Digitally Controlled Variation Tolerant Timing Generation Technique for SRAM Sense Amplifiers

K R Viveka, Bharadwaj Amrutur
Department of ECE, Indian Institute of Science, Bangalore-560012, India
E-mail: {krviveka, amrutur}@ece.iisc.ernet.in

Abstract— Embedded memories occupy increasingly greater portion of SoC area, significantly affecting system performance metrics such as speed and power. The adverse effects of variation, that is accompanying technology scaling, is however making design of these high density memories increasingly challenging. The speed and power consumption of memories is greatly affected by the technique employed to generate timing signals, specifically the sense-amplifier enable (SAE) signal. A BIST based post-silicon tunable approach is known to provide the best tracking with process variation with minimum margins. This paper proposes an improved tuning algorithm that utilizes random-sampling to achieve faster tuning. The algorithm also enables increased utilization of redundancy repair infrastructure to further lower power consumption and improve access speeds.

Keywords— SRAM timing, random sampling, sense-amplifier timing, statistical-approach, delay-tuning algorithm

I. Introduction

With technology scaling, the demand for embedded memories with high access rates and lower power consumption continues to increase [1]. The increased variation in these technologies, however, is forcing designers to add greater margins to ensure profitable yields. These margins translate into increased power consumption and sub-optimal performance.

One of the key challenges in design of Static Random Access Memories (SRAMs) is the accurate generation of sense amplifier enable (SAE) timing signal. If the sense amplifier is enabled too early, the insufficient differential voltage on the bitlines will results in an erroneous read. A delayed enable signal, on the other hand, will result in greater voltage swings on the bitlines, than necessary, causing increased power consumption and longer access times. Thus, SAE generation directly affects both the performance and power consumption of memories. As SRAMs continue to occupy increasingly greater portion of SoC area [1], their yield and power consumption significantly impact the system performance.

With increased variation effects such as Random Dopant Fluctuation (RDF), accurate generation of timing signal is proving to be extremely challenging. The conventional way of generating SAE is to use a replica bitline (RBL) [2] that consists of an additional column of SRAM cells that tracks the process (global) variation in SRAM array (Fig. 1). However, the increased local variation, due to RDF,

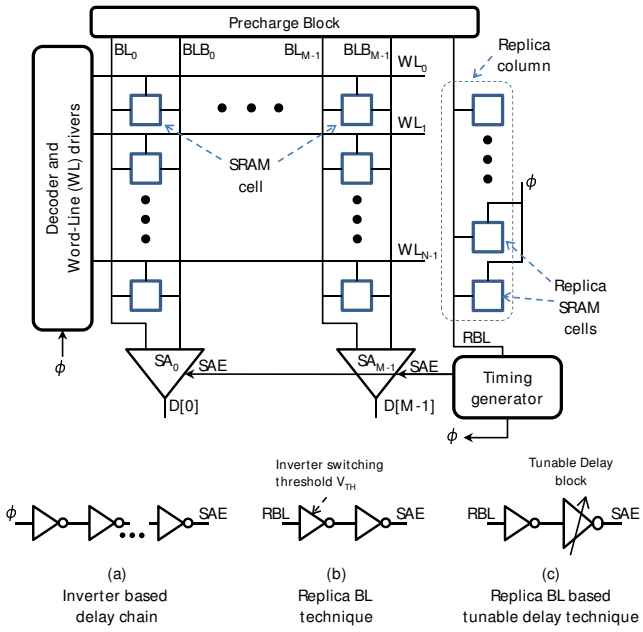


Fig. 1. Timing generation technique used in SRAMs for SAE generation

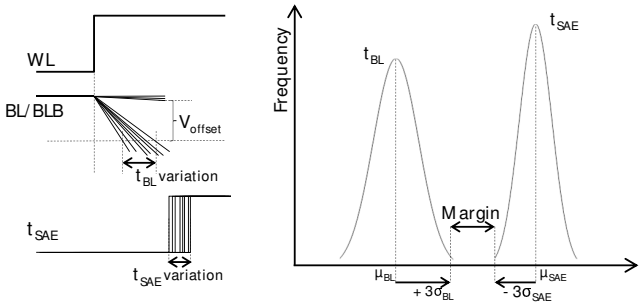
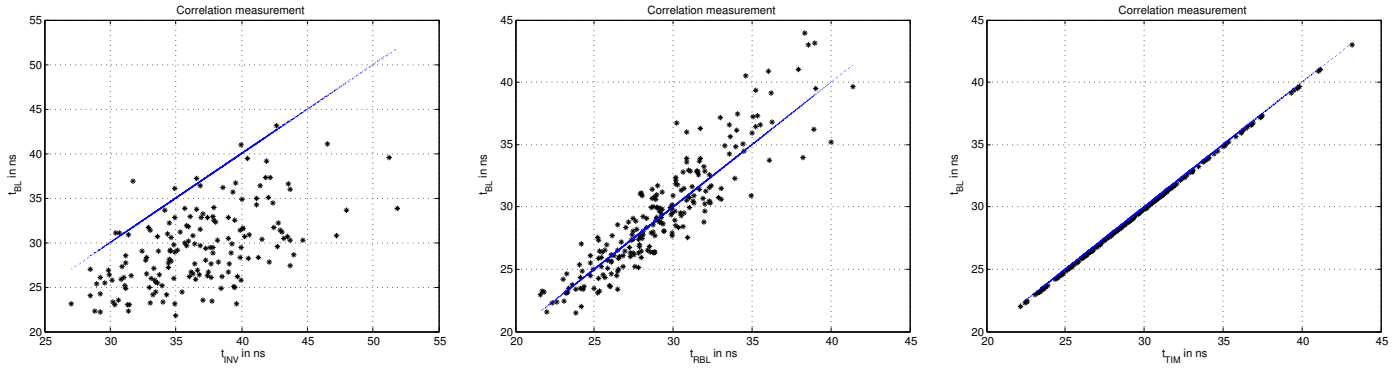


Fig. 2. Process variation causes uncertainity in bitline falltime and SAE generation

causes the replica column's characteristics to vary significantly. In order to achieve higher yields, designers trade-off performance by adding margins for these variations. Several modifications to this technique have been proposed that improve tracking, such as the use of averaging to limit variation [3] in RBL. Another approach proposed in [4], employs a ranking circuit to track a fixed yield point by monitoring all the bitlines of the memory. While each of these provide some improvement in tracking, their effectiveness remains limited.

Another approach to accurately generate timing signals is to use a programmable delay line and tune the delay post fabrication [5][6][7]. This enables minimizing of margins to



(a) (b) (c)

Fig. 3. Correlation between bitline fall time and SAE timing, with variation in process conditions. SAE is generated using (a) Inverter delay chain (b) Replica bitline and (c) Tunable replica bitline.

track SRAM delay accurately while maintaining yield targets. Programming of the delay line however requires additional tester time which in-turn increases the cost per chip. Hence the algorithm used in tuning the delay-line plays is significant role in determining the effectiveness of this technique. The algorithms proposed in literature [5][6][7] however consume large amounts of time in tuning and do not exploit the tunable delay technique completely.

This paper proposes a tuning algorithm that takes advantage of the random nature of the variation to reduce the sample-set used to tune the delay line. This translates to lower number of reads during tuning and hence shorter tester time. It is also shown that performing tuning before redundancy repair enables reduction in power consumption and faster access times in memories that have lower failure rates than expected.

The rest of this paper is organized as follows. Section II compares the performance and limitations of various SAE timing generation techniques available in literature. This is followed by the description of existing and proposed tuning algorithms in section III. Section IV then presents simulation results evaluating the effectiveness of the proposed techniques. Section V then concludes the paper.

II. SAE Generation Techniques

The SAE signal is required to enable the sense-amplifier to read the data on bitlines during a memory read operation. A read is performed by first precharging the bitlines, and then activating the wordline corresponding to the address being read as shown in Fig. 2. Depending on the data stored in a particular SRAM cell, one of the bitlines (per bit being read - assuming a SRAM cell with differential read) begins to discharge. The sense-amplifier is then activated, after a sufficient differential voltage develops between the bitlines, to determine the data stored in the cell. Bitlines are highly capacitive due to large number of SRAM cells connected to them. The SRAM cell, which contains mostly minimum sized transistor, thus requires a large amount of time to discharge the BL. Also to conserve power, we would like to minimize the voltage swing on these highly capacitive bitlines. Ideally the sense-amplifier is therefore activated immediately after the bitlines develop a differential voltage greater than the offset voltage of the sense-amplifiers.

Process variation however causes the bitline fall-time to vary across the memory array (local-variation) and from one chip to another (global variation), causing the bitline fall time to have a normal distribution as shown in Fig. 2 [5]. The timing generation circuit, used to generate SAE, also undergoes similar variation and may be modeled by a normal distribution. To ensure error free functionality, the SAE must arrive after a differential voltage greater than the sense-amplifier's offset voltage is developed on the bitlines. This is done by adding appropriate margins during design, depending the trade-offs between yield requirements and power consumption, as shown in Fig 2.

In order to minimize margins the variance in SAE generation needs to be reduced. Several techniques have been proposed in literature to address this issue. The remainder of this section examines and evaluates some of these techniques shown in Fig. 1.

A. Standard Logic Based Delay Line

This technique employs a standard logic based delay chain, whose configuration is determined at design time. Although this approach is seldom used, it has been included here to illustrate the mismatch between logic and memory circuits.

Figure 3(a) shows the scatter-plot between bitline fall time and inverter chain based delay line, with variation in process conditions (global mismatch) for 130nm UMC SRAM cells at 500mV. The memory is run at a lower voltage to enhance the effects of variation in order to mimic the increased variability in deep submicron processes.

Each point in the plot corresponds to a 1000-point Monte-Carlo simulation at a given global process point, simulating variation corresponding to only local mismatch (not global variation). The process corner (global mismatch) is then varied randomly (with Gaussian distribution) and a Monte-Carlo variation for local mismatch is performed for each of the process points to obtain the various points in the plot. At each process-point the bitline fall

time and delay of inverter chain corresponding to a fixed yield point (99.73%) are noted and plotted as the x and y-axis respectively. With local mismatch corresponding to variation in a given chip and global mismatch corresponding to variation across different chips, the plot enables us study the tracking capability of the SAE generation technique. Good tracking manifests as higher correlation and thus implies lower margins required during design.

As seen from Fig. 3(a), the standard logic based delay line offers poor tracking with a correlation of just 56.01%.

B. Replica Bitline

The conventional technique used commonly in SRAMs currently, is the Replica Bitline technique [2]. This technique uses an additional column in the SRAM array to track process variations in the memory. The bitline on the additional column is known as the replica bitline (RBL). Multiple SRAM cells are activated on RBL together and the time-taken by the RBL to fall below a preset threshold voltage is used to generate SAE signal. This techniques provide better tracking as can be seen in Fig. 3(b) with a correlation of 90.99%.

C. Other Circuit Techniques

Another approach [3] is to use a greater multiple of SRAM cells on the RBL, to provide averaging against random variation followed by a timing multiplier circuit to obtain the required timing. [4] proposes yet another technique that monitors all the bitlines in memory and ranks them in the order of speed using order extraction circuits. This ranking is used to estimate the correct timing to obtain a predetermined yield. These techniques however, provide limited improvement in tracking and reduction in variance of the SAE timing. Also they offer little flexibility and provide no insight into silicon's performance.

D. Replica Bitline with Tunable Delay

An alternative approach is to use a replica bitline along with a tunable delay controller to modify the timing generator after fabrication to achieve close tracking in the presence of process variation [5][6][7]. This technique allows reduction of the margins to the maximum extent, limited only by the delay tuning resolution. The tuning can be performed based on yield targets providing post fabrication flexibility. The delay setting finally used also readily enables binning of chips. Another advantage, is the capability to maintain functionality with slow varying changes such as aging.

The tracking obtained using this technique is evaluated using Monte-Carlo simulations similar to the previous scatter plots. For a given global process point, the tuning algorithm sets a switched capacitor based delay-chain to obtain a target yield of 99.73%. This is then repeated at various global process points (corresponding to different chips) and the actual delay required and target delay set by the tuning controller are plotted as the y and x-axis respectively in Fig 3(c). This technique clearly offers the best tracking with nearly ideal correlation ($\approx 100\%$).

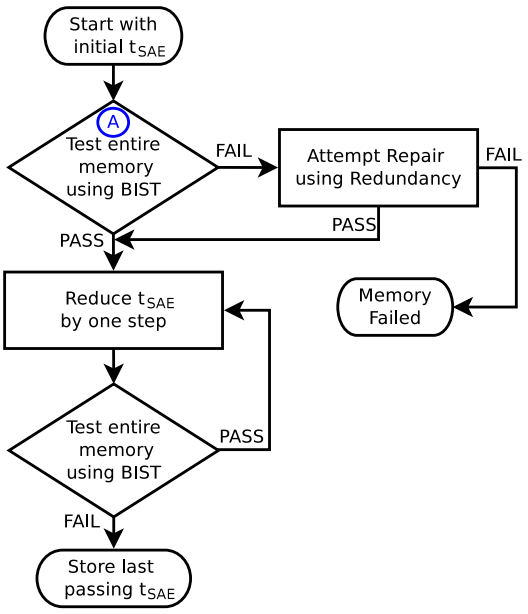


Fig. 4. Existing delay tuning algorithm [5][6]

As mentioned earlier, the tuning algorithm used here plays an important role in reducing the tester time required to set the delay controller. The issues related to these algorithms is examined in the following section.

III. Optimized Repair and Tuning

Delay tuning algorithms, used to set the SAE timing (t_{SAE}), are iterative in nature and can take a significant amount of time (measured as number of reads) depending on the implementation. We would like to minimize this time, especially if tuning requires time on the tester, as this adds to the cost of the chip. Also the effectiveness of the delay-tuning technique in minimizing power and increasing access speeds is determined by the algorithm. Hence the tuning algorithm plays a significant role in determining the efficiency of the delay tuning technique.

A. Conventional Approach: Repair followed by tuning

Figure 4 shows the generalized flowchart of algorithms proposed in [5][6]. Here the controller starts off with a worst case estimate for SAE timing (t_{SAE}), based on simulations. The entire memory is then tested for correct functionality using the memory's built in self test (MBIST) state-machine. Any failures at this stage are corrected, if possible, using redundancy. Once the memory passes with the initial t_{SAE} setting, the controller then iteratively reduces t_{SAE} and determines the minimum t_{SAE} for which the entire memory functions correctly.

This treatment of post-silicon tuning algorithms in previous works is, however, brief. While they serve as a good starting point, they fail to take advantage of several flexibilities enabled by this tunable technique. This work proposes an enhanced algorithm that integrates several im-

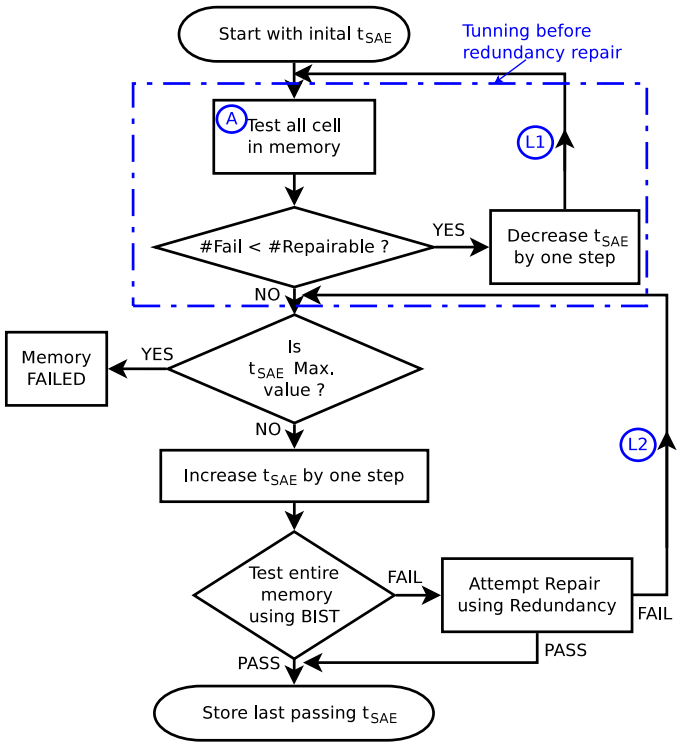


Fig. 5. Proposed delay tuning algorithm. A further optimization in block A is to "Test N_{sample} Cells" where N_{sample} ¡¡ total number of cells

provements that significantly reduce tester-time requirement and improves the effectiveness of this tunable technique.

B. Proposed Approach: Delay tuning followed by redundancy repair

The algorithms proposed in literature [5][6] perform redundancy repair prior to delay tuning as discussed earlier. However for chips in which the number of failures is lower than that repairable using redundancy, the additional redundant SRAM cells remain unused. The tuning technique may be utilized to improve the performance of such chips by performing delay tuning before redundancy repair as shown in Fig. 5 (loop L1).

As shown in the figure, the controller starts with a conservative estimate of t_{SAE} obtained through simulations and tests the entire memory for this value of t_{SAE}. If the number of cells failing is less than that repairable using redundancy, then the controller continues to reduce t_{SAE} until the available redundancy is just sufficient to repair all the failing cells. The algorithm, of course, declares the chip as failed in the worst-case scenario where the redundancy available is insufficient to repair the memory even when the maximum t_{SAE} timing is used. For chips in which available redundancy is higher than failure rates (when a conservative t_{SAE} is used), the proposed approach utilizes the remaining redundancy infrastructure to further reduce t_{SAE}.

The memory array read power consumption is given by [5]:

$$P_{array} = N_{BL} I_c t_{WL} V_{dd} f \tag{1}$$

where N_{BL} is the number of bitlines in the array, I_c denotes the average read current per SRAM cell, V_{dd} is memory supply voltage, f is the operating frequency and t_{WL} is the wordline pulse width, which closely tracks t_{SAE}. Hence a reduction in t_{SAE} translates directly to dynamic power savings of the array.

C. Random Sampling: Reducing number of reads

The conventional tuning algorithms (Fig. 4) check the entire memory at each iteration of delay setting (Block A). While this approach ensures no loss in yield during tuning, the process may require a large number of reads, depending on the initial t_{SAE} setting and the delay step used in the delay-line. Performing delay tuning before redundancy repair provides us with an additional margin for error in setting the sense amplifier timing. We propose to take advantage of this margin to reduce the number of reads in Block A of Fig. 4, via random sampling during delay tuning. This significantly reduces the time required for tuning.

From statistics [8] it is known that, in order to estimate the probability of success p in a binomial distribution, from a sample of size n with at least $100(1-\alpha)\%$ probability of being within a distance d of p, the sample size n should be no smaller than

$$n = \frac{z^2_{\alpha/2}}{4d^2} \tag{2}$$

where $z_{\alpha/2}$ is the value for which $P(Z \geq z_{\alpha/2}) = \alpha/2$. If p is known to be greater than some number p', then this information can be used to further reduce the number of samples required. n is then given by:

$$n = \frac{z^2_{\alpha/2}}{d^2} p'(1-p') \tag{3}$$

The above theorem can applied to the tuning algorithm by expressing the problem as follows. During tuning, at each iteration, we are trying to estimate the number of SRAM cells passing for a given t_{SAE} setting. We would like make this estimate with a high level of confidence, as any violation would cause the controller (Fig. 5) to iterate through the entire memory to find the correct t_{SAE} setting, as explained before. The error tolerance d, is set equal to the amount of redundancy r available. This enables us to repair any errors in our estimate using redundancy. However for low values of redundancy, setting d to a higher value yields significant reduction in number of samples while causing an insignificant increase in error of the estimate. We thus set

$$d = \begin{cases} 2 * r, & \text{if } r \leq 3\% \\ \\ r, & \text{if } r > 3\% \end{cases} \tag{4}$$

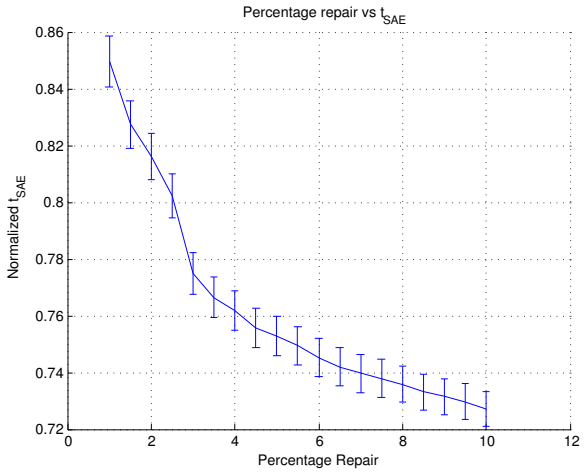


Fig. 6. Normalized t_{SAE} vs Percentage Repair

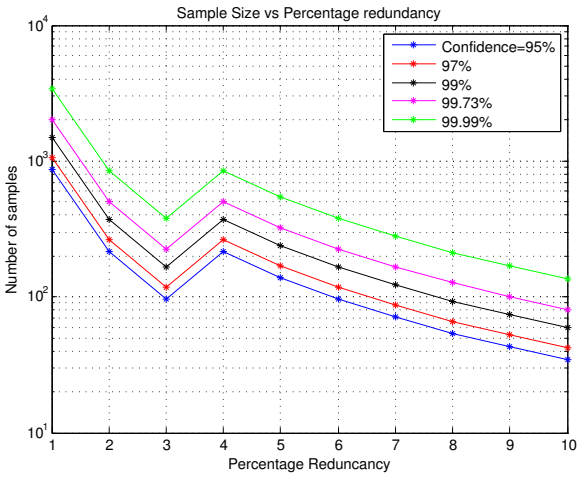


Fig. 7. Number of samples vs Percentage redundancy for various values of confidence

D. Proposed Algorithm: Tuning using random-sampling followed by repair

The proposed delay tuning algorithm combines the above two techniques to effectively utilize available redundancy and significantly speed up the tuning process. The final algorithm is obtained by replacing the content of the block labeled "A" in Fig. 5 with "Randomly test N_{sample} cells", where N_{sample} is the sample size obtained for random sampling. The controller starts off with an estimate for SAE timing t_{SAE} based on simulations, similar to the conventional technique. This value is then tuned iteratively, in loop L1, using random sampling until the available redundancy is just sufficient to repair all cells failing for the current t_{SAE} setting.

This is then followed by redundancy repair using MBIST, similar to one done in conventional algorithms, to set the final SAE timing. Random sampling may however provide a slightly aggressive estimate for t_{SAE}. Thus redundancy repair is done iteratively in loop L2, where t_{SAE} is increased if necessary. While the proposed approach significantly improves over conventional approaches, loop L2 ensures that it at least matches the conventional technique in the worst case.

Note that the proposed approach requires a pseudo random number generator (PNRG), in addition to the resources required by existing algorithms. MBIST controllers typically contain PNRGs, hence this technique incurs no area overhead.

IV. Results and Discussion

The proposed approach is tested on UMC 130nm process using extensive Monte-Carlo simulations. The design employs a 6T SRAM cell, however the results are valid for other SRAM cells that have a similar two transistor read-path. The effects of variation were exacerbated by running the circuits at a lowered supply of 500mV, making the results applicable to lower technology nodes. The models allow for independently varying parameters to simulate either local variation which correspond to different instances of a circuit on a single chip, or global variation which represents variation across multiple chips.

Figure 6 shows the effect of preforming delay tuning before redundancy repair, on t_{SAE} with varying amount of redundancy. The results are obtained by first simulating local variation using Monte-Carlo runs. This step is then repeated at over 200 process points (corresponding to memories on different chips) with a Gaussian distribution specified by the manufacturer. The proposed algorithm is then applied to each of the process points (representing the tuning algorithm algorithm on different chips). The average across these process points then gives a measure of the typical t_{SAE} used. These steps are then repeated for varying amounts of redundancy repair capability, while the failure-rate (due to manufacturing, is fixed at 1%). If the redundancy capability is lower than the failure-rate, the chip is discarded.

The values shown in Fig. 6 are normalized with the value of t_{SAE} obtained using conventional technique. For example, if the redundancy capability is 5%, the proposed algorithm enables a 25% reduction in t_{SAE}. The reduction in t_{SAE} for different failure rates can also be obtained from the figure. For instance, if the failure rate is 2%, then the same 5% redundancy would provide about 6.5% reduction in t_{SAE}.

Figure 7 shows the variation of number of samples required with the redundancy value r used in Eqns. (3) and (4). As the yield requirements are generally high in memories we set p' in eqn. 3 conservatively at 90%. It can be seen that as the amount of redundancy available decreases, the number of samples required in estimation increases exponentially as a higher accuracy in estimation is necessary. Also the sample size increases if a greater confidence is required in estimation. However this increase is not very significant, hence a large confidence value can used. Note that the discontinuity observed at $r = 3\%$ is contributed by Eqn. (4).

The effectiveness of the above random sampling is evaluated on normal distributions with 10% coefficient of vari-

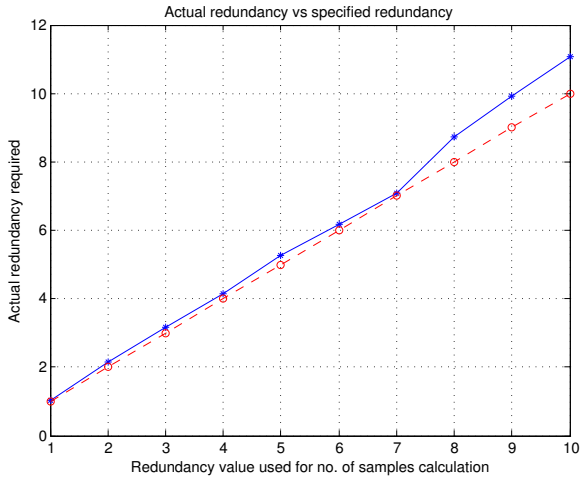


Fig. 8. Actual redundancy used vs specified redundancy

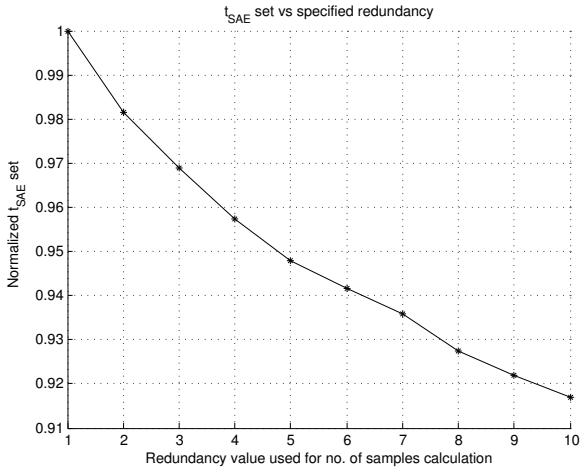


Fig. 9. t_{SAE} vs redundancy specified during calculation of N_{sample}

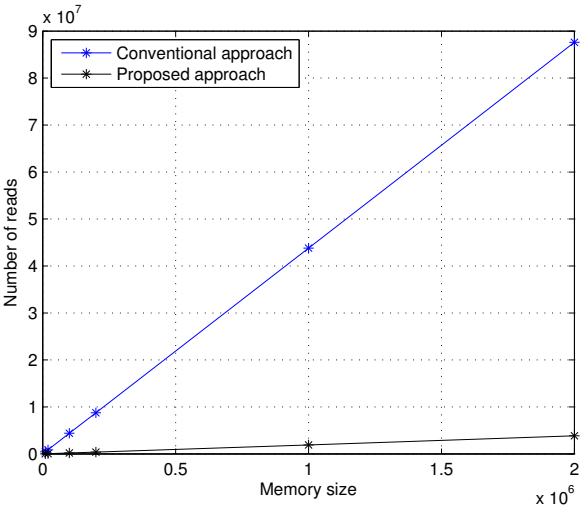


Fig. 10. Number of samples vs Memory size

ance (σ/μ). A confidence of 99.73% was used to obtain the results shown in Fig. 8. The figure plots the redundancy value r set to compute the number of samples and the actual amount of redundancy required when the t_{SAE} setting, obtained from random sampling based tuning, was applied to the complete 10Kb memory. It may be seen that the results track very well, verifying the effectiveness of the above technique. The good tracking ensures that the loop L2 in Fig. 5 is executed a very small number of times (at most 3) if redundancy percentage is sufficient to repair the chip.

The variation of SAE pulse width with redundancy is shown in Fig. 9. As expected, t_{SAE} reduces if higher redundancy is available. This can be used to trade redundancy for reduction in power consumption and access time in SRAMs. The reduction in tuning time however is a weak function of the amount of redundancy r available. The technique provides approximately 95% reduction in tuning time when a step size equal to 10% σ_{BL} is used for 10Kb memory. Figure 10 compares the time taken for t_{SAE} tuning (Block A in Fig. 5) by the conventional and proposed technique for different memory sizes. As the number of samples required is independent of the memory size, the technique is highly effective for large blocks of memory.

Thus random sampling can be used to significantly reduce the time taken to perform post silicon delay tuning. The amount of redundancy available and confidence required, is used to determine the number of samples ($Nsample$) which is used in the proposed algorithm shown in Fig. 5. Note that setting $Nsample$ equal to the total number of cells in memory would cause the proposed algorithm to perform identical to existing algorithms.

V. Conclusions

The paper demonstrates the effectiveness of a tunable delay line, in tracking memory performance and minimizing margins, for generation of SAE in SRAMs. The proposed delay tuning algorithm, that employs random-sampling, is shown to significantly reduce tester time thus directly contributing to reduction in cost. Further, the use of redundancy after delay tuning enables maximum utilization of redundancy infrastructure to reduce power consumption and enhance performance.

Acknowledgement

The authors would like to thank Rajath V for valuable discussions, and the Department of Electronics and Information Technology, MCIT, Government of India for funding this research.

References

[1] http://www.itrs.net
[2] Amrutur, B.S.; Horowitz, M.A., "A replica technique for wordline and sense control in low-power SRAM's," Solid-State Circuits, IEEE Journal of , vol.33, no.8, pp.1208,1219, Aug 1998.
[3] Niki, Y.; Kawasumi, A.; Suzuki, A.; Takeyama, Y.; Hirabayashi, O.; Kushida, K.; Tachibana, F.; Fujimura, Y.; Yabe, T., "A Digitized Replica Bitline Delay Technique for Random-Variation-Tolerant Timing Generation of SRAM Sense Amplifiers," Solid-

State Circuits, IEEE Journal of , vol.46, no.11, pp.2545,2551, Nov. 2011.

[4] Kawasumi, A.; Takeyama, Y.; Hirabayashi, O.; Kushida, K.; Tachibana, F.; Niki, Y.; Sasaki, S.; Yabe, T., "A 47% access time reduction with a worst-case timing-generation scheme utilizing a statistical method for ultra low voltage SRAMs,", VLSI Circuits (VLSIC), 2012 Symposium on , vol., no., pp.100,101, 13-15 June 2012.

[5] Abu-Rahma, M.H.; Anis, M.; Sei-Seung Yoon, "Reducing SRAM Power Using Fine-Grained Wordline Pulsewidth Control,", Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.18, no.3, pp.356,364, March 2010.

[6] Ya-Chun Lai; Shi-Yu Huang, "Robust SRAM Design via BIST-Assisted Timing-Tracking (BATT),", Solid-State Circuits, IEEE Journal of , vol.44, no.2, pp.642,649, Feb. 2009.

[7] Neale, A. and Sachdev, M., "Digitally programmable SRAM timing for nano-scale technologies", 2011 12th International Symposium on , vol., no., pp.1,7, 14-16 March 2011.

[8] Larsen, R.J. and Marx, M.L., An Introduction to Mathematical Statistics and Its Applications, Prentice Hall, 2001.