current requirement is so small, the power configuration transistors can be very small and dense. Therefore, four minimum size pmos transistors can be used as the power

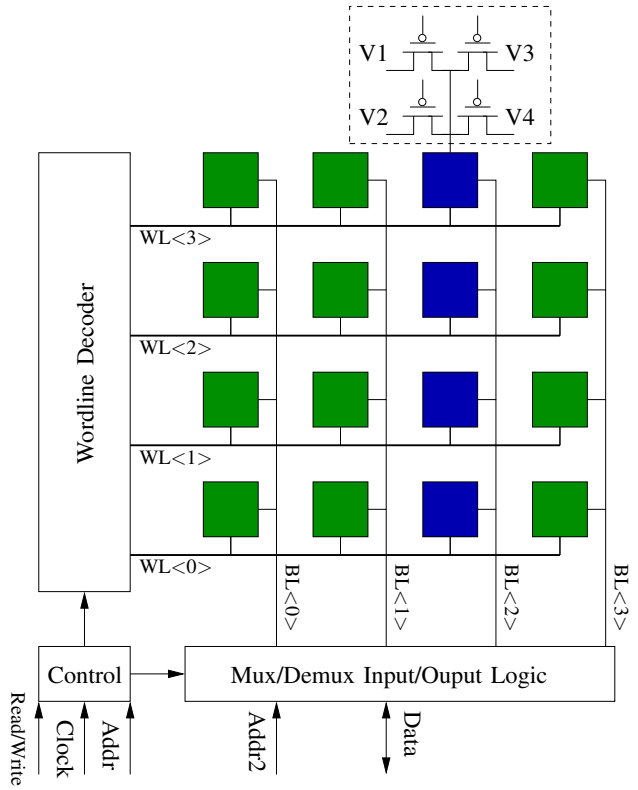

Fig. 5. 16-bit SRAM array. The dotted box at the top of BL<2> is the configurable voltage supply for one column of SRAM cells; the other columns would have a similar structure. V1 through V4 are the four different voltage supplies. By turning on one pmos power transistor, the SRAM voltage for that column can be adjusted. The pmos power transistors are configured using fuses (not shown).

transistors. Then, an electrical fuse can be programmed with the configuration to turn on one of the pmos transistors [4]. In terms of area, the four minimum sized pmos transistors would be 0.12 μm^2. The fuses would use 0.46 μm^2. Compared to a column containing 16 SRAM cells, the area overhead would be about 15%. In terms of the entire memory array, the SRAM columns contribute to about 50% of the total array area. Therefore, the adaptive memory for 16 SRAMs per column would be about 8% of the total memory array size.

In addition to the power transistor and configuration, this adaptive memory also requires four voltage supplies. Current designs usually have a Power Management IC (PMIC) that is already capable of generating multiple voltages. Since very little current is used in the SRAM supply voltages, there is no need for a full voltage grid. The SRAM supply voltages can use signal routing from the PMIC to the memory array. We believe that, similar to scan signals, these signals can be routed at the end of the physical design stage using any available metal layers with only small incremental cost.

D. Voltage Assignment Algorithm

Built-in Self Test (BIST) is the predominant way to test memory. Testing is done at full speed with algorithmic pattern generator and cycle-by-cycle response comparison with pass or fail signature. The BIST engine has a simple

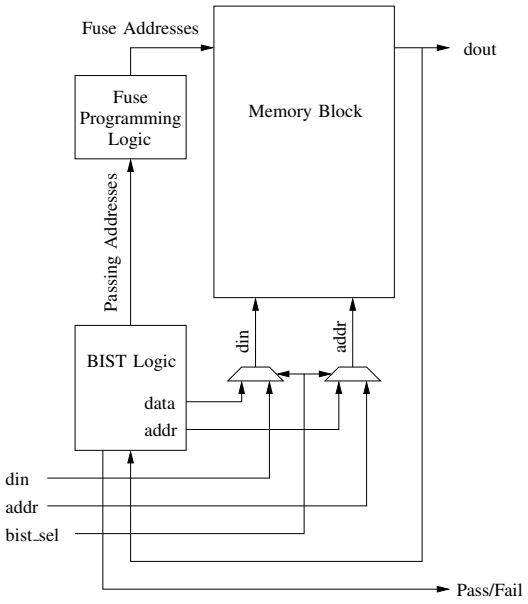

Fig. 6. BIST block diagram.

interface with input, output, address, enable signal, and pass or fail signal. It also keeps track of the failing addresses for use by the failure analysis and redundancy logic.

Our approach, shown in Fig. 6, assumes a BIST engine that has similar functionality; essentially, we propose running BIST multiple times, once for each supply voltage, to determine the minimum supply voltage for a given group of cells. Since BIST is already required for testing, the added overhead for our approach is the wiring from the passing group ids to the fuses and the time it takes to run BIST during the configuration iterations.

The sequence of testing starts by setting all of the SRAM cells to the highest voltage. If a group of cells does not pass testing at the highest voltage, that group is marked as failing because there is nothing more that can be done for that group. If the group passes testing, the second iteration of BIST testing starts by assigning all of the remaining memory groups to the lowest voltage. The ids of the groups that pass testing at the lowest voltage are sent to the fuse programming logic to be assigned to the lowest voltage group. The remaining memory cells that did not pass at the lowest voltage are then tested at the two other voltage levels using this same procedure.

IV. Experiments

A. Setup

We performed HSPICE simulations for the SRAM from Fig. 1 using transistor models from a large foundry. We created 10,000 instances of the SRAM cell—each instantiation had random device parameters, in accordance with the foundry's data.

We measured the resulting amount of bitline development for each instance across a range of supply voltages. With a 1.0 V supply voltage, 13 SRAM instances had less than the required 200 mV of bitline development. The SRAM instance with the weakest bitline development required 1.08 V in order to meet the sense amp requirement. From a power perspective, though, when using a nominal 1.0 V supply voltage, the average SRAM instance produced 110 mV more bitline development than needed. As shown in (4), this 110 mV of excessive bitline development results in 45% more bitline energy than is needed by the sense amp.

TABLE I
Array Energy

	Average Energy (fJ)		
	32 SRAMs per group	16 SRAMs per group	8 SRAMs per group
configurable supply (4)	1480	1422	1373
configurable supply (∞)	1412	1356	1294
fixed supply	1758	1758	1758

Next, we created a 1 KB SRAM memory array by selecting SRAM cells with uniform probability from the set of 10,000 instances. The array had 256 columns, and each column contained 32 SRAM cells. Each 1 bitline column had 20 fF of interconnect plus diffusion capacitance.

After the memory array was constructed, we applied the voltage configuration algorithm from the previous section to the array. We measured the energy for each bitline using (4).

Each column's bitline was pre-charged to 1 V. We then determined the amount of development for each SRAM instance in a particular column. The final voltage of the bitline was the average amount of development for all 32 SRAM instances in a column. This process was repeated for all 256 columns in the array.

B. Results

Table I shows the average array energy for varying supply voltages and varying SRAM configuration group sizes. The first line is our work where each SRAM group can be assigned to four discrete voltage supply values (0.92 V, 0.96 V, 1.00 V, and 1.08 V). The second line is our work where each SRAM group can be assigned to an arbitrary supply voltage; this gives us a bound on the maximum possible energy savings with our approach. The last line of the table is a conventional design that uses a fixed supply voltage of 1.08 V for all of the SRAMs cells. We also tried varying the size of the SRAM configurable group. The first column has all 32 SRAMs in a bitline column connected to the same supply voltage. The second column has 16 SRAMs in a bitline column connected to the same supply voltage. For the 16 SRAMs per group case, each bitline column would need to have two sets of configuration fuses and pmos pass transistors. Similarly, 8 SRAMs per group would need to have four sets of configurations fuses and pmos pass transistors.

C. Key Takeaways

Our adaptive SRAM shows significant advantages compared to a fixed voltage supply design.

- 32 SRAM cells per group with 4 configurable supplies uses 18.8% less energy than a fixed supply design. 16

SRAM cells per group uses 23.6% less energy than a fixed supply, and 8 SRAM cells per group uses 28.0% less energy than a fixed supply. As expected, smaller configuration groups allow for greater energy savings, since the supply voltage is determined by the slowest cell in the group.

- The ability to set group voltages arbitrarily resulted in roughly 5% additional savings compared to 4 configurable supplies. Therefore, the incremental energy savings from trying to use additional supplies would probably be offset by the area overhead associated with more supplies.
- The highest configurable supply voltage could be set to a value larger than the fixed supply voltage since this highest configurable supply is not used to power the entire SRAM array. This could result in greater yield by enabling correct operation from even very slow SRAM cells that are greater than 5σ away from nominal.
- While we only presented data for active energy, reducing the supply voltage should also allow corresponding savings in leakage energy.

Overall, an adaptive SRAM is effective in reducing SRAM array energy.

V. Present Approaches to Robust SRAM and Cache Design

In addition to finding the right balance for transistor sizing and cell area to achieve the design target, many proposals exist to address the minimum supply voltage requirement and parametric yield loss due to SRAM failure. These proposals can be generally characterized into the following four categories: (1) SRAM cell modification, (2) voltage islands, (3) body and well biasing, and (4) circuit techniques.

As described in Section II, the complex interactions between the SRAM cell parameters that affect both reading and writing may create optimization difficulties. Voltage islands have been proposed as a method by which to decouple the cache memory voltage supply from the logic voltage supply, resulting in lower overall voltage supply levels [5][6]. Since voltage has a quadratic relationship to power, any voltage supply reduction has a large effect on both active and leakage power. Voltage islands also enable additional power-saving operating modes, like standby and sleep. The limitation of voltage islands is that they require multiple power supplies and additional physical design resources. The disadvantage of this approach is that the memory supply has to support the weakest cell in the entire SRAM array.

Mukhopadhyay *et al.* [7] used nmos body bias and pmos well bias to shift the threshold voltage higher or lower based on the overall speed of a particular die. A ring oscillator is used to measure the speed of a particular die. Then, a corresponding amount of biasing is applied. The main purpose of Mukhopadhyay's work was to apply body bias to reduce the number of parametric failures due to random doping fluctuation. Lowering V_T affects read and hold failures, while raising V_T affects access and write failures. Because this approach shifts all nmos transistor threshold voltages the same way, it is not very effective in addressing SNM.

Yabuuchi *et al.* [8] propose special read and write assist circuits to improve SRAM operation. During SRAM read, voltage dividers are used to reduce the wordline voltage, which increases static noise margin. The wordline voltage is reduced by adding contention, which increases power. During SRAM write, dummy bitline capacitance is added to reduce V_{ddmem}, which also increases the write margin. Also, the V_{ddmem} reduction is a result of charge sharing between the V_{ddmem} column and the dummy metal capacitance. In terms of power, the dummy metal capacitance needs to be discharged after each access. Besides the power considerations with Yabuuchi's approach, process variation makes it difficult to balance capacitance in order to reliably reduce V_{ddmem}.

Yamaoka *et al.* [3] increase the write margin by floating V_{ddmem} during writes. This approach works well for low frequencies, but it is limited to only helping the write margin due to the fact that the V_{ddmem} capacitance is comparably big and the discharge path has to go through the pull up transistor PU of the 6T cell, which is a very small device.

VI. Conclusion

In conclusion, we proposed a SRAM architecture that allows us to compensate for variability by powering up devices that, after manufacturing, are slower than designed. Our approach avoids the need for guard banding in the design phase and leads to savings of up to 28% in bitline energy with moderate area overhead.

In the future, we will investigate more efficient ways to implement supply voltage configuration. For instance, floating gate power transistors might allow us to merge the power transistor with the configuration bit. We will also apply the adaptivity principle to overcoming variation in other building blocks, e.g., clock trees, buses, and combinatorial logic.

References

[1] S. Rusu, S. Tam, H. Muljono, D. Ayers, J. Chang, B. Cherkauer, J. Stinson, J. Benoit, R. Varada, J. Leung, *et al.*, "A 65-nm Dual-Core Multithreaded Xeon Processor With 16-MB L3 Cache," *IEEE Journal of Solid-State Circuits*, 2007.

[2] J. Massey, I. Microeletronics, and V. Essex Junction, "NBTI: What we know and what we need to know—A tutorial addressing the current understanding and challenges for the future," *Integrated Reliability Workshop Final Report, 2004 IEEE International*, pp. 199–211, 2004.

[3] M. Yamaoka, N. Maeda, Y. Shinozaki, Y. Shimazaki, K. Nii, S. Shimada, K. Yanagisawa, and T. Kawahara, "90-nm Process-Variation Adaptive Embedded SRAM Modules With Power-Line-Floating Write Technique," *IEEE Journal of Solid-State Circuits*, 2006.

[4] A. Hoefler, C. Henson, C.-N. Li, and D.-G. Lin, "Analysis of a Novel Electrically Programmable Active Fuse for Advanced CMOS SOI One-Time Programmable Memory Applications," *Solid-State Device Research Conference*, 2006.

[5] P. Royannez, H. Dahan, F. Wagner, M. Streeter, M. Bouetel, L. Blasquez, J. Clasen, H. Semino, G. Scott, D. Pitts, *et al.*, "90nm low leakage SoC design techniques for wireless applications," *IEEE International Solid-State Circuits Conference*, 2005.

[6] K. Zhang, U. Bhattacharya, Z. Chen, F. Hamzaoglu, D. Murray, N. Vallepalli, Y. Wang, B. Zheng, and M. Bohr, "A 3-GHz 70mb SRAM in 65nm CMOS technology with integrated column-based dynamic power supply," *IEEE International Solid-State Circuits Conference*, 2005.

[7] S. Mukhopadhyay, K. Kim, H. Mahmoodi, and K. Roy, "Design of a Process Variation Tolerant Self-Repairing SRAM for Yield Enhancement in Nanoscaled CMOS," *IEEE Journal of Solid-State Circuits*, 2007.

[8] M. Yabuuchi, K. Nii, Y. Tsukamoto, S. Ohbayashi, S. Imaoka, H. Makino, Y. Yamagami, S. Ishikura, T. Terano, T. Oashi, *et al.*, "A 45nm Low-Standby-Power Embedded SRAM with Improved Immunity Against Process and Temperature Variations," *IEEE International Solid-State Circuits Conference*, 2007.

APPENDIX

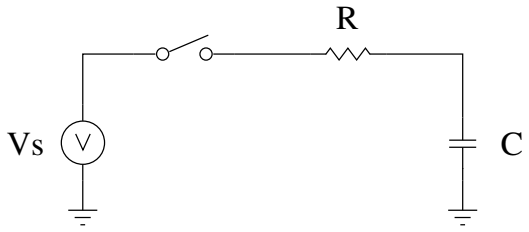

Fig. 7. RC circuit schematic used to derive the energy expended for precharing and evaluting the bitlines.

Fig. 7 show the schematic that is used to derive the total energy expended in pre-charging and evaluating the bitlines. Since only one bitline is evaluated during a cycle, C is the total capacitance of one bitline. The resistance R is the equivalent resistance of the additional circuit elements that appear between the voltage supply and the bitline. Since the bitline does not fully discharge to 0 V, the energy is calculated for a bitline that starts at voltage V_1 and rises to voltage V_2. The total energy is derived by adding the energy through the capacitor with the energy through the resistor using the following equations:

$$v_{cap}(t) = (V_2 - V_1)\left(1 - e^{-t/RC}\right) + V_1 \tag{5}$$

$$i_{cap}(t) = \frac{(V_2 - V_1)}{R}\left(e^{-t/RC}\right) \tag{6}$$

$$\text{Energy}_{cap} = \int_0^\infty v_{cap}(t)\, i_{cap}(t)\, dt = \frac{1}{2} C\, (V_2 - V_1)\, (V_2 + V_1) \tag{7}$$

Notice that (5) and (6) are the normal capacitor equations except that the voltage range has been shifted to V_2 through V_1 (instead of going all the way to 0). When solving for the energy of the resistor, the voltage supply is set to V_2. Additionally, $i_{res}(t) = i_{cap}(t)$.

$$v_{res}(t) = V_2 - v_{cap}(t) \tag{8}$$

$$\text{Energy}_{res} = \int_0^\infty v_{res}(t)\, i_{res}(t)\, dt = \frac{1}{2} C\, (V_1 - V_2)^2 \tag{9}$$

It is interesting to note that the energy through the resistor is independant of the size of the resistor. And now the total energy:

$$\text{Energy}_{total} = \text{Energy}_{res} + \text{Energy}_{cap} = C\, (V_2 - V_1)\, V_2 \tag{10}$$

Lastly, for this work V_2 is V_{dd} and $(V_2 - V_1)$ is V_{bl}, which results in:

$$\text{Energy}_{bitline} = C\, V_{bl}\, V_{dd} \tag{11}$$