A Reverse Write Assist Circuit for SRAM Dynamic Write V_{MIN} Tracking using Canary SRAMs

Arijit Banerjee[1], Mahmut E. Sinangil[2], John Poulton[3], C. Thomas Gray[3], Benton H. Calhoun[1]
[1]Dept. of ECE, University of Virginia, Charlottesville, VA 22904, USA
[2]NVIDIA, 2 Technology Park Drive, Floor 3, Westford, MA 01886, USA
[3]NVIDIA, 2700 Meridian Pkwy, Suite 100, Durham, NC 27713, USA
[1]E-mail: {ab9ca, bhc2b}@virginia.edu [2, 3]E-mail: {msinangil, tgray, jpoulton}@nvidia.com

Abstract

SRAMs occupy a large amount of area in modern system on chip circuits. With the growing trend of device scaling in deep sub-micron technologies, the 6T SRAM write operation is more vulnerable than the read operation from a failure standpoint. In order to make the SRAMs operate correctly, we must design them with some guard band above the minimum operating voltage (V_{MIN}) by designing for the worst case. In this paper, we investigate a reverse write assist circuit scheme that enables the tracking of SRAM write V_{MIN} by using canary SRAM bitcells to track dynamic voltage, temperature fluctuations and aging effects. This circuit ultimately allows us to lower the write V_{MIN} below the worst case corner (SF_85C) V_{MIN}, which saves a minimum of 30.7% energy per cycle at the SS_85C, and a maximum of 51.5% energy per cycle at the FS_85C corner.

Keywords

Canary, SRAM, reverse write assist, dynamic V_{MIN}.

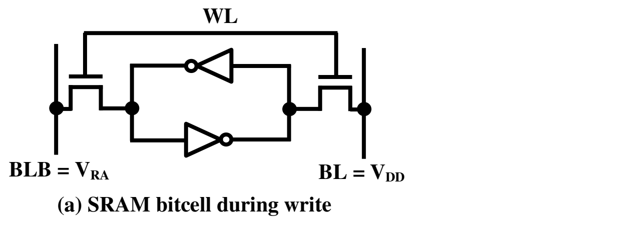
1. Introduction

SRAM energy varies quadratically with the supply voltage, so lowering supply voltage lowers energy. There are various ways to lower SRAM supply voltage to lower energy, such as dynamic voltage and frequency scaling (DVFS), using dual rail design for SRAMs etc. DVFS is widely used in system on chips (SOCs) to lower the energy [1][2][3] by adjusting the supply voltage and frequency from time to time as required, and SOC level design cost for DVFS is excluded from the SRAM design cost. On the other hand, dual rail [4] designs can be used for energy savings and avoiding readability issues by keeping SRAMs on a higher supply; however, this technique is complicated to implement, and increases design cost in SRAMs and area cost for SOCs. In spite of aforementioned voltage lowering techniques, the SRAM minimum operation voltage (V_{MIN}) poses a bottleneck for SRAM voltage scaling. The SRAM V_{MIN} is a function of the operating frequency, and it is hard to predict in a real design. So, we design with voltage and timing guard bands. On the other hand, local and global variation make scaling down SRAM V_{MIN} more challenging than for logic [5][6], and existing research work shows that SRAM write failure will increase during further scaling [6]. One solution for SRAM read and write V_{MIN} improvement is to use assist circuits, such as wordline boosting [7][8][9], negative bitline [7][8][9][10], V_{DD} lowering [8][9], V_{SS} raising [8][9] etc. for write improvement, and wordline under drive [9], partially suppressed wordline [10], V_{DD} boosting [9], negative V_{SS} [9] etc. for read improvement. Assist methods require extra silicon area and power consumption, but can allow for significantly lower SRAM V_{MIN}. With time, SRAM circuits age [11][12][13] like all other circuits, and the V_{MIN} gets higher and higher, which further adds to the margin necessary for the worst case design [14][15][16].

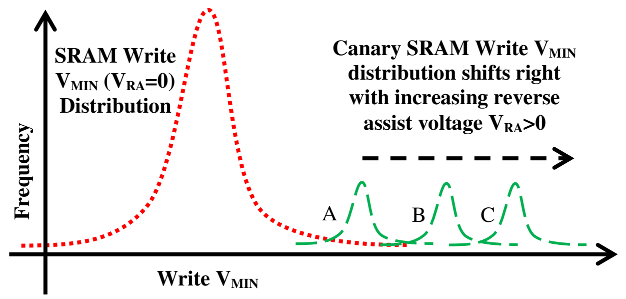
Hence, predicting the V_{MIN} by detecting failures during DVFS can allow corrections to address functional problems. So, a closed loop solution is ideally required to turn off or on assists or to dynamically adjust the assist voltage when required. Closed loop control can also track the effect of voltage and temperature fluctuations. Hence, there is a need to detect read or write failure dynamically. In this paper, we investigate the use of canary cells to detect failure and to track V_{MIN}.

The idea of canary circuits has been studied widely in different fields in circuits [17][18][19]. In SRAMs, the use of canary circuits has been studied by Wang and Calhoun [19][20][21] for predicting the data retention voltage (DRV) during standby, but canaries have not been presented in depth for write or read V_{MIN} tracking. This paper mainly focuses on the study of canary SRAMs for dynamic write V_{MIN} tracking, as device scaling makes a write failure more probable than a read failure [6].

In this paper, Section 2 discusses assists and reverse assists. In Section 3, we discuss the effect of reverse assist in



(a) SRAM bitcell during write



(b) SRAM write V_{MIN} distribution with reverse assist

Figure 1: SRAM write operation using bitline type reverse assist and write V_{MIN} distributions with reverse assist (A, B, C's are canary V_{MIN} distributions).

canary SRAMs. We develop a methodology based on probability theory and use the concept of canary SRAMs to quantify the output metrics in Section 4. Section 5 gives simulation results using our methodology. Section 6 describes the circuit implementation. We propose a canary SRAM architecture using BL type reverse assist and an algorithm to track SRAM V_{MIN} with canary reverse assist in Section 7. Section 8 describes the power and area tradeoffs of this scheme, and we conclude in Section 9.

2. Peripheral assist methods and reverse assists

There are many ways to create canary circuits in SRAMs. One method is to modify the SRAM core bitcell to fail earlier than a population of SRAM bitcells during the read or write operation. We can have built in control in the canary bitcell to tune the canary to change the failure point. However, this type of canary bitcell may not track same as core bitcells over variation. Another option is to use a shorter wordline pulse width modulator circuit for canaries to make the write/read operation more difficult to fail them earlier than the core SRAM cells. In order to get a precisely controlled wordline pulse width, extra wordline delay control circuit is required, which will increase the area overhead in SRAM decoder and may cause abutting problems in layout.

An assist in the SRAM context means an auxiliary circuit that helps improve write-ability [4][7][8][9][10], readability [9][10], or read stability [4]. We define a reverse assist as an auxiliary circuit that degrades the write-ability or readability of an SRAM cell. In this work, we use the same core SRAM bitcell as a canary SRAM, but we apply a reverse assist to degrade the canary SRAM bitcell write-ability.

The advantage of a reverse assist for a canary SRAM is to use the same SRAM core bitcells as canaries to track the core cells better. Also, a user can control the reverse assist with low overhead to fine tune the failure point of the canary SRAM bitcells dynamically.

In the context of this paper, assist or reverse assist will always refer to a bitline (BL) type assist or reverse assist. In core SRAMs during a write without any assist, either BL or BLB is pulled down (Figure 1 (a)) to V_{RA}=0V, while the other node (BLB or BL) is kept at V_{DD}, and then the wordline (WL) is triggered. Usually a BL type assist [7][8][9][10] is used to improve the dynamic write-ability of the SRAM bitcells by pulling the bitline or bitline bar (BLB) node below the ground voltage (V_{SS}). On the other hand, in canary SRAMs, a reverse assist will pull the BL/BLB node to a positive voltage, say for example V_{RA}=0.1V, while the other BLB/BL is kept at V_{DD} as shown in Figure 1 (a). This will degrade the dynamic write-ability of the canary SRAM bitcells.

3. Effect of reverse assist on canary SRAMs and canary design metrics

The ability to write in an SRAM bitcell is called bitcell write-ability. There are two widely used metrics for write-ability as write static noise margin (WSNM) known as the static write-ability metric, and another metric called critical wordline pulse width for write (T_{CRIT}) known as the dynamic write-ability metric for SRAMs. WSNM assumes the wordline pulse width to be infinite which overestimates the static write-ability metric, but T_{CRIT} assumes a realistic finite wordline pulse width as SRAM write operation is a dynamic process. Using a write assist in SRAMs causes the spread of the distribution of T_{CRIT} to decrease and to shift the V_{MIN} to a lower value [8]. Hence, applying a reverse assist to the canary bitcells, relative to the core SRAM cells, will cause the canary write V_{MIN} distribution 'A' to shift to a higher V_{MIN} distribution 'B' or 'C' as shown in the Figure 1 (b). Thus, the V_{MIN} of the canary SRAM bitcells will increase to cause canary failures earlier than the core SRAM bitcells.

Our goal is for the canary SRAMs to start to fail before a single failure in a given number of SRAM bits, say a million bits. Figure 2 shows the simulated dynamic write failure probability (P_{fail}) vs. write V_{MIN} plots for the core SRAM cells and the canary SRAM cells with varying degrees of reverse assist. Here, we use the extracted 6T bitcell netlist with the setup shown in Figure 1 (a) and simulate transient write operation using a commercial 28nm technology with HSPICE. We generate the P_{fail}-V_{MIN} data using an importance sampling algorithm [5][22][23][24]. For the input slews and the WL pulsewidth timings, we use a FO4 delay table data across voltages. We can see that for the same P_{fail}=10^{-10}, the canary SRAM using reverse assist has a higher write V_{MIN} than the core SRAMs without any assist. .

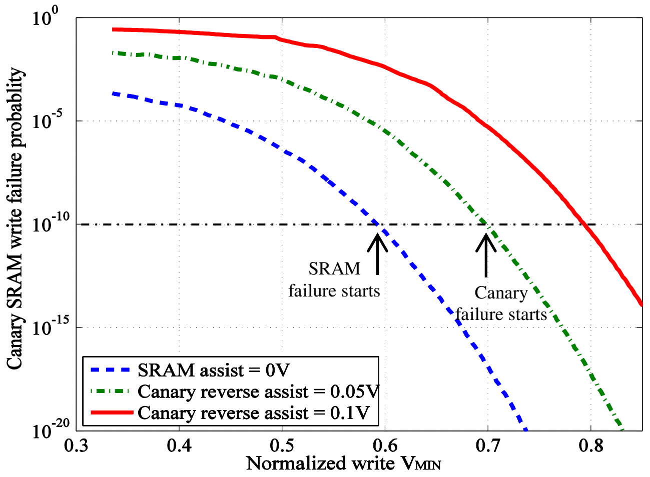


Figure 2: Canary SRAM dynamic write failure probability vs. normalized write V_{MIN}.

Table 1: Input and output design metrics for the canary SRAM design.

Input Metrics	
N	Number of SRAM bits on a chip
Y_{SRAM}	Core SRAM target yield
C	Number of canary SRAM bits
F_{th}	Canary failure threshold condition
V_{RA}	Canary BL type reverse assist voltage
Output Metrics	
P_{fc}	Canary SRAM chip failure probability

Table 1 defines the input and output design metrics for the canary SRAM design. A write failure probability for the core SRAMs corresponds to the number of SRAM bits (N) on a chip with a target yield (Y_{SRAM}). Similarly, tracking dynamic write failure of core SRAM bits requires a certain number of canary bits (C). Other important input knobs are the canary failure threshold condition (F_{th}) and the reverse assist voltage (V_{RA}). F_{th} condition is the number of canary cells allowed to fail before one in N SRAM core bits fails. For example, if a user defines F_{th}=8 for C=32 canaries in a chip, then an action can be taken if 8 canaries fail to write out of 32 canaries. As an action, either assists can be turned on for the core SRAMs or DVFS can be stopped. Also, the amount of degradation of the write-ability in canaries can be controlled by tuning the V_{RA}. The two input metric knobs available post-fabrication to a user are V_{RA} and F_{th} for the canary SRAMs. All other input metrics are set at design time. For the output metric, we define P_{f_c} as *canary SRAM chip failure probability*, which is the probability that the canary SRAM bitcells will be unable to fail earlier than one in N SRAM core bitcells. For example, if P_{f_c}=10^{-6} for a given N=10^7 SRAM bits with Y_{SRAM}=99%, C=32, and F_{th}=1, then the canary chip failure probability will denote that in one in a million 10Mb chips, the 32 canary cells will *not* experience a single bit failure prior to the first failure from ten million SRAM bits on the chip.

Now, for the core SRAMs, if the bit failure probability in a write operation is given by P_f, then the core SRAM bitcell success probability is given by $P = (1 - P_f)$, the SRAM chip success probability is given by $P_{chip} = P^N$, and the SRAM chip failure probability is given by $P_{f_{chip}} = (1 - P_{chip})$. Now, the SRAM chip yield for 'k' or less chip failures out of 'J' chips can be given by:

$$Y_{SRAM_{(J,k)}} = \sum_{i=0}^{i=k} P_{chip}^{(J-i)} * (1 - P_{chip})^i * \binom{J}{i} \qquad (1)$$

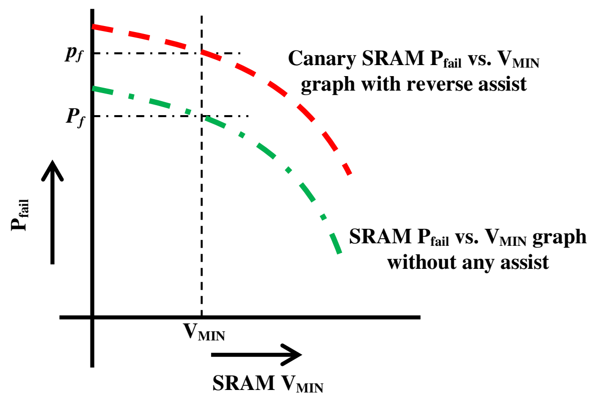
From (1), for a given value of Y_{SRAM} and N, we can calculate the corresponding SRAM bit failure probability P_f for write failures. Similarly, if the canary bit failure probability is given by p_f, then the probability of canary bits being unable to fail earlier than a given number of core SRAM bits with C number of canary bitcells with F_{th}=k condition can be given by:

$$P_{f_c} = \sum_{i=0}^{i=k} p_f^i * (1 - p_f)^{C-i} * \binom{C}{i} \qquad (2)$$

Hence, (1) and (2) relate the input metrics N, Y_{SRAM}, P_f, C, F_{th}, and p_f to canary chip failure probability P_{f_c}, which is our final output metric for observation.

4. Calculation Methodology for Canary Chip Failure Probability

Figure 3 shows the methodology to calculate the canary bit failure probability p_f using SRAM bit failure probability P_f. The plot shows P_{fail} vs. V_{MIN} for the core SRAM bitcells without any assist, and canary SRAM P_{fail} vs. V_{MIN} with reverse assist. For a given value of Y_{SRAM} and N, the research questions we are addressing here are: what is the



p_f
P_f
P_{fail}
Canary SRAM P_{fail} vs. V_{MIN} graph with reverse assist
SRAM P_{fail} vs. V_{MIN} graph without any assist
V_{MIN}
SRAM V_{MIN}

Figure 3: Methodology to calculate canary chip failure probability.

corresponding SRAM bit failure probability P_f, and what should be the corresponding bit failure probability p_f for the canary SRAM bits? We also want to know how the input metric C influences the canary chip failure probability P_{f_c}. In order to connect the two equations (1) and (2), we use the setup mentioned in Section 3 to get the P_{fail} vs. V_{MIN} data for different reverse assist voltages, which represents the data for the canary bitcells. We also got the P_{fail} vs. V_{MIN} data for the core bitcells without any assist using the same simulation setup. First of all, we calculate the corresponding P_{fail} P_f for the core bitcells using (1), and then we calculate the corresponding V_{MIN} for the core bitcells using the P_{fail} vs. V_{MIN} simulated data (Figure 3). After that, we calculate the corresponding P_{fail} p_f for the canary bitcells with same V_{MIN} obtained from the canary SRAM P_{fail} vs. V_{MIN} simulated data, which is shown in Figure 3. Finally, we plug the value of p_f into (2), and calculate the corresponding canary chip failure probability P_{f_c}.

5. Simulation Results for Canary SRAM Chip Failure Probability

In order to get the trends of the input metric vs. the output metric variation, we use the calculation method described in Section 4 and calculate the output metric for the reverse assist voltages of V_{RA}=0V, 0.05V, 0.1V, 0.15V and 0.2V. Figure 4 shows that same canary chip failure

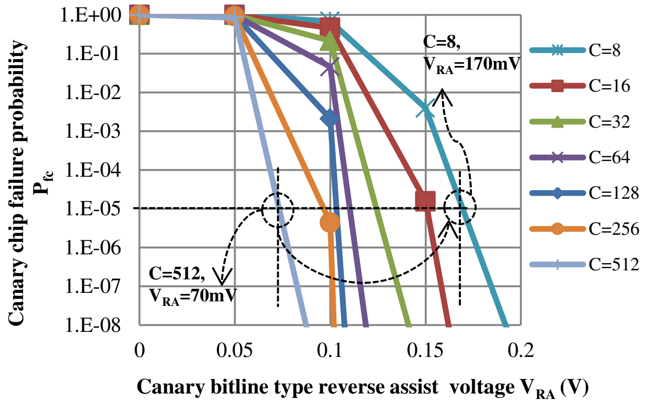


Figure 4: Canary chip failure probability vs. reverse assist voltage for 1 million SRAM bitcells with 95% yield @ TT_85C.

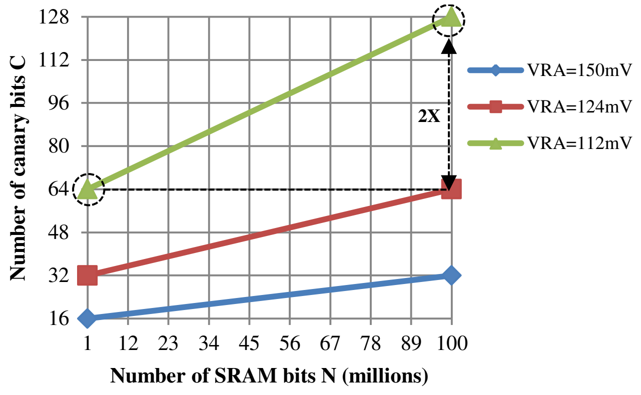


Figure 5: Trend for C vs. N with 95% SRAM yield at constant P_{f_c}=10^{-5} for different V_{RA} voltages @ TT_85C.

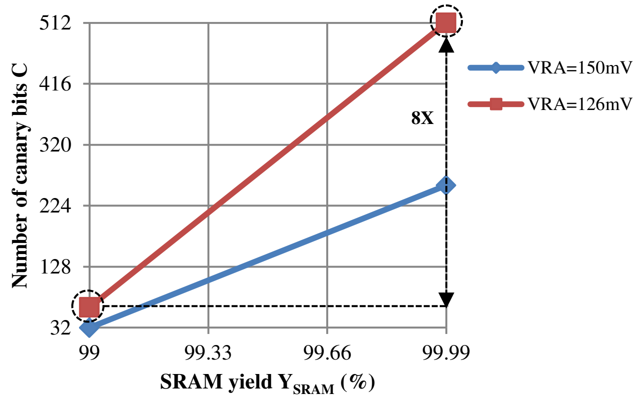


Figure 6: Trend of C vs. Y_{SRAM} with 100 million SRAM bitcell at constant P_{f_c}=10^{-5} for different V_{RA} voltages @ TT_85C.

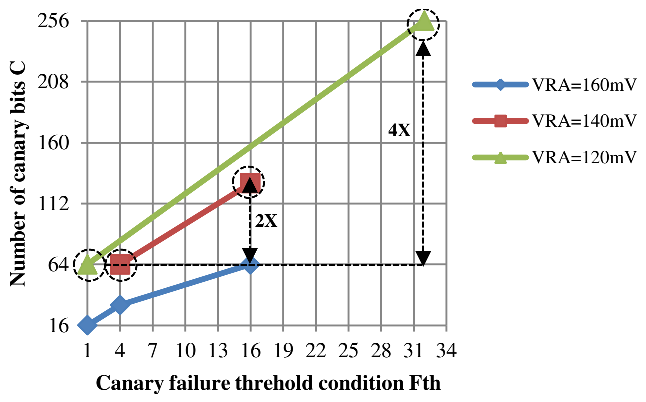


Figure 7: Trend of C vs. F_{th} with 100 million SRAM bitcell at constant P_{f_c}=10^{-5} for different V_{RA} voltages @ TT_85C.

probability of P_{f_c}=10^{-5} can be achieved by either increasing the number of canaries to C=512 with a lower V_{RA}=70mV or decreasing it to C=8 with a higher V_{RA}=170mV. In order to get the trends of C vs. N, C vs. Y_{SRAM}, and C vs. F_{th}, for a constant P_{f_c}=10^{-5} with different values of V_{RA}, we interpolated the data for V_{RA} in between known V_{RA} values.

Figure 5 shows the trend of the number of canary bits C vs. the number of SRAM bits N. We can see that increasing N two orders of magnitude from 1 million to 100 million bits, causes the number of canaries C required to maintain the same canary chip failure probability of P_{f_c}=10^{-5} (at different reverse assist voltages) to double. Figure 6 shows the trend of the number of canary bits C vs. SRAM yield Y_{SRAM}. We can see that in order to keep the same canary chip failure probability of P_{f_c}=10^{-5}, while increasing the SRAM yield from 99% to 99.99%, the number of canary bits have to be increased by 8X from C=64 to C=512 for V_{RA}=126mV BL type reverse assist.

Similarly, Figure 7 shows the trend of the number of canary bits C vs. canary failure threshold condition F_{th} while keeping other input metrics constant. We can see that to maintain the same canary chip failure probability roughly at P_{f_c}=10^{-5} with V_{RA}=140mV, increasing the failure threshold from F_{th}=4 to F_{th}=16, requires 2X more canary cells than that of the C=64. On the other hand, for the reverse assist voltage of V_{RA}=120mV, a change of 32X in F_{th} condition requires a 4X increase in C from C=64 to C=256 to maintain the same P_{f_c}=10^{-5}.

6. Circuit implementation of BL type reverse assist

We assume that a reverse assist will be integrated inside the existing core SRAM I/O as canary I/O; therefore, it requires additional circuitry. Possible ways of creating a reverse assist are to use a positive charge pump, or an analog closed loop voltage reference, or a voltage divider circuit, etc. to generate the reverse assist voltage for the BL type reverse assist. A charge pump and analog closed loop variable voltage reference would have caused much higher design and area overhead per canary I/O. Also, we found that a PMOS-NMOS voltage divider has much higher variation in the output voltage than an NMOS-NMOS voltage divider. To write canaries, we propose a novel

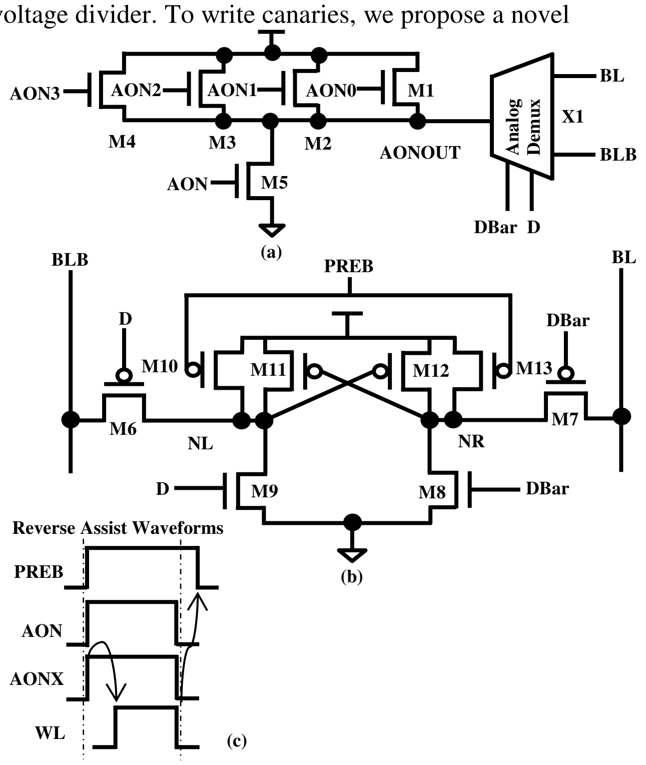


Figure 8: (a) Canary SRAM reverse assist circuit. (b) Canary write driver. (c) Reverse assist waveforms.

reverse assist (Figure 8 (a)) which generates the positive bias voltage (V_{RA}) for BL/BLB, and a write driver (Figure 8 (b)) which pulls up the other node BLB/BL to V_{DD}. Here, signals AON0, AON1, AON2, and AON3 are cumulatively represented by the name AONX in Figure 8 (c). Signals AON and AONX create the V_{RA} at node AONOUT (Figure 8 (a)) by selecting M5 and M1-M4 accordingly. Here, the AONOUT node is either get connected to BL or BLB using an analog de-multiplexer X1 controlled by D/DBar. During a write operation using reverse assist, D or DBar turns on M9/M8 to pull down one of the NL/NR nodes to ground (Figure 8 (b)). This pulls up NR/NL accordingly through cross coupled M12 and M11. However, only the pulled up node NR/NL gets connected to desired BLB/BL nodes by M7 or M6. Thus, M6 and M7 separate the internal pulled down node NL/NR from BL/BLB by turning off M6 or M7. This allows us to connect the reverse assist voltage node AONOUT to BL/BLB node using the analog de-multiplexer X1. For this experiment, we propose to size the analog demultiplexer, M1-M5 sufficiently to discharge the BL/BLB and to generate a minimum of 50mV and maximum of 200mV of reverse assist in a write operation.

7. Block diagram of canary SRAM architecture and an algorithm to track SRAM V_{MIN}

In order to implement the circuit proposed in Section 6, we propose a canary SRAM architecture and an algorithm to track SRAM V_{MIN} in this Section. The block diagram of the proposed canary architecture is shown in Figure 9. In this block diagram, the canary I/Os, canary control, and single canary bitcell rows constitute the canary SRAM. This canary block can adjoin a core SRAM macro as shown in Figure 9, which has two SRAM core arrays, a decoder, I/Os, and SRAM control logic. In this case, the canary control can directly talk to the SRAM control logic. The wordlines are oriented horizontally, and bitlines vertically in the SRAM core array and in canary bitcell rows. In order to operate the canaries independently of the SRAM, the bitlines break at the junction of the SRAM core array and canary row. Also, the canary can sit distantly from the SRAM macros. In the first case, if the canary SRAM is integrated in all the

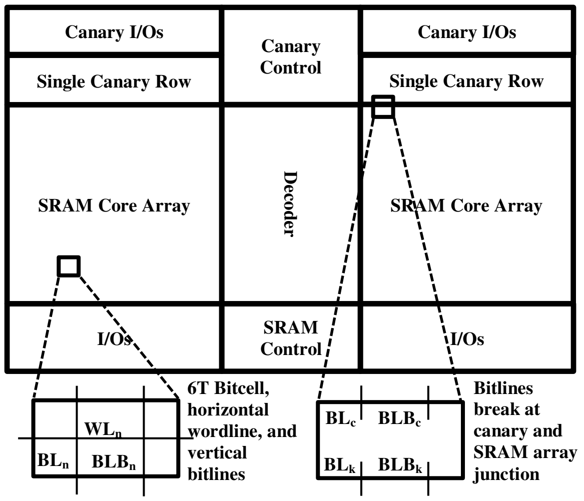


Figure 9: Block diagram of the canary SRAM inside SRAM macro (not in scale).

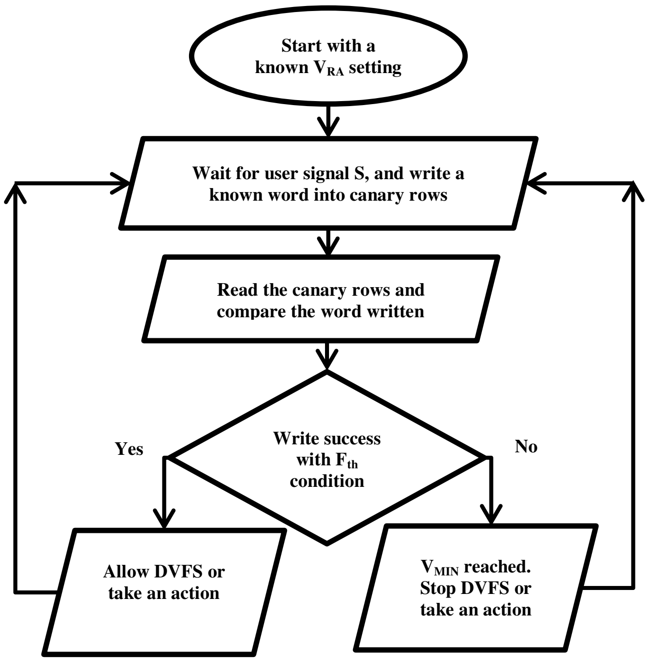

Start with a known V_{RA} setting
Wait for user signal S, and write a known word into canary rows
Read the canary rows and compare the word written
Write success with F_{th} condition
Yes
No
Allow DVFS or take an action
V_{MIN} reached. Stop DVFS or take an action

Figure 10: SRAM V_{MIN} tracking algorithm using canary SRAMs with reverse assist.

SRAM macros, it can track local and global voltage, frequency, temperature fluctuations on the power grid, variation in corners, and aging effects in a large SOC. On the other hand, a standalone single canary SRAM macro can only track the global variation effects in corners, aging etc. in an SOC. The reverse assist circuit described in Figure 8 (a) sits inside each individual canary I/O, and to reduce the effect of local variation, the AONOUT (Figure 8 (a)) signal is shared among the canary I/Os.

Figure 10 shows our proposed algorithm for tracking the SRAM V_{MIN}. Initially, the Canary Control logic State Machine (CCSM) starts with a known setting of V_{RA} during boot up. This known V_{RA} setting corresponds to the SRAM V_{MIN} at a certain process corner with a specific SRAM size of N bits, number of integrated canaries C, a constant P_{f_c} (Figure 4) etc. parameters. After applying the initial V_{RA} setting, the canary state machine waits for a user signal 'S.' If the user allows canary operation by setting the signal 'S,' then the CCSM writes a known word into the canary rows in the first cycle and reads it back from the canary rows to compare with the existing known word value in the second cycle. Word matching with less than or equal to F_{th} number of canary failures signifies a successful write in canaries in the previous cycle, else write fails. Write failure in canaries with an F_{th} condition indicates an imminent SRAM failure. In this situation, the CCSM can signal the DVFS control logic in SOC to stop voltage scaling further, or take a user defined action like stalling the memory access for a couple of cycles or turn on assist in SRAMs, etc. Otherwise, further voltage scaling is allowed or a user defined action like turning off the SRAM assist etc. can be taken. Thus, this algorithm can track the V_{MIN} of each individual SRAM macro with built in canaries. Also, the CCSM can quantify the number of failures and use this value to set the F_{th} value.

Moreover, a user can update the initial V_{RA} setting using on-chip temperature or aging sensor data and simulation data of P_{f_c} to track the write V_{MIN} more precisely.

8. Power and area tradeoff for the BL type reverse assist circuit with write driver

All the power (total of dynamic and leakage) and area tradeoff numbers related to P_{f_c} are calculated with the assumption that the total number of SRAM bits N is 100 million in an SOC with SRAM yield Y_{SRAM} of 99%. Elsewhere, tradeoff numbers are calculated using some assumption of the wordline driver width, I/O height, average bitline energy and bitcell energy per bit etc. parameters. A single canary SRAM, whether or not integrated inside a core SRAM macro, will not be able to track local fluctuations of voltage in the power bus, frequency, and temperature in all 100 million core SRAM bits. This is because those bits are distributed all over the SOC, and the voltage etc. fluctuations will vary from place to place in each macro. As this total of N number of SRAM bits can be divided into an M number of equal or unequal sized SRAM macros, we need to quantify the effect of area and energy overhead of canaries vs. the average size of SRAM macros.

If the canary SRAM is integrated inside a core SRAM macro, the number of canary I/Os, assuming column mux (CM) 4 scenario, required are the same as the number of SRAM I/Os to make a rectangular shaped symmetric total SRAM macro (Figure 9). Hence, C is dependent on the number of I/Os in the SRAM. If a designer chooses a logical macro size of 128 words, 64 bits with CM 4 (128x64x4), he has to use C=64x4=256. Hence, the SRAM size fixes the number of canaries in integrated canary SRAM macros; however, we can use standalone canary SRAMs of user defined size in between core SRAM macros. On the other hand, canary I/Os occupy much bigger area compare to the canary bitcells, which dominates the canary SRAM area overhead. Figure 11 shows that increasing the number of canaries increases the area overhead, and with same C=512 canaries (128 I/Os with CM=4) the overhead is 87% more in smaller 128Kb macros than the bigger one of size 1024Kb. Hence, area can be traded off for better tracking of small sized SRAM macros' V_{MIN} across an SOC. Figure 12 shows that for the same C=512 number of canaries, the 128Kb SRAM macro has roughly 45% higher total power (dynamic and leakage) overhead than a 1024Kb SRAM macro with the same V_{RA}=50mV at the TT_85C corner with an operating frequency of 1GHz. On the other hand, Figure 13 shows that for the change of V_{RA}=50mV to V_{RA}=150mV, the canary power overhead in SRAMs increases by 30% for

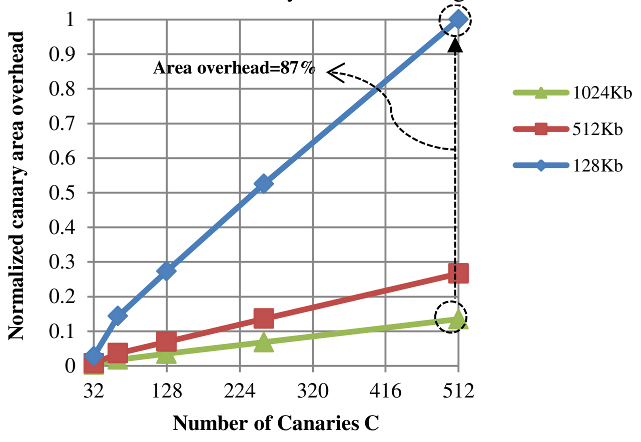


Figure 11: Normalized canary area overhead vs. number of canaries for different SRAM sizes.

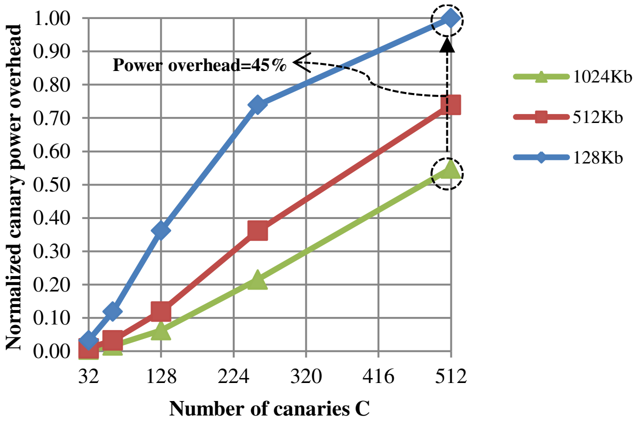


Figure 12: Normalized canary total power overhead vs. number of canaries C with constant V_{RA}=50mV for different SRAM sizes at 1GHz TT_85C corner.

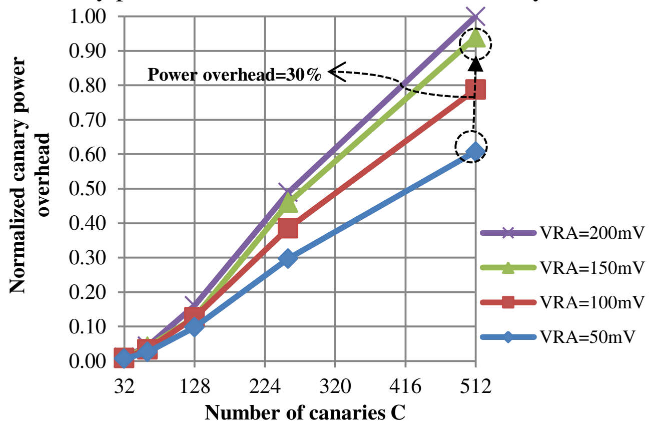


Figure 13: Normalized canary total power overhead vs. number of canaries C with N=512Kb SRAM for different V_{RA} voltages at 1GHz TT_85C corner.

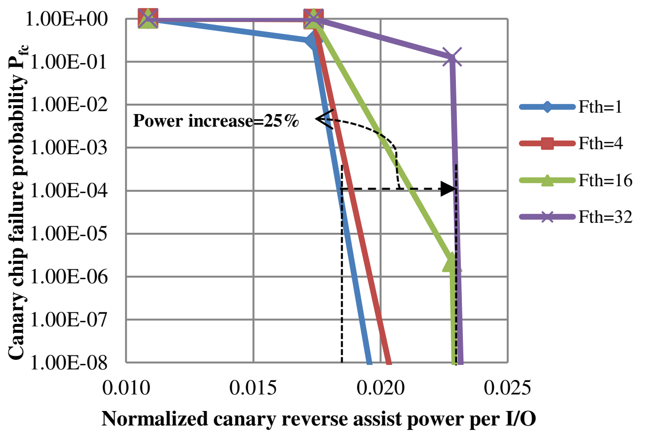


Figure 14: Canary chip failure probability vs. normalized reverse assist total power for increasing F_{th} conditions at 1GHz TT_85C corner (C=128).

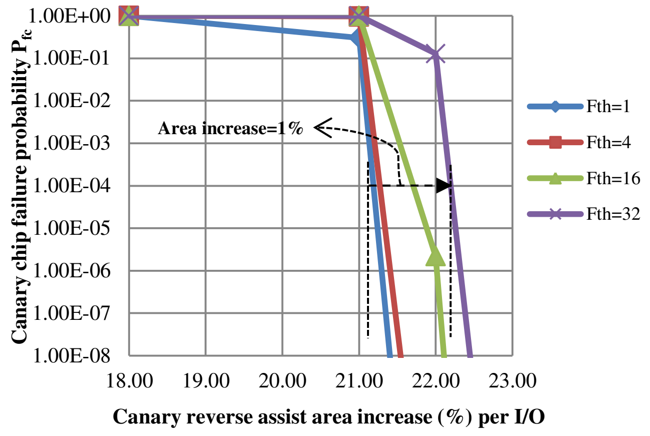


Figure 15: Canary chip failure probability vs. canary reverse assist area increase per I/O for increasing F_{th} conditions (C=128).

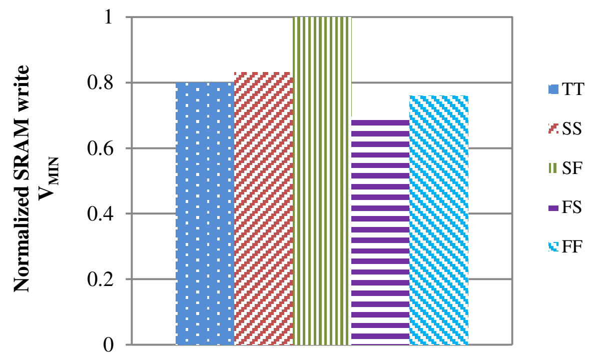


Figure 16: Normalized SRAM write V_{MIN} for 100 million SRAM bits with 99% yield constraints at 85C.

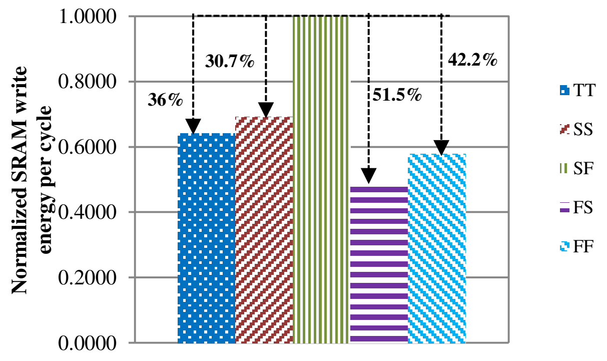


Figure 17: Normalized SRAM write energy per cycle at V_{MIN} for 100 million SRAM bits with 99% yield constraints at 85C.

a 512Kb macro with C=512 at 1GHz operating frequency. Figure 14 and Figure 15 show the P_{f_c} vs. power cost and area cost for increasing F_{th} values. We can see that at TT_85C with C=128 and for $P_{f_c}=10^{-4}$ at 1GHz, increasing the F_{th} condition from $F_{th}=1$ to $F_{th}=32$ increases the power cost by 25%, and the canary I/O area cost by 1%.

Ultimately, the normalized SRAM V_{MIN} for 100 million SRAM bits with 99% SRAM yield at 85C temperature is simulated and shown in Figure 16. As per our simulation results, the canary SRAMs can be used to track the write V_{MIN} of the SRAM bits with a specified confidence, and the normalized write energy corresponding to the core SRAM V_{MIN} is shown in Figure 17. We can see that at the TT_85C corner we can operate SRAMs with 36% lower write energy cost than that of the worst case V_{MIN} at the SF_85C corner, which would set the guard band. The least energy savings can be achieved at the SS_85C corner as 30.7%. The maximum energy savings from Figure 17 can be found to be at the FS_85C corner, which is 51.5% lower than the worst case. Furthermore, at the FF_85C corner, the energy savings can reach up to 42.2% with respect to the worst case energy at the SF_85C corner.

9. Conclusion

We conclude that the canary SRAM concept using a reverse assist is a promising solution to predict core SRAM failure resulting from write-ability problems. Canary SRAM enables the tracking of SRAM write V_{MIN} by using reverse assist to track dynamic voltage, frequency, temperature fluctuations and aging effects. It also allows us to take necessary actions which can be in the form of turning on assists, stalling the memory access, slowing down operating frequency, or boosting supply voltage. Here, we do all the probability calculations based on importance sampling algorithm. However, choosing an incorrect importance sampling distribution can mispredict the V_{MIN} to cause higher energy dissipation or SRAM failures before canaries. The area and power overhead of the canary SRAMs are lower for the bigger SRAM macros, and they depend on the number of SRAM I/Os in integrated canary SRAM macros. We can qualitatively say that the canary failure threshold condition F_{th} rejects the extreme canary outliers. Moreover, using canaries, SRAM write V_{MIN} in different corners can be reduced below the traditional worst case V_{MIN}, which saves energy. Finally, we conclude that the canary SRAM for dynamic write V_{MIN} tracking works in simulation and theory.

10. Acknowledgements

We thank DARPA and NVIDIA for supporting this research work.

11. References

[1] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, pp. 38-43, 2007

[2] A. Genser et al., "Power emulation based DVFS efficiency investigations for embedded systems," in *Proc. Int. Symp. Syst. Chip (SoC)*, pp. 173-178, 2010

[3] R. Airoldi et al., "Improving Reconfigurable Hardware Energy Efficiency and Robustness via DVFS-Scaled Homogeneous MP-SoC," in *Proc. IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW)*, pp. 286-289, 2011.

[4] J. Pille et al., "Implementation of the CELL Broadband Engine in a 65nm SOI Technology Featuring Dual-Supply SRAM Arrays Supporting 6GHz at 1.3V," in

IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, pp. 322–323, 606, 2007.

[5] B. Zimmer et al., "SRAM Assist Techniques for Operation in a Wide Voltage Range in 28-nm CMOS," in *IEEE Trans. Circuits Syst. II*, vol. 59, no. 12, pp. 853-857, 2012.

[6] A. Bhavnagarwala et al., "Fluctuation limits & scaling opportunities for CMOS SRAM cells," in *IEDM Tech. Dig*., 2005, pp. 659-662, 2005.

[7] N. Shibata et al., "A 0.5-V 25-MHz 1-mW 256-kb MTCMOS/SOI SRAM for solar-power-operated portable personal digital equipment—Sure write operation by using step-down negatively overdriven bitline scheme," in *IEEE J. Solid-State Circuits*, pp.728 -742, 2006.

[8] V. Chandra et al., "On the efficacy of write-assist techniques in low voltage nanoscale SRAMs," in *Proc. Des. Autom. Test Eur.*, pp. 345-350, 2010.

[9] R.W. Mann et al., "Limits of Bias Based Assist Methods in Nano-Scale 6T SRAM," in *Proc. Quality Electronic Design Symposium (ISQED)*, pp. 1-8, 2010.

[10] Jonathan Chang et al., "A 20nm 112Mb SRAM in High-κ Metal-Gate with Assist Circuitry for Low-Leakage and Low-VMIN Applications," in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, *2013 IEEE International*, pp. 316 - 317, 2013.

[11] A. Shah and H. Mahmoodi, "Thermal estimation for accurate estimation of impact of BTI aging effects on nano-scale SRAM circuits," in *SOC Conference (SOCC), 2010 IEEE International*, pp. 230-235, 2010.

[12] A. Bansal et al., "Impact of NBTI and PBTI in SRAM bit-cells: Relative sensitivities and guidelines for application-specific target stability/performance," in *IEEE IRPS*, pp. 745-749, 2009.

[13] S. C. Yang et al., "Timing control degradation and NBTI/PBTI tolerant design for write-replica circuit in nanoscale CMOS SRAM", in *Proc. IEEE Int. Symp. VLSI Design, Autom., Test*, pp. 162-165, 2009.

[14] S. Nalam et al., "Dynamic write limited minimum operating voltage for nanoscale SRAMs," in *Proc. Des. Autom. Test Eur.*, pp. 1-6, 2011.

[15] J. Wang et al., "Two Fast Methods for Estimating the Minimum Standby Supply Voltage for Large SRAMs," in *Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 29, issue 12, pp. 1908-1920, 2010.

[16] E. Karl et al., "A 4.6ghZ 162Mb SRAM design in 22nm trigate CMOS technology with integrated active V_{min}-enhancing assist circuitry," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers (ISSCC)*, San Francisco, CA, pp. 230-232, 2012.

[17] B. H. Calhoun and A. P. Chandrakasan, "Standby power reduction using dynamic voltage scaling and canary flip-flop structures," in *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1504-1511, 2004.

[18] Y. Otsuka et al., "Multicore energy reduction utilizing canary FF," in *Communications and Information Technologies (ISCIT), 2010 International Symposium*, pp. 922- 927, 2010.

[19] J. Wang and B. Calhoun, "Canary replica feedback for near-DRV standby V_{DD} scaling in a 90 nm SRAM," in *Proc. Custom Integrated Circuit Conf. (CICC '07)*, pp. 29–32, 2007.

[20] J. Wang and B. H. Calhoun, "Techniques to Extend Canary-based Standby V_{DD} Scaling for SRAMs to 45nm and Beyond," in *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 2514-2523, 2008.

[21] J. Wang et al., "An Enhanced Canary-based System with BIST for SRAM Standby Power Reduction," in *Transactions on VLSI Systems (TVLSI)*, pp. 909-914, 2011.

[22] L. Dolecek et al., "Breaking the simulation barrier: SRAM evaluation through norm minimization," in *Proc.IEEE/ACM Int. Conf. Comput.-Aided Des.*, pp. 322-329, 2008.

[23] R. Kanj et al., "Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events," in *Proc. ACM/IEEE Des. Autom. Conf.*, pp. 69-72, 2006.

[24] A. Singhee et al., "Recursive Statistical Blockade: An Enhanced Technique for Rare Event Simulation with Application to SRAM Circuit Design," in *International Conference on VLSI Design*, India, pp. 131-136, 2008.