Abstract

As CMOS technology scales down, more and more circuits can be put on a single chip so that power management becomes a big concern for a circuit designer. Also, denser integration and shrinking geometries can trigger circuit reliability problems. Static Random Access Memories (SRAM) occupies a large area on the chip so it plays a significant role in chip power consumption and the reliability of circuits.

This project has two main components: the first part of the research carried out and reported in this thesis consisted of a comparison and evaluation of the five different SRAM cells (i.e. from six transistor size (6T) design to ten transistor size (10T) design). The five different SRAM blocks are designed in 45nm high-k/metal gate CMOS technology, high threshold voltage process for on-chip caches of a low power processor. The comprehensive study of each design is carried out in terms of their access time, power, read stability, write stability and cell layout. The improved data node isolation technique is proposed, as is the better bit line power saving method. The results of statistical characterization of read-, write- and hold- SNM, and cell failure rates are presented for all cells in the presence of process variability. The cell static noise margin (SNM) degradation against temperature is also analysed.

The second part of the project is focused on ECC-protected SRAM design. A double error detecting and single error correcting hamming code is chosen to mitigate reliability problems. A hamming encoding and decoding schemes are implemented and a low power hamming encoder and decoder are proposed. The decoder is also embedded with sense amplifiers so the power can be saved during a read cycle. This encoder requires 0.6812ns to generate parity bits and the decoder requires 1.2661ns to detect errors and generate error bit position at 0.6V using 45nm CMOS for 8-bit word length. The proposed hamming encoder and decoder have half of the area, faster decoding speed and much less power consumption compared to the conventional iterative decoding scheme.

Declaration

This dissertation is submitted to the University of Bristol in accordance with the requirements of the degree of Master of Science in the Faculty of Engineering. It has not been submitted for any other degree or diploma of any examining body. Except where specifically acknowledged, it is all the work of the Author.

Ying Ying Zhang, September 2011

Contents

List of Figures

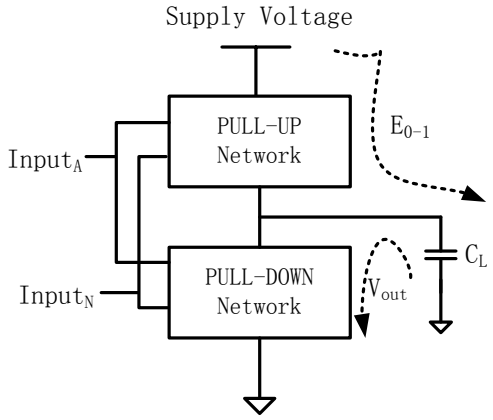
1. Motivation

1.1 Low power CMOS Design

Battery-powered embedded systems and mobile devices are very popular in modern life, and for designers power management becomes the primary concern. CMOS technology is used in most high performance, complex chips because of their favourable low power consumptions.

There are four components of the power consumed in the CMOS circuit: switching power is related to charging and discharging load capacitors; leakage power is due to the transistors being imperfectly switched; short circuit power is consumed when both pull-up and pull-down networks are on during transition; and static currents are due to the biasing current only appearing in certain circuit topology.

Dynamic power consumption is the most important component. Figure 1 shows how this power is associated with the switching of the logic state. When we want to establish a voltage level at the capacitor C_l then we have to pull some of the energy $E_{0\to1}$out of the supply voltage; half of the energy E_c is stored in the capacitor and the other half E_r of it vanishes as the thermal dissipation in the resistance of the pull-up network. When there is $1 \to 0$ transition, then we will discharge C_l and the energy from C_l is dissipated on the pull-down network.



$$E_{0\to1} = C_L V_{DD}^2$$

$$E_{R=}\frac{1}{2}C_L V_{DD}^2$$

$$E_{c=}\frac{1}{2}C_L V_{DD}^2$$

Figure 1: Dynamic power consumption in CMOS.

The dynamic power dissipation is data dependent and it is dependent on the switching probability $P_{0\to1}$ so the switched capacitance is $C_{switched} = C_L \cdot P_{0\to1}$ s. Power is described by:

$$\begin{aligned} P &= \frac{energy}{transition} \cdot transition\ rate \\ &= C_l V_{DD}^2 f_{0\to1} \\ &= C_l V_{DD}^2 f_{0\to1} \cdot P_{0\to1} \\ &= C_{switched} V_{DD}^2 f_{0\to1} \end{aligned} \quad (1)$$

Energy consumed in N cycles is $E = n_{0\to1} C_L V_{DD}^2$, where $n_{0\to1}$is the number of transitions from 0 to 1 in N cycles. So the average dynamic power dissipated is:

$$P_{average} = \lim_{N\to\infty} \frac{E}{N} \cdot f = (\lim_{N\to\infty} \frac{n_{0\to1}}{N} C_L V_{DD}^2 f_{0\to1}) \quad (2)$$

Assume input switch every cycle:

$$\partial_{0\to1} = \lim_{N\to\infty} \frac{n_{0\to1}}{N} \cdot f_{0\to1} \quad (3)$$

$$P_{average} = \partial_{0\to1} C_L V_{DD}^2 f \quad (4)$$

From the above equation, it shows that this power dissipation is proportional to the computation rate of the circuit, so we can adjust the computation rate to meet power requirements, but it is also proportional to the supply voltage. Reducing the supply voltage to the sub-threshold range of the CMOS transistors is a very effective way to meet the low power requirement and this method is independent of assumptions about the speed or architecture of the system.

Transistor leakage is becoming an important issue and it has three components: drain leakage, junction leakage and gate leakage.

From a design perspective, threshold voltage (V_T) is often used to control leakage; a high V_T is very desirable to use in memory because memory is very inactive so it leaks most of the time. Leakage is also dependent on the V_{DD}.

1.2 Process Variations

Process variations are manufacturing variations that are usually modelled with normal (Gaussian) distribution. The main subjects that have variations are devices and interconnect; they both have variation in film thickness, lateral dimensions and doping concentrations.

Devices have variations on channel length L, oxide thickness t_{ox} and threshold voltage V_T. Channel length variation is caused by photolithography proximity effects. Oxide thickness is only significant between wafers and sometimes lumped into the channel length variation, but it is normally well controlled. Threshold voltage variation is caused by different doping concentrations and annealing effects. If a wafer has a greater dose near its centre than near its periphery, when the dose is delivered by an ion implanter then the threshold voltages might tilt across the wafer.

Interconnect has variations that occurs in wire width and spacing, metal and dielectric thickness, and contact resistance. Wire width and spacing vary due to photolithography and etching proximity effects. Metal and dielectric thicknesses are influenced by polishing. Contact resistance is determined by contact dimensions and varies through etching and cleaning steps.

The process variations may occur from one wafer to another and across an individual die; it also occurs between dice on the same wafer, and due to certain parameters it varies slowly and symmetrically. The variation is generally smaller across a die than between wafers.

The reasons for why process variations draw so much attention are that the ratio of variation over the device size is not constant, it affects speed, and it makes power and primary leakage less predictable.

Table 1 from the International Technology Roadmap for Semiconductors (ITRS) shows that the variation on threshold voltage is increasing along the feature scaling. The figures are shown in the table are for high performance designs.

Channel Length L(nm)	**250**	**180**	**130**	**90**	**65**	**45**
Threshold voltage V_T (mV)	450	400	330	300	280	200
Variation on V_T (mV)	21	23	27	28	30	32
(Variation on V_T)/V_T	4.7%	5.8%	8.2%	9.3%	10.7%	16%

Table 1: Higher fractional variability with feature size scaling down.

The Figure 2 shows the Intel 65nm 6T SRAM cell; the red circles highlight the two transistors that should match each other but the variations can occur between a few nanometres in the width or length, which will affect the SRAM cell behaviour.

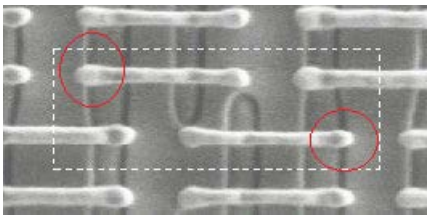
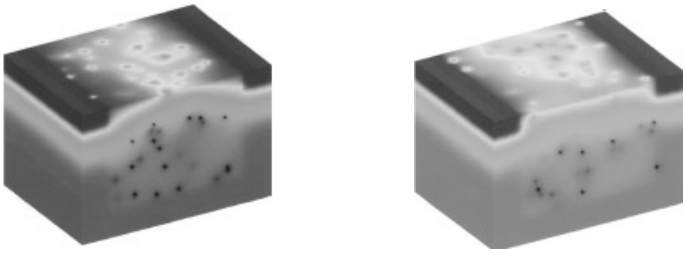


Figure 2: An Intel 65nm 6T SRAM Cell.

Random dopant can induce threshold voltage (V_T) variation. The devices are so small, we can actually count the number of dopant put on the channel, and there are usually 100-150 or less than 100 atoms. The threshold voltage (V_T) is determined by the position of the dopant atom. Figure 3 shows the two MOSFETs, which have the same physical layout and the same number of dopant atoms, but the positions of the dopant atoms are different, resulting in a different threshold voltage (V_T).



(a): V_T =0.78V (b): V_T =0.56V

Figure 3:

Figure 3: Two microscopically MOSFETs, both with 170 dopant atoms

in the channel depletion region. [1]

1.3 Aims and Objectives

The primary aim of this project is to evaluate the different memory cells under nano CMOS technology against nano-scale process variations. It is also intended to provide useful and detailed information that gives a full picture of SRAM cell design. The experiment results are validated through the extensive HSPICE and Matlab simulations. Cell layouts are studied by using the Cadence tool. In addition, the secondary aim is to develop new approaches to design a process-tolerant SRAM cell.

The main objectives and work are described as follows:

1. Literature Review and implantation: Study the relevant literature on different SRAM cells and selects the more promising and representative designs to analyse and design simulation models. Next, implement a framework for each design that can give a correlate write and read operation, and critically evaluate them against process variations.
2. Simulation Method: Research a method to simulate memory cells in a process variation environment and to analyse their performance from the output of the simulation.
3. Evaluation Method: To compare different cells at the same standard and test them in the same environment, then study their differences and elicit strengths and weaknesses.
4. Error Correct Code: Experimental evaluation of the efficiency of the existing ECC techniques used in memory cells in terms of error correction capability, power consumption, time and area overhead. ECC design suitability for SRAM design is investigated.
5. Design ECC-protected SRAM cell: Hardware implementation of hamming encoder and decoder is performed. The ECC-protected SRAM is capable of doing two error detection and one error correction.

1.4 Thesis Structure

This report is organised into six chapters as follows:

In the second chapter, the existing methods used in low-power CMOS design and the issues caused by these technologies are reviewed. The write and read operation for each design is described here. The SRAM yield limitations and scaling challenges are also introduced in this chapter.

In chapter three, the detailed lower power SRAM design technique we propose in this project is described and the implementation of five different SRAM blocks is presented. Also, their peripheral circuit designs are proposed in this chapter.

In chapter four, a comprehensive study and evaluation of the SRAM blocks are carried out. Several questions are answered here: what is the cost to achieve low power dissipation, how to cope scaling challenges of SRAM, what parameters are affected by process variation, what are the failure rates of each cell under process variation, do we get better performance by increasing the number of transistors in each cell, and what can be done next besides changing the cell schematic.

Chapter five introduces an alternative approach to dealing with errors occurring in the memory array. An ECC-protected SRAM array is presented which consumes less power and operates at a higher speed, and also occupies less area compared to other ECC schemes used in SRAM design.

In chapter six, we will conclude our work in this dissertation and highlight the major contributions of this research.

The final chapter provides suggestions for future work.

2 Introduction

Here we briefly review the five SRAM cells that are analysed in this project, a performance metric for evaluating these SRAM cells is introduced and how process variations affect memory cell performance will be explained. The main source of the reliability degradation is the system fault which will be explained in this chapter. The error correct code and the error correction scheme will also be introduced.

2.1 Embedded SRAM Memory Cell

Embedded memory is very important because we are now using high integration systems. All portable devices need huge amounts of memory. There are two approaches to meet the large memory requirement. One is to put the memory and microprocessor/DSP in the same package, which is called system in the package (SiP), and the other approach is to integrate memory on the same die along with other logics.

2.1.1 6T SRAM Memory Cell

This section gives a brief overview of the traditional 6T SRAM bit cell and its operation. Figure 4 shows a schematic for a conventional 6T SRAM cell. Once the reader understands a 6T SRAM cell then they can easily understand how other SRAM operate.

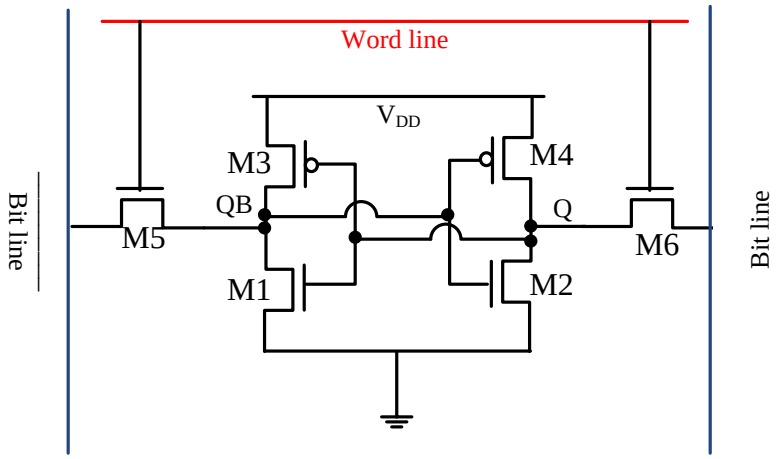


Figure 4: 6T standard SRAM memory cell.

Each bit can be stored in a SRAM cell, and M1, M3, M2, M4 transistors are combined together to provide two data nodes (Q and QB) to store a single bit and complement of this bit. The two extra transistors, M5and M6, are called access gates and are used to control access to the storage node during write and read operation. Access to the cell is enabled by word line (WL), which controls when the cell should be connected to bit lines: BL and BL_bar. The bit lines are used to transfer data from the read and write operation. The symmetrical structure of this SRAM allows for differential signalling, which makes small signal swings easier to detect.

An SRAM cell usually has three different states: hold state is when the circuit is idle, read state is when the data stored in the cell is requested and write state is when the data is needed to write into the cell.

Write Operation: The start of the write operation is applying the value to be written to the bit lines. Assume we want to write a "1" into the cell: BL is set to "1" and BL_bar is set to "0" then WL is asserted, so two access gates are enabled. The current from BL flows into data node Q and so data "1" is stored in this node; no current flows from BL_bar, so data "0" is stored in node QB. The feedback between these two inverters will help to write a perfect "0" and "1".

Hold State: When WL is not asserted, the access transistors M5 and M6 are disconnected from BL and BL_bar. But the two cross-coupled inverters continue to reinforce each other to keep the data stored in the data nodes as long as they are connected to power supply.

Read Operation: Assume we want to read what we stored in the cell during the write operation. The read cycle starts by pre-charging the BL and BL_bar (i.e. set both bit lines to "1"), and then assert word line (WL) to enable the access gates. Previously we stored "1" in the data node Q and "0" in QB, so the current will flow through M5 and M1 into ground because the input of M3 and M1 is the data stored in Q, which is "1", so the M1 transistor is enabled. There will be a voltage difference between BL and BL_bar that will be detected by the sense amplifier.

2.1.2 7T SRAM Memory Cell

This selected 7T SRAM bit cell is designed to reduce write power consumption. Figure 5 shows the schematic of this memory cell. The cell is formed by seven transistors and the functionality is slightly different to a conventional 6T SRAM cell. An extra transistor M5 is used to cut off the feedback connection during a write operation between the two cross-coupled inverters.

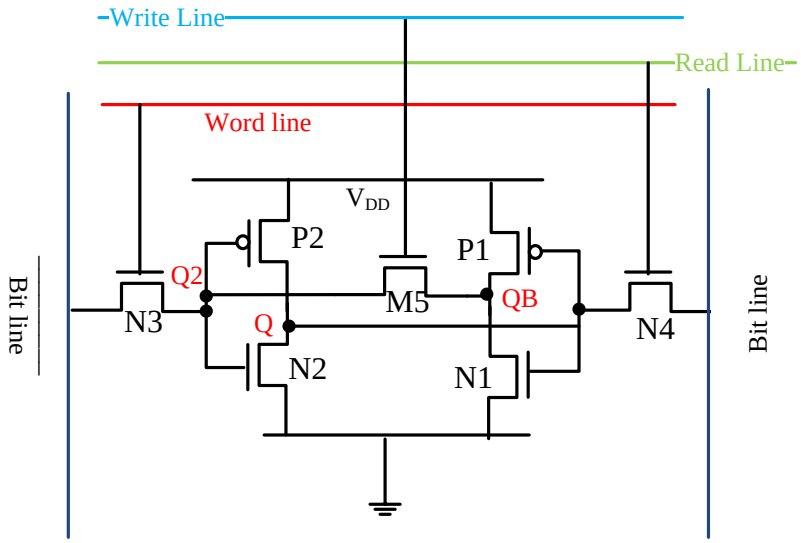


Figure 5: The schematic of a 7T SRAM bit cell.

Write Operation: Before starting a write cycle, first cut off the feedback between two cross-coupled inverters by disabling M5. The write operation is only performed by BL_bar, so set BL_bar to the complement value of the input data. The word line

(WL) is asserted which enables N3, and then the complement data will be stored in data Q2, driving P2 and N2 together to develop the input data which will be stored in data node Q. Q drives P1 and N1 to develop data which will be stored in QB. The data stored in Q2 and QB is the same. Then the WL is turned off so the memory cell will be disconnected from the BL_bar. The write line is asserted now to enable M5 to reconnect the feedback between two inverters.

Hold State: The word line, read line and write line are all turned off so the memory cell will be disconnected from the bit lines. The two inverters will be reinforcing each other to keep the data stored in the data node but there is voltage degradation between Q2 and QB.

Read Operation: The read operation for 7T SRAM is the same as 6T SRAM. But 7T SRAM will have a longer read time because the critical read path for this memory cell is through N1, N5 and N3.

2.1.3 8T SRAM Memory Cell

The 8T SRAM introduced here has a high read static noise margin (SNM). Also, the memory cell only requires a single bit line for read and write operations, which will consume less power. Figure 6 shows the schematic of this cell.

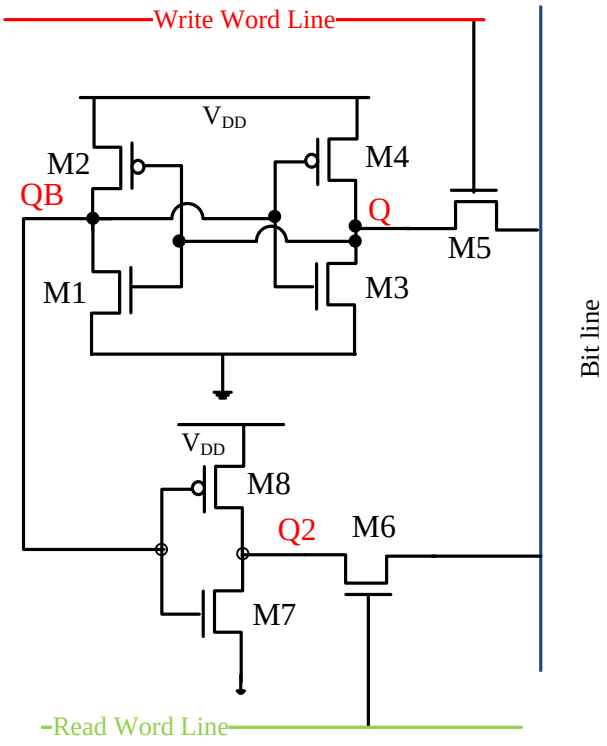


Figure 6: The schematic for 8T SRAM cell.

The reason that this memory cell has high read SNM is because the read path is separated from the write path by three extra transistors, M6, M7 and M8.

Write Operation: In this memory cell, a single bit line (BL) is used. BL is set to the value of the input data, and then the write word line is asserted while the read word line is kept off so M5 is enabled. Assume we want to write data "1" into the cell: data node Q starts charging and enables M1, which develops "0" on data node QB. M4 is then turned on to help write a good logic "1" into data node Q.

Hold State: During the hold cycle, the three inverters are disconnected from the BL. Data stored in the QB node needs to drive two inverters so the hold SNM for this cell actually decreases.

Read Operation: The read path is through M6, M7 and M8. Once the read word line is asserted, M6 is enabled then data can be read from data node Q2.

2.1.4 9T SRAM Memory Cell

The 9T SRAM cell introduced here has a very high read and writes SNM. This memory cell has two separate data access mechanisms for read and write operation. During a read cycle, the data nodes are isolated from bit lines so the read SNM is enhanced compared to a conventional 6T SRAM cell. The schematic of this memory is shown in Figure 7.

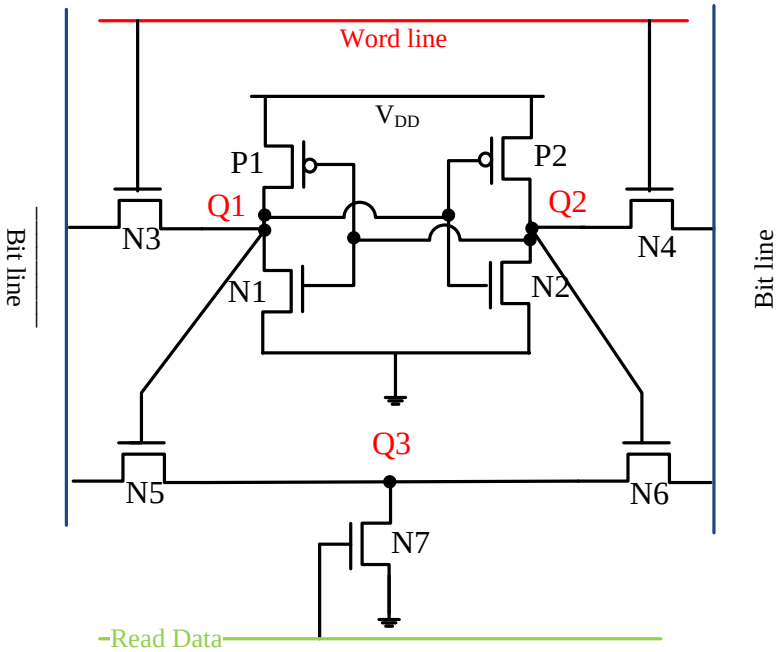


Figure 7: The schematic of 9T SRAM Cell.

The access gates N3 and N5 are controlled by the word line. The data will be stored in the data node Q1 and Q2. The bit line accessing transistors N5 and N6 is controlled by the data stored in the data nodes. N7 is controlled by the read data line.

Write Operation: The bit lines BL and BL_bar are set to the input data value and the complement of the input data respectively. Then the word line is asserted to enable N3 and N4 while the read data line is maintained low. The input data are transferred through N3 or N4 to data nodes to write a "0"or "1".

Hold State: When the word line and read line are turned off, the one side of the data node will still be connected to the bit lines, providing a less stable hold state.

Read Operation: The read word line is turned on at the beginning of the read cycle and the word line is kept off. Assume the data node Q2 has data "1" then M6 is enabled, the current flows from BL_bar to ground through M6 and M7. The voltage difference between BL and VL_bar can be detected by the sense amplifier, and then the data is read.

2.1.5 10T SRAM Memory Cell

The SRAM cell introduced here allows efficient bit-interleaving for soft error immunity and should provide better read stability. Figure 8 shows a schematic for this memory cell.

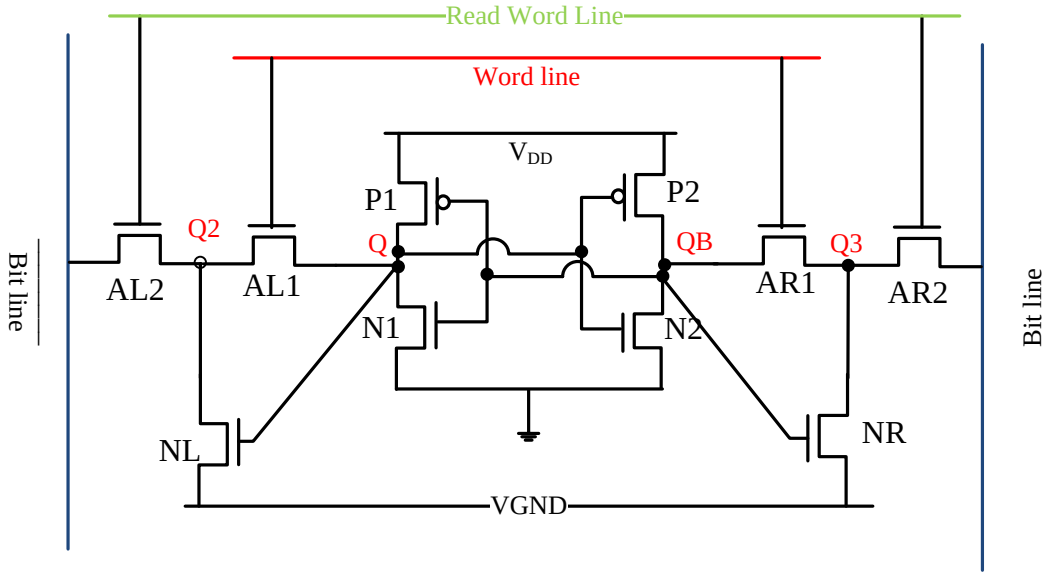


Figure 8: A schematic of a 10T SRAM cell.

This memory has a similar structure as the 9T SRAM introduced earlier, but it has quite different read and write mechanisms.

Write Operation: During a write cycle, the word line and read word line are both enabled and virtual ground (VGND) is set to high, because this memory has two extra access gates which give weak writability, so the bit lines BL and BL_bar are boosted stronger than V_{DD}. The input data and complement of this data are stored in the data node Q and QB.

Hold State: When the word line and read word line are both disabled, the VGND is kept high. The cell will be disconnected from the bit lines.

Read Operation: In read mode, the word line is enabled while the read word line is kept off, and VGND is forced to ground. Since the read word line being disabled made the data node Q and QB isolated from the bit lines, it enhanced the read SNM. Assuming that data node QB has data "0", the BL is discharging through AR2 and NR

and then the sense amplifier will detect the voltage difference between BL and BL_bar to output the data stored in the memory cell.

For SRAM memory cell scaling down to 45nm/35nm feature sizes, there are some challenges in terms of signal noise margin, read and write stability. We will use conventional 6T SRAM to explain these issues.

2.2 SRAM Yield Limitations Due to Process Variation

Memory technology is challenged with scaling as we face consequences from reducing supply voltage and process variations [2]. There are four elements that will determine how memory works: read stability, hold stability, access time and write stability.

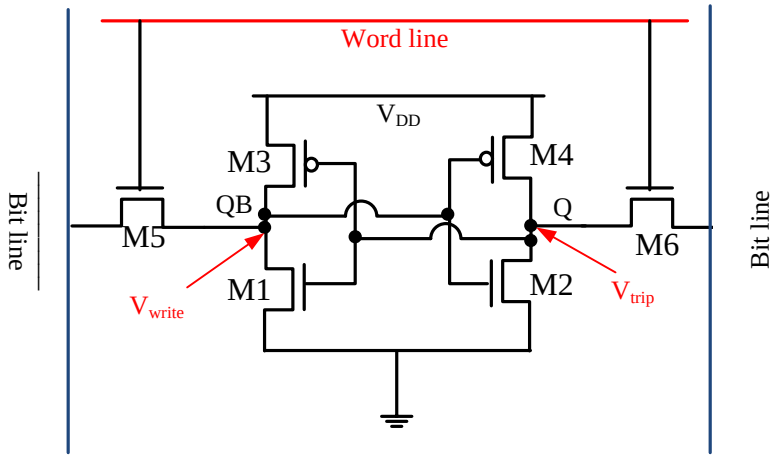


Figure 9: 6T standard SRAM memory cell.

For reading a 6T memory as shown in Figure 9, assume we have "0" stored on the data node Q and "1" stored on the data node QB. We pre-charge both bit lines to V_{DD}, which creates a large capacitance on the bit line then one of the bit lines is discharged through the pass transistor M6 and one through the inverter transistor M2; during this read operation we may flip the memory cell.

When we turn on transistor M6, we will see a glitch voltage at node A, which is shown in Figure 10, and this voltage goes so high it may trip the inverter then the data stored here may flip. When designing a memory cell we need to make sure performing a read operation will not destroy the content; this is called read stability.

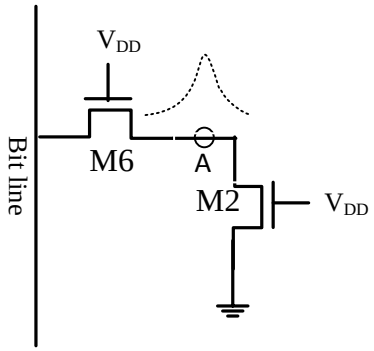


Figure 10: Memory cell can flip due to an increase in the "0" storage node A above the trip voltage of the other inverter during read operation.

Figure 9 shows that in a typical 6T SRAM memory cell we have a left pair of inverters, a right pair of inverters and two access devices. To perform a read operation, we put V_{DD} on both sides, the read stability will be determined by the voltage transfer characteristic (VTC) so we can draw a diagram of the left voltage V_{Left} versus the right voltage V_{right} where V_{left} is describing the behavior of the left inverter and V_{right} is describing the behavior of the right inverter. The noise margin is determined by the stable state on the V_{left} axis; once the state disappears, we will not have a base state then we cannot store data any more.

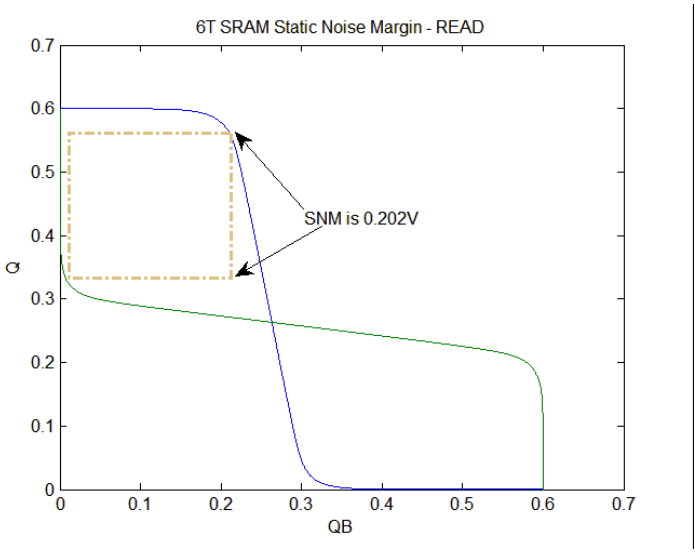


Figure 11: Read signal noise margin of a 6T SRAM memory cell.

The left side inverter is described by function FI and the right side inverter is described by function FII. The noise margin is defined by the size of the opening window and is created by function FI and function FII; assume that if we move the curve closer then we will narrow the opening window. Eventually we will lose the entire noise margin, which is shown on the cross-section of the graph.

The relationship between function I, function II and signal noise margin can be described by

$$FI(V_{left}) = V_{left} + C \tag{5}$$

$$FII(V_{left} - S) = V_{left} - S + C \tag{6}$$

$$S = FII(V_{left} - S) - FI(V_{left}) \tag{7}$$

$$\frac{\partial FI(V_{left})}{\partial(V_{left})} - \frac{\partial FII(V_{left}-S)}{\partial(V_{left})} = 0 \tag{8}$$

Hold stability means that when we turn on the cell while it is on standby mode, the data stored does not change; however, we may lose data if the data retention current is not able to compensate for the leakage current.

Hold stability is similar to read stability analysis but without access transistors; Figure 5 shows the curve of the V_{left} versus the voltage V_{right} without the access transistors. In order to keep the memory cell stable in the standby mode, the PMOS transistor M4

must provide enough leakage to compensate for the leakage leaking in NMOS pull-down and access the transistors.

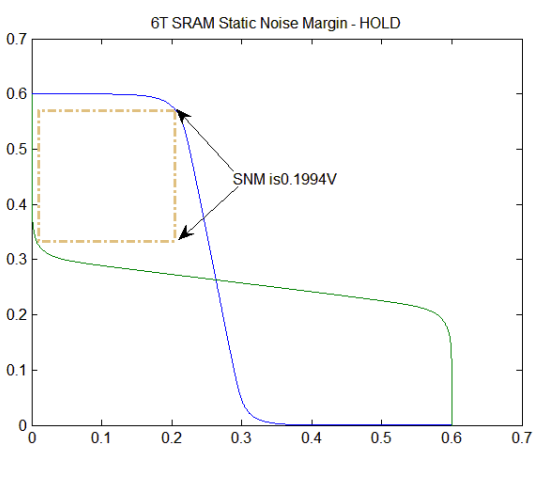


Figure 12: Hold signal noise margin analysis of a 6T SRAM memory cell without access transistors.

When performing a read operation, the cell must provide ΔV between the bit lines for a sensing amplifier to come in within the maximum tolerable time; the time required to produce this pre-specified ΔV is called access time. The access time can be formulised by the sum of the bit line current and integrate them, which is shown below.

$$T_{access} = \left| \int_{V_{DD}}^{V_{DD}-\Delta V_{BL,R}-\Delta V_{min}} \frac{C_{bl,l}\partial V_{bl,l}}{I_{bl,l}} \right| = \left| \int_{V_{DD}}^{V_{DD}-\Delta V_{BL,R}} \frac{C_{bl,R}\partial V_{bl,R}}{I_{bl,R}} \right| \quad (9)$$

Write stability can be defined as during a write operation, the storage node that has "1" may not be reduced below the trip voltage of the other inverter before the word line is discharged, which is shown in Figure 9. We need V_{trip} greater than V_{write} for good write-ability. Figure 13 shows write SNM for the curve of V_{left} versus voltage V_{right} while writing "0" into date node.

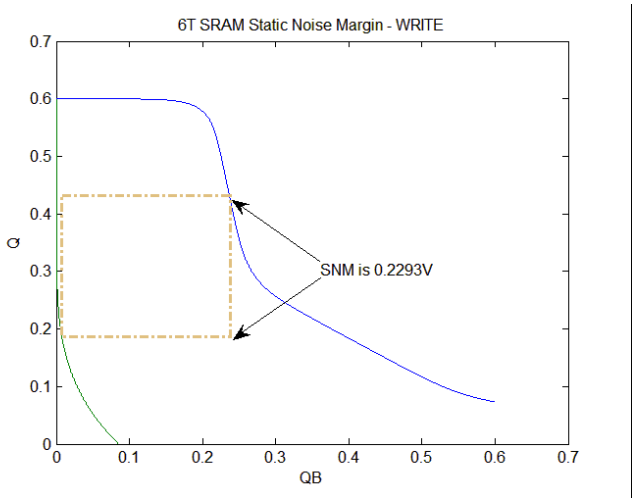


Figure 13: Write signal noise margin analysis of a 6T SRAM memory cell.

The write time can be described by:

$$T_{write} = \left| \int_{V_{DD}}^{V_{trip}} \frac{C_R(V_R)\partial V_R}{I_{in(R)}(V_R) - I_{out(R)}(V_R)} \right| < T_{WL} \quad (10)$$

The analysis techniques and methods used in this project to find out the access time and SNM are described in next chapter.

2.3 SRAM cell Scaling Challenges

As we move into smaller feature sizes, there are some stable operation issues occurring in the SRAM cell design. For instance, in a standard 6T SRAM cell we lose the write and read margin as we scale down supply voltages, because, at the same time, the minimal operation voltage is going up because of the process variation. Signal noise margin is getting smaller as well. It may be that a separate power rail for SRAM is needed, but this is bad for leakage.

As for leakage, there are some other concerns as well: on every node we get more transistor leakage components. Standby leakage and active leakage need to be dealt with to meet different design requirements.

From a layout perspective, a SRAM cell has specific design rules like gate extension past active, gate-end to gate-end space, N^+/P^+ space, contact-gate space and contact/active overlap. In order to meet these design rules we need the manual tuning of layout/SRAFs/OPC and bit swap, stretch contacts and staggered poly.

There are some possible solutions from the process point of view for 6T SRAM cell scaling. For operation stability issues, we can add one mask for deep N-well to isolate Nch so we can well tune voltage to adjust the threshold voltage, separating the array and word line supply voltage to optimise the signal noise margin over the trip voltage, and reducing channel doping in order to reduce transistor mismatch.

For leakage problems, by repairing weak retention bits so even lower retention voltage can be enabled, data retention can be enabled for selective blocks on the system level design, and by using high-k gate dielectric material to reduce the growing gate leakage component.

2.4 Low Voltage Nanoscale Embedded SRAM Cell

Challenges to low-voltage embedded SRAM cells are: maintaining signal charges that are used to fight soft errors, but with scaled supply voltage and reduced capacitance we have a smaller amount of charges on the node; maintain signal voltage because we want stable sensing and also a wide voltage margin for a readable signal; and reduced cell size as all other logic circuits are getting really small.

Signal charge Q_S is approximately equal to soft error Q_{crt}, the cell with smaller Q_S will be more vulnerable to soft error.

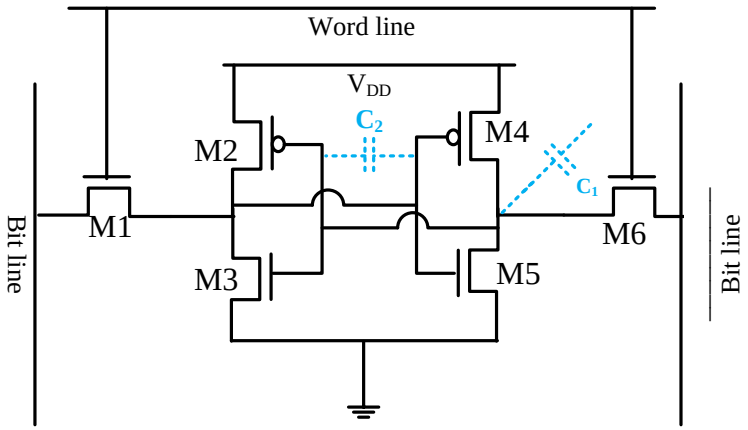


Figure 14: Parasitic capacitance on the 6T SRAM Cell.

Signal charge is the product of parasitic capacitance and supply voltage which is shown in equation (11), and the two parasitic capacitors are shown in Figure 14.

$$Q_s = (C_1 + 2C_2) \cdot V_{DD} = C_s \cdot V_{DD} \tag{11}$$

C_1 is diffusion capacitance and C_2 is the gate capacitance. These two capacitors are getting smaller with device scaling.

There are some signal and noise issues in the SRAM cell. As shown in Figure 14, the capability of having enough current I_L going to NMOS transistor M3 in order to discharge the bit line to get a sizable signal to read out the cell is determined by the signal voltage V_s at this transistor where V_S is equal to$V_{DD} - (V_T - \delta V_T)$, so the threshold voltage variability of M3 transistor is very important.

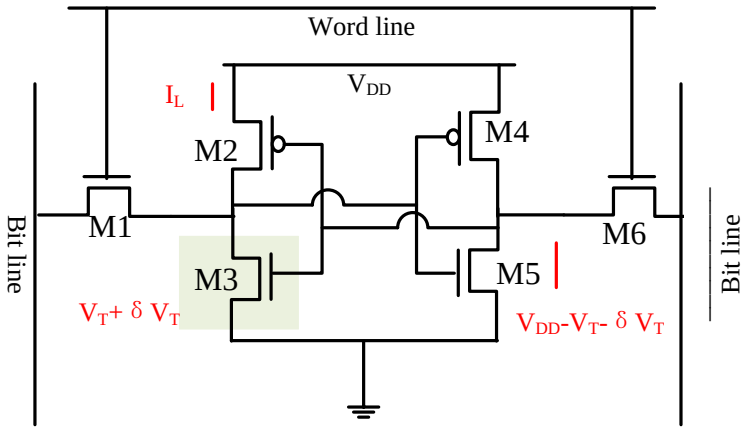


Figure 15: Signal voltage showing in 6T SRAM Cell

Cross-coupled MOS transistors need a high threshold voltage usually around 0.5V to 0.7V to reduce leakage δV_T ; δV_T is the V_T mismatch between paired MOS transistors and the value of this δV_T is physically and statistically large due to many cells with small transistors in a chip. With δV_T added on top of V_T plus scaled V_{DD}, we will not be able to get enough signal to read the cell out in a reasonable amount of time.

δV_T increases with device scaling so it makes the nanoscale SRAMS difficult to design. It can mathematically be described as:

$$\delta V_T = \sqrt{2}\,\Delta V_T \quad \text{Where } \Delta V_T \text{ is the variation of } V_T \tag{12}$$

2.4 Variation-tolerant Layout

Standard cells tend to be placed regularly on the grid in rows and have the same height, but when it comes to routing there are many problems generated by the router like density variations, variations in coupling capacitance, resistivity and interference. All these problems are because the router tries to run a short path in a random fashion. In future work, the metal wires may be pre-positioned and placed really regularly on the grid, and then designers make the choice on how to connect the circuit structures, where to break them and where to place vias. By doing this, the density structures will be more regular and potentially a lot more predictable. In order to minimise variations, the fabrics idea is doing atomic regularity, which is starting from circuits to logic then to routing.

In this project, we have drawn all memory cell layouts in a custom fashion and analysed their structure complexity.

2.5 Variation-tolerant Design

In order to overcome the process variation problems, variation-tolerant design is one of the main concerns in this project. Figure 15 shows that there are tradeoffs we can make, using large transistors so we will get better target frequency probability, but this is paid for in terms of power and size. Using high-V_T devices will get better target frequency probability, but will cost in power and performance. In terms of logic depth, we should add more pipeline stages (less logic depth) or less pipeline stages. Assuming all the variations are random, a long path will have a smaller ratio of variation of propagation delay/propagation delay than the short path. But note that if the variation is a systematic variation that is die to die, then this adjustment will not work. The smaller number of critical paths in micro-architecture will have better target frequency probability than larger numbers.

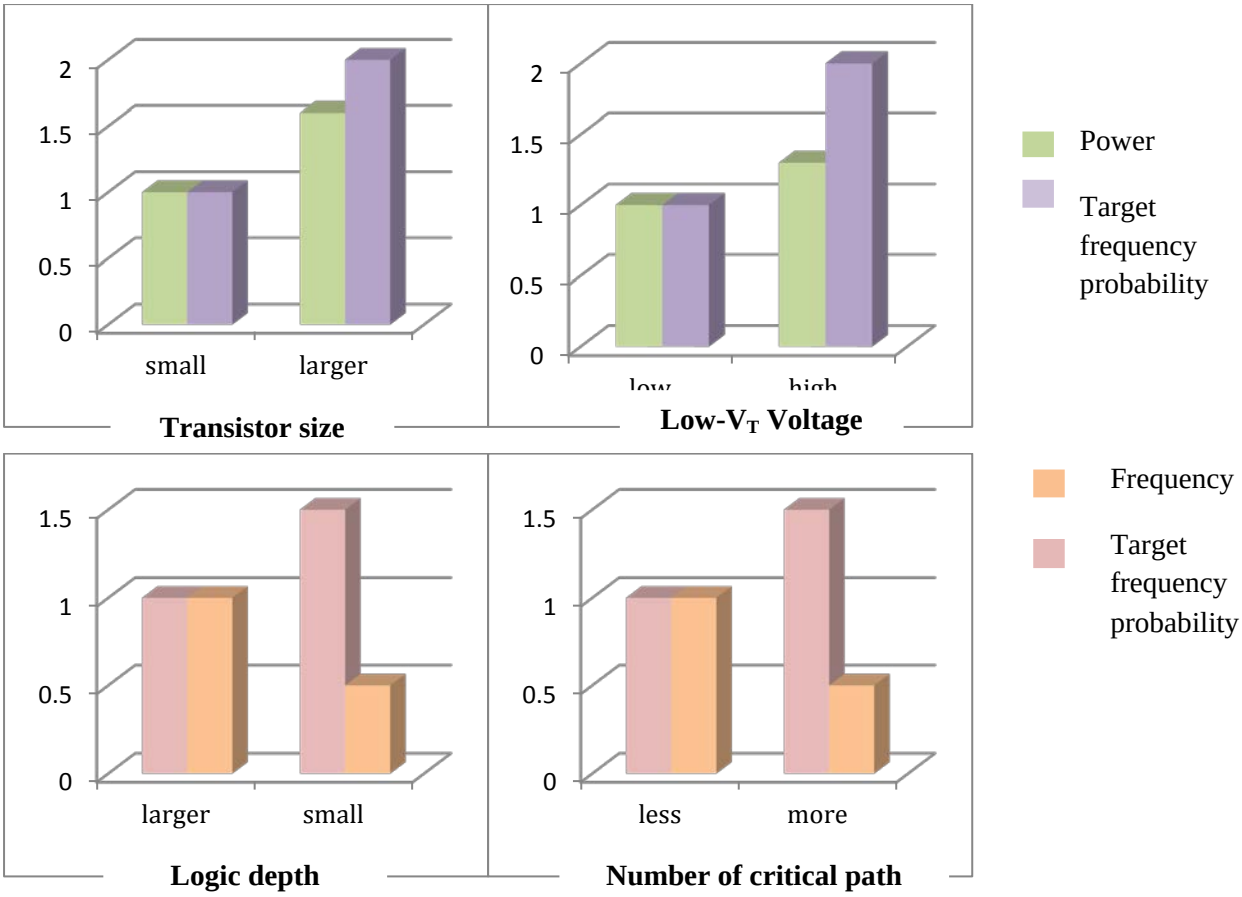


Figure 15: Overview of balance power and frequency with variation tolerance. [3]

2.6 Error Correct Schemes

Fault-tolerant design is designing the system so that it will continue to operate correctly even in the presence of certain specified faults, and this can be achieved by using hardware redundancy, software redundancy or time redundancy. One of the most common methods to improve memory yield is by using the redundant elements to replace the faulty cells.

Error detection and correction are techniques that enable reliable delivery of digital data over an unreliable system. The SRAM memory cell can become highly unreliable due to process variation and soft errors. Error detection techniques can be used to detect incorrect data stored in the memory while error correction enables reconstruction of the original information. Error detection and correction schemes can be either systematic or non-systematic, while in this project we will focus on the systematic scheme in which the data is stored in memory with a fixed number of check bits (or parity data) and some deterministic algorithm will be used to detect errors. The reason for using this scheme is that the errors caused by process variation and soft errors occur randomly, and with enough information we can predict a certain probability of error occurrence.

ECC technique has widely used in memories such as L2 and L3 caches of Itanium processor, L2 cache of Power4, and L2 cache of UltraSpace processor.

In general, there are two structurally different types of codes used: block codes and convolution codes.

Block codes process information on a block by block basis, treating each block of data independently. For example, the encoder receives a block code that will divide this block information into message blocks of K bits. There are a total of 2^K different possible message blocks. Each message block will be encoded into an N-bit codeword, therefore for 2^K different possible message blocks there are 2^K different possible codewords. These 2^K different codewords of length N are called an (N, K) block code and $K \leq N$. Hamming code, Bose-Chaudhuri-Hockquengem (BCH) and Reed Solomon codes are block codes.

Hamming code can correct single errors and is suitable for fault tolerant SRAM design. BCH code can correct random errors. Reed Solomon code can correct blocks of errors and is used for CD and DVD scratches. In this project, Hamming code is investigated in detail.

Convolution codes are not usually used in memory designs because each encoded message block is not only dependent on the corresponding K-bit information block, but also on the previous message blocks.

3. Low Power SRAM Designs

3.1 Overview of SRAM Block Structure

How can we implement a 1MB memory? We do not want a linear array that is one million bits long and has a skinny shape. We want to put them in a square shape structure with row ID and column ID to indicate bit cells, which will reduce the number of access pins and the length of the word line.

For 1MB = 2^{23} bits = 2^{12} rows × 2^{11} columns, we need a 12-bit row address decoder to select a single row and 2^{11} columns means 256 numbers of 8-bit words so we need a column address decoder to select one of these words. Normally the rows lines are called word lines and column lines are called bit lines. Figure 16 shows an example of SRAM architecture. To select a single row, we will use a row decoder and only one of the word lines can be high at the same time. This complete row will then be read by sense amplifiers; the reason we use sense amplifiers is that the voltage difference will be a small ΔV which will be detected by the sense amplifier and amplified to a full swing CMOS level output signal. These signals will be feed into a column decoder to select m-bit words.

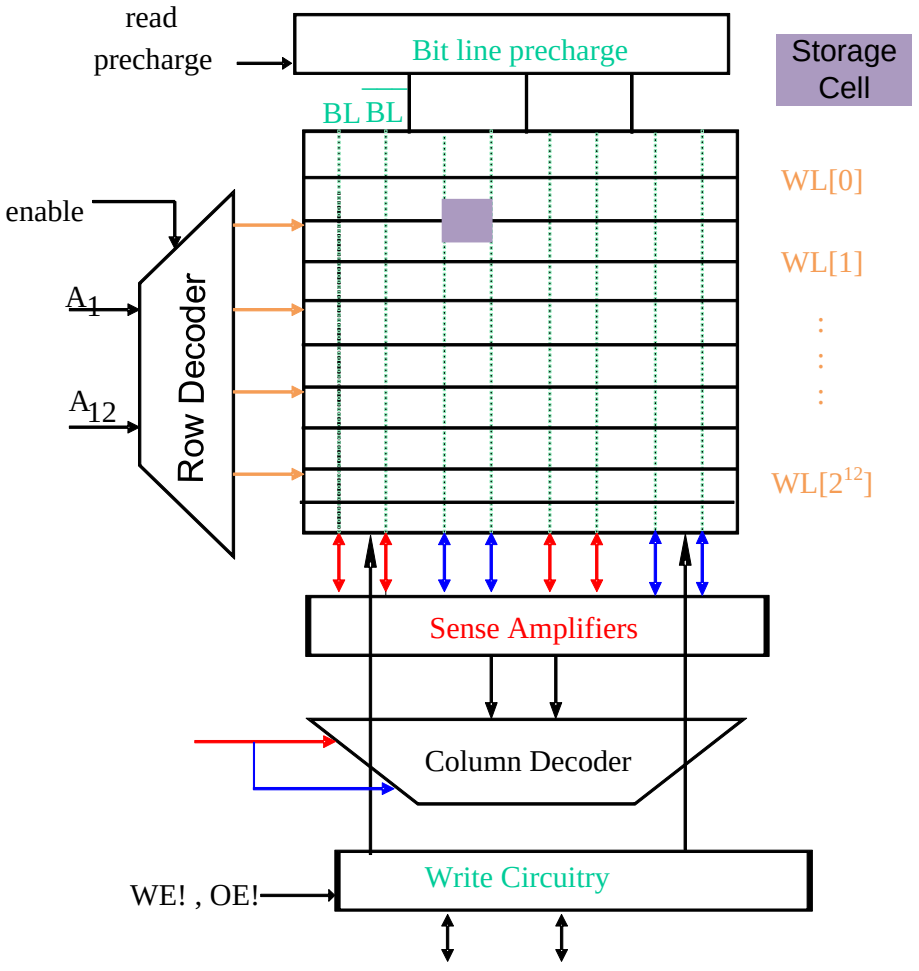


Figure 16: Overview of SRAM architecture.

3.2 Peripheral Circuit Design

In this project, we only analysed bit cell so we do not need a row decoder and column decoder, but we need other peripheral circuits to simulate the read operation, write operation and hold state. These peripheral circuits will be introduced here.

3.2.1 Sense Amplifier and Bit Line Pre-charge Circuit

A sense amplifier is an important component in memory design because it affects read speed and power. The primary function of a sense amplifier is to detect the small voltage difference between bit lines during a read cycle and amplify this small analog differential voltage into a full swing digital signal, therefore helping to reducing the time required for a read operation. SRAMs do not feature data refresh after data required so the sensing operation must be non-destructive.

A modified latch-type voltage sense amplifier (SA) is used to read the content of the memory cell. It was introduced by Kobayashi *et al*. in 1993 [2]. The conventional latch-type SA has two nodes that are used as input and output terminals at the same time so the bit lines cannot directly connect to the SA since the circuit will try to discharge the bit lines during the decision phase (i.e. SA starts to have voltage difference on the two input nodes and decides to flip in one or the other direction), which will cause longer delays and larger power dissipation.

This modified SA is shown in Figure 17. This SA has a strong positive feedback, the bit lines BL and BL_bar are connected to the input of transistor M5 and M6 and these two transistors control the latch circuit. The small difference of current flows into M5 and M6 converts to a full swing voltage signal at the output SO and SON. For this design, we can directly connect it to bit lines.

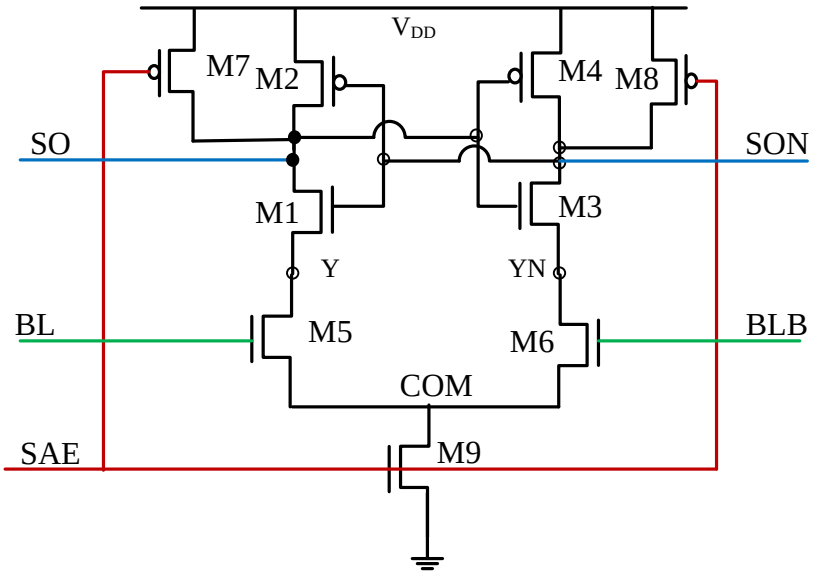


Figure 17: Modified latch-type voltage sense amplifier.

The time to enable SA is important because once the sensing process starts, it cannot recover unless the circuit resets to the meta-stable point where v (so) = v (son). But from the delay point of view, the SA should be enabled at the smallest possible input voltage difference then the output will be generated when the voltage swing ΔV_{in} is reached.

During the off state, the outputs SO and SON are kept high by turning on transistor M7 and M8. The sensing operation starts by setting the SAE signal high to enable transistor M9; the COM node will immediately be pulled down then transistor M5 and M6 will be enabled. Because of the voltage difference at the input of M5 and M6, these two transistors start to discharge SO and SON nodes at different discharging speed. Transistor M2 and M4 are kept off until the output voltage reaches (V_{DD}-V_{tp}). Then one of the NMOS transistors in the latch circuit will be cut off. The current flow stops automatically after the sensing process is complete so there is no static power dissipation.

The transistor sizes are considered here. For relatively small V_{DD}, the transistor M9 needs to discharge the COM node quickly once the SAE signal goes high; M5 and M6 should be stronger than the latch circuit to develop drain current difference so the W/L ratio is set to 5 for M9 and M5, and for M6 the W/L ratio is set to 2. The latch circuit is used for minimal transistor sizes.

The bit line differential voltage comes in two different values: the small bit line differential can reduce total read access time but the larger bit line differential is beneficial for more reliable sensing, which will be more tolerant to process variation. Figure 18 shows the bit line conditioning circuitry used in this project.

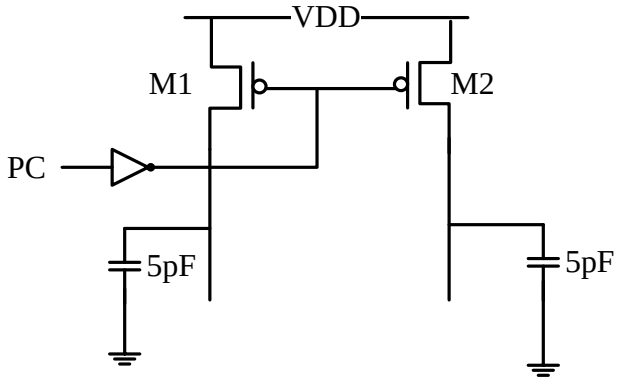


Figure 18: Bit line conditioning circuitry.

Bit line conditioning circuitry is used to pre-charge bit lines to V_{DD} before any write or read operation. In this project, the bit line conditioning circuitry we designed meets several requirements, such as charging the bit line from ground to V_{DD} in 7.18ns, the pre-charge signal is clocked in order to reduce power dissipation and very small bit line noise.

The reason we used a 5pF capacitor here is that normally bit lines are attached to more than one bit cells, which increases the bit lines' capacitance, and using a bigger capacitor here will help us to simulate the design in a more realistic environment.

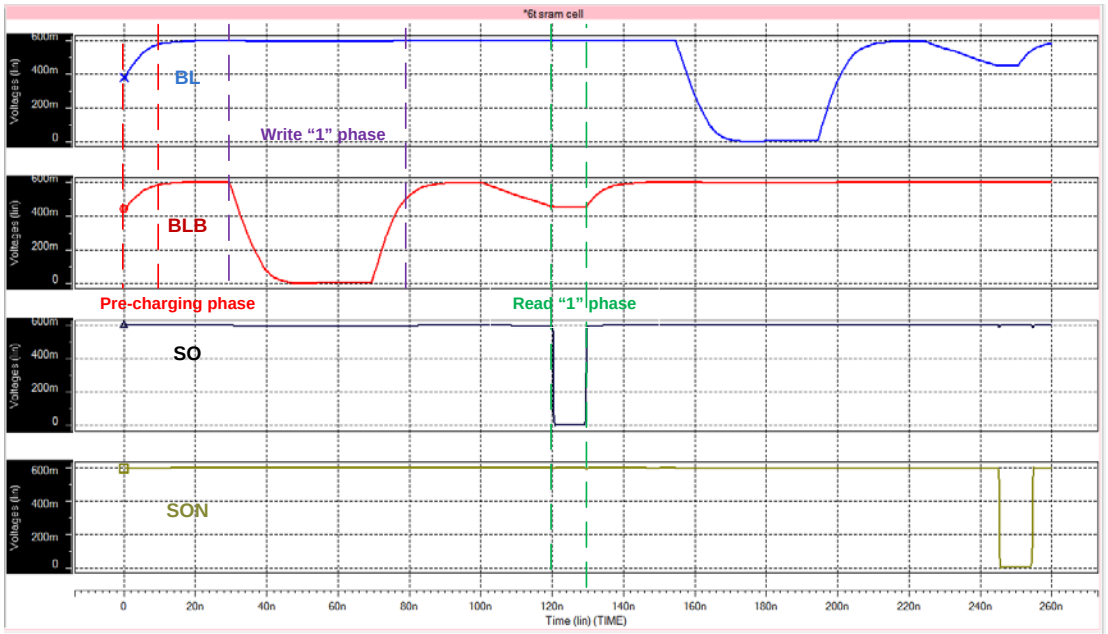


Figure 19: Bit line transient behaviours for write data (1, 0) and read data (1, 0) operation of a 6T SRAM.

Figure 19 shows bit lines' transient behaviours during the write and read operation of a 6T SRAM. The total read time is 0.1995ns and SA sensing time is 0.0385ns.

3.2.2 Write Driver

The function of the write driver in this design is to quickly discharge one of the bit lines from V_{DD} to ground once the data signal is available. The write driver is active first then enables the word line for a correct write operation. There are some typical write driver designs that use pass-transistors or AND gates, but here we used a NMOS combined write driver because NMOS has a better feature for discharging a circuit. This design is presented in Figure 20. When the write enable signal goes high then data becomes available, if data is logic "1" then through an inverter feed into transistor M2 to keep the bit line high, but the input of transistor M1 is directly connected to a data signal so bit line_bar is discharging to ground.

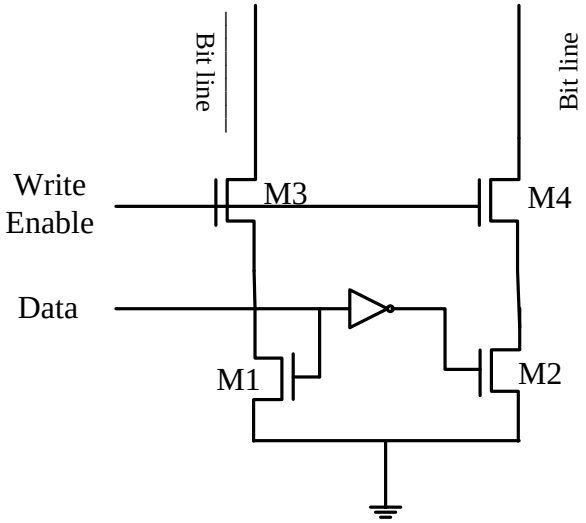


Figure 20: NMOS-type write driver circuit.

Even though we discharge one of the bit lines to ground level during a write operation, which is presented in Figure 19, a write operation can be faster than a read operation. Only one write driver is needed for each column so we can size up the transistors used in the write driver if necessary.

3.3 SRAM Designs

3.3.1 6T SRAM designs

In order to simulate a correct write and read operation, we have designed one bit cell with a pre-charged circuit and a write driver with sense amplifier as presented in Figure 21.

There are five input pins: PC, WL, Data, WE and SAE. The PC signal controls when we are going to pre-charge the bit lines to V_{DD}. The WL signal goes high when the data needs to write into the memory cell and when the data stored in the memory is required. The Data signal represents the logic data we want to write into the memory. The WE signal controls when to activate the write driver circuit. The SAE signal controls the sensing amplifier. These five input pins are connected to five pulse voltage sources that are manually clocked. The clock signals are designed to meet several requirements, such as the PC signal sets to high only when the memory cell is not connected to bit lines. The write driver is only enabled when both bit lines are pre-charged to V_{DD} then the WL signal is enabled. For the WL signal, the time is considered. SAE sets to high only when bit lines have developed a small ΔV. The simulation waveform is shown in Figure 22.

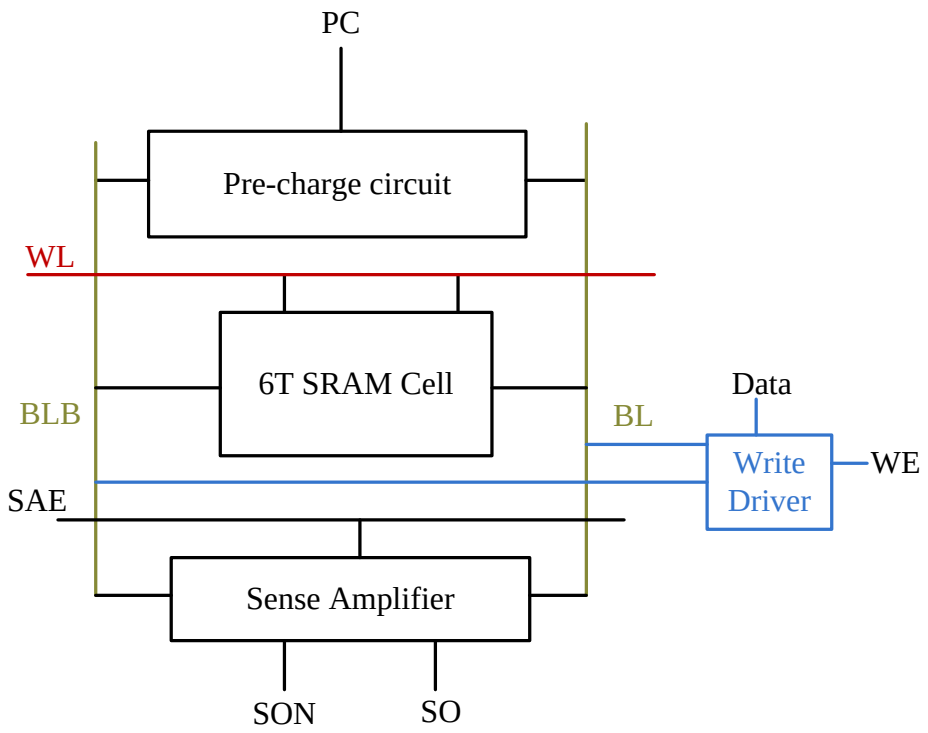


Figure 21: 6T SRAM bit cell with peripheral circuit.

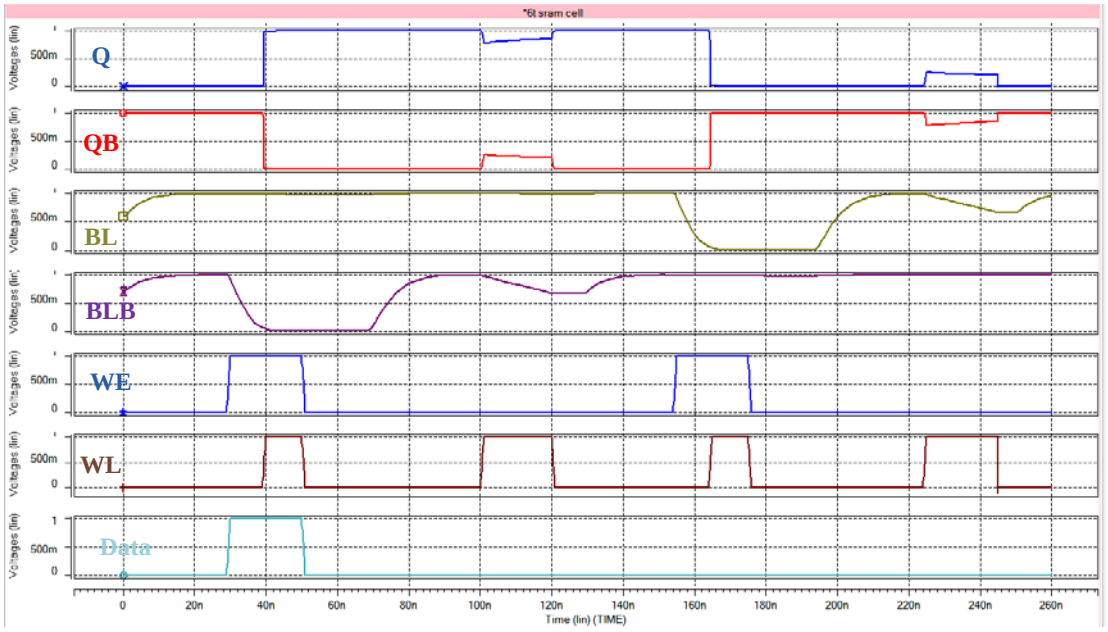


Figure 22: Write (1-0-1) and read (1-0-1) operation for 6T SRAM.

3.3.2 7T SRAM designs

7T SRAM has a different write and read mechanism that requires two extra input signals to simulate write and read operations. PC, Data, WE and SAE signals have the same functionality as a 6T memory block. WL is only enabled during the write operation. W is enabled after the write operation and kept high. R and WL are set to high to start a read cycle. A 7T SRAM block with peripheral circuits are shown in

Figure 23 and the simulation waveform for this block circuit is shown in Figure 24.

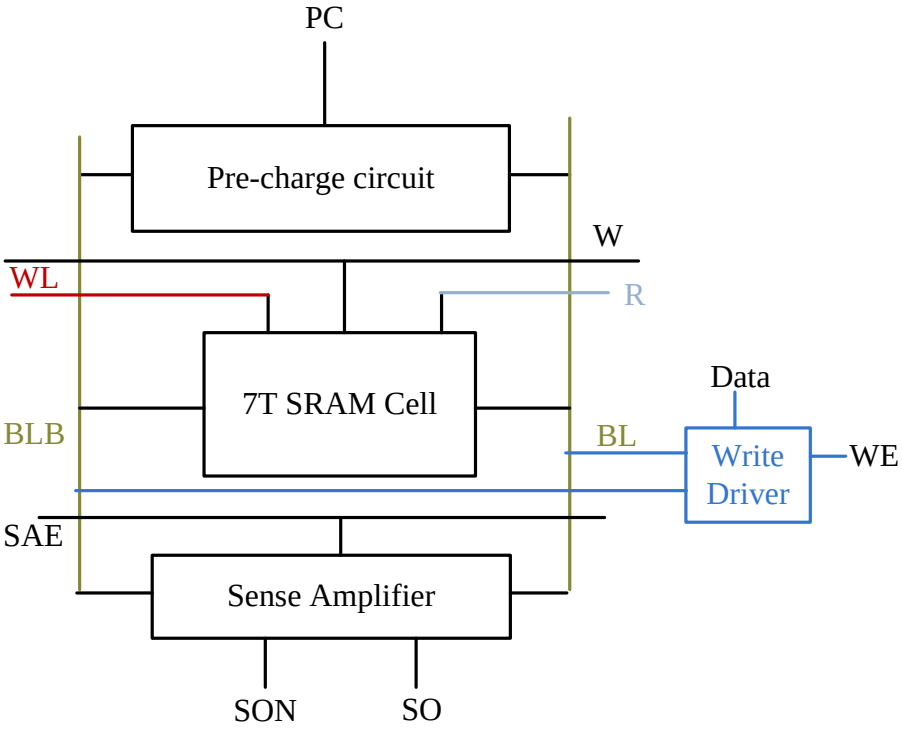


Figure 23: 7T SRAM bit cell with peripheral circuit.

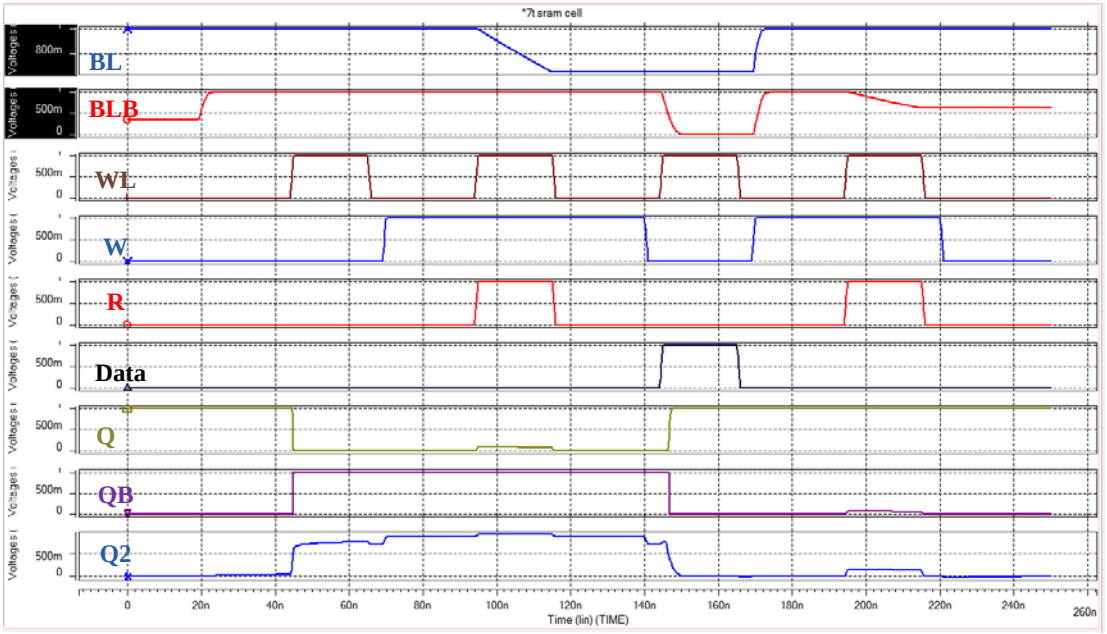


Figure 24: Write (0-1) and read (0-1) operation for 7T SRAM.

3.3.3 8T SRAM designs

8T SRAM has only one bit line for the write and read mechanism, and it has different input signals for write and read operations. PC, Data, WE and SAE signals have the same functionality as a 6T memory block. WWL is only enabled during the write operation. RWL is enabled to start a read cycle. Because there is only one bit line

from memory, a reference signal is used in the sense amplifier. An 8T SRAM block with peripheral circuits is shown in Figure 25 and the simulation waveform for this block circuit is shown in Figure 26.

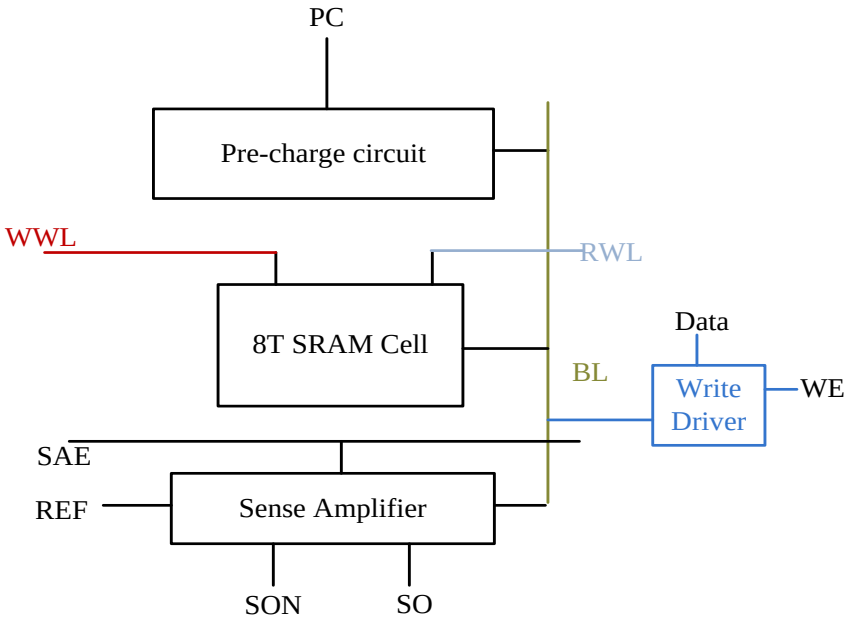


Figure 25: 8T SRAM bit cell with peripheral circuit.

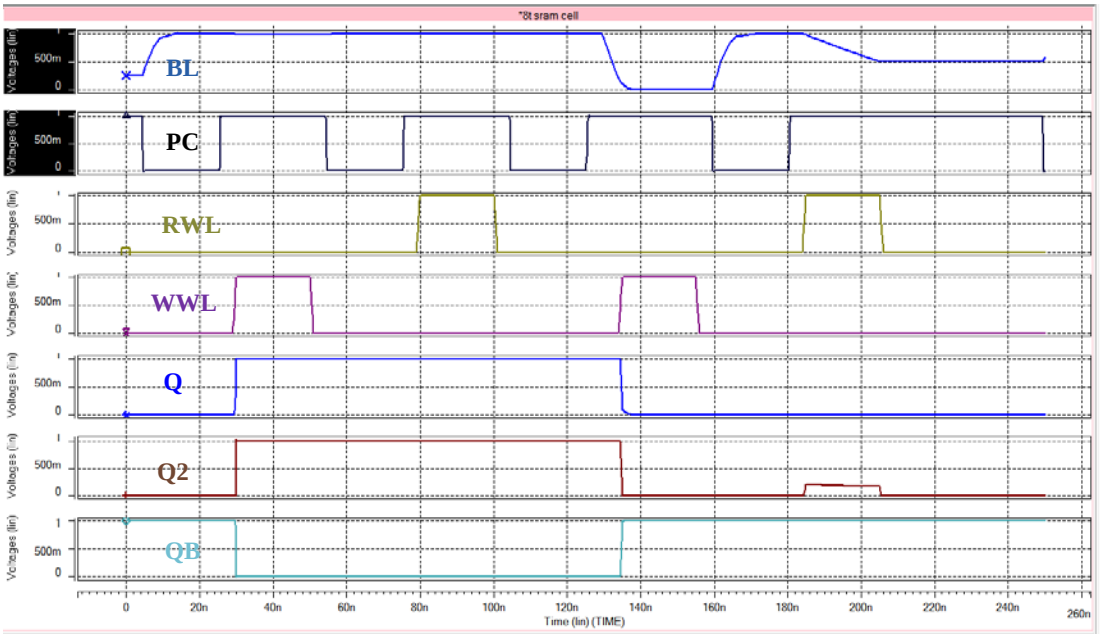


Figure 26: Write (1-0) and read (1-0) operation for 8T SRAM.

3.3.4 9T SRAM designs

9T SRAM has different read mechanism compared to other memory cells so it has different input signals for write and read operations. PC, Data, WE and SAE signals have the same functionality as a 6T memory block. WR is enabled during the write operation. RD is enabled to start a read cycle. Because the memory is not physically

disconnected from the bit lines, the PC signal should be considered carefully. A 9T SRAM block with peripheral circuits is shown in Figure 27 and the simulation waveform for this block circuit is shown in Figure 28.

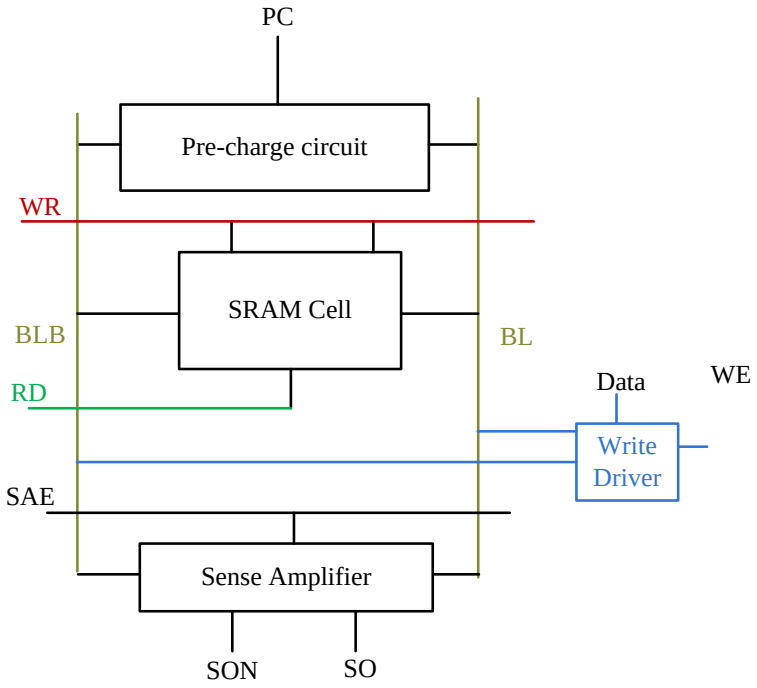


Figure 27: 9T SRAM bit cell with peripheral circuit.

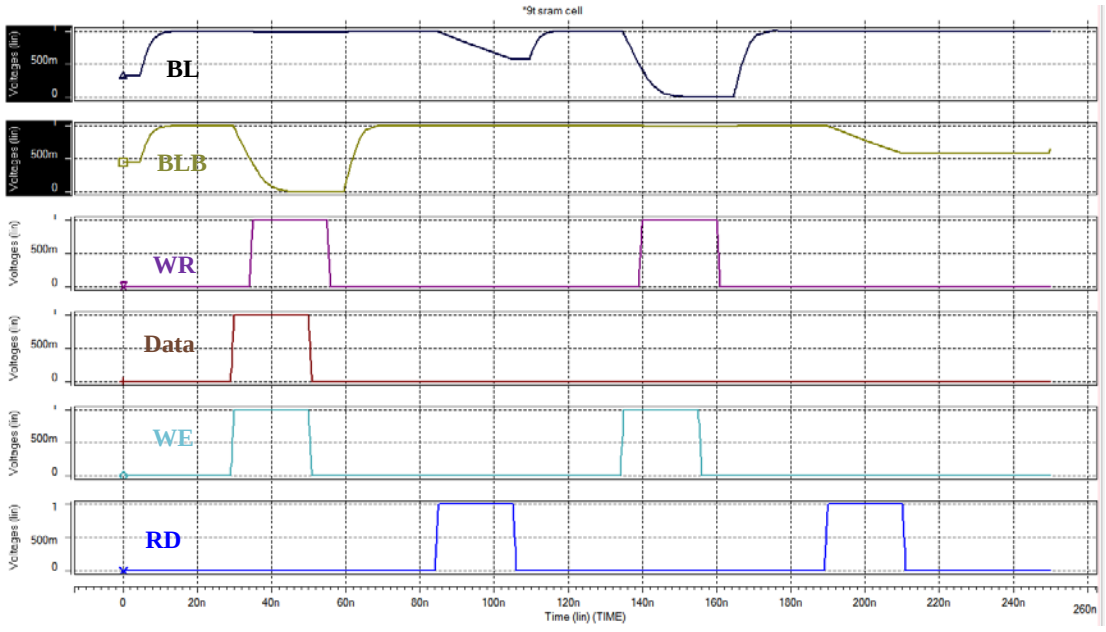


Figure 28: Write (1-0) and read (1-0) operation for 9T SRAM.

3.3.5 10T SRAM designs

10T SRAM has a very similar write and read mechanism compared to the 9T memory cell but it has adopted a virtual ground technique for read operations. PC, Data, WE and SAE signals have the same functionality as a 6T memory block. WWL and WL

are enabled during the write operation. WL is enabled to start a read cycle. The virtual ground control signal is normally generated by the read operation enabled (RD) signal and clock (CLK) signal together. A 10T SRAM block with peripheral circuits is shown in Figure 29 and the simulation waveform for this block circuit is shown in Figure 30.

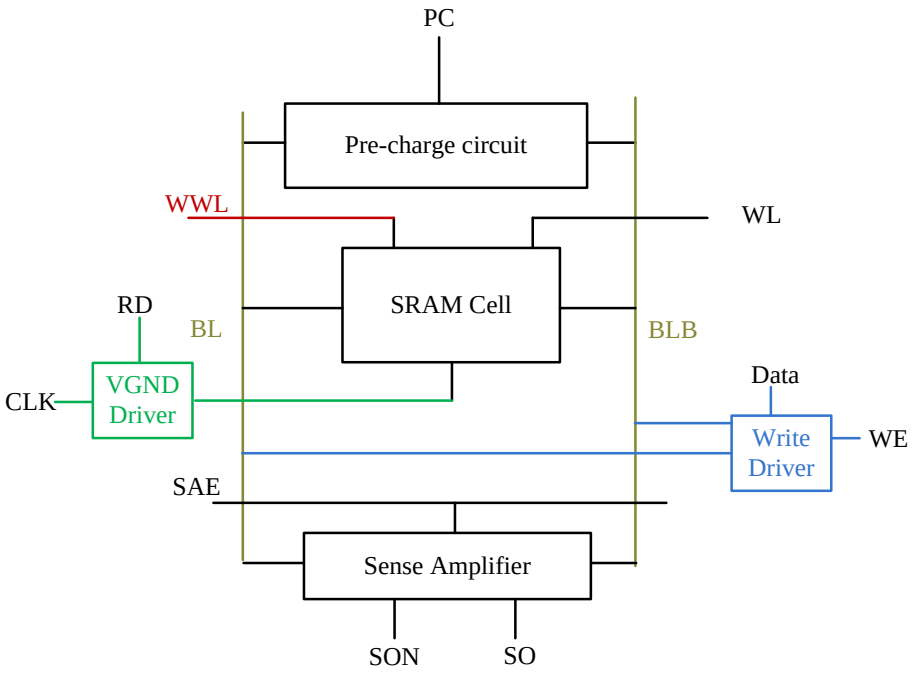


Figure 29: 10T SRAM bit cell with peripheral circuit.

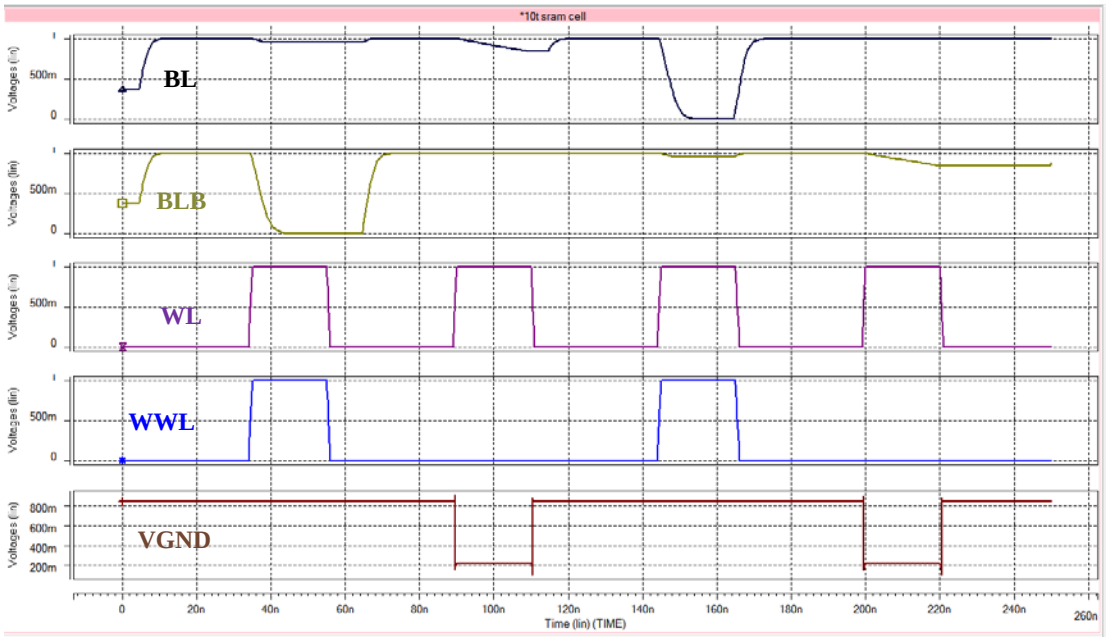


Figure 30: Write (1-0) and read (1-0) operation for 10T SRAM.

4. SRAM Cell Simulation Results

4.1 Simulation Technique

4.1.1 Monte Carlo Simulation

The Monte Carlo simulation is a random analysis that is used in this project. The method is to simulate the design circuit many times and vary all parameters simultaneously, and then study the statistical variability of power and delays. The flow chart of the Monte Carlo simulation is shown in Figure 31.

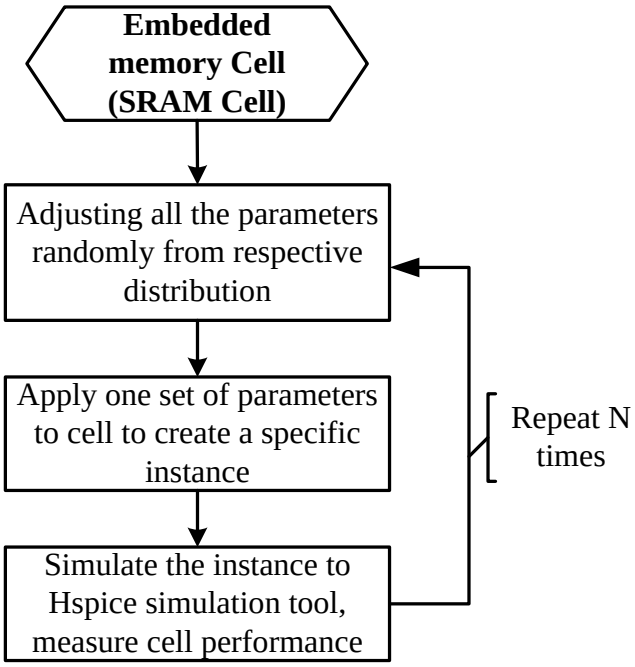


Figure 31: Monte Carlo Simulation flow chat.

The variable parameters that we can adjust are threshold voltage V_T, oxide thickness t_{ox}, supply voltage V_{DD} and temperature.

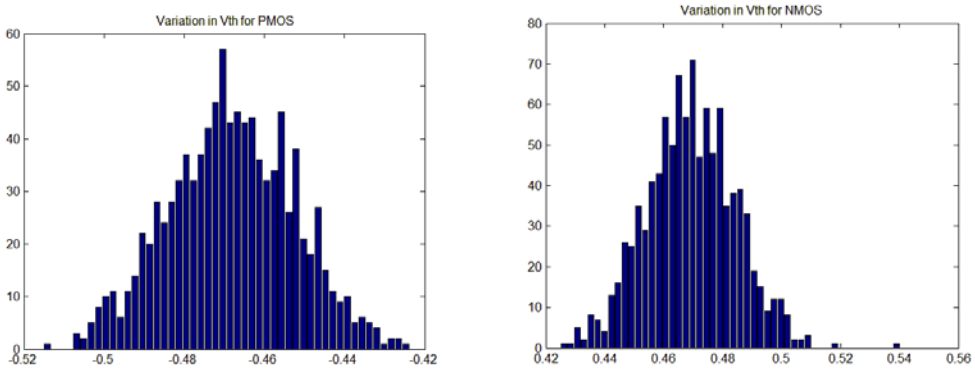
We will randomly pick a value for each parameter within the distribution and then simulate the design. Different sets of parameters will be simulated, and then we will measure circuit performance, power, stability, and read and write times of the SRAM cells. Looking at the parameters mentioned here, there is some interdependence between these parameters so random values will be picked and then reconciled; also, every set of parameters will remain entirely consistent. For the adjacent transistors, there is a spatial correlation coefficient that defines the parameter matching between these transistors. The perfect correlation means the spatial correlation coefficient is equal to one.

For the memory cells we looked at, the operation value of threshold voltage V_T is dependent on the L, t_{ox} and V_{DD}; one way to deal with this interdependence is to separate the operating threshold voltage into independent and dependent components.

The drawback of this model is that we can see the variations that appear in the simulations but we do not know where they come from. So we will also try the other model, which is to adjust one parameter and then do the same procedures from model I in order to see which parameter has the biggest impact on the cell performance.

4.1.2 Gaussian distribution

The testing parameters used in the Monte Carlo simulation should be randomly generated but within a certain interval. Here we used the Gaussian distribution statistical model to generate each parameter. Figure 32 (a) shows the sample distribution of V_{th} with a variation of 10% and Figure 32 (b) shows the sample distribution of T_{ox} with a variation of 20%. The peak value is a typical threshold voltage (T_{ox}) for 45nm high-k/metal gate NMOS within three standard deviations (3-sigma). Three sigma deviations can cover 97.9% of the total samples, which is usually used as industrial standard.



(a) PMOS and NMOS V_{th} samples in Gaussian Normal Distribution.

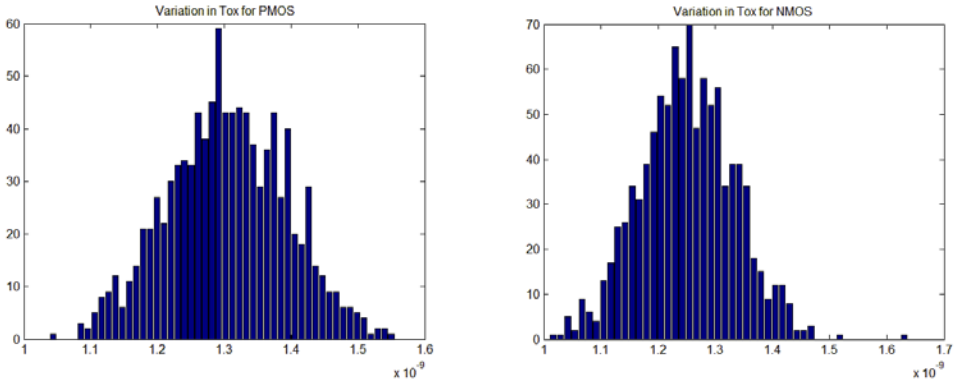


Figure 32: (b) PMOS and NMOS T_{ox} samples in Gaussian Normal Distribution.

4.1.3 Simulation Setup and Transistor Models

The performance of SRAM cells is affected by transistor size so here we will design each SRAM cell with a reasonable transistor size in order to get the best performance. These designs are implemented and simulated in HSPICE, and the simulation results are plotted onto graphs by using the MATLAB tool. The 45nm high-k/metal gate CMOS transistor models used in the simulation is adopted from Predictive Technology Model (PTM) website [14].

4.2 Assessments and Analysing Results

The implementation of each SRAM cell is presented in chapter 3 under section 3.3. Here we will show the simulation results and the comparison between each design. The performance metrics we used here are: access time, power, layout, write stability, read stability and hold stability. We will present our work in this order as well.

4.2.1 Write and Read Operation Test under Process Variation

Access time is the time used to write into the cell and to read from the cell. For write time, we will measure from when the WL signal first rises to 50% of the V_{DD} to when the data node Q falls or rises to 50% of V_{DD}. For read time, we will measure from when the WL signal rises a second time to 50% of V_{DD} to when one of the bit lines falls to 10% of V_{DD}. The access time depends on the supply voltage so before measuring the access time, the minimum operating supply voltage for each memory cell should be detected first.

Table 1 shows the power dissipation for write (1-0) and read (1-0) operations for each memory cell, and the read time and write time at the minimum operating supply voltage. Because each memory cell has a different write and read mechanism, they will operate at different values of supply voltage.

SRAM Cell (minimum Vdd)	Power (uw)	Read Time (ns)	Write Time (ns)
6T---0.6V	16.15	18.39	7.036
7T---0.65V	13.41	12.59	0.6214
8T---0.8V	21.56	5.496	0.5551
9T---0.65V	24.86	52.03	9.060
10T--1V	136	55.64	4.307

Table 1: Power and access time comparison between 6T-10T SRAM Cells.

The numerical results are not similar to the design proposed in [4] [5] [6] [7] because in this project we have used large bit line capacitors to give more realistic simulation results. In order to get a more accurate comparison between each cell, the analysis was also carried out at the same supply voltage level.

Table 2 shows the power dissipation for write (1-0) and read (1-0) operations for each memory cell, and the read time and write time at the 1V operating supply voltage.

SRAM Cell (1V Vdd)	Power (uw)	Read Time (ns)	Write Time (ns)
6T	53.25	7.638	0.254
7T	39.44	5.958e	0.3127
8T	35.29	4.110e	0.4034
9T	65.96	51.01	1.248
10T	136	55.64	4.307

Table 2: Power and access time comparison between 6T-10T SRAM Cells at 1v V_{DD}.

Scaling down supply voltage will consume less power but it also affects system stability. With a minimum supply voltage, we can run the Monte Carlo Simulation to

analyse how stable the memory cell is when it is against process variation. Figure 33 shows the failure rate of a conventional 6T SRAM cell. There are 1000 samples with a random threshold voltage and T_{ox} (as shown in Figure 32) to do the same write and read operation to see how many operations are successful.

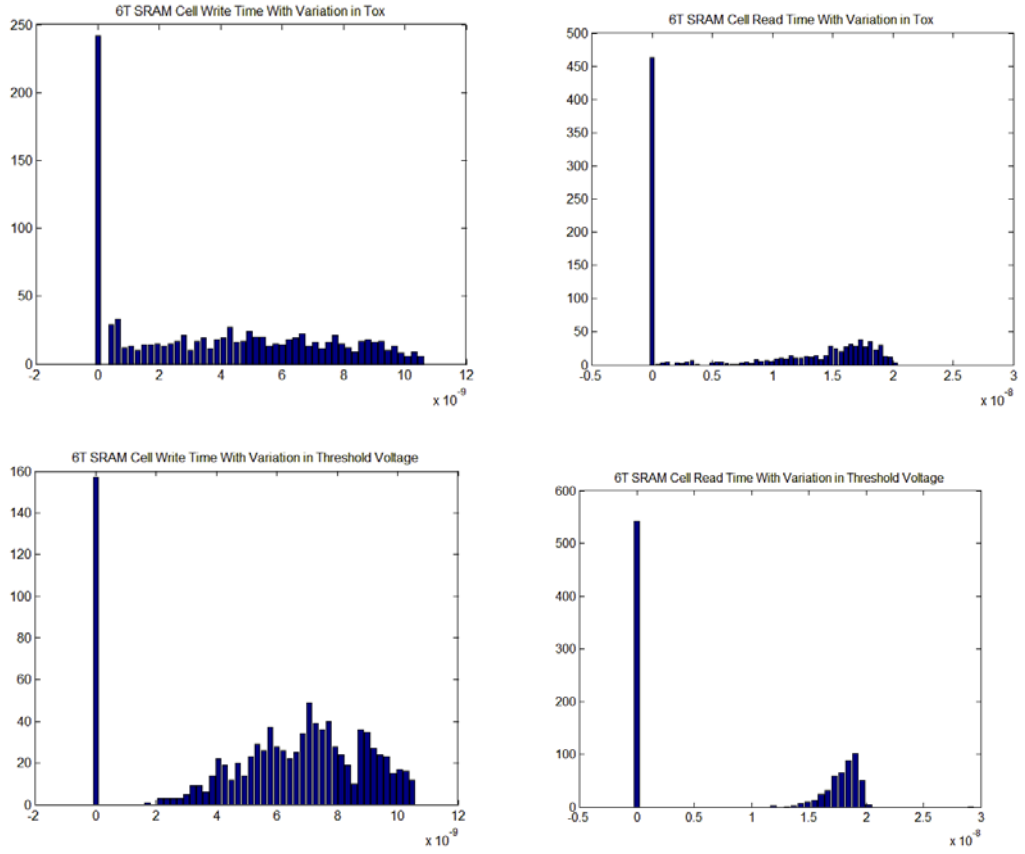


Figure 33: Access time failure rate of conventional 6T SRAM cell.

From the above figures, we can see that the read operation is worse than the write operation when the cell operates at minimum supply voltage (0.6V). The read operation failure rate is around 50%-60%. The write operation failure rate is around 15.6% for V_{th} variation and 24.8% for T_{ox} variation.

Figure 34 shows the failure rate of proposed 7T SRAM cell. The same number of samples are simulated with random threshold voltage and T_{ox} (as shown in Figure 32) to do the same write and read operations.

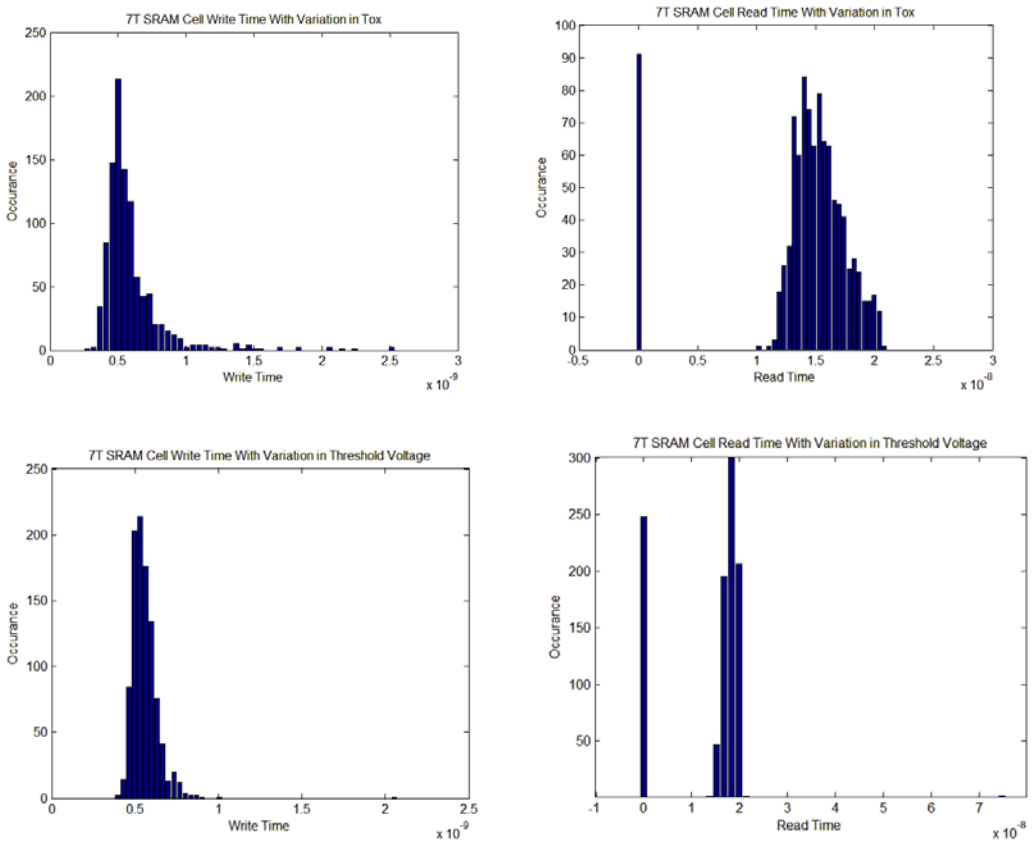
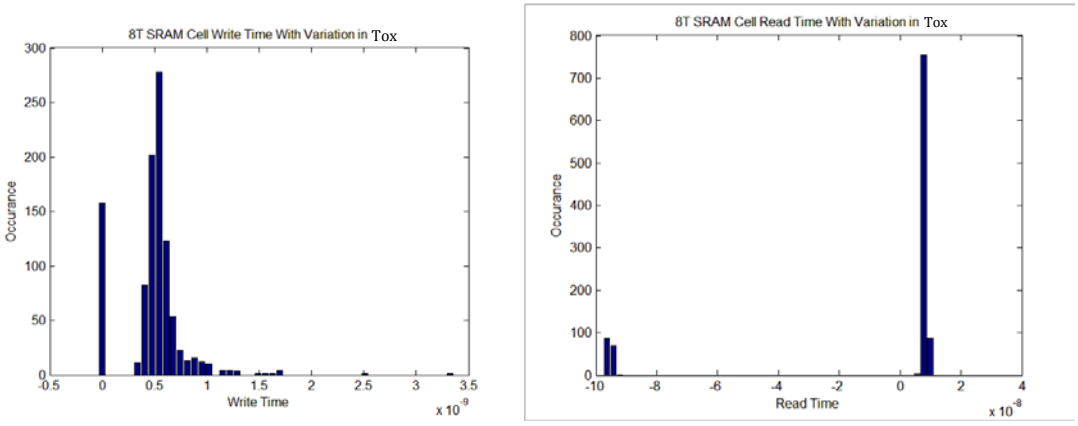


Figure 34: Access time failure rate of proposed 7T SRAM cell.

This 7T SRAM design has a very reliable write mechanism so even at the minimum operating voltage (0.65v) the write operation can perform correctly and gives a reasonable write time under process variation without any failure. But the read operation is quite vulnerable here. The read operation failure rate is around 25% for V_{th} variation and 9.3% for T_{ox} variation.

Figure 35 shows the failure rate of a proposed 8T SRAM cell with the same sample parameters as used in the simulation of 6T and 7T memory cells.



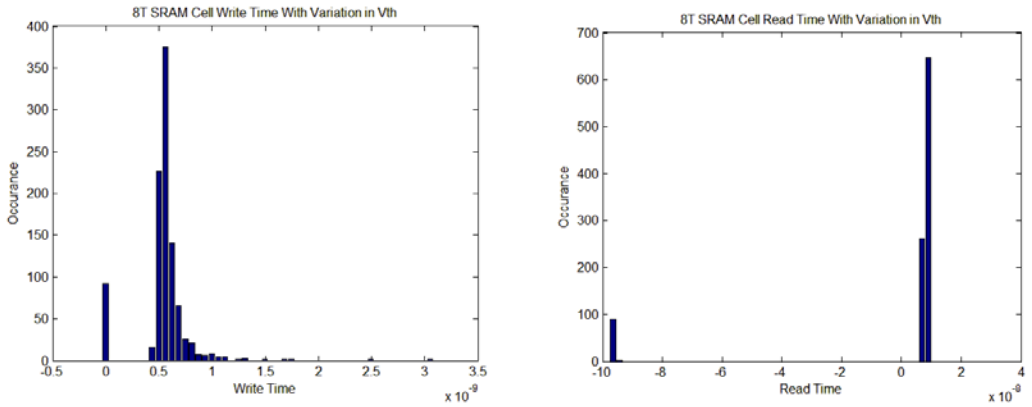


Figure 35: Access time failure rate of proposed 8T SRAM cell.

The proposed 8T SRAM design has a very different write mechanism because it only uses one bit line for the write operation. The data node is separated by an individual inverter during the read operation. The write access transistor has to be very strong in order to perform a correct write operation. The write operation failure rate is around 8.6% for V_{th} variation and 17% for T_{ox} variation. The read operation failure rate is around 10.1% for V_{th} variation and 17.8% for T_{ox} variation.

Figure 36 shows the failure rate of the proposed 9T SRAM cell with the same sample parameters as used in the 8T memory cell simulation.

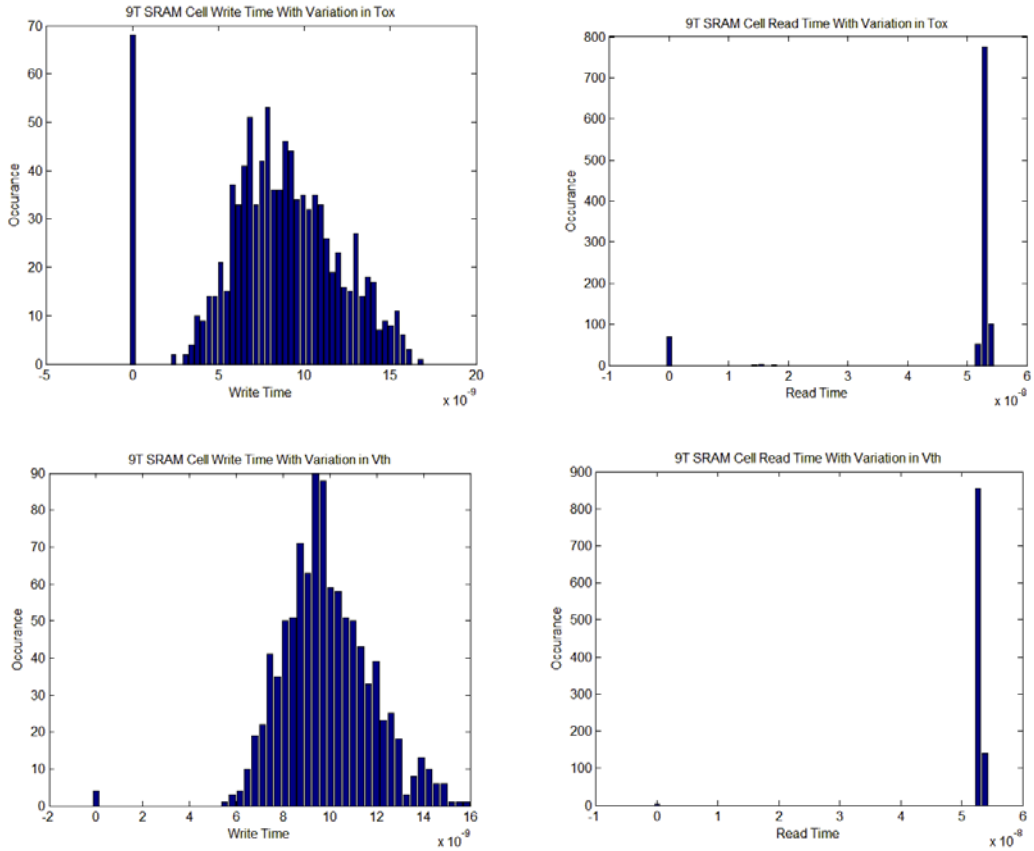


Figure 36: Access time failure rate of proposed 9T SRAM cell.

The proposed 9T SRAM design has a very different read mechanism compared to the above memory cells because it has two extra transistors used to isolate the data node from the bit lines. It is also why the cell requires more power for read operations. The write operation is the same as a conventional 6T SRAM. The write operation failure rate is around 0.6% for V_{th} variation and 6.8% for T_{ox} variation. The read operation failure rate is around 7.8% for T_{ox} variation and there are no failure samples with V_{th} variation. As presented in Figure 36, we can see that the proposed 9T SRAM cell has good stability during read operations as there is very little variation against process variations.

Figure 37 shows the failure rate of the proposed 10T SRAM cell with the same sample parameters as used in the 9T memory cell simulation.

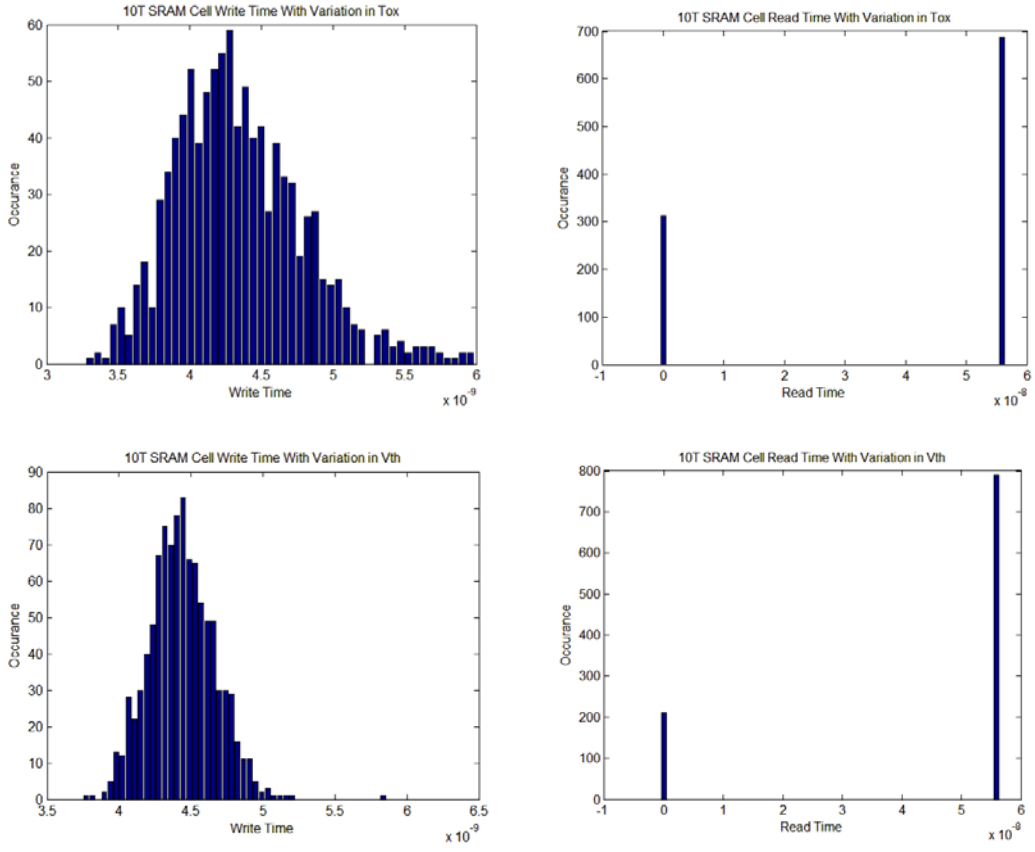


Figure 37: Access time failure rate of proposed 10T SRAM cell.

This proposed 10T SRAM cell has very similar write and read mechanisms compared to the 9T SRAM cell, and the access time variability under process variation also has a similar distribution. But the 10T SRAM cell has a more stable write operation compared to the 9T SRAM. Under the same simulation environment, there are no failure samples for the write operation. The read operation failure rate is around 21% for V_{th} variation and 31.4% for T_{ox} variation.

4.2.2 SRAM Cell Layout

The layout of an SRAM cell defines the density of the memory array, and the structure complexity is the key to the manufacturing yield of the SoC containing a

large area of memory arrays. As CMOS technology size scales down, the layout of the SRAM cell needs to be streamlined due to the limitation of lithograph, etching and CMP. In this project, all the SRAM cell layouts are drawn in 90nm technology as the 45nm model is not available. All the transistors are drawn in the minimum feature size. Here we will show a 2D view of the cell layout.

The PMOS transistor we used here is shown in Figure 38 and it consists of six layers; the N-well represents the bulk of the PMOS transistor. Oxide and Oxide tank layer is the thick field oxide layer. Pimp layer is used to define the type of doping for PMOS. Cont is the contact window of the PMOS. The poly layer creates the input gate of the PMOS, and the metal one layer is creates the source and drain of the PMOS.

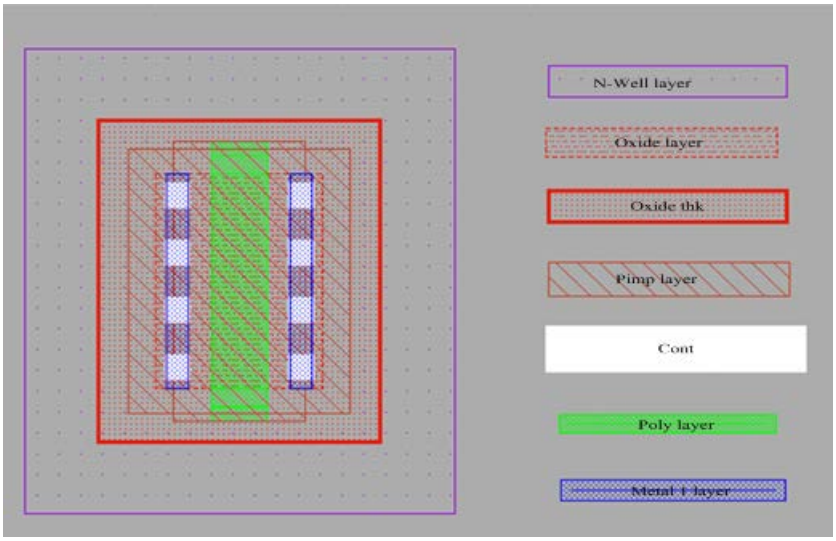


Figure 38: The layout of the PMOS device. [15]

The NMOS transistor we used in this project is shown in Figure 39 and it also consists of six layers; assume the PMOS and NMOS will be built in the P-substrate. The NMOS transistor starts from the oxide tank and oxide layer, which represents the thick field oxide layer. Nimp layer is used to define the type of doping for NMOS. Cont is the contact window of the NMOS. The poly layer creates the input gate of the NMOS and the metal one layer creates the source and drain of the NMOS.

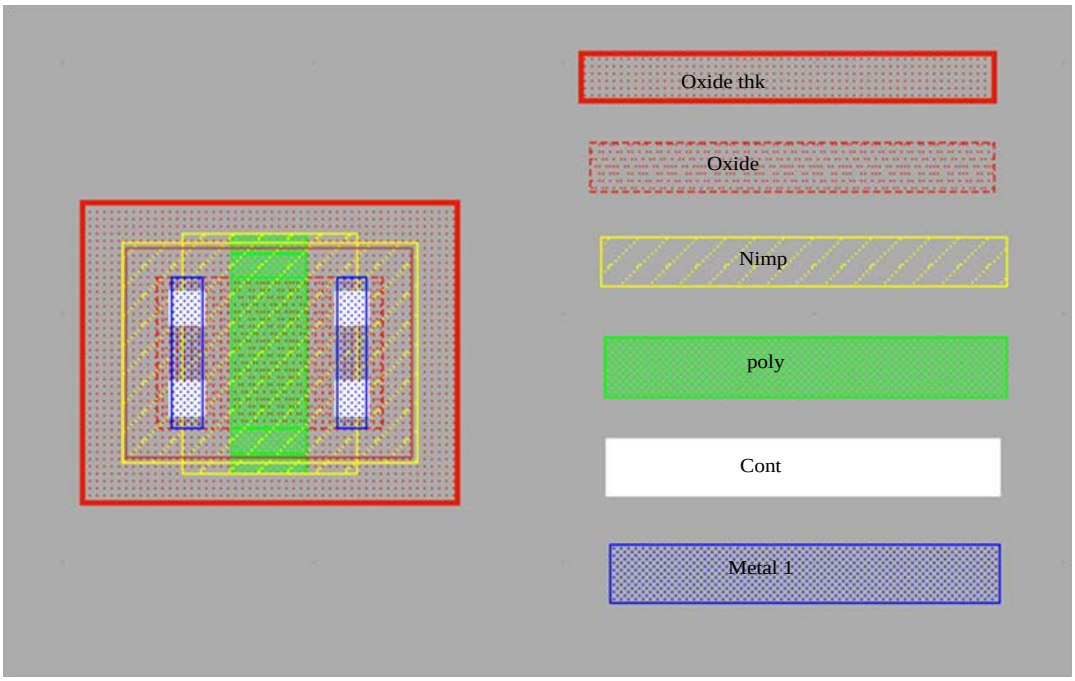


Figure 39: The layout of the NMOS device. [15]

A layout of a conventional 6T CMOS SRAM cell is shown in Figure 40 and this layout is consistent with the schematic we presented in Figure 4.

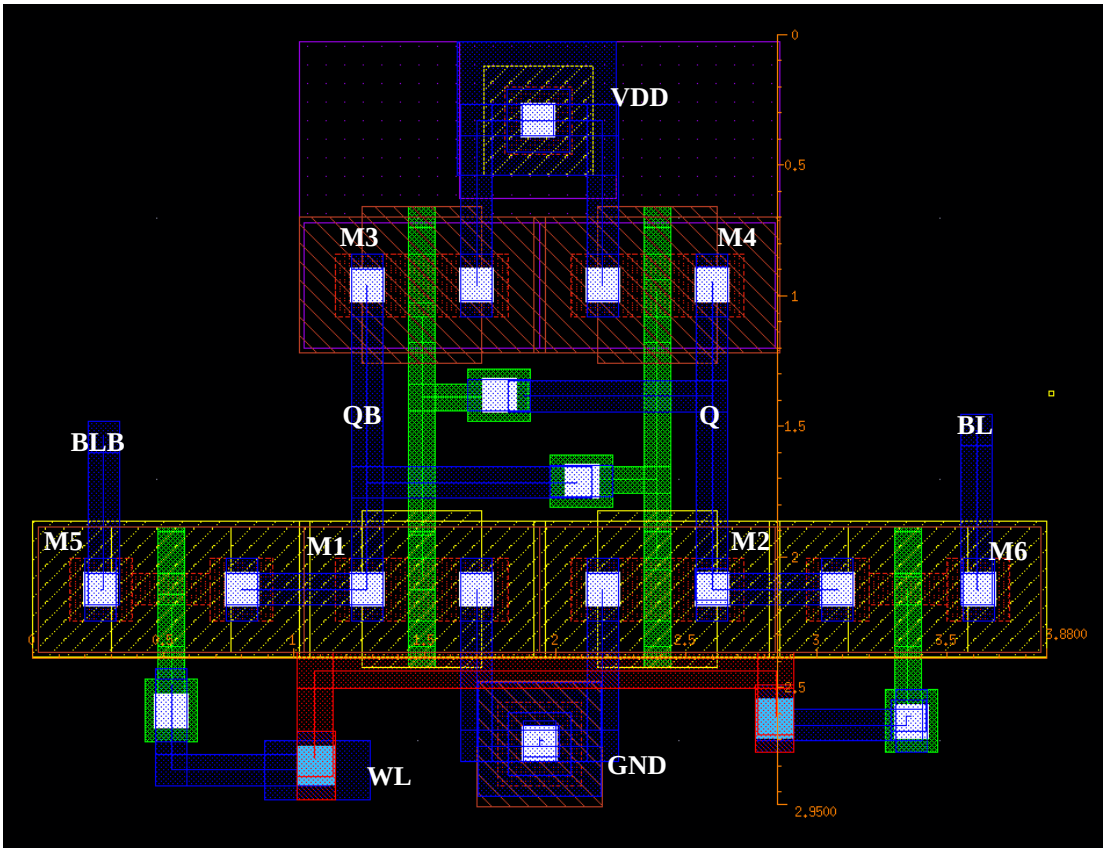


Figure 40: A layout type of conventional 6T SRAM cell.

In this design, the WL wire is drawn in metal_two (in red color), but it would be easier just to use poly (in green color) strip to connect the input of two access transistors. The reason for not using poly strip as a long wire is that poly has a very high resistivity; combining gate capacitance or drain source capacitance together will give a large RC time delay. From this layout, we can see the conventional 6T SRAM cell has a very symmetrical structure and is quite easy to implement. The area of this layout type is 11.446nm.

The layout of the proposed 7T CMOS SRAM cell is shown in Figure 41 and this layout is consistent with the schematic we presented in Figure 5. Assuming we have drawn an array of SRAM cells, we will then need two extra long wires to transfer RL and write line signals across the whole array, which will increase the area and the delay time, as well as add more parasitic capacitances to the design. All of this is an extra cost for adding one more transistor to the cell. This design has a symmetrical structure as well, which can be drawn in a streamlined pattern so it is quite easy to reproduce. The area of this layout is 11.7244nm.

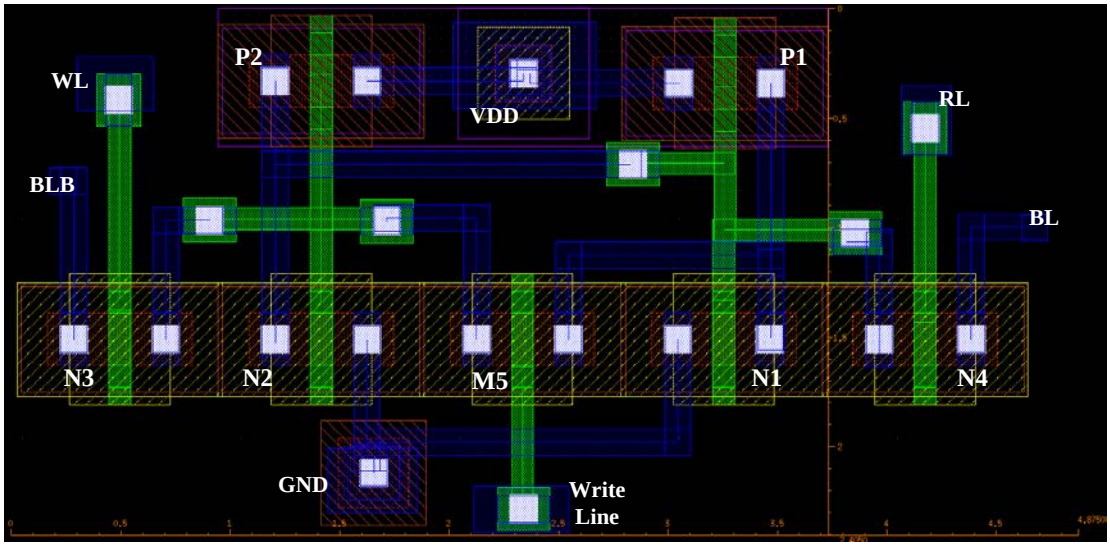


Figure 41: A layout type of proposed 7T SRAM cell.

The layout of proposed 8T CMOS SRAM cell is shown in Figure 42 and this layout is consistent with the schematic we presented in Figure 6.

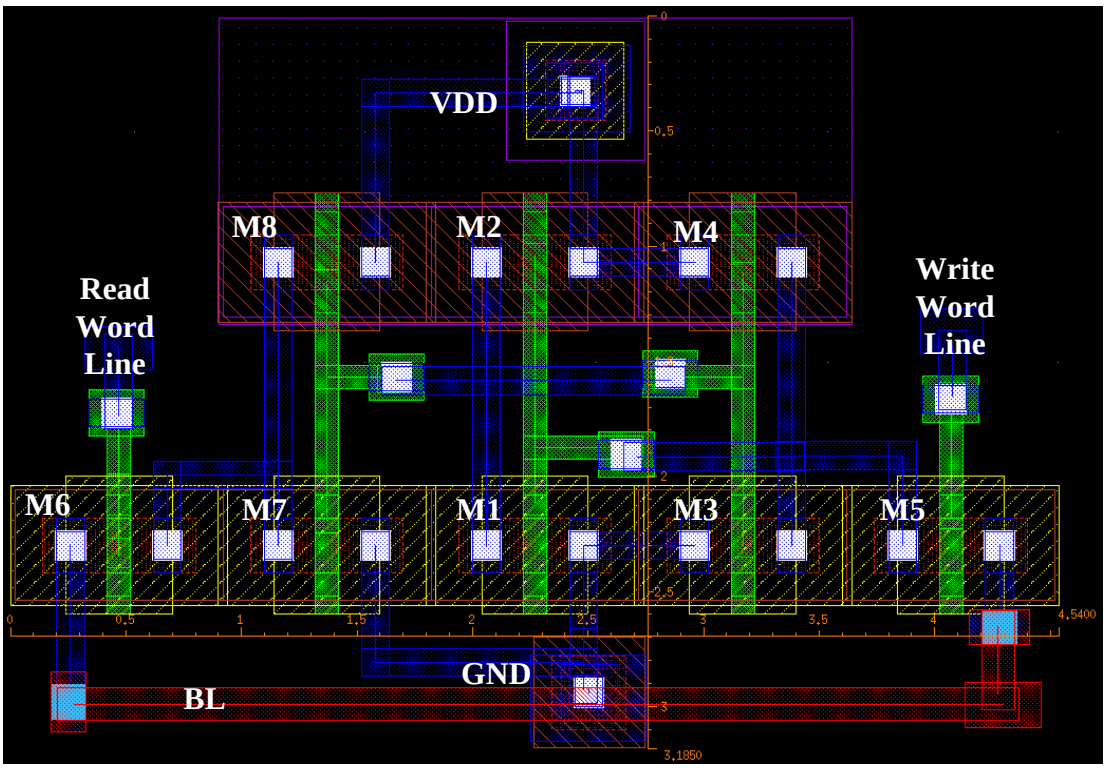


Figure 42: A layout type of proposed 8T SRAM cell.

This layout style has all the PMOS on top and all the NMOS at the bottom, which is the same as the others. The cell structure is not symmetrical due to one extra inverter connected to one side of the data node, which increases the complexity of the interconnection. However, it has fewer numbers of long wires compared to the proposed 7T SRAM cell. The single bit line wire will be twice as long as other cells that use two bit lines, and a longer metal wire will increase the RC time delay. The total area of this layout is 14.46nm.

The schematic of the proposed 9T SRAM cell is presented in Figure 7 and the layout is shown in Figure 43.

From this figure we can see that the cell structure is much more complex compared to a conventional 6T SRAM layout. The interconnection complexity has increased because of the data node separation technique and one extra interconnect long wire is used to connect data node Q3. This actually adds one more layer to the layout. There are four input/output pins, which means four very long wires cross the memory array so the RC time delay, the total area, and the total parasitic capacitance are increased. The power dissipation is increased due to the increased parasitic capacitance. Also, this structure will be hard to produce due to the complexity of the interconnection and the unsymmetrical structure. The total area of this layout is 20.89nm.

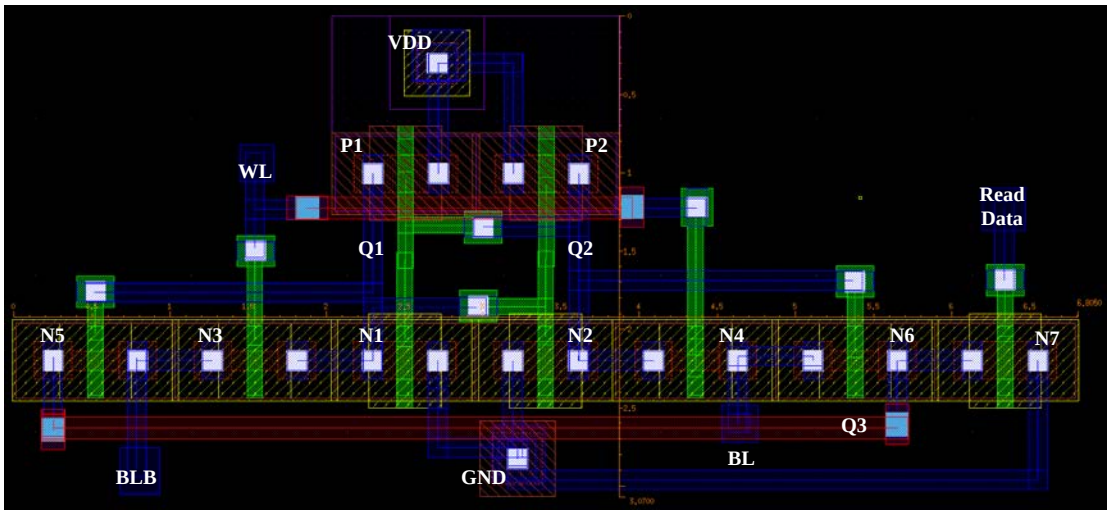


Figure 43: A layout type of proposed 9T SRAM cell.

The schematic of proposed 10T SRAM cell is presented in Figure 8 and the layout is shown in Figure 44.

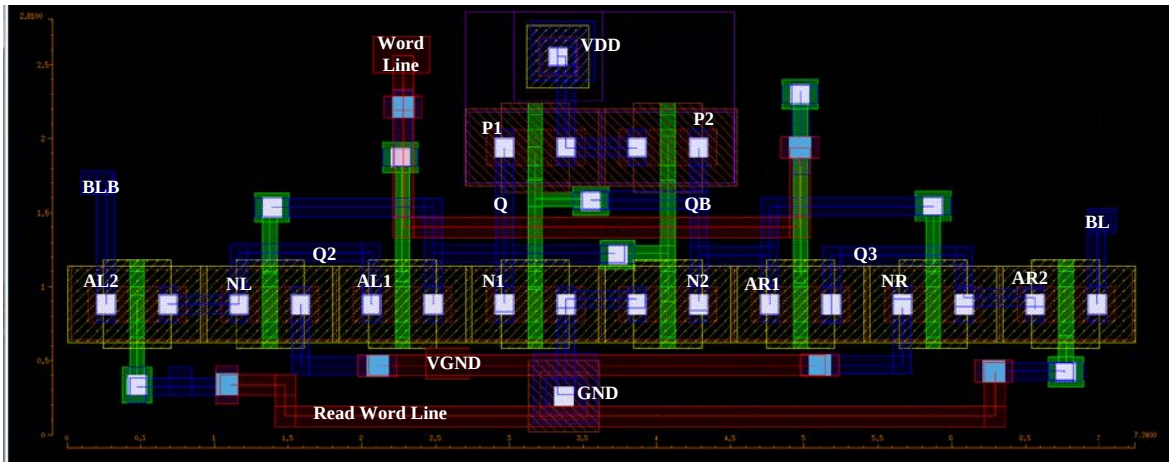


Figure 44: A layout type of proposed 10T SRAM cell.

The proposed 10T SRAM layout is more complex than the 9T SRAM layout. The two access transistors in the series and the data node isolation technique both increase the complexity of interconnection. The read word line wire and virtual ground rail are two long wires that cross the whole memory array. The cell structure is symmetrical but with a very complex interconnection. The total area of this layout is 20.34nm.

4.2.3 SRAM Cells Stability Test under Process Variation

The stability of SRAM cells is mainly investigated in terms of static noise margin (SNM). The static noise Vn is a DC disturbance that comes from offsets and transistors mismatch due to manufacturing and variation in operation conditions. The definition of SNM is the maximum value of Vn that can be tolerated by the cross-coupled inverters before changing state. The requirement for the SRAM designer is to design a memory cell that, under all operating conditions, the noise margin is reserved to fight against dynamic disturbances caused by α particles, capacitor cross talk, supply voltage noise and thermal noise.

The measurement of SNM is obtained by drawing and mirroring the cross-coupled inverters voltage transfer characteristic (VTC) under different operating states, and finding the maximum possible square between them. This is a graphical technique to determine the SNM that was used in this project. The methods used to measure SNM for each memory cell are similar, and here we will explain these methods in detail by using a 6T SRAM cell.

Figure 45 shows the circuit we used to measure the **hold state** SNM for 6T SRAM, and the VTC graph of each memory cell is shown in Figure 47. Hold state is when the cell is disconnected from bit lines and the data is stored in the data nodes. Here we introduce static noise Vn by using a DC voltage source and doing a DC analysis to measure output QB and then mirroring the output of Q. The simulation is carried out in HSPICE and the output of HSPICE is a binary file, which will be compiled by C complier then the VTC graph can be generated by the MATLAB tool. The maximum value of Vn is equal to V_{DD}.

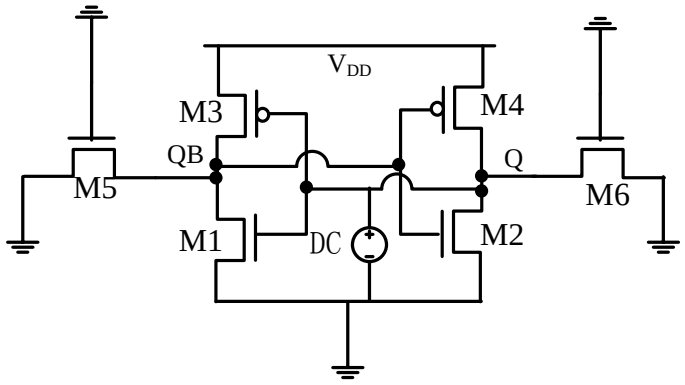
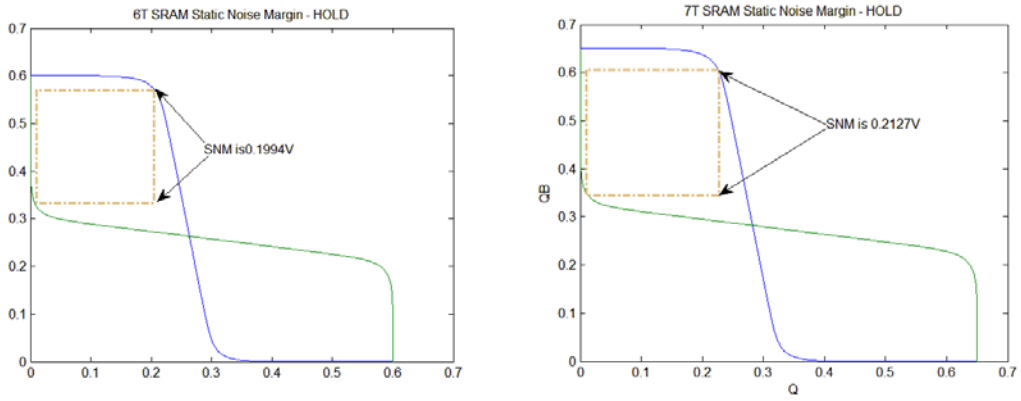


Figure 45: Circuit design for measure hold state SNM for 6T SRAM.



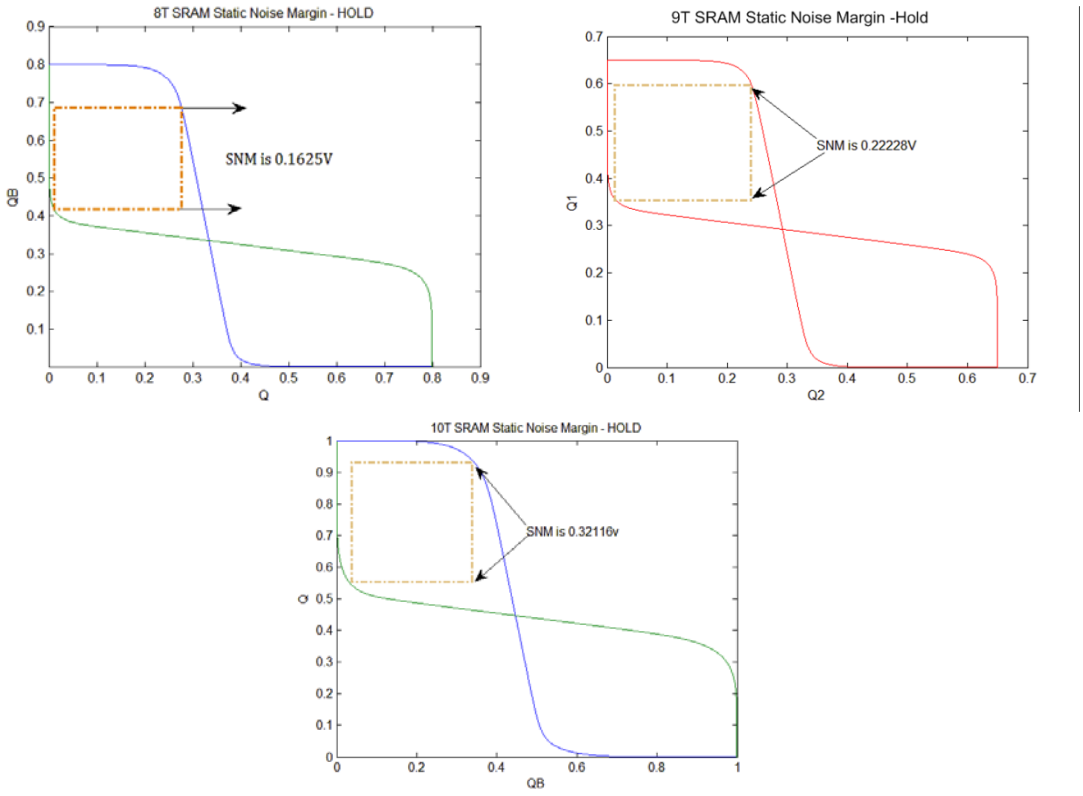


Figure 46: Hold state SNM of 6T-10T SRAM cell.

From the VTC curves shown in Figure 46, we can see that the proposed 10T SRAM cell has the highest hold-state SNM. However, it does not mean it is the most stable circuit during the hold state because we measured SNM at minimum operating voltage for each cell. The proposed 10T SRAM can only work at 1V. The proposed 9T SRAM actually has very high stability during a hold cycle at low power.

Figure 47 shows the circuit we used to measure **SNM during write operation** for a 6T SRAM and generated VTC curves of different cells are presented in Figure 48. The write SNM is defined by a write trip voltage, which is the maximum voltage on the bit line needed to flip the cell content. Normally the bit lines are pre-charged to the voltage level, which is lower than V_{DD} in order to save power. The write trip voltage is determined by the pull up ratio (i.e. the W/L ratio of transistor M3 and M4) of the cell. The write trip voltage is simulated by a DC voltage source.

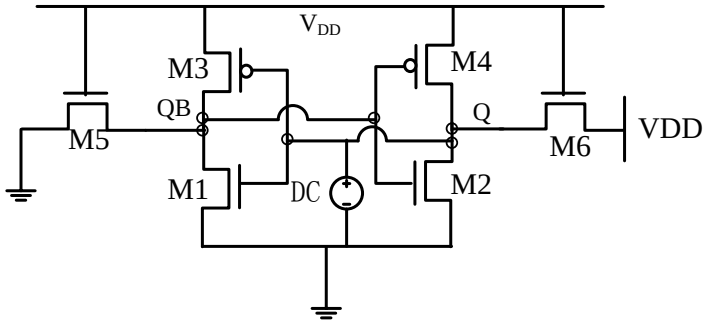


Figure 47: Write SNM for 6T SRAM.

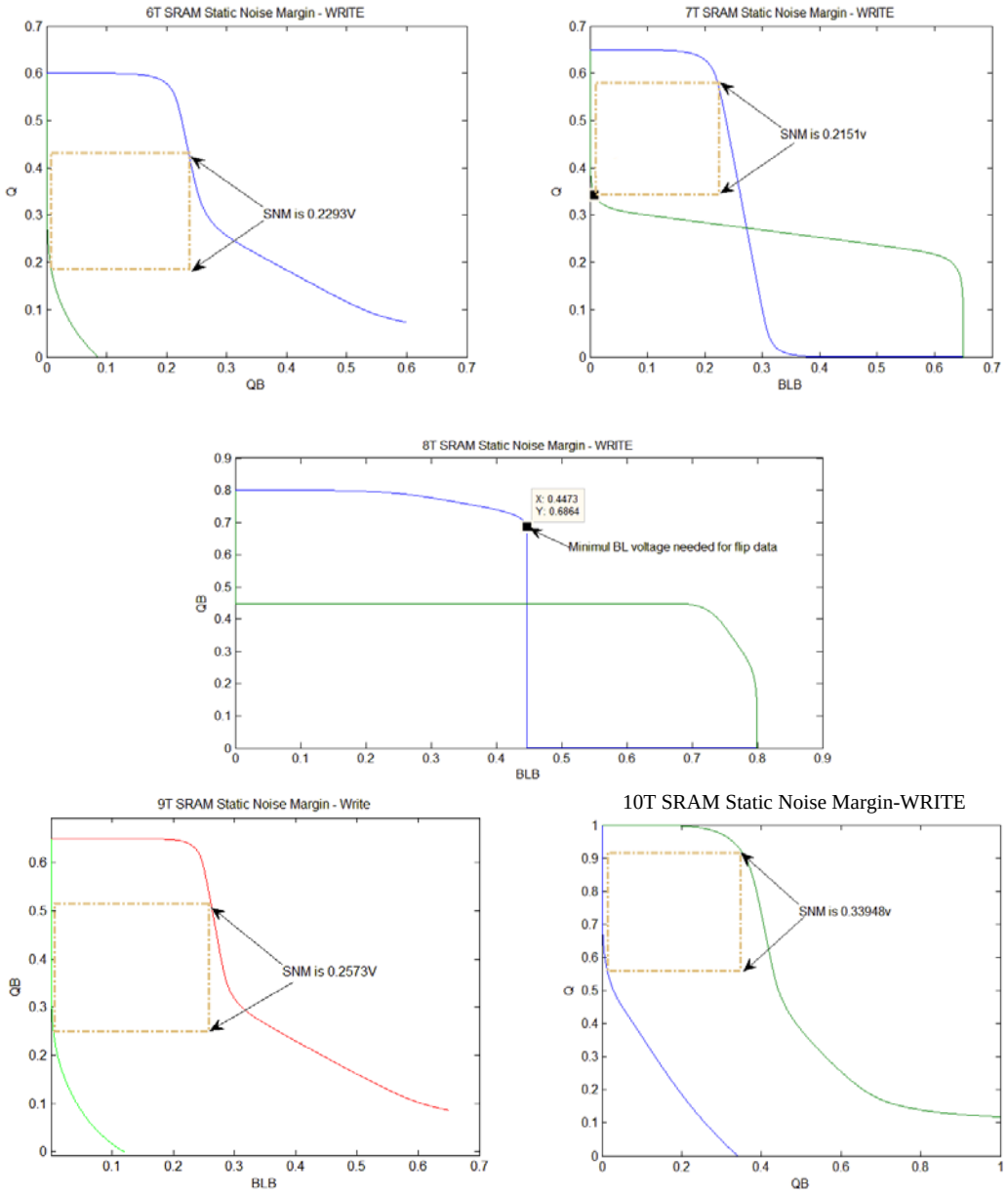


Figure 48: Write state SNM of 6T-10T SRAM cell.

The VTC curve of the 7T and 8T SRAM cells have very different shapes compared to other cells because 7T has a very different write mechanism: at the beginning of the write cycle it cuts off the feedback between the two cross-coupled inverters. But 8T also has a very different bit line arrangement because it only used one bit line so the write trip voltage is the minimum voltage required to flip the data on the data node Q2. The interpretation of the 8T SRAM VTC curve should be the amount of write triple voltage required. For SRAM cells the less write triple voltage required the less power consumed during the write cycle.

Figure 49 shows the circuit we used to measure **SNM during read operation** for 6T SRAM and generated VTC curves of different cells are presented in Figure 51. Assume we have logic "0" stored in data node Q. During a read cycle, the voltage on the data node Q will rise higher than ground and may flip the data stored in data node

QB. The read SNM is determined by the ratio of the pull down transistors (M1, M2) and access transistors (M5, M6), called cell ratio. The circuit presented here is in a read mode, and a DC voltage source Vn that will sweep from 0V to V_{DD} is applied at the "0" data node and the corresponding VTC is measured.

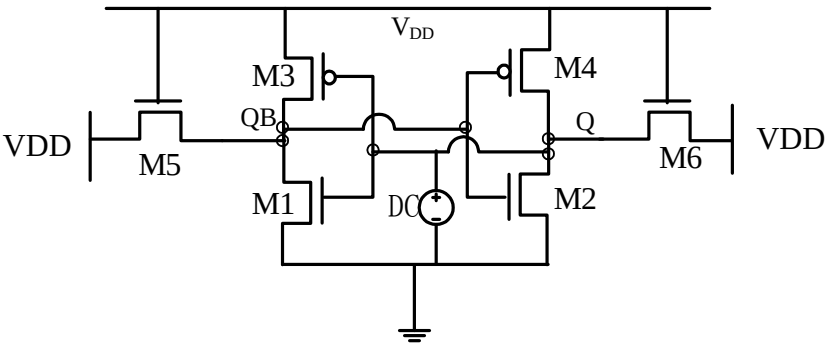
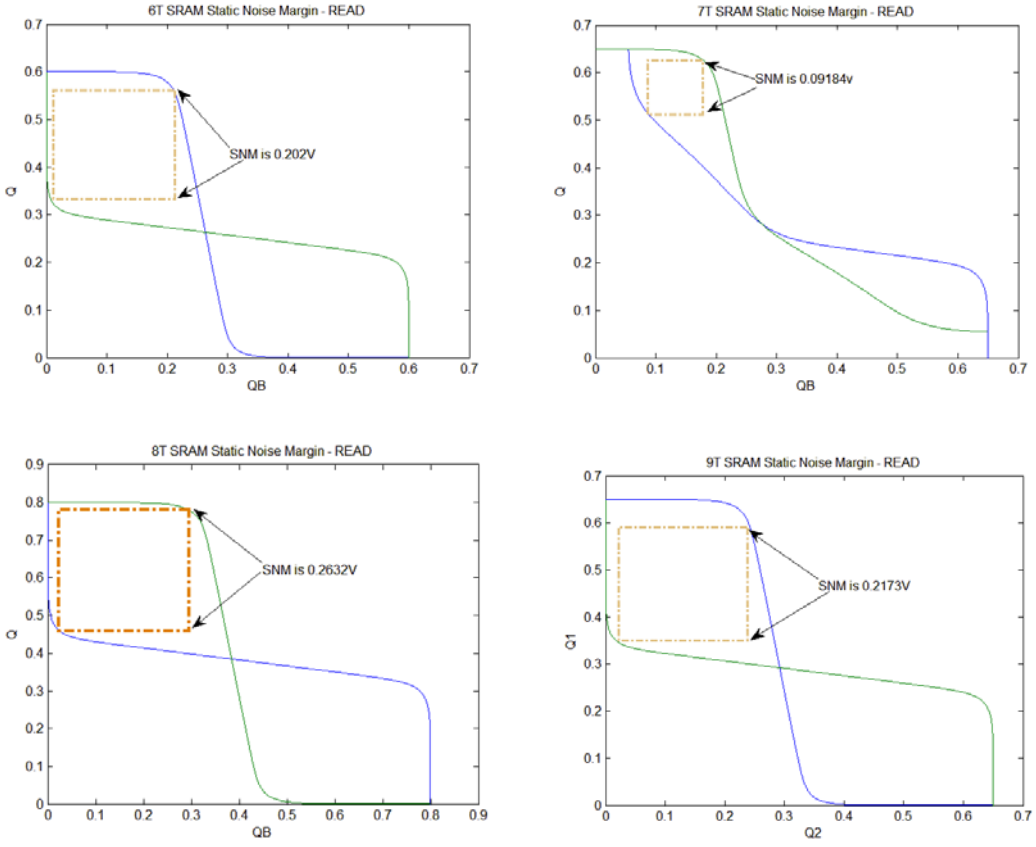


Figure 49: Circuit design for measure read SNM for 6T SRAM.



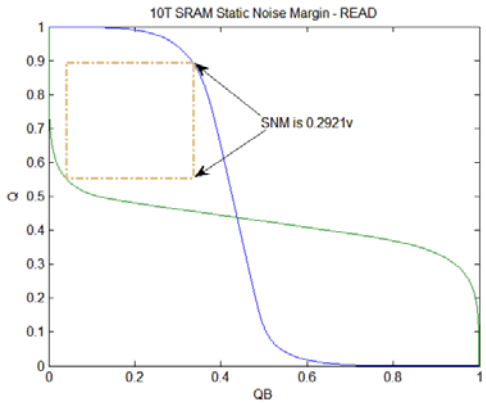


Figure 50: Read state SNM of 6T-10T SRAM cell.

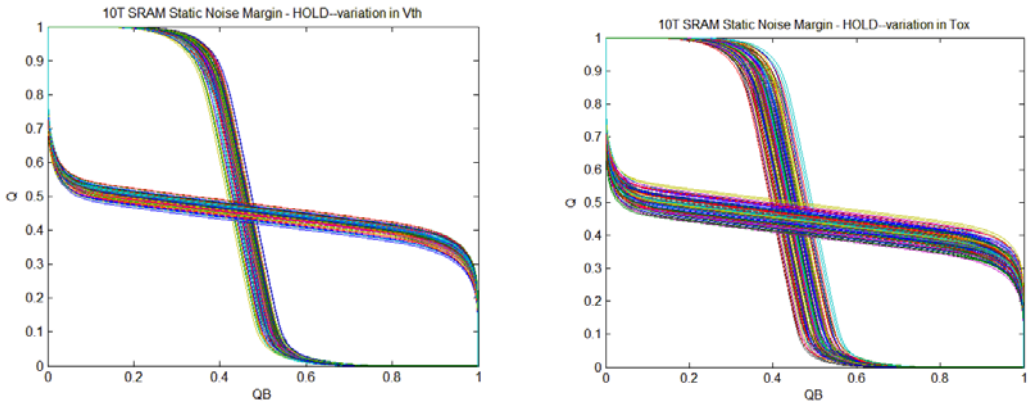
As shown in Figure 50, the 7T SRAM has a very small read SNM because of the extra transistor inserted in the feedback between the cross-coupled inverters. This adds extra burden on the pull down network. From the 9T and 10T SRAM VTC curve we can see that the data node isolation technique gives a good read stability.

SNM under process variation:

SRAM stability is measured in different states under process variation environment at minimum operating voltage. The influence of supply voltage and temperature on the memory SNM is also analysed. We will present our work in this order as well.

The above section has introduced how to measure the SNM in different operating modes and here we will show the SNM variation under process variation. The Monte Carlo simulation method is used here to investigate the variation in SNM and the simulation parameters are V_{th}, T_{ox}, V_{DD} and temperature.

The statistical analysis of **SNM** was performed using the Monte Carlo simulation of 100 samples with 10% deviation in the threshold voltage and a 15% deviation in T_{ox}; the results are shown in Figure 51. We will only present a VTC graph for the design cells that have the most variation due to the thesis page limitation, and the simulation results will be presented as a table. Table 3 shows the statistical analysis results of the SNM of the SRAM cell by using sample parameter V_{th}. Table 4 shows the statistical analysis results of the SNM of the SRAM cell by using sample parameter T_{ox}.



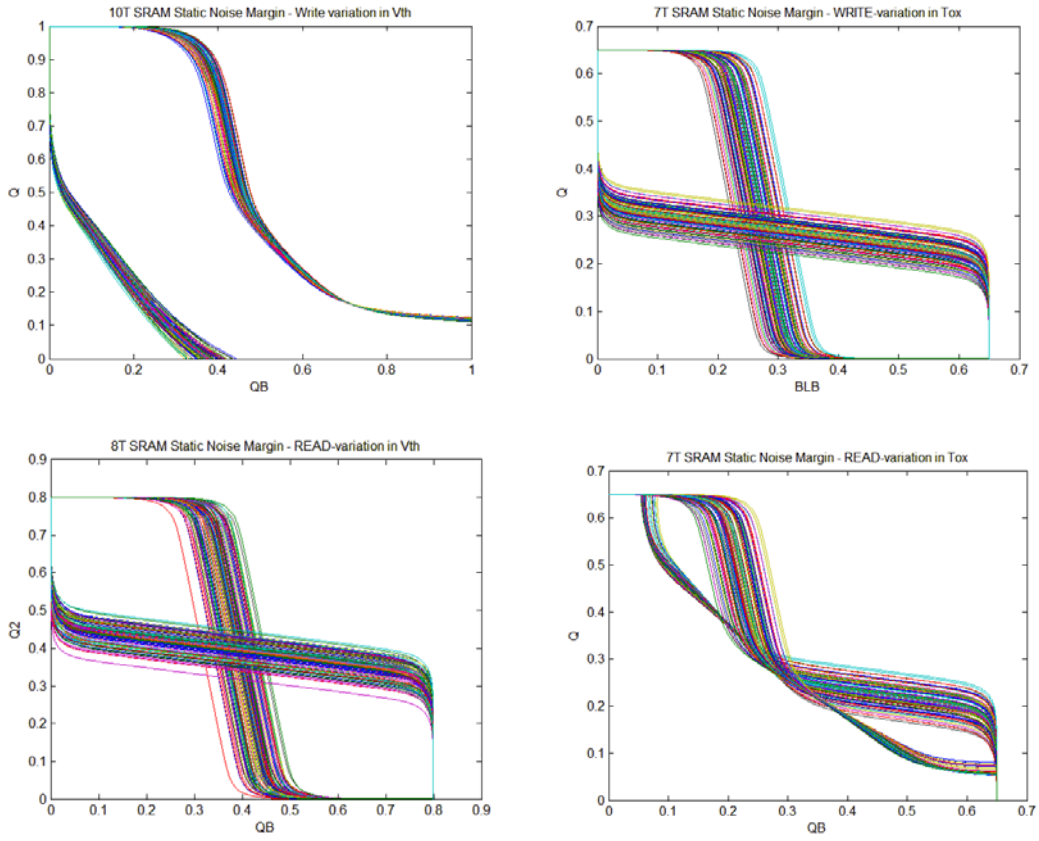


Figure 51: Monte Carlo simulation of SNM of 10T SRAM cell.

SRAM Cell (minimum V_{DD})	Hold-SNM (V_{th})	Write-SNM (V_{th})	Read-SNM (V_{th})
6T-0.6V	0.0538v	0.0516v	0.0537v
7T-0.65V	0.0482v	0.0545v	0.057v
8T-0.8V	0.0559v	0.45v*	0.1295v
9T-0.65V	0.0541v	0.0478v	0.0541v
10T-1V	0.0572v	0.0562v	0.0576v

*This value is write triple voltage variation of 8T SRAM cell.

Table 3: Hold-SNM variation comparison between SRAM cells by using sample parameter V_{th}.

SRAM Cell (minimum V_{DD})	Hold-SNM (T_{ox})	Write-SNM (T_{ox})	Read-SNM (T_{ox})
6T-0.6V	0.0972v	0.0947v	0.0974v
7T-0.65V	0.0968v	0.0976v	0.0998v
8T-0.8V	0.0953v	0.4941v*	0.0983v
9T-0.65V	0.0975v	0.0966v	0.0973v
10T-1V	0.0987v	0.0962v	0.0986v

*This value is write triple voltage variation of 8T SRAM cell.

Table 4: Hold-SNM variation comparison between SRAM cells by using sample parameter T_{ox}.

The 6T SRAM provides a relatively good hold-SNM and write-SNM, but these will be reduced due to process variation. To increase hold-SNM in this cell, we can increase supply voltage but this will increase power dissipation. To increase write-SNM, we can increase cell ratio but this will increase the cell area.

The 7T SRAM provides higher hold-SNM as well as write-SNM at almost the same operating voltage but very less read-SNM, and it is quite sensitive against T_{ox} variation so it is more prone to failure during read operation.

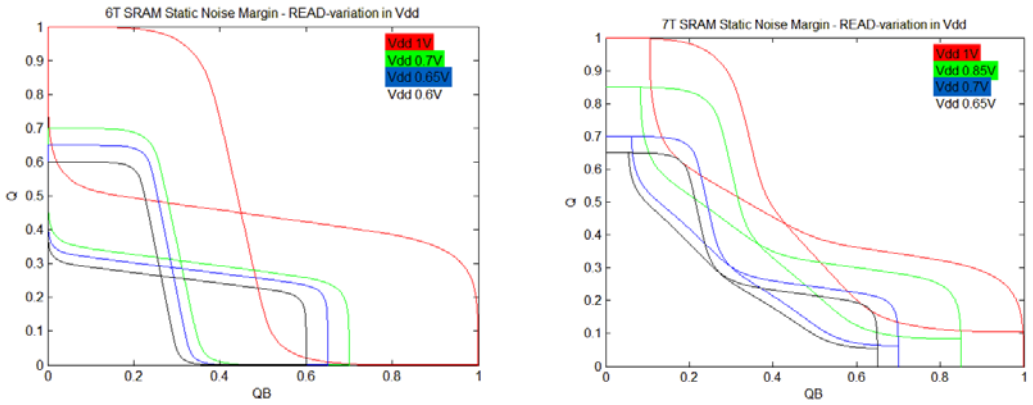
The 8T SRAM provides much less hold-SNM at a higher operating voltage but has very strong write stability and quite good read-SNM, and it is less sensitive to process variation. But the write-SNM and read-SNM will be reduced by bit line voltage noise.

The 9T SRAM provides the highest hold-SNM and good write-SNM and read-SNM at low operating voltage, and it is relatively stable under V_{th} variation but quite vulnerable to T_{ox} variation. For low power design, this cell will give a good performance in different operation modes. To increase SNM against V_{th} variation in this cell, we can increase supply voltage.

The 10T SRAM provides the highest SNM in different operating modes but it is most vulnerable against process variation. We can increase the cell ratio to increase stability under process variation. In terms of area, 10T SRAM can provide a very stable operation.

SNM with V_{DD} scale down:

An SRAM array can be fully functional at the full power supply voltage as we have shown in the previous section but, from the simulation results we presented in section 4.2.1, the cells become unstable and lose data at a reduced supply voltage. The SNM variation will be analysed within the supply voltage range from 0.6V to 1V. We will only present read-SNM VCT curves at the different supply voltage in Figure 52 as the read operation has the most critical operating condition. The numerical result will be shown in Table 5. The 10T SRAM can only operate at 1V supply voltage, which is the normal operating voltage.



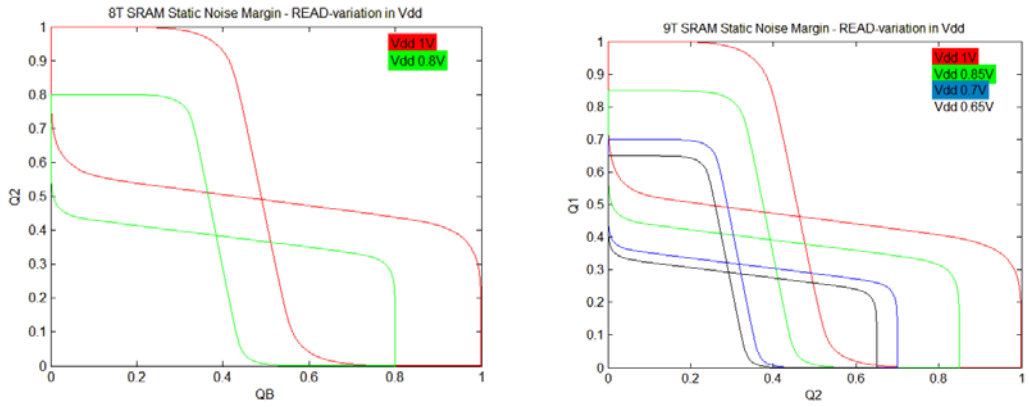


Figure 52: Read state SNM of 6T-10T SRAM cell under VDD variation.

SRAM Cell(1V)	Hold-SNM	Write-SNM	Read-SNM
6T	0.32216v	0.35139v	0.31043v
7T	0.32682v	0.3155v	0.1611v
8T	0.30373v	0.3919v	0.31716
9T	0.31973v	0.35822v	0.31868v
10T	0.3212v	0.33948v	0.2921v

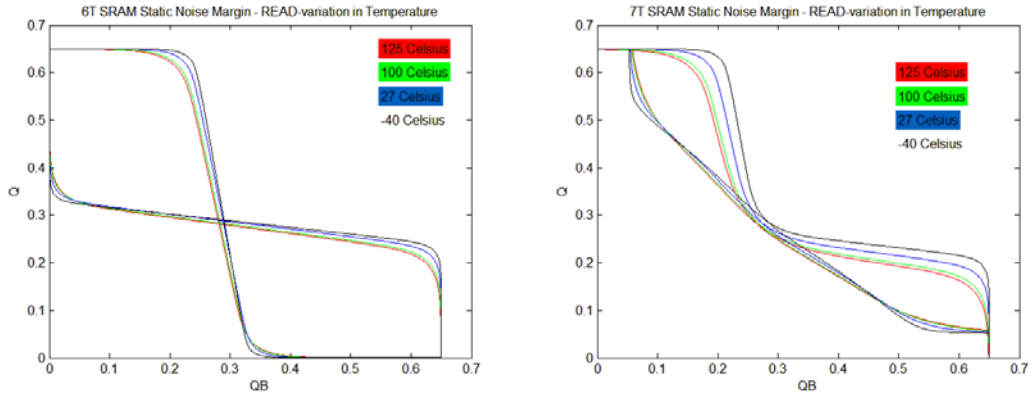
Table 5: SNM of 6T-10T SRAM cell at 1V supply voltage.

From the table above, we can see that the 8T and 9T SRAM cell provide a good overall performance at the same supply voltage. The read stability has been improved by isolating the cell nodes from bit lines and write stability is raised by either using a separate supply voltage or cutting off the feedback between two cross-coupled inverters. The common drawbacks of these designs are increasing cell size and power consumption, and complex interconnection of the cell.

SNM with different temperature:

In this section we analyse the effect of temperature variation on the reliability of the SRAM cell in terms of SNM with the same technology model as introduced before. The temperature range is from -40 °C to 125 °C at cell minimum operating voltage.

Due the page limitation, we will only present read-SNM VTC curves here and the rest of the results are shown in Table 6. The graphs are shown in Figure 53.



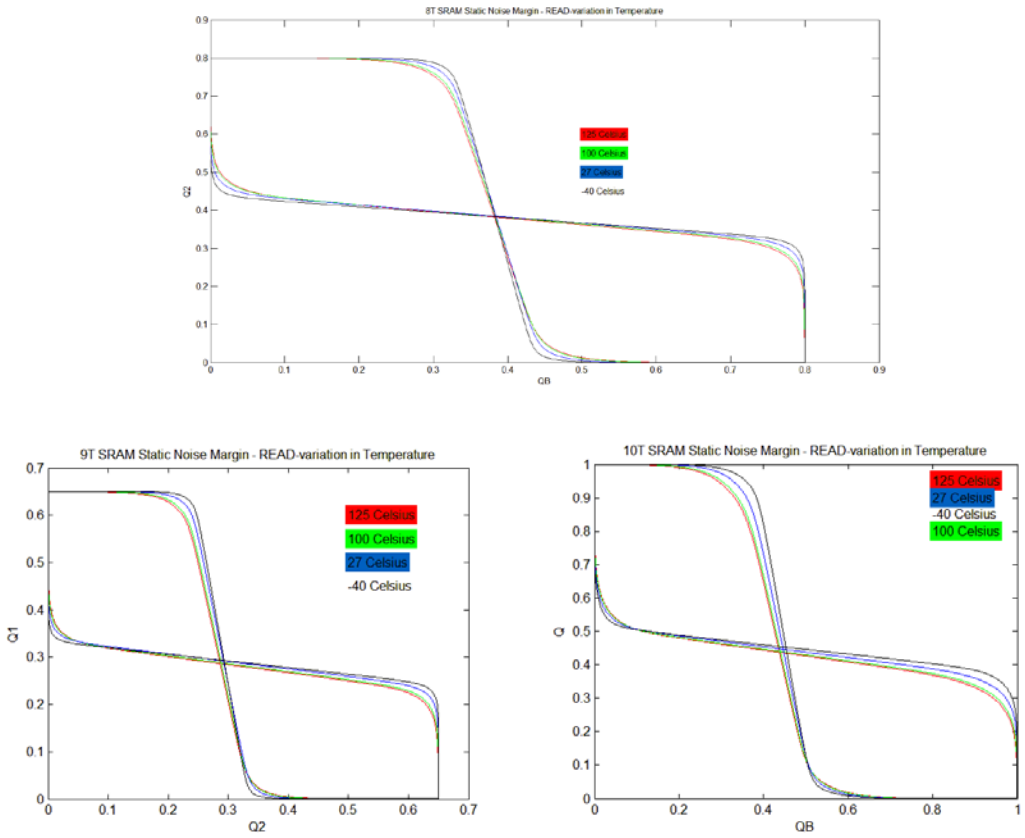


Figure 53: Read state SNM of 6T-10T SRAM cell under Temperature variation.

SRAM Cell (minimum V_{DD})	Hold-SNM (degradation percentage)	Write-SNM (degradation percentage)	Read-SNM (degradation percentage)
6T-0.6V	16.19%	15.7%	14.05%
7T-0.65V	15.47%	23.07%	28.96%
8T-0.8V	17.5%	35.8%*	4.87%
9T-0.65V	9.96%	10.9%	11.76%
10T-1V	15.98%	17.39%	11.8%

*This value is write triple voltage variation of 8T SRAM cell.

Table 6: SNM degradation % of 6T-10T SRAM cell under temperature variation.

SNM degrades due to temperature induced Vt mismatch and 7T has more degradation compared to rest of cells, especially for write and read stability at low supply voltage. Overall the 9T SRAM has much less degradation under all operating modes at a low supply voltage.

4. 3 Summary

This chapter presented a comprehensive study on five different SRAM cells in terms of access time, SNM, power and layout. The cell performances are also analysed under process variation. We found that the 6T SRAM has a very high failure rate under threshold voltage and T_{ox} variation at low voltage supply, and the cell stability will be reduced more with increases in temperature. The 7T SRAM has a small write

failure rate under the same simulation environment but with very poor read stability, and the read-SNM will be further reduced by increasing the temperature. The 8T SRAM cell gives a relatively small read and write failure rate but the write stability is affected most by temperature. The 9T SRAM has a better read and write failure rate compared to other designs, and it also gives good SNM under process variation. The SNM degradation percentage is quite small but it is at the cost of a larger cell area, more complex structure and more power consumption. The 10T SRAM has more power consumption and a more complex cell structure compared to the 9T SRAM, but its performance is worse than the 9T SRAM cell, especially for read stability.

5 ECC for SRAM Design

The previous chapter studied different cell schematics to see what can be improved by changing circuit schematics. However, the simulation results have shown that with CMOS technology scaling down only alters circuit design, it does not give us a big improvement in cell performance. The SRAM array is most likely to fail some bits under process variation and this malfunction cannot be prevented by increasing the cell area, so here we have adopted a different approach, which is error correction technique into SRAM cell.

5.1 Definition of Hamming Code

Hamming code is one of the most popular ECC codes and widely used in SRAM design; it was introduced by R.W. Hamming in 1950. The code works with a number of bits. A Hamming codeword is constructed by data bits and parity bits. Hamming codes detect two error bits and correct one bit. The minimum distance d_{min} is defined as two data strings of equal length with the number of positions with corresponding data being different. The following definitions describe Hamming code with a minimum distance equal to 3 and the parity bits length j≥3:

Error correction capability: $t = \left[\frac{(d_{min}-1)}{2}\right] = 1$

Data-word length: k

Parity bits length: $j = [\log_2 k] + 1$

Code-word length: $n = k + j$

For example, if we have data bits at 8 bits and the length of the parity bits is 4 bits, then the total codeword length is 12 bits, with an error correction capability t=1. This can be expressed as Hamming (12, 8, 1) code.

5.2 Hamming Encoding

In order to generate a Hamming codeword, we based on k bits data to encode j bits parity bits and combined these two parts together to form a Hamming codeword. The following steps describe how to generate parity bits and their positions in the final codeword.

I. **Find out where we can distribute the parity bit in data-word.** The data string is expanded by inserting parity bits P_i at the position 2^i, where i=0, 1, 2, 3, 4…
II. **How to generate parity bits position.** Once we received a data string, we will mark their position from LSB bits where position 2^i will be left out. The order of the data string should not be changed. The total number of bit positions in this sequence is depending on the length of the codeword.
III. **Determine the parity bits.** For parity bits P_i, the first 2^i-1 bits are marked then the next 2^i bits are marked, and then the next 2^i bits are left out. The last two steps are repeated until the end of the sequence is reached. All the marked bit positions are summed by using modulo-2 addition, except the first marked bit

position. The first marked bit position is not included in the calculation because it is the parity bit position. Here where i=0, 1, 2, 3, 4…

IV. **Insert parity bits into the marked data sequence.** Now the codeword can be constructed by inserting the parity bit P_i into the appropriate position.

The following example will illustrate the above encoding procedures in Table 7. Assuming we have an 8 bit data string D=10101110 and data length k=8, then the parity bits j= 4, the codeword length is 12 bits.

Here we have parity bits P_0, P_1, P_2, P_3; we want to insert this parity bit into the data sequence at position 1, 2, 4 and 8 so the data sequence looks like $D_7D_6D_5D_4P_3D_3D_2D_1P_2D_0P_1P_0$ according to steps I and II. In order to generate the value of parity bits, we need to mark the checking sequence for each parity bit by following step III. For parity bit P_0 the bit positions 1, 3, 5, 7, 9, 11 are marked. For parity bit P_1 the bit positions 2, 3, 6, 7, 10, 11 are marked. For parity bit P_2 the bit positions 4, 5, 6, 7 are marked. For parity bit P_3 the bit positions 8, 9, 10, 11, 12 are marked. Each marked sequence are summed by using modulo-2 addition, which is to calculate how many logic "1" are in the sequence; if there are an odd number "1" then the parity bit is set to "1", otherwise the parity bit is set to "0".

Finally, by inserting these parity bits into the data-word as described in step I, a non-systematic codeword is created.

Bit position	12	11	10	9	8	7	6	5	4	3	2	1	Parity bit Value
	D_7	D_6	D_5	P_3	D_4	D_3	D_2	P_2	D_1	D_0	P_1	P_0	
Dataword	1	0	1	0		1	1	1		0			
P_0		⊕		⊕		⊕		⊕		⊕			0
P_1		⊕	⊕			⊕	⊕			⊕			1
P_2	⊕					⊕	⊕	⊕					0
P_3	⊕	⊕	⊕	⊕									0
Codeword	1	0	1	0	0	1	1	1	0	0	1	0	

Table 7: The encoding process of hamming codeword (12, 8, 1).

5.3 Hamming Decoding

Based on the parity bit we generated before, we are able to detect and correct error bits by using a Hamming decoding algorithm. The procedure of how to perform this algorithm is described below:

I. **Identify the data information from the codeword.** It would be easier to identify the parity bit first then eliminate the parity bits from the codeword; the remaining bits are the data-word. The parity bits are the bits sitting at position 2^i for i=0, 1, 2, 3, 4…

II. **Calculating new parity bits.** Calculate new parity bits by applying the same encoding algorithm as introduced in last section.

III. **Generate syndrome bits.** The syndrome bits are generated by comparing the new parity bits with the parity bits previously inserted in the codeword. The comparison can be performed by using the XOR bitwise operation.

IV. **Check the syndrome value**. If the result of the comparison from step III is equal to 0, then there is no error in the codeword. If there is an error, the value of the syndrome represents the error bit position. The error can be corrected by flipping the data at this bit position.

Assume we have received a codeword we generated from the last section but the data is flipped at the bit position 3. So the codeword sequence now is 101001110**1**10. Following the above steps, first we separate the data-word and parity bits from the codeword by identifying the parity bits from position 2^i: data-word=10101111 and parity bits=0010. Step II is to calculating the new parity bits by applying the encoding algorithm. The new parity bits are 0001. The syndrome value is generated by (0010) XOR (0001) = 0011 and this binary representation translated to a decimal number is 3, which is the error bit position.

5.4 Hardware Implementation and Simulation Results

The encoding and decoding algorithm are explained in section 5.1 and 5.2. An XOR gate is frequently used in Hamming encoders and decoders. Several two-input XOR gates are studied and verified in [9]. The number of transistors in different two-input XOR gates vary from 12 to 5 transistors and these designs use either CMOS logic or pass-transistor logic. Here we adopted a two-input XOR gate with two PMOS transistors connected in a series and two NMOS transistors connected in parallel, and there are three paths connected to the output to give an inverted output. The XOR gate circuit schematic is shown in Figure 54 and the simulation waveform is shown in Figure 55.

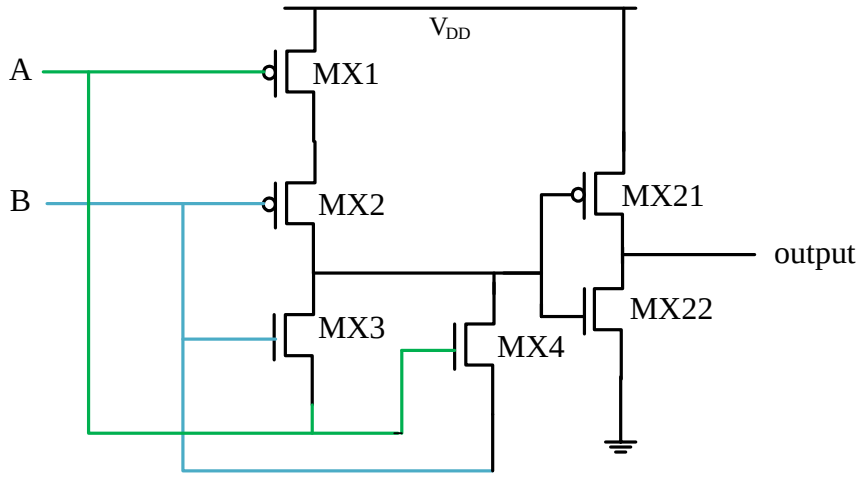


Figure 54: A two-input XOR gate design.

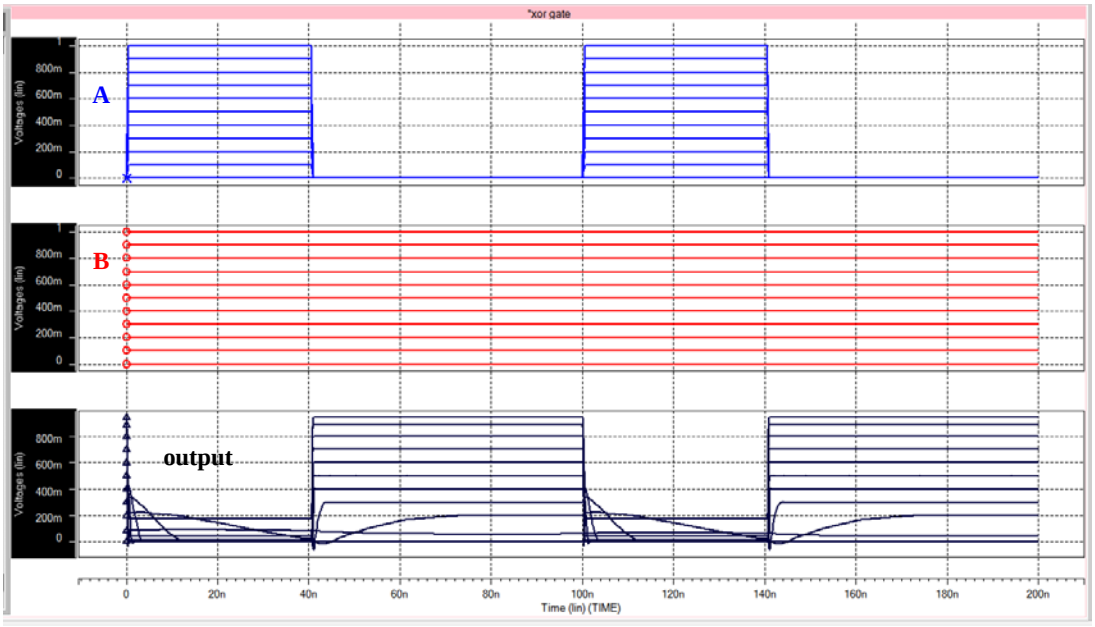


Figure 55: Simulation waveform of the proposed XOR gate.

The proposed XOR gate with the same CMOS technology model as introduced before was able to operate at 0.3V voltage supply. The time delay is 6.578ns. In this project, the minimum operating voltage for SRAM cell is 0.6V so the Hamming encoder and decoder will have the same supply voltage, and the time delay for the XOR gate is 0.1433ns.

As described in the above section, to construct a Hamming code for an 8 bit data string, we need 4 bits parity bit to form a 12 bit codeword. Figure 56 shows the write and read operation for an ECC protected memory array.

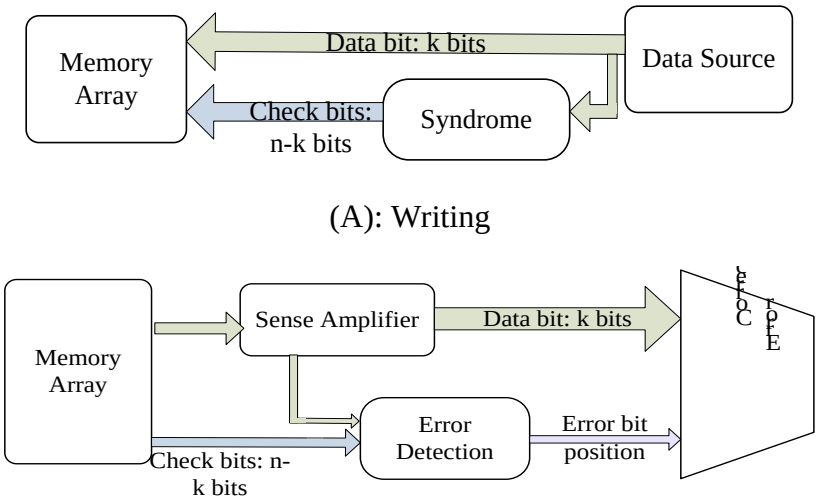


Figure 56: An ECC-protected SRAM during write and read cycle.

The data-word will be sent to the syndrome generating circuit to generate parity bits then store the data-word and parity bits in an SRAM array at the same time. At the beginning of the read cycle, the codeword is first read by the sense amplifiers then the data will be sent to the error detection circuit; the syndrome value is generated at the same time the data-word is sent to an error correction circuit by the sense amplifiers. The error correction circuit will flip the error bit position based on the syndrome value.

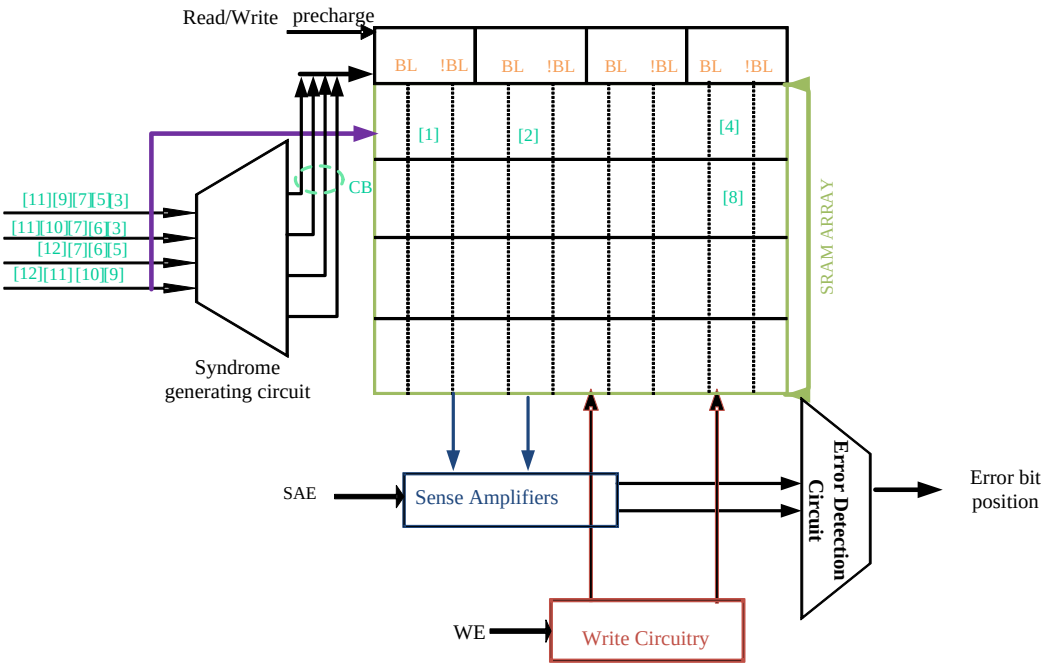
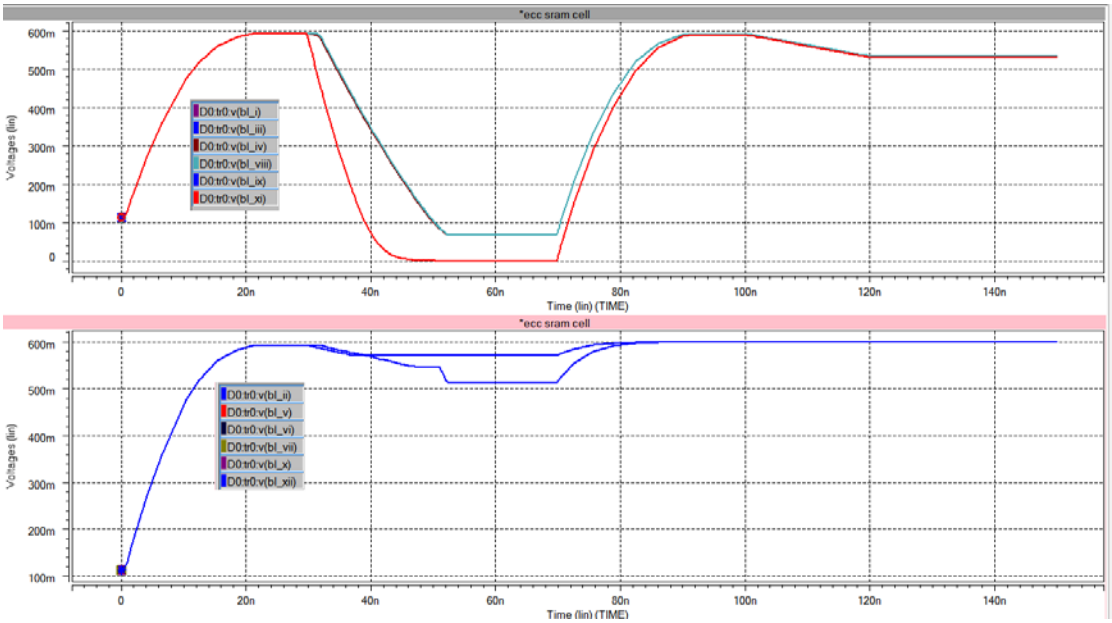
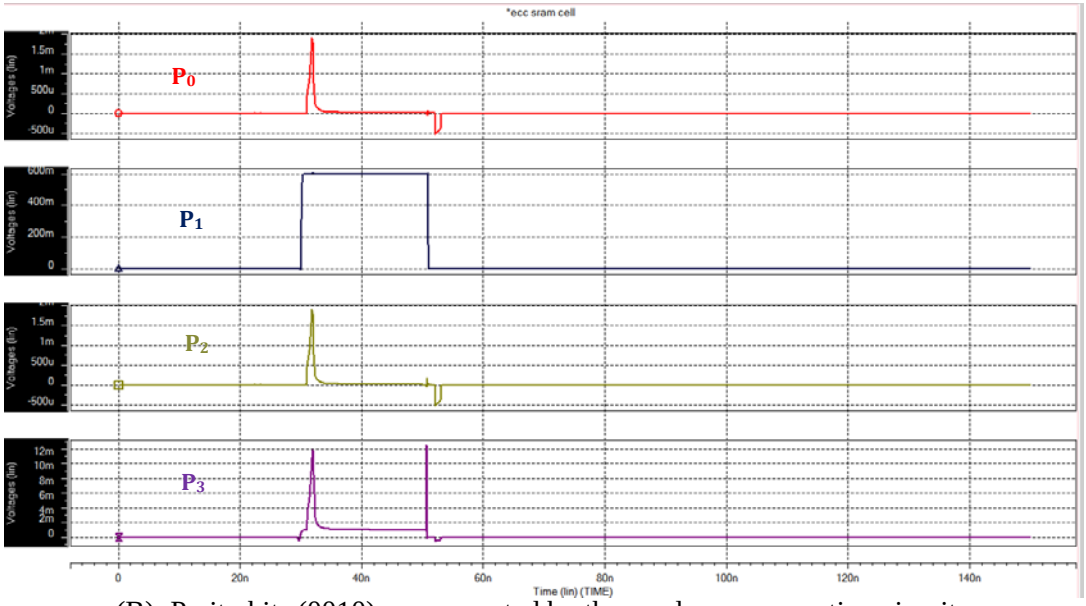


Figure 57: Block diagram of the ECC-protected SRAM memory array.

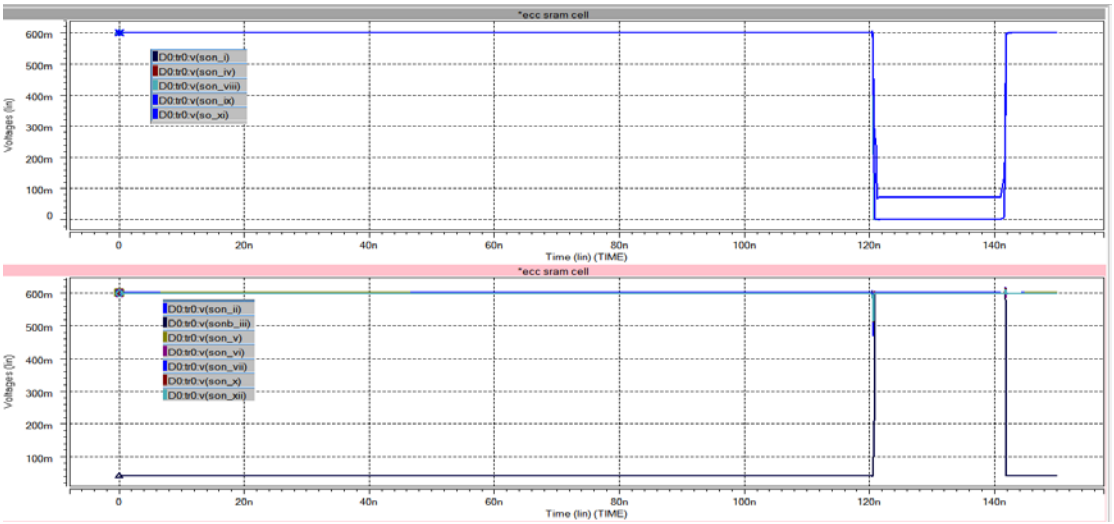
The block diagram of the proposed ECC-protected SRAM array is shown in Figure 57. The syndrome generating circuit consists of 12 two-input XOR gates and the internal connection of these XOR gates follows the encoding algorithm step III in section 5.1. The word line (WL) signal is only enabled after the parity bits are ready to write into the memory arrays. The parity bits and data bits are written into memory arrays by the write driver. In order to check the result of our design, the bit position 3 is connected to an inverter so the data is flipped during a read cycle. After pre-charging bit lines to V_{DD}, the WL signals are set to high again then the data is read by the sense amplifier. The codeword is sent to the Hamming decoder, which consists of a Hamming encoder and four extra XOR gates to create the syndrome value. The simulation result is shown in Figure 58.



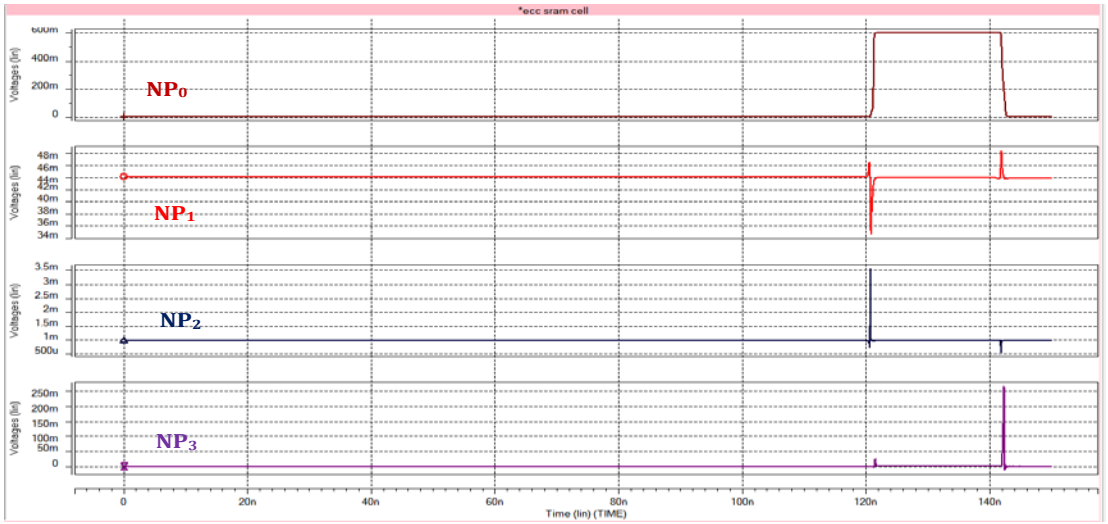
(A): The data stored in the SRAM array, data-word (10101110) and parity bits (0010).



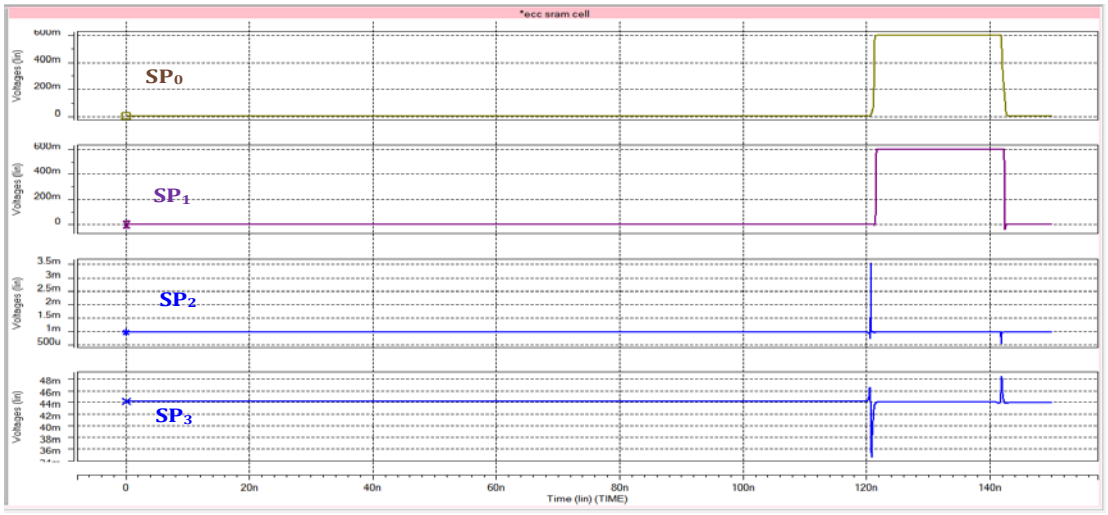
(B): Parity bits (0010) are generated by the syndrome generating circuit.



(C): The data read from the SRAM array, data is flipped at bit position 3(SONB_III).



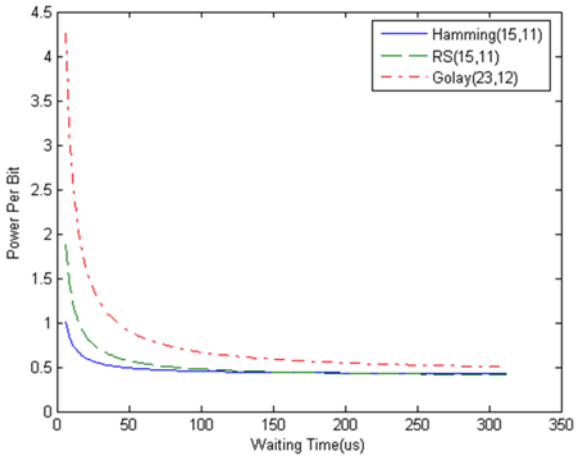
(D): The new parity bits (0001) are generated by the error detecting circuit.



(E): The syndrome value (0011) is generated by the error detecting circuit.

Figure 58: the simulation waveform of the ECC-protected SRAM array.

The time delay for the parity bits to be generated is measured from when the bit line for bit position 5 is raised to 50% of V_{DD} to when the output of parity bit P_2 is raised to 50% of V_{DD}, and it is 0.6812ns. The time delay for the syndrome value to be generated is the measure from when the new parity bits NP_0 is raised to 50% of V_{DD} to when the syndrome bits SP_0 is raised to 50% of V_{DD}, which is 0.5761ns. The total time for error detecting and correcting is 1.2661ns. The total power dissipation is 0.3527mW.



Number of columns	Standby Voltage (mV)		
	Hamming	RS	Golay
256	201.36	194.78	176.36
512	206.53	198.59	179.26
1024	210.04	201.76	182.12
Hardware overhead (gates/bit)	9.091	22.727	58.333

Figure 59: Comparison between different ECC codes used in SRAM Design. [10]

From Figure 61, we can see the Hamming code requires less power with a bigger memory array compared to other ECC codes. For the Hamming code, every time the data bits double, the bit number of parity bits requires one more bit, so the ratio of the bit number of parity bits to that of the data decreases with the bits number of data. Even the two-input XOR gate has the same number of transistors as a 6T SRAM cell but the XOR gate cell area is only 10.21nm and the Hamming code total overhead is around 20%. The hamming encoder and decoder can apply to different SRAM designs and embedded with sense amplifiers.

6 Conclusions

This thesis presents five different low-voltage SRAM designs in 45nm high performance CMOS. The cell ratio of each SRAM design is calculated to give the biggest SNM at their minimum operating voltage.

We found that the 6T SRAM has a very high failure rate under threshold voltage and T_{ox} variation at low voltage supply, and the cell stability will be reduced relatively more with the increase in temperature. But it is still a good design in terms of power and cell area.

The 7T SRAM has a small write failure rate under the same simulation environment but with very poor read stability; the read-SNM is further reduced by increasing the temperature. The cell can be improved by giving it a right cell ratio and bit-line voltage scaling method to give a better read operation. The cell can work at very low power and has a relatively simple cell structure but quite a bigger cell area.

The 8T SRAM cell gives relatively small read and writes failure rates but the write stability is affected most by temperature. The cell SNM can be improved by sizing two access transistors and a large amount of power can be saved depending on the probability of the number of logic “1” in the data information.

The 9T SRAM has better read and write failure rates compared to other designs; it also gives good SNM under process variation and the SNM degradation percentage is quite small but it is at the cost of a larger cell area, more complex structure and more power consumption.

The 10T SRAM has more power consumption and a more complex cell structure compared to the 9T SRAM, but its performance is worse than the 9T SRAM cell, especially for read stability.

The Hamming code for double error detection and single error correction is investigated, and the ECC-protected SRAM array is proposed. The proposed encoder requires 0.6812ns to generate parity bits and the decoder requires 1.2661ns to detect errors and generate error bit position at 0.6V using 45nm CMOS for 8-bit word length. The proposed Hamming encoder and decoder have half of the area, faster decoding speed and much less power consumption compared to the conventional iterative decoding scheme. The encoder and decoder can connect to the column address decoder and the parity bits can be stored in the single column of the memory array rather than inserted into the data sequence.

7 Future Work

SRAM memory cells are capable of storing one bit of data and these cells are generally laid out as arrays, and each bit is addressed as a row/column intersection. Errors usually occur in three ways: row failures, column failures and individual cell failures. The method used to control these errors during manufacture is row/column replacement. In paper [11], this row/column replacement strategy is examined and the limitation of this method is due to the large amount of memory arrays required for current microprocessors. Different approaches of redundant rows/columns replacement are introduced in [12] and in [13] to improve memory performance, but they have limitations in the number of fault row/columns that the designs can handle. Achieving a better algorithm for error correction blocks is an open question.

References

1. Asenov, A. “Random Dopant Induced Threshold Voltage Lowering and Fluctuations in Sub-0.1 um MOSFET's: A 3-D 'Atomistic' Simulation Study” *IEEE Trans. On Electron Devices*, vol. 45, no. 12, Dec 1998.
2. Rabaey, J.M. Process Variations In: EE241: Advanced Digital Integrated Circuits lecture notes in Electrical & Electronics and Computer Science, University of California at Berkeley, 2006.
3. Liu, D. and Svensson, C. “Trading speed for low power by choice of supply and threshold voltages,” *IEEE Journal of Solid-State Circuits*, vol. 28, no. 1, pp. 10-17, January 1993.
4. Aly, R.E.; Bayoumi, M.A. "Low-Power Cache Design Using 7T SRAM Cell," Circuits and Systems II: Express Briefs, *IEEE Transactions* on , vol. 54, no. 4, pp. 318-322, April 2007.
5. Sil, A.; Ghosh, S.; Bayoumi, M. "A novel 8T SRAM cell with improved read-SNM," Circuits and Systems, 2007. NEWCAS 2007. *IEEE Northeast Workshop on* , pp. 1289-1292, 5-8 Aug. 2007.
6. Zhiyu Liu; Kursun, V.; "Characterization of a Novel Nine-Transistor SRAM Cell," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* , vol. 16, no. 4, pp.4 88-492, April 2008.
7. Ik Joon Chang; Jae-Joon Kim; Park, S.P.; Roy, K. "A 32kb 10T Subthreshold SRAM Array with Bit-Interleaving and Differential Read Scheme in 90nm CMOS,"*Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International* , pp. 388-622, 3-7 Feb. 2008.
8. Aditya Bansal *et al.*, “Impacts of NBTI and PBTI on SRAM Static/Dynamic Noise Margins and Cell Failure Probability”
9. Shen-Fu Hsiao; Chia-Sheng Wen; Ming-Yu Tsai; Ming-Chih Chen. "Automatic generation of high-performance multiple-input XOR/XNOR circuits and its application in Advanced Encryption Standard (AES)," *Next-Generation Electronics (ISNE), 2010 International Symposium on* pp.77-80, 18-19 Nov. 2010
10. Hsin-I Liu, “On-Chip ECC for Low-Power SRAM Design”
11. Horiguchi, M. “Redundancy techniques for high-density DRAMs,” Innovative Systems in Silicon, 1997. Proceedings. Second Annual IEEE International Conference on , pp. 22-29, 8-10 Oct 1997.
12. H. L. Kalter, *et al*. “A 50-ns 16-Mb DRAM with a 10 ns data rate and on-chip ECC,” IEEE Journal of Solid-State Circuits, vol. 25, pp. 1118–1128, Oct.1990.
13. Agarwal, A.; Paul, B.C.; Mahmoodi, H.; Datta, A.; Roy, K. “A process-tolerant cache architecture for improved yield in nanoscale technologies,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol. 13, no. 1, pp. 27- 38, Jan. 2005.
14. http://ptm.asu.edu/
15. Ying Ying Zhang Thesis “VLSI implementation of add-compare-select unit for high performance Viterbi decoder”. 2009

Appendix: HSPICE Source code

Due to page restriction, only five SRAM designs and ECC protected SRAM arrays are shown in this appendix.

```
*6T SRAM Cell
*parameter define
.param cload=5pf
.param vdd=1
.param l=45n
.param w=90n
.param nvt= 0.46893
.param pvt= -0.49158v
*.param nvt= agauss(0.469v,0.046v,3)
*+pvt= agauss(-0.469v,0.046v3)
.param ntox=1.25n
.param ptox=1.3n
*.param ntox= agauss(1.25n,0.25n,3)
*+ptox= agauss(1.3n,0.26n,3)
.GLOBAL 1
*set initial state
.IC V(Q)=0
.IC V(QB)='vdd'
VCC 1 0 'vdd'
Vcharge  Charge 0  'vdd'
*********************************************************************************
*supply voltage source define
VPC PC 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 129n 0v, 130n 'vdd', 150n  'vdd', 151n 0v,  194n  0v,
+195n 'vdd', 215n 'vdd', 216n 0v, 250n 0v,R 0
*VLPC LPC 0 pwl  0n 0v, 69n 0v,  70n   'vdd',  90n 'vdd', 91n 0v,
*+194n  0v, 195n 'vdd', 215n 'vdd', 216n 0v, 260n 0v,R 0
VWL WL 0 pwl 0n 0v,  39n 0v, 40n 'vdd', 50n 'vdd', 51n  0v ,100n  0v , 101n 'vdd',
+120n 'vdd', 121n 0v,164n 0v,  165n 'vdd',175n 'vdd',176n  0v,
+224n 0v , 225n   'vdd', 245n 'vdd', R 0
```

```
VW W 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
+154n 0v ,155n 'vdd', 175n 'vdd', 176n 0v,245n 0v,R 0
VD  D  0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
+154n 0v ,176n 0v,245n 0v,R 0
VDB  DB  0 pwl 0n 0v, 154n 0v ,155n 'vdd',  175n 'vdd',176n 0v,245n 0v,R 0
*VSA SAE 0 pwl 0n 0v,  120n 0v,121n 'vdd', 129n 'vdd',  130n 0v, 245n 0v,246n 'vdd',
*+254n 'vdd', 255n 0v, 260n 0v,  R 0
************************************************************
*Precharge Circuit
XIPC PC PCB INV
MG2 BLB  PCB Charge Charge      PMOS      W='18*w'  L='l'
MG3 BL PCB  Charge Charge                  PMOS      W='18*w'  L='l'
************************************************************
CBL       BL 0 cload
CBLB    BLB 0 cload
************************************************************
*6T SRAM netlist
M1 Q QB 1 1                PMOS     W='w'          L='l'
M2 Q QB 0 0                NMOS     W='1.2*w'  L='l'
M3 QB Q 1 1                PMOS     W='w'          L='l'
M4 QB Q 0 0                NMOS      W='1.2*w' L='l'
M5 Q WL BL 0               NMOS      W='w'              L='l'
M6 QB WL BLB 0             NMOS      W='w'              L='l'
************************************************************
*Write Circuit
MW    BL W Qw 0            NMOS W='10*w' L='l'
MWB BLB W Qwb 0 NMOS W='10*w' L='l'
MRB Qwb D 0 0 NMOS W='10*w' L='l'
MR    Qw DB 0 0            NMOS W='10*w' L='l'
************************************************************
.SUBCKT INV in out1
```

```
*subcircuit of a  inverter
MI1 out1 in 1 1 PMOS W='2*w' L='l'
MI3 out1 in 0 0 NMOS W='w' L='l'
.ENDS INV
*********************************************************
************************************************************
*Sense Amplfier
*MQ7  SO SAE 1 1                  PMOS      W='2*w'  L='l'
*MQ2  SO SON 1 1          PMOS      W='2*w'  L='l'
*MQ1  SO SON Y 0          NMOS     W='w' L='l'
*MQ8  SON SAE 1 1                 PMOS      W='2*w'  L='l'
*MQ4  SON SO 1 1          PMOS      W='2*w'  L='l'
*MQ3  SON SO YN 0         NMOS     W='w' L='l'
*MQ5  Y BL COM 0          NMOS      W='8*w'  L='1.5*l'
*MQ6  YN BLB COM 0                NMOS     W='8*w' L='1.5*l'
*MQ9  COM SAE 0 0         NMOS     W='6*w' L='l'
************************************************************
.TRAN 0.1n 260n  *Sweep monte=1000
.meas tran pow AVG power
*****************************Measure read time with SA**************
*.MEASURE ReadTimeSA     trig v(WL) VAL='0.5*vdd'  Rise=2
*+                  targ v(SO)   VAL='0.5*vdd' fall=1
*****************************Measure read time without SA**************
.MEASURE ReadTime    trig v(WL) VAL='0.5*vdd'  Rise=2
+                  targ v(BLB)   VAL='0.9*vdd' fall=2
*****************************Measure write time **************
.MEASURE WriteTime   trig v(WL) VAL='0.5*vdd'  Rise=1
+                  targ v(QB)   VAL='0.5*vdd'  Fall=1
.option  nopage nomod  post
.END
*******************************
*7T SRAM Cell
```

```
* Define supply voltage sources
*VPC  PC 0          pulse 0 'vdd'  10n 1n 1n 50n  300n
*VWL WL 0        pulse 0 'vdd'   60n 1n 1n 50n  102n
*VW    W 0        pulse 0   'vdd'   60n 1n 1n 50n  200n
*VR     R  0          pulse 0 'vdd'  110n 1n 1n 100n 152n
*VWR WR 0        pulse 0 'vdd' 800n 1n 1n 100n  202n
*VWRB WRB 0   pulse 0 'vdd'  110n 1n 1n 100n  252n
*VD     D  0           pulse 0 'vdd' 110n 1n 1n 100n  202n
*****************************************************************
VPC        PC 0    pulse 0 'vdd'  10n 1n 1n 40n  300n
VWL       WL 0 0
VW         W 0 1
VR          R  0 1
VWR        WR 0 0
VWRB    WRB 0 1
VD           D  0 1
*********************************************************************
*Precharge Circuit
*XIPC PC PCB INV
*MPC  BL PCB 0 0  NMOS W='w' L='l'
*MPCB BLB PCB 0 0  NMOS W='w' L='l'
XIPC PC PCB INV
MPC    BL  PCB 1 1  PMOS W='w' L='l'
MPCB BLB PCB 1 1  PMOS W='w' L='l'
*7T SRAM netlist
MN3 Q2 WL  BLB 0  NMOS      W='1.3*w'  L='l'
MP2  Q Q2 1 1           PMOS      W='w'  L='l'
MN2  Q Q2 0 0          NMOS      W='2.6*w' L='l'
MN5  Q2 W QB 0       NMOS      W='4*w'  L='l'
MP1   QB Q 1 1          PMOS      W='2*w'   L='l'
MN1   QB Q 0 0         NMOS      W='3.3*w'   L='l'
MN4    Q R BL  0        NMOS       W='w'   L='l'
```

```
*Write Circuit
MW    BL WR Qw 0        NMOS W='w' L='l'
MWB BLB WRB Qwb 0 NMOS W='w' L='l'
XIOUT D DB INV
MR    Qw D 0 0 NMOS W='w' L='l'
MRB Qwb DB 0 0 NMOS W='w' L='l'
.SUBCKT INV in out1
*subcircuit of a  inverter
MI1 out1 in 1 1 PMOS W='2*w' L='l'
MI3 out1 in 0 0 NMOS W='w' L='l'
.ENDS INV
CBL BL 0 cload
CBLB BLB 0 cload
.TRAN 0.1n 500n $SWEEP cload 0.1p 1p 0.1p (incre)
.MEASURE rms_pow rms power
.meas tran pow AVG power
*.MEASURE ReadTime    trig v(SAE) VAL='0.5*vdd'  Rise=1
*+                targ v(OUT2)   VAL='0.5*vdd' rise=1
*.MEASURE WriteTime   trig v(WL) VAL='0.5*vdd'  Rise=1
*+                targ v(QB)   VAL='0.5*vdd'  Fall=1
.option  nopage nomod  post
**********************
8T SRAM Cell
* Define supply voltage sources
VPC       PC 0          pwl 0n 0v, 4n 0v, 5n 'vdd', 25n  'vdd', 26n  0v,
                                          +54n 0v ,55n 'vdd', 75n 'vdd', 76n 0v,104n 0v,
                                          +105n 'vdd', 125n  'vdd', 126n  0v,  159n 0v ,
                                          +160n 'vdd', 180n 'vdd',181n 0v,245n 0v,R 0
VD     D  0              pwl 0n 0v,  129n 0v,130n 'vdd', 150n 'vdd',
                                          +151n 0v,245n 0v,R 0

VWWL  WWL 0      pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
```

```
                                        +134n 0v,135n 'vdd', 155n 'vdd',156n 0v,245n 0v,R 0

VRWL   RWL  0       pwl 0n 0v, 79n 0v, 80n 'vdd', 100n  'vdd', 101n  0v,
                                        +184n 0v,185n 'vdd', 205n 'vdd',206n 0v,245n 0v,R 0
************************************************************
*Precharge Circuit
XIPC PC PCB INV
MPC    BL  PCB 1 1  PMOS W='15*w' L='l'
CBL BL 0 cload
*8T SRAM netlist
M6 Q2 RWL  BL 0          NMOS      W='2*w'  L='l
M8  Q2 QB 1 1            PMOS      W='4*w'  L='l'
M7  Q2 QB 0 0            NMOS      W='2*w' L='l'
M2  QB Q 1 1             PMOS      W='w'  L='l'
M1   QB Q 0 0            NMOS      W='4*w'   L='l'
M4   Q QB 1  1           PMOS      W='2*w'   L='l'
M3    Q QB 0  0          NMOS       W='w'   L='l'
M5   BL WWL Q 0          NMOS       W='7*w'   L='l'
*Write Circuit
MD    BL D 0 0 NMOS W='9*w' L='l'
.SUBCKT INV in out1
*subcircuit of a  inverter
MI1 out1 in 1 1 PMOS W='2*w' L='l'
MI3 out1 in 0 0 NMOS W='w' L='l'
.ENDS INV
.TRAN 0.1n 250n * SWEEP Vdd 0v 1v 0.05v
*.TRAN 0.1n 250n Sweep monte=1000
*.MEASURE rms_pow rms power
.meas tran pow AVG power
**************************Measure read time without SA*************
.MEASURE ReadTime    trig v(RWL) VAL='0.5*vdd'  Rise=2
+                    targ v(BL)   VAL='0.9*vdd' Fall=2
```

```
**************************Measure write time *************
.MEASURE WriteTime   trig v(WWL) VAL='0.5*vdd'  Rise=1
+                    targ v(Q)   VAL='0.5*vdd'  Rise=1
.option  nopage nomod  post
*********************************************************************************
*supply voltage source define
9T SRAM cell
VPC      PC 0      pwl 0n 0v, 4n 0v, 5n 'vdd', 25n  'vdd', 26n  0v,
                                                +59n 0v ,60n 'vdd', 80n 'vdd', 81n 0v,109n 0v,
                                                +110n 'vdd', 130n  'vdd', 131n  0v,  164n 0v ,
                                                +165n 'vdd', 185n 'vdd',186n 0v,245n 0v,R 0
VWR      WR 0      pwl 0n 0v, 34n 0v, 35n 'vdd', 55n  'vdd', 56n  0v,
                                                +139n 0v,140n 'vdd', 160n 'vdd',161n 0v,245n 0v,R 0
VD       D  0      pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
                                                +245n 0v,R 0
VWE      WE 0      pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
                                                +134n 0v,135n 'vdd', 155n 'vdd',156n 0v,245n 0v,R 0
VRD      RD 0      pwl 0n 0v, 84n 0v, 85n 'vdd', 105n  'vdd', 106n  0v,
                                                +189n 0v,190n 'vdd', 210n 'vdd',211n 0v,245n 0v,R 0
*Precharge Circuit
XIPC PC PCB INV
MPC      BL  PCB 1 1       PMOS W='15*w' L='l'
MPCB     BLB PCB 1 1       PMOS W='15*w' L='l'
*9T SRAM netlist
MP1 Q1 Q2 1 1         PMOS     W='w' L='l'
MN1 Q1 Q2 0 0         NMOS     W='w'  L='l'
MP2 Q2 Q1 1 1         PMOS      W='w'   L='l'
MN2 Q2 Q1 0 0         NMOS      W='w'   L='l'
M3 Q1 WR BL 0         NMOS      W='w'   L='l'
M4 Q2 WR BLB 0        NMOS       W='w'   L='l'
M5 Q3 Q1 BL 0         NMOS       W='1.5*w'   L='l'
M6 Q3 Q2 BLB 0        NMOS       W='1.5*w'   L='l'
```

```
M7 Q3 RD 0 0          NMOS       W='2*w'   L='l'
*Write Circuit
MWE    BL   WE Qw 0 NMOS W='9*w' L='l'
MWEB BLB WE Qwb 0 NMOS W='9*w' L='l'
XIOUT D DB INV
MR    Qw DB 0 0 NMOS W='9*w' L='l'
MRB Qwb D 0 0 NMOS W='9*w' L='l'
.SUBCKT INV in out1
*subcircuit of a  inverter
MI1 out1 in 1 1 PMOS W='2*w' L='l'
MI3 out1 in 0 0 NMOS W='w' L='l'
.ENDS INV
CBL BL 0 cload
CBLB BLB 0 cload
.TRAN 0.1n 250n  *SWEEP Vdd 0v 1v 0.05v
*.TRAN 0.1n 250n Sweep monte=1000

.meas tran pow AVG power
*****************************Measure read time without SA*************
.MEASURE ReadTime    trig v(RD) VAL='0.5*vdd'  Rise=1
+                    targ v(BL)   VAL='0.9*vdd' Fall=2
*****************************Measure write time *************
.MEASURE WriteTime   trig v(WR) VAL='0.5*vdd'  Rise=1
+                    targ v(Q1)   VAL='0.5*vdd'  Rise=1
.option  nopage nomod  post
*********************************************************************
* Define supply voltage sources
VPC       PC 0                  pwl 0n 0v, 4n 0v, 5n 'vdd', 25n  'vdd', 26n  0v,
                                          +64n 0v ,65n 'vdd', 85n 'vdd', 86n 0v,114n 0v,
                                          +115n 'vdd', 135n  'vdd', 136n  0v,  164n 0v ,
                                          +165n 'vdd', 185n 'vdd',186n 0v,245n 0v,R 0
```

```
VD      D  0            pwl 0n 0v, 34n 0v, 35n 'vdd', 55n  'vdd', 56n  0v,
                                    +245n 0v,R 0

VDB     DB  0          pwl 0n 0v, 144n 0v,145n 'vdd',
                                    +165n 'vdd',166n 0v,245n 0v,R 0

VWL     WL  0          pwl 0n 0v, 34n 0v, 35n 'vdd', 55n  'vdd', 56n  0v,
                                    +89n 0v ,90n 'vdd', 110n 'vdd', 111n 0v,144n 0v,
                                    +145n 'vdd', 165n  'vdd', 166n  0v,  199n 0v ,
                                    +200n 'vdd', 220n 'vdd',221n 0v,245n 0v,R 0

VWWL  WWL 0            pwl 0n 0v, 34n 0v, 35n 'vdd', 55n  'vdd', 56n  0v,
                                    +144n 0v,145n 'vdd',165n 'vdd',166n 0v,
                                    +245n 0v,R 0

VRD     RD 0           pwl 0n 0v, 89n 0v, 90n 'vdd', 110n  'vdd', 111n  0v,
                                    +199n 0v,200n 'vdd', 220n 'vdd',221n 0v,245n 0v,R 0

VCLK    CLK 0          pwl 0n 0v, 89n 0v, 90n 'vdd', 110n  'vdd', 111n  0v,
                                    +199n 0v,200n 'vdd', 220n 'vdd',221n 0v,245n 0v,R 0

**********************************************************************

*Precharge Circuit
XIPC PC PCB INV
MPC     BL  PCB 1 1     PMOS W='20*w' L='l'
MPCB    BLB PCB 1 1     PMOS W='20*w' L='l'

**********************************************************************

*Write Data Circuit
MR    BL DB 0 0                 NMOS W='9*w' L='l'
```

```
MRB BLB D 0 0 NMOS W='9*w' L='l'

**********************************************************************

*VGND Circuit

M18 C1 CLK 1 1                  PMOS     W='w'  L='l'

M19 C1 RD 1 1                   PMOS     W='w'  L='l'

M20 C1 RD C2 0                  NMOS     W='w'  L='l'

M21 C2 CLK 0 0                  NMOS     W='w'  L='l'

M23 C3 C1 1 1                   PMOS     W='2*w'  L='l'

M22 C3 C1 0 0                   NMOS     W='w'  L='l'

M24 VGND C3 1 C3        PMOS     W='2*w'  L='l'

M25 VGND C3 0 C3        NMOS     W='w'  L='l'

**********************************************************************

*10T SRAM netlist

MAL2  Q2  WL  BLB 0             NMOS     W='3*w'  L='l'

MAL1  Q  WWL  Q2 0              NMOS     W='w'  L='l'

MP1     Q QB 1 1                PMOS     W='w'  L='l'

MN1     Q QB 0 0                   NMOS     W='1.5*w' L='l'

MNL     Q2  Q  VGND 0              NMOS     W='w'  L='l'

MP2     QB Q 1 1                PMOS     W='w'  L='l'

MN2     QB Q 0 0                NMOS     W='1.5*w'   L='l'

MAR1    QB  WWL  Q3 0           NMOS     W='w'  L='l'

MAR2     Q3  WL  BL 0           NMOS     W='3*w'  L='l'

MNR     Q3  QB  VGND 0          NMOS     W='w'  L='l'


CBL BL 0 cload

CBLB BLB 0 cload

**********************************************************************

.SUBCKT INV in out1

*subcircuit of a  inverter
```

```
MI1 out1 in 1 1 PMOS W='2*w' L='l'
MI3 out1 in 0 0 NMOS W='w' L='l'
.ENDS INV


**************************************************************************
*.TRAN 0.1n 250n  *SWEEP Vdd 0v 1v 0.05v

.TRAN 0.1n 250n *Sweep monte=1000

.meas tran pow AVG power
*****************************Measure read time without SA**************
.MEASURE ReadTime    trig v(WL) VAL='0.5*vdd'  Rise=2
+                    targ v(BL)   VAL='0.9*vdd' Fall=2
*****************************Measure write time **************
.MEASURE WriteTime   trig v(WL) VAL='0.5*vdd'  Rise=1
+                    targ v(QB)   VAL='0.5*vdd'  Rise=1
.option  nopage nomod  post
**************************************************************
*ECC protected SRAM Design
*set initial state
.IC V(Q_I)=0
.IC V(QB_I)='vdd'
.IC V(Q_II)=0
.IC V(QB_II)='vdd'
.IC V(Q_III)=0
.IC V(QB_III)='vdd'
.IC V(Q_IV)=0
.IC V(QB_IV)='vdd'
.IC V(Q_V)=0
.IC V(QB_V)='vdd'
.IC V(Q_VI)=0
```

```
.IC V(QB_VI)='vdd'
.IC V(Q_VII)=0
.IC V(QB_VII)='vdd'
.IC V(Q_VIII)=0
.IC V(QB_VIII)='vdd'
.IC V(Q_IX)=0
.IC V(QB_IX)='vdd'
.IC V(Q_X)=0
.IC V(QB_X)='vdd'
.IC V(Q_XI)=0
.IC V(QB_XI)='vdd'
.IC V(Q_XII)=0
.IC V(QB_XII)='vdd'
VCC 1 0 'vdd'
Vcharge  Charge 0  'vdd'
********************************************************************************
*supply voltage source define
VPC_I PC_I 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_II PC_II 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_III PC_III 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_IV PC_IV 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_V PC_V 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_VI PC_VI 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_VII PC_VII 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
VPC_VIII PC_VIII 0 pwl  0n 0v, 1n 'vdd',  21n 'vdd', 22n  0v,  69n 0v,  70n   'vdd',
+90n 'vdd', 91n 0v, 150n 0v,R 0
```

VPC_IX PC_IX 0 pwl 0n 0v, 1n 'vdd', 21n 'vdd', 22n 0v, 69n 0v, 70n 'vdd',

+90n 'vdd', 91n 0v, 150n 0v,R 0

VPC_X PC_X 0 pwl 0n 0v, 1n 'vdd', 21n 'vdd', 22n 0v, 69n 0v, 70n 'vdd',

+90n 'vdd', 91n 0v, 150n 0v,R 0

VPC_XI PC_XI 0 pwl 0n 0v, 1n 'vdd', 21n 'vdd', 22n 0v, 69n 0v, 70n 'vdd',

+90n 'vdd', 91n 0v, 150n 0v,R 0

VPC_XII PC_XII 0 pwl 0n 0v, 1n 'vdd', 21n 'vdd', 22n 0v, 69n 0v, 70n 'vdd',

+90n 'vdd', 91n 0v, 150n 0v,R 0

VWL WL 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v ,100n 0v , 101n 'vdd',

+120n 'vdd', 121n0v,150n 0v, R 0

VWL_P WL_P 0 pwl 0n 0v, 31n 0v, 32n 'vdd', 52n 'vdd', 53n 0v ,100n 0v , 101n 'vdd',

+120n 'vdd', 121n0v,150n 0v, R 0

VWE WE 0 pwl 0n 0v, 31n 0v, 32n 'vdd', 52n 'vdd', 53n 0v, 150n 0v,R 0

VRE RE 0 pwl 0n 'vdd', 99n 'vdd' , 100n 0v,120n 0v, 121n 'vdd' ,151 'vdd',R 0

**********Data sequence is "101001110010"(most Left bit is zero position bit)**********

VD_III D_III 0 0

VDB_III DB_III 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v, 150n 0v,R 0

VD_V D_V 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v, 150n 0v,R 0

VDB_V DB_V 0 0

VD_VI D_VI 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v, 150n 0v,R 0

VDB_VI DB_VI 0 0

VD_VII D_VII 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v, 150n 0v,R 0

VDB_VII DB_VII 0 0

VD_IX D_IX 0 0

VDB_IX DB_IX 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v, 150n 0v,R 0

VD_X D_X 0 pwl 0n 0v, 29n 0v, 30n 'vdd', 50n 'vdd', 51n 0v, 150n 0v,R 0

VDB_X DB_X 0 0

```
VD_XI          D_XI    0                    0

VDB_XI         DB_XI   0                    pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
150n     0v,R 0


VD_XII         D_XII   0                    pwl 0n 0v, 29n 0v, 30n 'vdd', 50n  'vdd', 51n  0v,
150n     0v,R 0

VDB_XII        DB_XII 0                     0


VSA            SAE     0                    pwl  119n  0v , 120n 0v, 121n 'vdd', 141n 'vdd',
142n 0v, 145n 0v, R 0

************************************************************

*Precharge Circuit_1

XIPC_I PC_I PCB_I INV

MG2_I  BLB_I   PCB_I   Charge Charge          PMOS     W='18*w'  L='l'

MG3_I  BL_I    PCB_I   Charge Charge          PMOS     W='18*w'  L='l'

************************************************************

************************************************************

*Precharge Circuit_2

XIPC_II PC_II PCB_II INV

MG2_II BLB_II          PCB_II  Charge Charge          PMOS     W='18*w'  L='l'

MG3_II BL_II   PCB_II  Charge Charge          PMOS     W='18*w'  L='l'

************************************************************

************************************************************

*Precharge Circuit_3

XIPC_III PC_III PCB_III INV

MG2_III  BLB_III  PCB_III Charge Charge       PMOS     W='18*w'  L='l'

MG3_III  BL_III PCB_III  Charge Charge        PMOS     W='18*w'  L='l'

************************************************************

************************************************************

*Precharge Circuit_4

XIPC_IV PC_IV PCB_IV INV

MG2_IV BLB_IV  PCB_IV Charge Charge           PMOS     W='18*w'  L='l'

MG3_IV BL_IV PCB_IV  Charge Charge            PMOS     W='18*w'  L='l'
```

```
************************************************************
************************************************************
*Precharge Circuit_5
XIPC_V PC_V PCB_V INV
MG2_V BLB_V  PCB_V Charge Charge     PMOS     W='18*w'  L='l'
MG3_V BL_V PCB_V  Charge Charge              PMOS     W='18*w'  L='l'
************************************************************
************************************************************
*Precharge Circuit_6
XIPC_VI PC_VI PCB_VI INV
MG2_VI BLB_VI  PCB_VI Charge Charge          PMOS     W='18*w'  L='l'
MG3_VI BL_VI PCB_VI  Charge Charge           PMOS     W='18*w'  L='l'
************************************************************
************************************************************
*Precharge Circuit_7
XIPC_VII PC_VII PCB_VII INV
MG2_VII BLB_VII  PCB_VII Charge Charge       PMOS     W='18*w'  L='l'
MG3_VII BL_VII PCB_VII  Charge Charge        PMOS     W='18*w'  L='l'
************************************************************
************************************************************
*Precharge Circuit_8
XIPC_VIII PC_VIII PCB_VIII INV
MG2_VIII BLB_VIII  PCB_VIII Charge Charge            PMOS     W='18*w'  L='l'
MG3_VIII BL_VIII PCB_VIII  Charge Charge             PMOS     W='18*w'  L='l'
************************************************************
************************************************************
*Precharge Circuit_9
XIPC_IX PC_IX PCB_IX INV
MG2_IX BLB_IX  PCB_IX Charge Charge          PMOS     W='18*w'  L='l'
MG3_IX BL_IX PCB_IX  Charge Charge           PMOS     W='18*w'  L='l'
************************************************************
************************************************************
```

```
*Precharge Circuit_10
XIPC_X PC_X PCB_X INV
MG2_X BLB_X  PCB_X Charge Charge              PMOS     W='18*w'  L='l'
MG3_X BL_X PCB_X  Charge Charge               PMOS     W='18*w'  L='l'
************************************************************
************************************************************
*Precharge Circuit_11
XIPC_XI PC_XI PCB_XI INV
MG2_XI BLB_XI  PCB_XI Charge Charge           PMOS     W='18*w'  L='l'
MG3_XI BL_XI PCB_XI  Charge Charge            PMOS     W='18*w'  L='l'
************************************************************
************************************************************
*Precharge Circuit_12
XIPC_XII PC_XII PCB_XII INV
MG2_XII BLB_XII  PCB_XII Charge Charge                PMOS     W='18*w'  L='l'
MG3_XII BL_XII PCB_XII  Charge Charge         PMOS     W='18*w'  L='l'
************************************************************
CBL_I           BL_I            0       cload
CBLB_I          BLB_I           0       cload

CBL_II                  BL_II           0       cload
CBLB_II         BLB_II          0       cload

CBL_III         BL_III          0        cload
CBLB_III        BLB_III                 0        cload

CBL_IV          BL_IV           0       cload
CBLB_IV         BLB_IV                  0       cload

CBL_V                   BL_V            0       cload
CBLB_V          BLB_V           0       cload
```

```
CBL_VI          BL_VI           0       cload
CBLB_VI         BLB_VI                  0       cload

CBL_VII         BL_VII          0       cload
CBLB_VII        BLB_VII                 0        cload

CBL_VIII        BL_VIII         0       cload
CBLB_VIII       BLB_VIII        0       cload

CBL_IX          BL_IX           0       cload
CBLB_IX         BLB_IX                  0       cload

CBL_X                   BL_X            0       cload
CBLB_X          BLB_X           0       cload

CBL_XI          BL_XI           0       cload
CBLB_XI         BLB_XI                  0       cload

CBL_XII         BL_XII          0       cload
CBLB_XII        BLB_XII         0       cload
************************************************************
*6T SRAM_1 netlist
M1_I Q_I QB_I 1 1               PMOS    W='w'         L='l'
M2_I Q_I QB_I 0 0               NMOS    W='1.2*w'  L='l'
M3_I QB_I Q_I 1 1               PMOS    W='w'         L='l'
M4_I QB_I Q_I 0 0               NMOS     W='1.2*w' L='l'
M5_I Q_I WL_P BL_I 0            NMOS    W='w'           L='l'
M6_I QB_I WL_P BLB_I 0          NMOS    W='w'           L='l'
************************************************************
************************************************************
*6T SRAM_2 netlist
M1_II   Q_II    QB_II   1               1               PMOS    W='w'         L='l'
```

```
M2_II   Q_II    QB_II   0               0               NMOS    W='1.2*w'  L='l'
M3_II   QB_II   Q_II    1               1               PMOS    W='w'      L='l'
M4_II   QB_II   Q_II    0               0               NMOS    W='1.2*w' L='l'
M5_II   Q_II    WL_P    BL_II   0               NMOS    W='w'           L='l'
M6_II   QB_II   WL_P    BLB_II  0               NMOS    W='w'           L='l'
**********************************************************
**************************************************************
*6T SRAM_3 netlist
M1_III Q_III QB_III 1 1             PMOS    W='w'       L='l'
M2_III Q_III QB_III 0 0             NMOS    W='1.2*w'  L='l'
M3_III QB_III Q_III 1 1             PMOS    W='w'       L='l'
M4_III QB_III Q_III 0 0             NMOS    W='1.2*w' L='l'
M5_III Q_III WL  BL_III 0               NMOS    W='w'           L='l'
M6_III QB_III WL  BLB_III 0     NMOS    W='w'           L='l'
**********************************************************
**************************************************************
*6T SRAM_4 netlist
M1_IV Q_IV QB_IV 1 1            PMOS    W='w'       L='l'
M2_IV Q_IV QB_IV 0 0            NMOS    W='1.2*w'  L='l'
M3_IV QB_IV Q_IV 1 1            PMOS    W='w'       L='l'
M4_IV QB_IV Q_IV 0 0            NMOS    W='1.2*w' L='l'
M5_IV Q_IV WL  BL_IV 0              NMOS    W='w'           L='l'
M6_IV QB_IV WL  BLB_IV 0    NMOS    W='w'           L='l'
**********************************************************
**************************************************************
*6T SRAM_5 netlist
M1_V Q_V QB_V 1 1               PMOS    W='w'       L='l'
M2_V Q_V QB_V 0 0               NMOS    W='1.2*w'  L='l'
M3_V QB_V Q_V 1 1               PMOS    W='w'       L='l'
M4_V QB_V Q_V 0 0               NMOS    W='1.2*w' L='l'
M5_V Q_V WL  BL_V 0         NMOS    W='w'           L='l'
M6_V QB_V WL  BLB_V 0       NMOS    W='w'           L='l'
```

```
**********************************************************
**************************************************************
*6T SRAM_6 netlist
M1_VI Q_VI QB_VI 1 1            PMOS    W='w'         L='l'
M2_VI Q_VI QB_VI 0 0            NMOS    W='1.2*w'  L='l'
M3_VI QB_VI Q_VI 1 1            PMOS    W='w'         L='l'
M4_VI QB_VI Q_VI 0 0            NMOS    W='1.2*w' L='l'
M5_VI Q_VI WL  BL_VI 0                  NMOS    W='w'           L='l'
M6_VI QB_VI WL  BLB_VI 0    NMOS    W='w'           L='l'
**********************************************************
************************************************************
*6T SRAM_7 netlist
M1_VII Q_VII QB_VII 1 1                 PMOS    W='w'         L='l'
M2_VII Q_VII QB_VII 0 0                 NMOS    W='1.2*w'  L='l'
M3_VII QB_VII Q_VII 1 1                 PMOS    W='w'         L='l'
M4_VII QB_VII Q_VII 0 0                 NMOS    W='1.2*w' L='l'
M5_VII Q_VII WL  BL_VII 0               NMOS    W='w'           L='l'
M6_VII QB_VII WL BLB_VII 0  NMOS    W='w'           L='l'
**********************************************************
************************************************************
*6T SRAM_8 netlist
M1_VIII Q_VIII QB_VIII 1 1              PMOS    W='w'         L='l'
M2_VIII Q_VIII QB_VIII 0 0              NMOS    W='1.2*w'  L='l'
M3_VIII QB_VIII Q_VIII 1 1              PMOS    W='w'         L='l'
M4_VIII QB_VIII Q_VIII 0 0              NMOS    W='1.2*w' L='l'
M5_VIII Q_VIII WL_P BL_VIII 0                   NMOS    W='w'           L='l'
M6_VIII QBVIII WL_P BLB_VIII 0      NMOS    W='w'           L='l'
**********************************************************
************************************************************
*6T SRAM_9 netlist
M1_IX Q_IX QB_IX 1 1            PMOS    W='w'         L='l'
M2_IX Q_IX QB_IX 0 0            NMOS    W='1.2*w'  L='l'
```

```
M3_IX QB_IX Q_IX 1 1            PMOS     W='w'        L='l'
M4_IX QB_IX Q_IX 0 0            NMOS     W='1.2*w' L='l'
M5_IX Q_IX WL  BL_IX 0                   NMOS     W='w'           L='l'
M6_IX QB_IX WL  BLB_IX 0     NMOS     W='w'           L='l'
*********************************************************
************************************************************
*6T SRAM_10 netlist
M1_X Q_X QB_X 1 1               PMOS    W='w'        L='l'
M2_X Q_X QB_X 0 0               NMOS    W='1.2*w'  L='l'
M3_X QB_X Q_X 1 1               PMOS     W='w'        L='l'
M4_X QB_X Q_X 0 0               NMOS     W='1.2*w' L='l'
M5_X Q_X WL  BL_X 0             NMOS     W='w'           L='l'
M6_X QB_X WL  BLB_X 0           NMOS     W='w'           L='l'
*********************************************************
************************************************************
*6T SRAM_11 netlist
M1_XI Q_XI QB_XI 1 1            PMOS    W='w'        L='l'
M2_XI Q_XI QB_XI 0 0            NMOS    W='1.2*w'  L='l'
M3_XI QB_XI Q_XI 1 1            PMOS     W='w'        L='l'
M4_XI QB_XI Q_XI 0 0            NMOS     W='1.2*w' L='l'
M5_XI Q_XI WL  BL_XI 0                   NMOS     W='w'           L='l'
M6_XI QB_XI WL  BLB_XI 0     NMOS     W='w'           L='l'
*********************************************************
************************************************************
*6T SRAM_12 netlist
M1_XII Q_XII QB_XII 1 1                  PMOS    W='w'        L='l'
M2_XII Q_XII QB_XII 0 0                  NMOS    W='1.2*w'  L='l'
M3_XII QB_XII Q_XII 1 1                  PMOS     W='w'        L='l'
M4_XII QB_XII Q_XII 0 0                  NMOS     W='1.2*w' L='l'
M5_XII Q_XII WL BL_XII 0                 NMOS     W='w'           L='l'
M6_XII QB_XII WL BLB_XII 0   NMOS     W='w'           L='l'
*********************************************************
```

*Generating Check Bits

*Synorome Generating Circuit

XA1 D_VII D_IX PA1 XORGate

XA2 D_V D_III PA2 XORGate

XA3 PA1 PA2 PA3 XORGate

XA4 PA3 D_XI PA XORGate

XB1 D_III D_VI PB1 XORGate

XB2 D_VII D_X PB2 XORGate

XB3 PB1 PB2 PB3 XORGate

XB4 PB3 D_XI PB XORGate

XC1 D_V D_VI PC1 XORGate

XC2 D_VII D_XII PC2 XORGate

XC3 PC1 PC2 PC XORGate

XD1 D_IX D_X PD1 XORGate

XD2 D_XI D_XII PD2 XORGate

XD3 PD1 PD2 PD XORGate

****

*Write Input Data and Check bits into SRAM CELLS

XI PA WE BL_I BLB_I WriteP

XII PB WE BL_II BLB_II WriteP

XIII D_III DB_III BL_III BLB_III Write

XIV PC WE BL_IV BLB_IV WriteP

XV D_V DB_V BL_V BLB_V Write

XVI D_VI DB_VI BL_VI BLB_VI Write

XVII D_VII DB_VII BL_VII BLB_VII Write

XVIII PD WriteP WE BL_VIII BLB_VIII

XIX D_IX DB_IX BL_IX BLB_IX Write

XX D_X DB_X BL_X BLB_X Write

XXI D_XI DB_XI BL_XI BLB_XI Write

XXII D_XII DB_XII BL_XII BLB_XII Write

****

```
*Read Data through Sense Amplfier

XSAI            BL_I            BLB_I           SAE     SON_I           SO_I    SA

XSAII           BL_II           BLB_II          SAE     SON_II          SO_II   SA
XSAIII          BL_III          BLB_III         SAE     SON_III                 SO_III  SA
XSAIV           BL_IV           BLB_IV          SAE     SON_IV                  SO_IV   SA

XSAV            BL_V            BLB_V           SAE     SON_V           SO_V    SA
XSAVI           BL_VI           BLB_VI          SAE     SON_VI                  SO_VI   SA
XSAVII          BL_VII          BLB_VII                 SAE     SON_VII         SO_VII  SA
XSAVIII                 BL_VIII                 BLB_VIII        SAE     SON_VIII
        SO_VIII SA
XSAIX           BL_IX           BLB_IX          SAE     SON_IX                  SO_IX   SA
XSAX            BL_X            BLB_X           SAE     SON_X           SO_X    SA
XSAXI           BL_XI           BLB_XI          SAE     SON_XI                  SO_XI   SA
XSAXII          BL_XII          BLB_XII                 SAE     SON_XII         SO_XII  SA
**********************************************************************

*Call Error Decting Circuit

*XRC    RE      REB     INV
*MRE_I          BL_I            RE              DE_I            1       PMOS W='4*w' L='l'
*MCE_I          BL_I            REB     DE_I            0       NMOS W='2*w' L='l'
*MRE_II                 BL_II           RE              DE_II           1       PMOS
W='4*w' L='l'
*MCE_II                 BL_II           REB     DE_II           0       NMOS W='2*w' L='l'
*MRE_III                BL_III          RE              DE_III          1       PMOS
W='4*w' L='l'
*MCE_III                BL_III          REB     DE_III          0       NMOS W='2*w' L='l'
*MRE_IV                 BL_IV           RE              DE_IV           1       PMOS
W='4*w' L='l'
*MCE_IV                 BL_IV           REB     DE_IV           0       NMOS W='2*w' L='l'
*MRE_V                  BL_V            RE              DE_V            1       PMOS
W='4*w' L='l'
*MCE_V                  BL_V            REB     DE_V            0       NMOS W='2*w' L='l'
*MRE_VI                 BL_VI           RE              DE_VI           1       PMOS
W='4*w' L='l'
*MCE_VI                 BL_VI           REB     DE_VI           0       NMOS W='2*w' L='l'
```

```
*MRE_VII        BL_VII          RE              DE_VII          1       PMOS W='4*w' L='l'
*MCE_VII        BL_VII          REB     DE_VII          0       NMOS W='2*w' L='l'
*MRE_VIII       BL_VIII                 RE              DE_VIII         1       PMOS
W='4*w' L='l'
*MCE_VIII       BL_VIII                 REB     DE_VIII         0       NMOS W='2*w' L='l'
*MRE_IX                 BL_IX           RE              DE_IX           1       PMOS
W='4*w' L='l'
*MCE_IX                 BL_IX           REB     DE_IX           0       NMOS W='2*w' L='l'
*MRE_X                  BL_X            RE              DE_X            1       PMOS
W='4*w' L='l'
*MCE_X                  BL_X            REB     DE_X            0       NMOS W='2*w' L='l'
*MRE_XI                 BL_XI           RE              DE_XI           1       PMOS
W='4*w' L='l'
*MCE_XI                 BL_XI           REB     DE_XI           0       NMOS W='2*w' L='l'
*MRE_XII        BL_XII          RE              DE_XII          1       PMOS W='4*w' L='l'
*MCE_XII        BL_XII          REB     DE_XII          0       NMOS W='2*w' L='l'
*XCretError             DE_III  DEB_III         INV
*Detecing Error Bit Position
*XEA0           DEB_III DE_V    PNA0    XORGate
*XEA1           DE_IX   DE_VII  PNA1    XORGate
*XEA2           DE_IX   DE_XI   PNA2    XORGate
*XEA3           PNA0    PNA1    PNA3    XORGate
*XEA4           PNA3    PNA2    PNA     XORGate
*XEB0           DEB_III DE_II   PNB0    XORGate
*XEB1           DE_VI   DE_VII  PNB1    XORGate
*XEB2           DE_X    DE_XI   PNB2    XORGate
*XEB3           PNB0    PNB1    PNB3    XORGate
*XEB4           PNB3    PNB2    PNB     XORGate
*XEC0           DE_IV   DE_V    PNC0    XORGate
*XEC1           DE_VI   DE_VII  PNC1    XORGate
*XEC2           PNC0    DNE_XII         PNC2    XORGate
*XEC3           PNC1    PNC2    PNC     XORGate
*XED0           DE_VIII DE_IX   PND0    XORGate
*XED1           DE_X    DE_XI   PND1    XORGate
```

```
*XED2           PND0    DE_XII PND2    XORGate
*XED3           PND1    PND2   PND     XORGate
*********************************************************************
XCretError              SON_III SONB_III            INV
*Detecing Error Bit Position
XEA0            SONB_III        SON_V           PNA0    XORGate
XEA1            SON_IX          SON_VII         PNA1    XORGate
XEA2            PNA0            SON_XI          PNA2    XORGate
XEA3            PNA2            PNA1            PNA     XORGate
XEB0            SONB_III        SON_VI          PNB0    XORGate
XEB1            SON_X           SON_VII         PNB1    XORGate
XEB2            PNB0            SON_XI          PNB2    XORGate
XEB3            PNB2            PNB1            PNB     XORGate
XEC0            SON_VI                  SON_V           PNC0    XORGate
XEC1            SON_XII         SON_VII         PNC1    XORGate
XEC2            PNC0            PCN1            PNC     XORGate
XED0            SON_X           SON_IX          PND0    XORGate
XED1            SON_XII         SON_XI          PND1    XORGate
XED2            PND0            PND1            PND     XORGate
*********************************************************************
.SUBCKT Write   D DB BL BLB
*Write Circuit
MRB     BLB     D       0       0       NMOS W='10*w' L='l'
MR      BL              DB      0       0       NMOS W='10*w' L='l'
.ENDS Write
*********************************************************************
.SUBCKT WriteP   D WE BL BLB
*Write Check Bits  Circuit
MW              BL              WE      Qw      0       NMOS W='10*w' L='l'
MWB             BLB     WE      Qwb     0       NMOS W='10*w' L='l'
MWR1            DB              D               1               1       PMOS W='2*w' L='l'
MWR2            DB              D               0               0       NMOS W='1*w' L='l'
```

```
MRB             Qwb     D               0               0       NMOS W='10*w' L='l'
MR              Qw      DB              0               0       NMOS W='10*w' L='l'
.ENDS WriteP
********************************************************************************
.SUBCKT XORGate   A  B          XOR
*XOR Gate NetLsit
MX1     2       A       1       1       PMOS    W='4*w'         L='l'
MX2     3       B       2       0       PMOS    W='4*w'         L='l'
MX3     3       B       A       1       NMOS    W='1*w'         L='l'
MX4     3       A       B       0       NMOS    W='1*w'         L='l'
MX21    XOR     3       1       1       PMOS    W='w'         L='l'
MX22    XOR     3       0       0       NMOS    W='0.5*w'         L='l'
.ENDS XORGate
********************************************************************************
.SUBCKT SA              BL              BLB     SAE     SON     SO
*Sense Amplfier
MQ7  SO         SAE     1               1       PMOS            W='2*w'         L='l'
MQ2  SO         SON     1               1       PMOS            W='2*w'         L='l'
MQ1  SO         SON     Y               0       NMOS            W='w'           L='l'
MQ8  SON        SAE     1               1               PMOS            W='2*w'         L='l'
MQ4  SON        SO              1               1       PMOS            W='2*w'         L='l'
MQ3  SON        SO              YN              0       NMOS            W='w'           L='l'
MQ5  Y          BL              COM     0       NMOS            W='2*w'         L='l'
MQ6  YN         BLB     COM     0       NMOS            W='2*w'         L='l'
MQ9  COM        SAE     0               0       NMOS            W='5*w'         L='l'
.ENDS SA
********************************************************************************
.SUBCKT INV    in          out1
*subcircuit of a  inverter
MI1 out1 in 1 1 PMOS W='4*w' L='l'
MI3 out1 in 0 0 NMOS W='2*w' L='l'
.ENDS INV
```

```
**************************************************************************
*.TRAN 0.1n 150n  *Sweep monte=1000
.TRAN 0.1n 150n  *Sweep  Vdd 0v 1v 0.2v
 .meas tran pow AVG power
*****************************Measure SyndromeTimeDelay*************
.MEASURE STDelay     trig v(D_V) VAL='0.5*vdd'  Rise=1
+                    targ v(PB)   VAL='0.5*vdd' Rise=1
*****************************Measure ErrorDectingTimeDelay**************
.MEASURE EDDelay    trig v(SAE) VAL='0.5*vdd'  Rise=1
+                    targ v(PND)   VAL='0.42*vdd' fall=1
*****************************Measure write time *************
.MEASURE WriteTime   trig v(WL_P) VAL='0.5*vdd'  Rise=1
+                    targ v(Q_II)   VAL='0.5*vdd'  Rise=1
.option  nopage nomod  post
```