A Low-Power SRAM for Viterbi Decoder in Wireless Communication

Shin-Pao Cheng and Shi-Yu Huang *Member*, IEEE

Abstract — *In a consumer electronic device, the embedded memories often consume a major portion of the total power. In this paper, we present a low-power SRAM design for a Viterbi decoder, featuring a quiet-bitline architecture with two techniques. Firstly, we use a one-side driving scheme for the write operation to prevent the excessive full-swing charging on the bitlines. Secondly, we use a precharge-free pulling scheme for the read operation so as to keep all bitlines at low voltages at all times. Silicon results shows that such architecture can lead to a significant 70% power reduction over a self-designed baseline low-power SRAM macro.*

Index Terms — SRAM, embedded memory, low-power[1]

I. INTRODUCTION

Static Random Access Memory (SRAM) has found its way into almost every IC as an embedded component. In a consumer electronic device, the embedded SRAM is often not operated to its maximum speed. As a result, it is very likely that one can sacrifice the memory's performance slightly without affecting the system's throughput, while saving significant power dissipation. This is the basic strategy of this work – by giving up 27% speed degradation we gain 76% power savings as compared to the state-of-the-art lower-power SRAM design.

For example, in a Viterbi decoder, which has been popularly used as the channel decoding scheme to guard the signal's integrity over a wireless channel from the disturbance of noise, the power consumption spent by the SRAM unit accounts for up to 70% of the total power [5]. As a result, minimizing the power consumption of the SRAM is a critical issue in a low-power Viterbi design.

Traditionally, an SRAM macro is mainly formed by an array of cells consisting of four or six transistors and a number of periphery circuits such as row decoder, column decoder, sense amplifier, write buffer, etc. Information access from/to this macro consume power in both dynamic and static ways. The dynamic power involving the switching of signals is consumed in operations such as wordline decoding, bitline charging/discharging, sense amplification, etc. The static power is consumed when there is a direct path from V_{DD} to ground during memory access.

The power consumption of the address decoders and the sense amplifiers can usually be tamed quite satisfactorily. In a low-power SRAM design, the question is often how to minimize the excessive power due to the following sources: (1) wordline switching power, (2) bitline switching power, and (3) static power [7].

Divided wordline or divided bitline architecture [3][6][9] partitions the entire array into smaller pieces. The access of the memory is thereby confined to a smaller sub-array and the total capacitance that needs to be switched during a memory operation is thus reduced to save power. Nevertheless, divided architecture also requires significant area overhead for the extra decoding and controlling circuitry. Wordline pulse control [8] is an important scheme aimed at reducing the excessive static current flowing through the cells that are open for access. To achieve this goal, the wordline is only activated (i.e., raised to high voltage) for a minimal period of time. Also, current-mode access schemes [10] have reportedly achieved both high speed and low power consumption because of its instant read operation and low bitline swing nature. However, current-mode SRAM tends to be less reliable as compared to the conventional voltage-mode counterparts because of its higher sensitivity to the noise. For sub-90 nanometer technologies, the leakage power poses another threat since it sinks significant power in the standby mode, and thus affecting a device's battery lifetime. Numerous techniques like [1][4] have been proposed for this issue.

In this paper, we explore low-cost low-power architecture using *quiet bitlines*. By *quiet*, we mean that the voltages of the bitlines stay as low as possible at all times. The immediate reward is that all charging/discharging power associated with the bitlines can be eliminated dramatically. To achieve this goal, both the *write* and *read* operations need to be modified. For the *write* operation, a *one-side driving scheme* is devised in such a way that only a strong '0' signal is forced into the cell being accessed from one side, while leaving the other side floating. For the *read* operation, we use a *pulling scheme* that operates in four stages: bitline equalization, wordline activation, and bitline pulling, and finally sense amplification. Unlike conventional ways, we rely on low-power wide-input-range sense amplifier [2] to abandon the precharge operation to further minimize the energy loss on the bitlines. A 1k-bit SRAM macro fabricated in 0.18μm TSMC technology has validated that this scheme indeed has potential to save the power up to 70%.

II. PRELIMINARIES

In this section, we review the basic structure and operations of a conventional 6-transistor SRAM.

[1] This work was supported in part by Taiwan's National Science Council, under Grant No. NSC 95-2220-E-007-040.

S.-P. Cheng is with Electrical Engineering Department, National Tsing-Hua University, HsinChu, 30013, Taiwan, (e-mail: spcheng@larc.ee.nthu.edu.tw).

S.-Y. Huang is with Electrical Engineering Department, National Tsing-Hua University, HsinChu, 30013, Taiwan, (e-mail: syhuang@ee.nthu.edu.tw).

Contributed Paper
Manuscript received November 20, 2007

0098 3063/08/$20.00 © 2008 IEEE

A. Basic Column Structure

The structure of one column – including a representative cell, bitline conditioning circuitry, and write buffer – is shown in Fig. 1. Inside the cell, there are two inverters that form a cross-coupled latch for storing one-bit complementary information as *Q* and *Q-bar*. The two nMOS transistors of the inverter pair are called *n-drivers*, while the two pMOS transistors *p-drivers*. Also, it takes two nMOS transistors, called *access transistors*, to guard the passage of this latch to the bitline pair, *bitline* and *bitline-bar*.

In the bitline conditioning circuitry, it is common that there are two *precharge transistors* and an *equalizer transistor*, denoted as *EQ*. The precharge transistors connect the bitline and bitline-bar to V_{DD} at all times. While the memory macro is idle, these two transistors will set the bitline pair to their default voltage levels. Regarding the equalizer, it is mainly used to balance the voltage levels at bitline and bitline-bar and resolve any mismatched initial conditions before the read operation.

In the write buffer, a simple logic circuit takes as input the data bit and decides which strong *write drivers* (i.e., transistors with large sizes) should be turned on to force a complementary signal on the bitline pair. Here, the *write drivers* are multiple times stronger than the precharging transistors and thus will dominate the final voltage of the bitline if fighting exists.

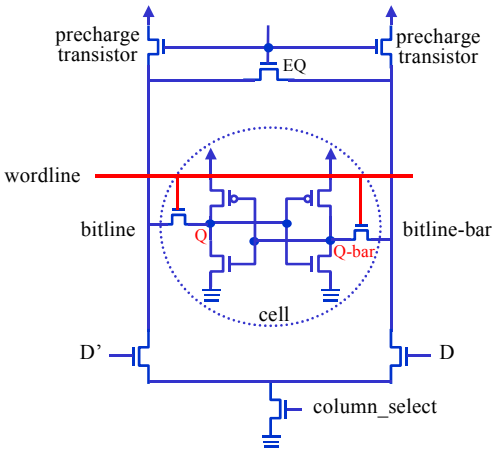

Fig. 1: Traditional 6-T SRAM column.

B. Basic Read Operation

When a read operation is issued, the memory macro will go through the following steps:

- (1) Row decoding: The row address is decoded to activate a selected wordline. For low-power purpose, two-stage NAND type decoder is often desired at moderate sacrifice of speed.
- (2) Bitline driving: After the wordline has been turned on, the target cell connects to the bitline and bitline-bar. The so-called *cell current* through an *n-driver* of the target cell (as indicated in Fig. 2) will discharge the voltage of either bitline or bitline-bar progressively. The process is normally complete once 100~200 mV difference has been established between the bitline and bitline-bar. During this bitline driving step, a direct current is formed from V_{DD} through a precharge transistor and the *n-driver* to the ground. Aggressive wordline pulse control will de-assert the wordline at the end of this step to suppress this static current.
- (3) Sensing: The sense amplifier is turned on to amplify the small difference voltage at the bitline pair into full-swing logic signals. There could be two stages of sense amplifiers performed in cascade to accelerate the process.
- (4) Precharging: At the end of the read operation, all bitline pairs will return to their normal default conditions through the precharge transistors and the equalizer and get ready for the next operation.

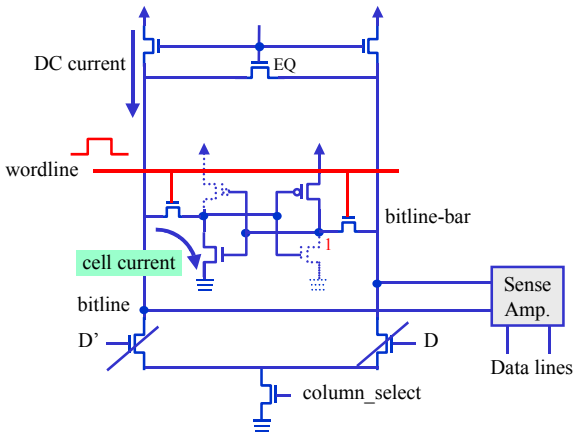

Fig. 2: Bitline discharging for the READ operation.

III. Bad Pixel Detection and Repair

A. Basic Write Operation

When a write operation is issued, the memory macro will go through the following steps:

- (1.1) Row decoding: Similar to the read operation, the row address decoding starts immediately to activate the selective wordline.
- (1.2) Bitline driving: For a write operation, this bitline driving conducts simultaneously with the row address decoding by turning on proper write buffer. After this step, the bitline pair will be forced into full-swing logic level.
- (2) Cell flipping: If the value of the stored bit in the target cell is the opposite to the value being written, then the cell flipping process will take place. Normally, the transistors in a cell are sized in a way that the cell only reacts to the '0' signal side of the bitline pair. For example, let us assume that the originally stored information in the target cell is {Q='1', Q-bar='0'}. After the access transistors have turned on, the strong '0' signal on the bitline will overwhelm the stored Q bit '1' and make it become '0'. Once this over-writing is successful on one side, internal latch will take over to force the cell flipping and finally make Q-bar to change from '0' to '1' as well.
- (3) Precharging: At the end of the write operation, all bitline pairs return to their normal default conditions

through the precharge transistors and the equalizer and get ready for the next operation just like the read operation.

In is worth mentioning that the above column circuits are simplified versions. In practice, a number of columns (e.g., 4 or 8, or even 16) could share the same read-write circuitry (such as the write buffer and the sense amplifier) via a multiplexer controlled by the output of the column decoder.

IV. Quiet-Bitline Architecture

In this section, we will first analyze the power dissipation components and then propose our quiet-bitline architecture.

A. Overall Strategy

The power of an SRAM macro is normally consumed in the row and column decoding, wordline driving, bitline charging, sense amplification, output driving, and the static current during the read or write operations.

It is known that the power dissipated on the address decoding and wordline driving can be controlled to a small amount if NAND-type decoders are used. Also, the power dissipation of the sense amplifiers can be tamed by using dynamic latch-based circuits [1]. Therefore, the dynamic power consumed in an SRAM macro is mainly on the bitline charging. This component, scaling with the number of bitlines, is the major target in our power optimization process.

For a read operation, the bitline charging power (consumed in the precharging step) can be reduced quite significantly if the aggressive wordline pulse control is used to limit the bitline swing to around 100~200 mV. However, all bitlines are usually forced to have full swings during a write operation, mainly to enable a quick cell flipping.

Our strategy for alleviating the excessive power on bitlines consists of two major tactics. First of all, we investigate a *one-side driving* scheme in which only strong '0' signal is forced into a bitline or bitline-bar for the write operation, while leaving the other side floating. Secondly, we *remove the precharging* for both read and write operations. The combination of these two tactics leads to a design with quiet bitlines. This strategy may cause speed penalty to some degree, however, the significant power reduction thereby achieved could more than compensate the side effects.

B. Proposed Read Operation

The proposed read operation now differs from the previously mentioned read operation in two aspects. (1) There is no bitline precharging any more. Consequently, the initial voltages of the bitline and bitline-bar before the read operation could be more diverse. Typically, the bitline pair is first equalized to a low voltage ranging from 0V to a few hundred mV as the starting condition. (2) Without the precharge transistors, the cells are the only drivers to the bitline pair after the wordline is activated. Normally, we rely on the *p-driver* to pull up the voltage of one side of the bitline pair. The voltage of the other side could remain unchanged when the initial voltage is low, or could become lower due to the discharging by the *n-driver*. In either way, there is one major advantage – there is no static power any more. In order to support such a precharge-free read operation, the sense amplifier that takes the difference voltage of the bitline and bitline-bar should be able to operate over a wide of common-mode input signal range, e.g., from 0V to 500mV.

C. Proposed Write Operation

The proposed write operation is similar to the basic write operation in terms of the bitline driving. The write buffer is still strong enough to force a value '0' into one side of the target cell. However, the cell flipping will proceed in a slightly different way. To be more specific, the cell flipping can be divided into two sub-steps:

- (Weak pulling): The storage node connected to the floating bitline or bitline-bar only rise slowly when the wordline is still activated. This is mainly because of the heavy loading of the bitline or bitline-bar and the relatively weak pulling strength of the *p-driver*.
- (Quick latching): Right after the wordline is de-activated, the target cell detaches the bitline pair and behaves like a static latch. Although the difference of Q and Q-bar at the beginning is relatively small, it will quick amplify them into complementary logic signal and thus complete a write operation.

In the conventional design in which the bitlines are precharged to high by default, the second sub-step, quick latching, is insignificant, because the Q and Q-bar may have reached their stable values even when the wordline is still activated.

Example 1: (New write operation) Fig. 3 shows our column architecture and simulation waveforms of signals {D, bitline, Q}. We assume that the data to be written to the target cell is D = 1, forcing a strong '0' signal into Q-bar through bitline-bar quickly, whiling making bitline floating. Once the stored signal Q-bar switches from '1' to '0', its complementary signal Q will rise only slowly as in the weak pulling stage. In the waveform of signal Q, this stage lasts from node A to node B. As the signal of D goes low, the signal Q rise quickly as in the quick latching stage, as indicated from node B to node C.

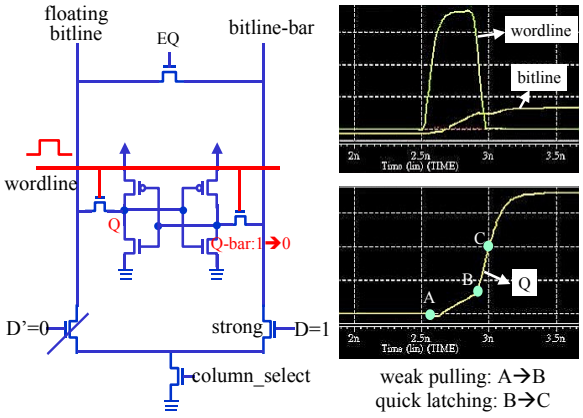

Fig. 3: A WRITE example.

On the average, we observed that such a one-side '0'-driving scheme reduces the bitline voltage swing to only 250mV – a 14% value of the power supply voltage 1.8V. Moreover, the power savings are beyond that. The extra direct current flowing through the precharge transistor, access transistor, and then to the *n-driver* has also been eliminated.

D. Worst-Case Scenarios

In our architecture, the read or write operations are both conducted with different initial conditions on the bitlines. Derivation of the worst-case scenarios is thus important in order to ensure the circuit's functionality.

To gauge the worst-case scenario, we investigate how high the voltages of bitlines could become under a sequence of 15 consecutive write '1' operations. The voltage waveforms of the bitline and bitline-bar are shown in Fig. 4. It can be seen that each write operation pumps certain charge into the floating bitline by the *p-driver* and cause its voltage to rise a few hundred mV. However, as the consecutive write operations proceed, the charge pumping strength gets weaker and weaker and finally the bitline voltage saturates at around 900 mV. This waveform indicates that the largest voltage a bitline or bitline-bar can get is around 900 mV. Under such a hostile situation, we apply a read and a write operation respectively to make sure that it can still function correctly. In the upper figure, we apply a read '0' operation from another cell in the same column. In the lower figure, we apply a write '0' operation. Both cases show correct response at the bitline pairs.

On the other hand, the read operation does not unilaterally increase charge on the floating bitline pair. A cell essentially pumps charges on one side of the bitline pair while discharges on the other. As a consequence, a sequence of consecutive read operation does not create hostile situation on the bitlines.

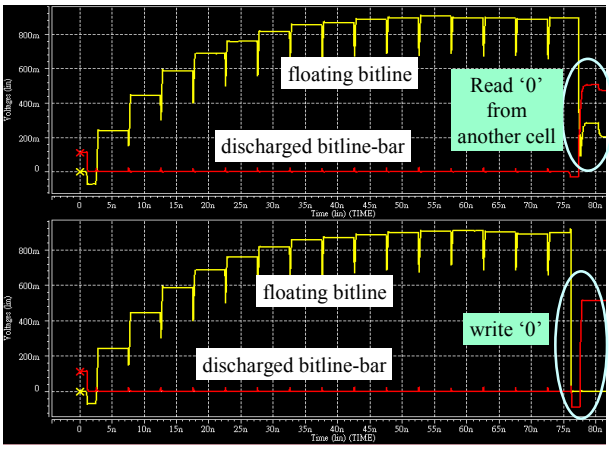

Fig. 4: Effects of 15 consecutive WRITE operations on bitlines.

V. Experimental Results

To evaluate the effectiveness, we have implemented the proposed architecture as an SRAM macro with *1K bits*. This macro is required in our in-house low-power Viterbi decoder as the so-called survivor path memory. This memory macro was taped out separately using *TSMC 1P6M 0.18μm CMOS process* for validating its power advantage.For comparison, we also implemented a conventional SRAM macro with the same configuration using the common low-power techniques, including a two-stage NAND-decoder, the wordline pulse control, and low-power sense amplifiers.

Table 1 lists the SPICE simulation results of these two versions. The simulation is performed under 1.8V supply voltage, 25^0C temperature, and a 200MHz clock signal. The meaning of each column in Table 1 is as follows:

- Read power: The average power for performing a read operation.
- Write power: The average power for performing a write operation.
- Average power: calculated by (read power + write power) / 2.
- Access Time: The time required to complete a read or write operation.

It can be seen that our quiet-bitline architecture consumes only 2.4mW on the average, which is only 24% of the power consumption of the reference design. That is, we have achieved 76% reduction. It is also notable that our scheme reduces the power consumption of the write operation (-86%) more than that of the read power (-61%). The access time of our design is 3.02ns, which is indeed longer than that of the reference design, 2.38ns. However, this macro is used in our Viterbi decoder designed to operate at 200MHz. The access time of the new SRAM macro, 3.02ns, is still well below the clock cycle time requirement of 5ns. This implies that such an access time increase does not actually cause any system performance loss in our application.

Table 1: Post-layout simulation results.

Type	Read Power (mW) @200MHz	Write Power (mW) @200MHz	Average Power (mW) @200MHz	Access Time (ns)
Ref. Design	7.74	12.24	9.99	2.38
Our Design	3.04	1.75	2.40	3.02
Comparison	-61%	-86%	**-76%**	**+27%**

The test chip has been fabricated successfully with the die photo shown in Fig. 5. Due to the wiring effects across the load board and bounding wire, our chip can only be tested up to 50MHz in our test environment. Our chips function correctly. The waveform of the data output pin logged by our tester is shown in Fig. 6. It can be seen that the data output rises to high after a two-operation sequence - WRITE '1' followed by READ '1' from the same cell, and it correctly goes low after another two-operation sequence - WRITE '0' followed by READ '0'.

We measure the average power dissipation of six fabricated chips operated at 50MHz under a common March test algorithm, going through the following stages:

- (1) WRITE every cell in the array to '0'.

- (2) READ every cell and check if all are '0'.
- (3) WRITE every cell in the array to '1'.
- (4) READ every cell and check if all are '1'.
- (5) Scanning the first cell towards the last cell in a row-major order. At each cell, we perform {READ(1), WRITE(0), READ(0)} operations.
- (6) Scanning backward from the last cell towards the first cell in a row-major order. At each cell, we perform {READ(0), WRITE(1), READ(1)}.

As shown in Table 2, the average of the measured power consumption over six fabricated chips is about 977 μW, which is very close to the data obtained by post-layout simulation, 932 μW. Since the power consumption of our reference design by post-layout simulation is 3456 μW, we thus draw a conclusion that our chips achieve (3456 – 977) / 3456 * 100% = 70.6% power savings.

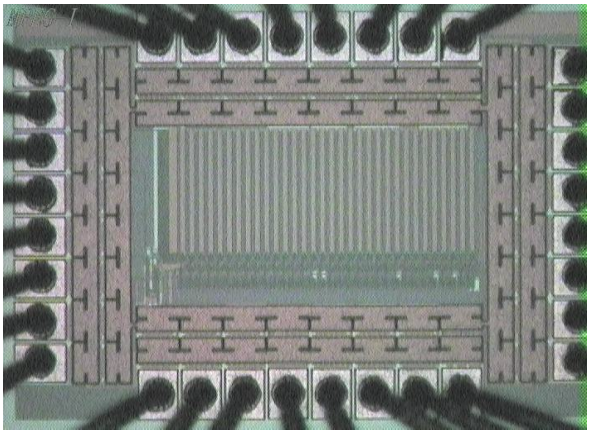

Fig. 5: Die photo of our test chip.

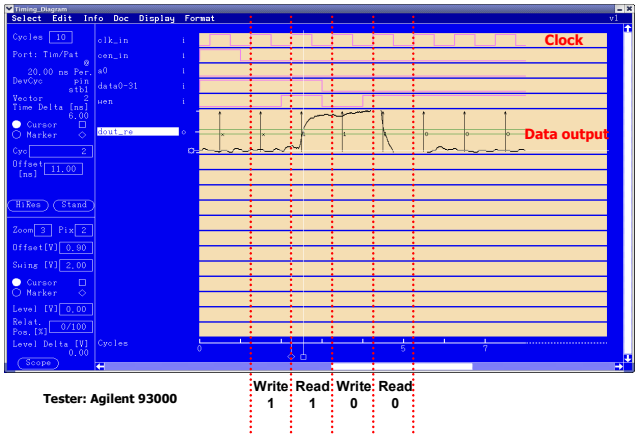

Fig. 6: Data output waveform observed by our tester.

VI. Conclusion

We present in this paper a low-power SRAM macro of a Viterbi decoder. In this Viterbi decoder design, the SRAM macro is the power hungry part, but not the performance bottleneck and therefore a low-power technique that degrades the SRAM's speed slightly does not actually degrade the system's performance at all. Our low-power strategy aims to keep bitlines at low voltages at all times. We achieved this goal by two schemes – the precharge-free pulling scheme for the read operation and the one-side driving scheme for the write operation. This strategy has been validated in silicon. The results show that up to 70% power reduction is indeed possible.

Table 2: Comparison of measured power and simulated power.

Chip or Design		Average Power (μW) @50MHz	Comparison
Reference Design		3456	100%
Our Quite-Bitline Design		932	**27%**
Fabricated Chips	**Chip #1**	1037	30.0%
	Chip #2	1051	30.4%
	Chip #3	1003	29.0%
	Chip #4	1040	30.1%
	Chip #5	1017	29.4%
	Chip #6	977	28.3%
	Average	**1021**	**29.5%**

Acknowledgement

The authors appreciate the kind support from *National Chip Implementation Center* of National Applied Research Lab., HsinChu, Taiwan, in the fabrication of the test chips.

References

[1] R. E. Aly and M. A. Bayourni, "Low-Power Cache Design Using 7T SRAM Cell," *IEEE Trans. on Circuits and Systems II: Express Brief*, Vol. 54, No. 4, pp. 318-322, April 2007.

[2] L. Benini, G. De Micheli, and E. Macii, "Designing Low-Power Circuits: Practical Recipes," *IEEE Circuits and Systems Magazine*, Vol. 1, No. 1, pp. 6-25, 2001.

[3] J. S. Caravella, "A 0.9V, 4K SRAM for Embedded Applications," *Proc. of Custom Integrated Circuit Conf.*, pp.119-122, May 1996.

[4] S. Cseryeny, L. Sumanen, J.-M. Masgonty, and C. Piguet, "Locally Switched and Limited Source-Body Bias and Other Leakage Reduction Techniques for a Low-Power Embedded SRAM," *IEEE Trans. on Circuits and Systems II: Express Brief*, Vol. 52, No. 10, pp. 636-640, Oct. 2005.

[5] Y.-S. Fang, "*Design of a Low-Power Viterbi Decoder with Multi-Stepping Traceback*," MS Thesis, National Tsing-Hua Univ., Taiwan, pp. 1-53, 2005.

[6] K. Itoh et al., "Trends in Low-Power RAM Circuit Technologies", *Proc. of the IEEE*, pp.524-543, April 1995.

[7] M. Margala, "Low-Power SRAM Circuit Design," *Proc. of IEEE Int'l Workshop on Memory Technology Design and Testing*, pp. 115-122, August 1999.

[8] H. Morimura and N. Shibata, "A Step-Down Boosted Wordline Scheme for 1-V Battery Operated Fast SRAMs," *IEEE Journal of Solid-State Circuits*, Vol. 33, No. 8, pp. 1220-1227, August 1998.

[9] M. Ukita et. al., "A Single Bitline Cross-Point Cell Activation (SCPA) Architecture for Ultra Low Power SRAMs," *Digest of Technical Papers, Int'l Solid-State Circuit Conf.*, pp.252-253, February 1994.

[10] J.-S. Wang, W. Tseng, and H.-Y. Li, "Low-Power Embedded SRAM with the Current-Mode Write Technique," *IEEE Journal of Solid-State Circuits*, Vol. 35, No. 1, pp. 119-124, Jan. 2000.

[11] C.-C. Wang, C.-L. Lee, and W.-J. Lin, "A 4-kb Low-Power SRAM Design with Negative Word-Line Scheme," *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, Vol. 54, No. 5, pp. 1069-1076, May 2007.

Shin-Pao Cheng was born in 1975 in Taiwan. He received his B.S. degree from Kun-Shan University of Technology in 2000, and M.S. degree from National Tsing-Hua University in 2005. From 2000 to 2002, he worked as a hardware engineer in Liteon Technology Corp., designing circuits for printers. Also, he was involved in the analysis and testing of ESD and EMI effects during that period of time. Currently, his interests are mainly in low-power digital IC design and low-power memory design.

Shi-Yu Huang received his BS, MS degrees in Electrical Engineering from National Taiwan University in 1988, 1992 and Ph.D. degree in Electrical and Computer Engineering from the University of California, Santa Barbara in 1997, respectively. From 1997 to 1998 he was a software engineer at National Semiconductor Corp., Santa Clara, investigating the System-On-Chip design methodology. From 1998 to 1999, he was with Worldwide Semiconductor Manufacturing Corp., designing the high-speed Built-In Self-Test circuits for memories. He joined the faculty of National Tsing-Hua University, Taiwan, in 1999, where he is currently an Associate Professor. Dr. Huang's research interests are mainly in design automation for VLSI, with an emphasis on power estimation, fault diagnosis, CMOS image sensor design, and nanometer SRAM design. He co-authored a book entitled "Formal Equivalence Checking and Design Debugging" published by Kluwer Academic in 1998.