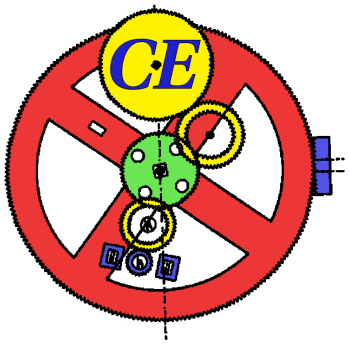


Computer Engineering
Mekelweg 4,
2628 CD Delft
The Netherlands
http://ce.et.tudelft.nl/

red**2008**

MSc THESIS

Experimental Analysis Of Design-For-Testability Techniques In SRAMs

ANNE NKENGAFEH FOMIN

CE-MS-2008-17

Abstract

Semiconductor memories most specifically SRAMs are becoming very popular in today's System-On-Chip(SOCs). As technology shrinks and the share of memories in these complex systems increases, memories become more susceptible to faults. One of the most important faults in SRAMs is Data Retention Fault (DRF). DRF is a fault which occurs as a results of an (partial) open in the V_{dd} path of the cell.This open defect causes the memory cell to be unable to keep its logic value after a certain time interval. Testing memories for these defects has therefore become a major issue to test engineers as they try to reduce the long test time. One of the ways of reducing this test time is by use of Design-For-Testability (DFT) techniques. DFTs are specialised additional hardware which are added to integrated circuits to make them testable. In DFT, the emphasis is not on the test itself but on the design. The question asked is always: "How can circuits in general or memories in particular be designed to make them testable"? Many DFT's have been proposed in the past for the detection of DRFs. These DFTs techniques are said to reduce this test time. Limited experimental data has been presented to proof the efficiency of these DFTs, thus qualifying them as better test methods over traditional functional test. In this Thesis a comparative study will be carried out to show the efficiency of different DFTs over functional test in the detection DRFs. A typical functional test (the Pause test) will be used as standard. To begin, an accurate simulation model is build. The SRAM model is designed using CMOS 90nm technology which is one of the most available process technology and implemented using HSPICE, the industrial version of SPICE (Simulation Program with Integrated Circuit Emphasis). The validated model is further used to implement three DFT techniques: Weak Write Test Mode (WWTM), No Write Recovery Test Mode (NWRTM) and PreDischarge Weak Test Mode (PDWTM). From the experimental results obtained we observed that all three DFTs presents great reduction in test time as compared to the functional test. In terms of test time reduction, PDWTM was a more preferable DFT to WWTM and NWRTM. WWTM presented the least fault coverage as it was not capable of detecting faults with resistance values of $\leq 15k\Omega$.

TUDelft

Faculty of Electrical Engineering, Mathematics and Computer Science

Experimental Analysis Of Design-For-Testability Techniques In SRAMs

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

by

ANNE NKENGAFEH FOMIN
born in BUEA, CAMEROON

Computer Engineering
Department of Electrical Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Experimental Analysis Of Design-For-Testability Techniques In SRAMs

by ANNE NKENGAFEH FOMIN

Abstract

Semiconductor memories most specifically SRAMs are becoming very popular in today's System-On-Chip(SOCs). As technology shrinks and the share of memories in these complex systems increases, memories become more susceptible to faults. One of the most important faults in SRAMs is Data Retention Faults (DRF). DRF is a fault which occurs as a results of an (partial) open in the V_{dd} path of the cell. This open defect causes the memory cell to be unable to keep its logic value after a certain time interval. Testing memories for these defects has therefore become a major issue to test engineers as they try to reduce the long test time. One of the ways of reducing this test time is by use of Design-For-Testability (DFT) techniques. DFTs are specialised additional hardware which are added to integrated circuits to make them testable. In DFT, the emphasis is not on the test itself but on the design. The question asked is always: "How can circuits in general or memories in particular be designed to make them testable"? Many DFT's have been proposed in the past for the detection of DRFs. These DFTs techniques are said to reduce this test time. Limited experimental data has been presented to proof the efficiency of these DFTs, thus qualifying them as better test methods over traditional functional test. In this Thesis a comparative study will be carried out to show the efficiency of different DFTs over functional test in the detection DRFs. A typical functional test (the Pause test) will be used as standard. To begin, an accurate simulation model is build. The SRAM model is designed using CMOS 90nm technology which is one of the most available process technology and implemented using HSPICE, the industrial version of SPICE (Simulation Program with Integrated Circuit Emphasis). The validated model is further used to implement three DFT techniques: Weak Write Test Mode (WWTM), No Write Recovery Test Mode (NWRTM) and PreDischarge Weak Test Mode (PDWTM). From the experimental results obtained we observed that all three DFTs presents great reduction in test time as compared to the functional test. In terms of test time reduction, PDWTM was a more preferable DFT to WWTM and NWRTM. WWTM presented the least fault coverage as it was not capable of detecting faults with resistance values of $\leq 15k\Omega$.

Laboratory : Computer Engineering
Codenumber : CE-MS-2008-17

Committee Members :

Advisor:	Dr.Ir. Said Hamdioui, CE, TU Delft
Advisor:	Nor Zaidi Haron, CE, TU Delft
Chairperson:	Dr. Koen Bertels, CE, TU Delft
Member:	Dr.Ir. Zaid Al-Ars, CE, TU Delft
Member:	Dr.Ir. Jaap Hoekstra, EE, TU Delft

Contents

List of Figures

List of Tables

Acknowledgements

Many persons have contributed to me getting to this stage of my Studies in TU Delft. I will like to mention some of those names. Firstly I will like to thank my Supervisor Dr. Said Hamdioui for the patience and time he exercised with me throughout this project. I am grateful to him for the much he has thought me. I also want to thank Nor Zaidi for his availability, and feedbacks whenever I needed help. My gratitude also goes to Dr. Zaid Al-Ars for his help and advice on the design of the SRAM simulation model, and to all the members of the CE group of the Delft university who in one way or the other helped me through my studies.

In addition to the above mentioned, I will like to give special gratitude to Ir. Egbert Bol for the encouragements he gave me especially in times of difficulties at TU Delft.

Most importantly I will like to thank my parents, family and friends who have stood by me through out my studies. To you all I owe credits for your support and encouragement.

To God almighty I give him all the glory for his unlimited blessings through out this period and my entire life.

ANNE NKENGAFEH FOMIN
Delft, The Netherlands
October 27, 2008

1 Introduction

Embedded memories are widely used in the realisation of todays complex systems known as SOCs. The forecast for 2013 from states that 90% of the area of SOCs will be made up of memories most specifically SRAMs[2][16]. This implies that test cost of memories will have a large impact on the test cost of the SOCs. For this reason adequate test methods have to be put in place in order to minimise the cost time while maintaining efficiency thereby increasing the quality of the product. This introductory Chapter starts out showing some memory basics and importance of testing them. Then it describes various memory faults and test techniques. Finally it discusses some basic concepts of Design-For-Testability (DFTs) and Data Retention Faults (DRFs) in memories which are the main focus of our research.

1.1 Memory Basics

A memory is a device, which is used in storing and retrieving information. This information presents a logic value of 1 and 0. Each logic value is stored in one memory element; the memory elements are together arranged in an array. The size of the memory array thus gives the capacity of the memory. Memories fall under the class of integrated circuits known as sequential circuits. These are integrated circuits whose output depends on the value of the input and its internal state.

Various types of memories can be distinguished and each of these memory type manifest different types of defects. This distinction is based on two factors. The first is the method used in storing information (volatility) and the second is based on the process technology used in manufacturing them. These two factors are explained next:

- Volatility: This classifies the memory on the basis of how it stores information. There exist two types which are:
  - **Non-Volatile Memories:** These are those type of memories that will keep their information even when the power supply is turned off. Memories that belong to this lass are called Read Only Memories(ROM). Some examples of ROMs are EPROMS, EEPROMS and Flash.
  - **Volatile Memories:** These memories will loose their stored information when the power supply is turned off. The most widely used volatile memories are Random Access Memories(RAMs). Two types of RAMs can be distinguished: Static Random Access Memories (SRAMs) and Dynamic Random Access Memories (DRAMs).
- **Process Technology:** There are three main types of process technology that are used in the manufacturing process of memories. They are:
  - **Complementary Metal - Oxide Semiconductor(CMOS) Process Technology:** This is the most widely used process technology. It consist of two types of Metal - Oxide Semiconductor Field Effect Transistors - MOSFETS; namely NMOS and PMOS. MOSFETs transistors are made up of three layers; Material, Oxide and Semiconductor. In processing of MOSFETs both the N-type and P-type materials are used. The type

of the MOSFET, that is NMOS or PMOS is determined by the typed of material used for the drain and source. The CMOS process is used in the fabrication of memories which require high density and low power consumption.

- **Bipolar:** This technology consists of a circuit made up of Bipolar junction transistors. These circuits are used in fast applications with low densities.
- **BiCMOS:** This process is a combination of the CMOS and the Bipolar process. It is used in circuits where a combination of both CMOS and Bipolar processes are needed such as radio frequency (RF) oscillators.

As the size of memories on chip increases it is important for new and faster test methods to be developed for testing them since they have a high impact on the overall yield and quality of today's SOCs. Tests used in the past are not suitable presently to test memories. This is because most of these memories are sometimes be deeply embedded on the chip and are hardly observable and controllable externally. The increasing complexity in testing memories has given rise to smarter test techniques. One of the most widely used techniques are DFTs which will be extensively covered in this thesis.

1.2 Types of Memory Faults

There are many types of memory faults which can be grouped under the following classes:

- **Hard Faults [29]:** These are memory faults whose sensitisation/detection is time independent. These faults are sensitised once the sensitisation operation is applied. Some examples of hard faults are, stuck-at faults and coupling faults.
- **Soft Faults [15]:** These are memory faults whose sensitisation does not only depend on memory operations but also depends on a time factor. For example a fault will be sensitised after a certain time period ΔT. Soft faults in SRAMs is modeled as DRFs [30].

1.3 Basic Concepts of DRFs and DFT in SRAMs

DRFs are faults which occur as a result of a leakage in the V_{dd} path of a memory cell array or in the memory peripheral circuits. A memory is said to have DRFs when its fails to retain its data after a certain time period ΔT. The wait period needed to detect this fault is usually not known since this value depends on the strength of the defect. For this reason, functional tests approach is not a suitable for testing these faults.

Test engineers have therefore resolved to the use of DFTs for the detection of DRFs. DFTs are techniques which use specialised circuits. These specialised circuits are added to the design of an integrated circuit to make it testable. Many of these DFTs have been proposed in the past. The efficiency of most these DFT techniques have not yet been evaluated.

In this thesis we will focus on investigating the challenges of testing DRFs. DFTs techniques will be evaluated to demonstrate their efficiency in resolving these challenges. At the end of the research the following issues should be resolved.

1. Determining the main causes of DRFs in the memories and the existing test challenges in DRFs testing.
2. Demonstrate the inefficiency of the traditional functional test by means of experimental results

3. Demonstrate the efficiency of DFTs in the detection of DRFs in SRAMs by implementing an evaluating and existing DFT technique.

To begin, we started off by carrying out a literature survey on existing DFTs for the detection of DRFs in SRAMs. This survey resulted in answering the first research question. To resolve the remaining two research questions, simulations needed to be performed on an SRAM simulation model. An accurate SRAM simulation model was thus designed and implemented using HSPICE. All experimental results obtained are based on the proposed SRAM simulation model. This thesis is thus organised as presented below.

1.4 Organisation

This thesis is divided into five chapters, a Conclusion and Appendices.

Chapter 1

Chapter 1 introduces the basics of semiconductor memories. Types of memory faults , an introductory concept of DFTs and DRFs will also be highlighted.

Chapter 2

This Chapter reviews exiting DFT techniques for DRFs in SRAMs. A brief description of each technique with its advantage and disadvantage will be presented. A comparison of these techniques is also presented at the end of the chapter.

Chapter 3

Chapter 3 describes the architecture of an SRAM memory, both functionally and electrically. All the peripheral circuits are presented stating the rule each of them play in carrying out Read and Write operations (Read/Write). The timing diagram for this two operations is also presented.

Chapter 4

This chapter presents an SRAM simulation model, implemented using HSPICE. Simulation results for a defect free memory are presented. The results presented are for two types of simulation; Operation simulation for normal read and write operations, and data retention simulation with an additional pause time to check the cell's ability to retain data.

Chapter 5

This chapter evaluates the efficiency of DFT over the traditional function pause test. The evaluation is based on ability of DFTs to detect both Symmetric and Asymmetric faults using a short pause time. Experimental results are presented after the evaluation.

Conclusion

The conclusion includes summary of the thesis and recommendations for future research.

Appendices

The Appendices consist of the following

- Alternative SRAM electrical model which could be used for SRAM simulation
- HSPICE input file for the SRAM model.

DFT Techniques For The Detection of DRFs in Semiconductor Memories

2

DFT techniques are gaining much popularity in the testing of embedded systems in general and embedded memories in particular. DFTs are most applicable in the test of faults whose detection is time dependent such as is the case of DRFs. DFTs usually involve some design modification which should not be detrimental to the original design and performance of the circuit. In this chapter the state-of-the-art review on existing DFT techniques for the detection of DRFs in SRAMs will be presented. For a basic understanding of the targeted defect, Section 2.1 is dedicated to the concept and the importance of DRFs in SRAMs. In Section 2.2 a classification of test solutions which are used in the detection of DRFs are outlined. In Section 2.3, existing DFT techniques for DRFs in SRAMs are explained, stating their advantages and disadvantages. In Section 2.4, a comparison of the presented DFT techniques is given followed by a conclusion summarising the chapter.

2.1 Concept and Importance of DRFs

DRFs are faults that occur as a result of a leakage in the V_{dd}path of memory cell array usually known as a weak cell. It is defined as a fault that occurs when a memory cell loses its stored information after a certain time, ΔT during which it has not been active [1][3]. Their occurrences are time dependent. For example a cell with logic 1(0) is supposed to hold its value as long as it is connected to supply voltage. However in the presence of a DRF, the cell flips after say 100ms from 1(0) to 0(1). DRFs can be caused by opens in the V_{dd} (Note that opens in the Ground path results

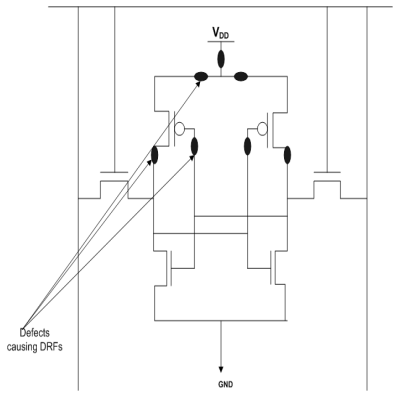


Figure 2.1: Cell showing possible Opens causing DRFs [21]

in Destructive Read Faults) due to fabrication defects such as missing connections to power supply terminal. The highlighted portions in Figure 2.1 represents those parts of the memory cell where opens can result in DRFs. The time period after which the cell flips its value can be modeled as ΔT as shown in Figure 2.2. The value of ΔT depends on the node capacitance and current leakage in the V_{dd} path. The amount of current leakage depends on the strength of the defect. The defect can result from a partially filled Via or a loose contact. As the defect resistance increases the value of ΔT decreases, thus a larger value of ΔT for small resistive defects (Rdefects), as illustrated

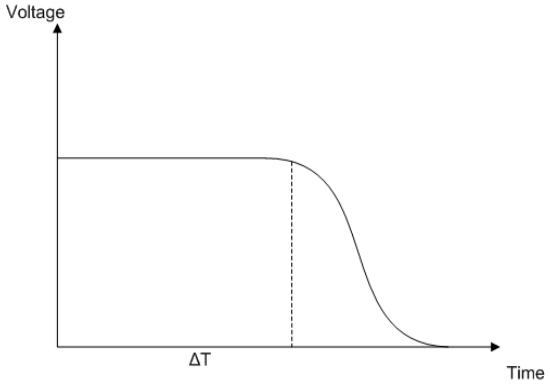


Figure 2.2: Cell flips due to leakage after time delay ΔT [21]

in Figure 2.3. The resistivity of these defects and the time the cell needs to flip its value is of

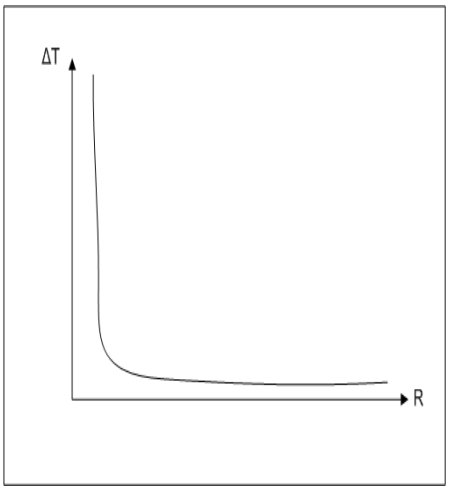


Figure 2.3: Flipping Time as a Function of R defect [21]

utmost importance to test engineers, who try as much as possible to get good quality devices at low cost using limited time period. There are seven possible electrical opens per cell that can cause DRFs. This might not be a realistic assumption since the distribution of opens depends on the cell layout. However, assuming one possible critical contact per cell, the probability of a critical contact is therefore given approximately as 10^{-9} [21], implying Test Escapes = 1M x 1 x 10^{-9} = 1000Defects Per Million (DPM). The occurrence probability of DRFs in embedded memories, has however been barely published. According to Philips , for a 64kbit memory, using $4\mu m$ technology the occurrence probability of DRFs is 2.27% [4]. For Intel's 32Kbits memory, using $0.18\mu m$ technology the occurence probability of DRFs is 0.59% [18]. Altough no adequate data is available to prove whether DRFs will be a challenge in future technology, they are probable to arise due to the following reasons:

- It is difficult to characterise the contact resistance of pull-ups.
- There is an increase number of missing Via/Contacts with the scaling down of technology.
- Failures are most likely to occur in Opens rather than Bridges/Shorts.

Obviously, the above mentioned challenges indicate the importance of DRFs in today's and future technologies.

2.2 Classification of Test Solutions for DRFs in Memories

There are two types of test solutions for detecting DRFs in memories. They are:

- **Functional Test:** These involve writing a background value to the memory cell, pausing for a given time and reading the cell. March test is suitable for functional test[3]
- **Structural Test:** These tests are implemented on-chip with the aid of specialised circuits, which can either be a Built-in-Self-Test (BIST) or a DFT circuit.

These two test solutions are explained below.

2.2.1 Functional Test Solutions

These are tests used to determine if the internal logic of a memory cell behaves as expected. Many functional test solutions for the detection of DRFs have been proposed. These functional tests are however not very efficient since they are time consuming. The following are some of the functional tests techniques:

- **Pause [4]:** The Pause test involves writing a background pattern in the memory cell usually using a checkerboard test [3] and pausing for a period of time say 100ms, then reading the memory cell. The following March test(tests that consist of a finite state of March elements, where March elements are finite sequence of operations applied to a memory cell [3]) is used for detecting faults [21].
  $\{\Updownarrow (w0)$; pause; $\Updownarrow (r0)$; $\Updownarrow (w1)$; pause; $\Updownarrow (r0)\}$. A background value of 0 is written to the entire memory cell array. a delay period of say 100ms is added followed by a normal read operation. The same procedure is performed for a write 1 and read 1.
- **Voltage Bump [20]:** This test proposes a method whereby a background pattern is written in the memory cell and two voltage patterns, a high and low voltage, are applied to read the memory cell. The following March test [21] can be used for detection of DRF.
  $\{\Updownarrow (w0)$; lower V_{dd}; $\Updownarrow (rx)$; Nominal V_{dd}; $\Updownarrow (r0)$; (w1);l ower V_{dd}; $\Updownarrow (rx)$; Nominal V_{dd}; $\Updownarrow (r1)\}$
- **Destructive Read [6]:** This test involves performing multiple reads to a particular memory cell. It is also known as *hammer* test.

In addition to the above mentioned functional tests methods, current measurement I_{ddq} can also be used in the detection of DRFs [7][17][19]. With this method the current drawn by the circuit after a read or write operation is monitored. With these measurements it is possible to determine any change in the V_{dd} which can be as a result of DRF.

2.2.2 Structural Test Solutions

These are, as oppose to functional tests, tests solutions which are implemented on chip by specialised circuitry added to the memory cell array. There exist two types of structural test methods. These are: Built-in-Self Test (BIST) and Design-for-Testability (DFT). These methods will be explained next.

- **Built-in-Self-Test (BIST):** These are tests which are implemented on chip. In this method all the functions required to perform the tests are contained on chip [3][22]. During test mode the BIST architecture enables the chip to test itself.
- **Design-for-Testability (DFT):** Using this method, parts of the test, usually those that have the largest reduction in test time, are implemented on chip. This enables the inner loop of a test algorithm, which might take a longer time, to be implemented by the DFT circuit. The other parts of the algorithms are implemented traditionally, by externally providing certain control and test data and /or observing certain response data [3].

In this chapter the attention is focused on DFT techniques used to detect DRFs in SRAMs

2.3 DFT Techniques for Detection of DRFs in SRAMs

The following DFT techniques are presented in this Section:

- **Weak Write Test Mode (WWTM) :** A Weak Ram Write (WRW) circuit is attached to the bitlines of the memory cell array and is used to weakly overwrite weak or unstable cells after a normal write operation [5].
- **Force Voltage Technique (FVT) :** Using the bitlines the memory cell is written with intermediate voltages. This enables the detection of faulty cells during the memorizing stage by measuring the I_{DDQ} [9].
- **Word Line Pulsing Technique (WLPT) :** This method entails gradually discharging the cell under test (CUT) with a pull-down of a reference cell, within the same column and comparing the logic values of the CUT after several pulses with that of the reference cell [10].
- **PreDischarge Write Test Mode (PDWTM) :** Memory cells are pre-discharged using a pre-discharge circuit to a floating ground (GND) and then overwritten. Only good cells are overwritten thereby easing the detection of weak cells [11].
- **No Write Recovery Test Mode (NWRTM) :** This technique is used to detect open defects in SRAMs. A special write cycle is used to distinguish bad cells from good cells. This special write cycle is carried out immediately after the normal write operation by driving the bitlines to opposite voltage, a weak and strong GND respectively [12].
- **Bitline Sensing (BLS) :** Set bitlines to a certain intermediate voltage, predefined and zero volts and measure voltage change on bitlines [13].

The succeeding sections explains in detail the above mentioned DFT techniques.

2.3.1 Weak Write Test Mode (WWTM)

In this DFT [5] a circuit known as the Weak Ram Write (WRW) circuit is used to weakly overwrite unstable cells in the presence of a defect. The Circuit as depicted on Figure 2.4 below involves the addition of 6 transistors to the bitlines of a memory cell. These transistors as shown on Figure 2.4 are designed as follows:

- 1 PMOS device acting as pull-up. The size of this transistor has to be carefully chosen so as to avoid overwriting of good cell. The ratio of the width / length of this device is at least 4:1 (where 4 = width and 1 = length). The gate of this device is connected to ground.

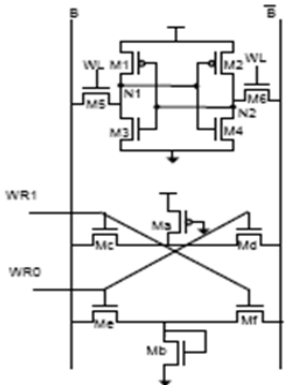


Figure 2.4: Conceptual Diagram of WWTM [5]

- 1 NMOS device acting as pull-down. The size of this device is at most 4:1. This is because larger sizes of this device increase the probability of weak cells to be overwritten.
- 4 NMOS pass deviced which connect the pull-up and pull-down devices to the bitlines. The sizes of this devices are similar to that of pass transistors of the cell array.

In addition to the above mentioned the following must be taken into consideration when designing the WRW circuit.

- The circuit should not be able of overwriting good cells. This is achieved by setting a margin which must be kept during the extra write cycle.
- The circuit should be able to detect open or weak connections in PMOS devices.
- The circuit should be able to detect weak PMOS.

The WRW circuit is controlled by two control signals *wro, wr1* connected to the NMOS pass devices. Using the DFT circuit on Figure 2.4 the test is performed as follows:

- A background of 1 or 0 is written to the memory using the appropriate write operation.
- The WRW circuit is enabled.
- Depending on the initial value in the cell the appropriate control signal is enabled. For example if the background data value was a 0 then the *wr1* to perform a long *wr1* operation of about 50ns.
- Disable the WRW circuit.
- A read operation is performed to determine if faulty cells were overwritten.

The above procedure is repeated with the opposite background value written to the memory. The timing diagram for the WWTM is presented in Figure 2.5. The extra write cycle *wr0* and *wr1* ensures the detection of faulty circuits. The timing diagram indicates how the test is carried out. The *nwr0* and *nwr1* operations are performed consecutively with the first one being the long weak write. This is because two types of defects results in a DRF. These defects known as symmetric and asymmetric defects results from opens at the source or drain of the PMOS pull-up device. In the case of a defective source both operations are needed to detect the defect. But in the case

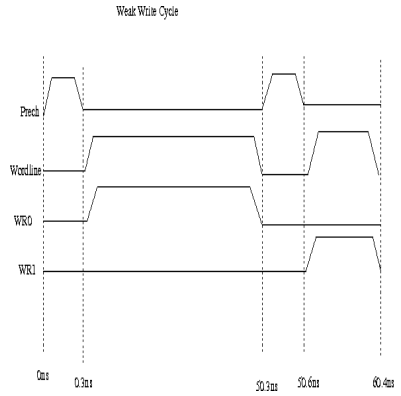


Figure 2.5: Timing Diagram of WWTM [5]

of a defective drain only one of the operations are required depending on the initial data value present in the cell.

The WWTM DFT technique is cost effective since the additional test circuitry has no effect on the original 6-transistor SRAM structure. This technique is an active defect oriented tests method. This is because it targets only cells which are defective and does not affect good cells. This test method is non intrusive [21] and can replace the pause test. However, this method requires a long write time, 50ns.

2.3.2 Forced-Voltage Technique(FVT)

In this technique the memory cell is written with intermediate voltages using the data buses BIT and $\overline{BIT}$ before the memorizing phase. During the memorizing phase fault free memories evolves to a stable quiescent state which has well defined logic levels while faulty cells evolves to a state with intermediate voltages. Thus causing the appearance of an intermediate voltage at the gate inverter of the memory cell which intern increases the quiescent current consumption. This enables easy detection of DRFs by sensing the current increase. The testability regions of the memory cell are obtained using state diagrams analysis [23]. The DFT circuit used for the FVT is depicted in Figure 2.6. The DFT circuit consist of four NMOS transistors devices, with two each

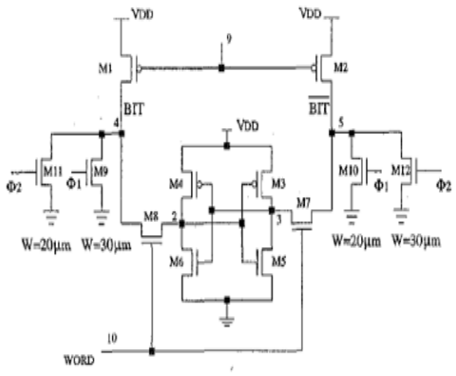


Figure 2.6: DFT circuit for FVT [9]

connected to BIT and $\overline{BIT}$ respectively. These transistors are controlled by two control signals $\Phi 1$ and $\Phi 2$. These signals are activated independently depending on what open defect is being

targeted. That is for an open defect at M_4 and M_5, $\Phi 2$ is activated before the access transistors are enabled. For an open defect at $M_3 and M_6, \Phi 1$ is activated. The test is carried out as follows:

- The pass transistors of the cell under test are enabled by turning on the wordlines.
- An initial condition is written to memory that is logic 1 or 0.
- The wordline is turn low thus disabling the pass transistors.
- Finally the quiescent current consumption is measured.

Critical sizing of the transistors of the DFT circuit are needed. This is to minimise area while also ensuring that the right voltage level is transmitted to the bitlines. The width of transistors M_9 and M_{12} should be the same as that of the precharge circuits. The sizes of M_{10} and M_{11} are designed to be in a ratio of at most 3:1 to that of the precharge circuits. These sizes thus ensures minimum area and reduce the amount of I_{ddq} consumption. Quiescent current measurement is performed twice once after every deactivation of the pass transistors.

The FVT DFT circuit adds four transistor columns to the normal six-transistor memory cell. Minor design modifications are needed but these modifications do not affect the performance of the memory. This test is however applicable only for asymmetric defects which can result to DRFs. Because this techniques involves two current measurement, there is a possibility of intermediate voltages undergoing drifts due to charge leakage. Thus good cells could also failing the test. Extra capacitance is also needed for the bitlines. Due to this extra capacitance the read and write cycles might turn to be slower.

2.3.3 Wordline Pulsing Technique (WLPT)

This technique requires two cells in a given column during the testing, connected to the same bitlines. One cell known as the reference cell and the other known as the cell under test (CUT). The node resistance of the CUT is gradually discharged through a pull-down of the reference cell. The test is performed as follows:

- A chosen background is written into the reference cell.
- The opposite of this written background is written into the CUT.
- Both bitlines are precharged and the wordline of the reference cell is enabled n times.
- The wordline of the CUT is enabled.
- A normal read operation is then performed on the CUT to determine if the CUT has flipped its value.
- Steps 1-5 are repeated by inverting the value written in the reference cell and the CUT. This is to enable the detection of asymmetric defects.

The discharge rate, which is the rate at which the bitlines are being discharged by applying different pulses to the wordline is given as follows:

$$\Delta V_{BL} = \frac{I_{read} * t_{WL-REF-pw}}{C_{BL}} \tag{2.1}$$

where ΔV_{BL} represents the discharge of the bitline at every time the wordline of the reference cell (WL_{REF}) is enabled. WL_{REF} and $t_{WL-REF-pw}$ represents the wordline and pulse width of the reference cell respectively . C_{BL} is the bitline capacitance. In Figure 2.7 the waveforms of

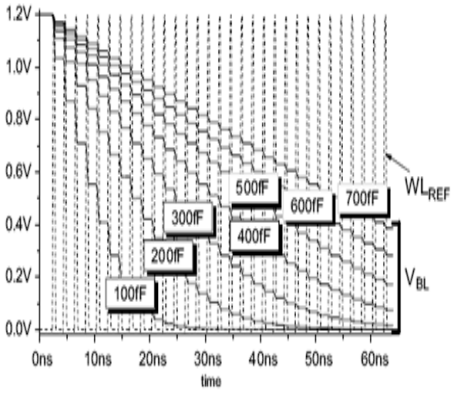


Figure 2.7: Waveform for wordline pulses of the reference cell after application of several bitline capacitances.

a step-wise discharge of V_{BL} after 32 WL_{REF} pulses is presented. It should be noted that from the above Figure that, the bitline discharge rate is slower for the more capacitive bitlines. This is because $\Delta\ V_{BL}$ is inversely proportional to C_{BL}.

When the wordline of the reference is enabled n times the voltage at the bitlines become unstable. The voltage transmitted from the bitlines to the PMOS device in case of a defective PMOS is not enough to compensate the leakage at the NMOS thus the cell flips is content. For fault free cells the voltage at he PMOs device remains stable enough and compensates for the leakage of the NMOS device when the wordline is turned on.

With this DFT techniques it is possible to perform test in parallel on a wordline by wordline basis by selecting one reference cell and one cell as CUT if these two cells share the same bitlines. Although this technique, requires the modification of the wordline, it has reduction in test time and does not involve high temperature requirements. It also ensures high level of fault coverage for DRFs especially as there is a possibility of increasing the pulse during test given that the pulse depends on the C_{BL}. Over killing could also occur if the write stress is too strong.

2.3.4 PreDischarge Write Test Mode Technique (PDWTM)

The goal of this technique is to achieve at speed testing of SRAMs. The PDWTM technique uses a special write operation known as PreDischarge Write (PDW) which aims at predischarging the bitline or bitlines to low. The DFT circuit used is presented in Figure 2.8. This DFT circuit

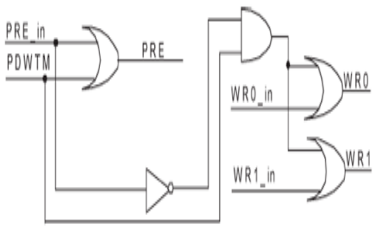


Figure 2.8: Memory control circuit modification for PDWMarch 9N [11]

presents a the modification which is implemented on the the precharge circuit of the memory. The circuit functions as follows. The PRE_{in} is generated after the read operation which triggers the precharge cycle. During test mode PDWTM is set to 1 and PRE_{in} is disabled using the inverter. These two signals now triggers the two controlling signals $wr1$ and $wr0$ which are used to mentained the low voltage level on the bitlines during the subsequent write operation. Using the above DFT circuit the test is carried out as follows:

- Initialise memory with particular value say 0(1).
- Predischarge bitlines to ground (GND) and perform a write 1(0) operation.
- Precharge bitlines to high and perform a read operation.

Two NMOS devices are used are use to act as the write control devices during the implementation of the PDWTM. The sizes of these NMOS device transistors should be the same as that of the precharge circuit. These two signals controlling these two NMOS devices are turned on high after the respective read operation. That is depending on the value to be written on the cell. For example to perform a write 0 operation $wr1$ is turned on high and $wr0$ is low. These signals are maintain low during the subsequent read operation. The purpose of the predischarge operation is to drive the bitlines low before the write operation. This DFT technique can be adapted to every March test. In the implementation with the March test all pause operations added to test for DRFs are disabled and replaced predischarge operation. An example of PDW implemented using with March 9N test is given as follows:
$\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w1, w0, r0)\}$
$\{\Updownarrow (w0); \Uparrow (r0, preD, w1); \Downarrow (r1, w1, preD, w0, r0)\}$

The cell modification needed for the implementation of the PDW circuit involves five extra logic gates added to the memory cell. This causes extra cell delay for write drivers and precharge circuits. This DFT also requires and additional control signal to the timing generation circuit used providing PWD_{In} signal. In this technique, predischarging is done on both bitlines. This entails a lot of power dissipation which is not necessary as far fault coverage is concern. It is preferable if predischarge could be done on one bitline at a time based on the data value in the cell. This might however, cause the realisation of the DFT circuitry to be more complicated.

2.3.5 No Write Recovery Test Mode Technique (NWRTM)

This DFT method aims at detecting all open defects which cannot be detected by typical March test. In this technique a special write operation is carried out to separate faulty cells from good cells. The implementation of this technique is based on Figure 2.9. This Figure presents an SRAM simulation model used in the implementation of the NWRTM technique. In the above model, two write control signals $nwr0$ and $nwr1$ are use to perform a no-write recovery to the bitlines after a read operation. This is achieved by turning on the appropriate signal high depending on the initial value in the cell. For example assuming a *W1* operation, *nwr0* operation is performed. The test is carried out as follows:

- Write a background value to the memory 0(1).
- Enable NWRTM
- Bitlines *BL* and *BLb* are set to a weak GND and strong GND respectively. The weak GND is a voltage level driven to GND but not driven by any source. This is achieved by activating the wordlines for a certain period of time and then turning them low. After the deactivation

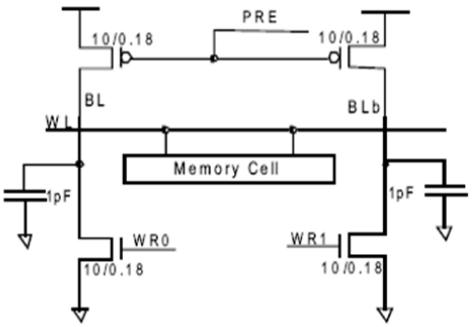


Figure 2.9: SRAM Simulation model for NWRTM Implementation [12]

of the wordlines no write recovery is performed. That implies the bitlines are not precharged back to high.

- Begin a normal write 1(0) cycle.
- Read 1: good cells flip the values while bad cells do not since the voltage in node A never exceeds that in node B.

It is easy to merge this technique with any typical March test algorithm. This is done using two steps. To begin, all test set specially designed to detect open defects/data retention faults are removed. This is followed by the addition of a No Write Recovery (NWR) circuit before the write cycle. The March 9N test algorithm with data retention test is used to illustrate the NWRTM [12]. This technique does not have any effect on performance of the memory and achieves reduced test time but requires some design and area efforts, due to the modification on the column and data input-output (I/O). This addition is used to disable the write recovery operation. A sensor for the measurement of I_{ddq} is also needed.

2.3.6 Bitline Sensing Technique

Bitline sensing techniques use specialised DFT circuits to measure the change in voltage of the cell. The memory cell is divided into two, the left and the right column, and test is carried out on each side of the cell. Each portion of the memory is driven by two bitlines BIT and $\overline{BIT}$. BIT is precharged to 0volts while $\overline{BIT}$ is assigned a predefined voltage. Then the precharged transistors are disabled and a read operation is performed on that memory cell of that column. The algorithm used in this DFT technique is as follows:

- A background 0 is written to all cells in a particular chosen column.
- BIT is precharged to 0 while $\overline{BIT}$ is precharged to a predefined voltage.
- The first cell of that column is read. Step 2 and 3 are repeated for all cells in the column.

The DFT reading circuit used is presented in Figure 2.10 below. During test mode, the NMOS transistor whose gate is connected to $\Phi 1 + \Phi 2$ is used to enable the DFT reading circuit. The signal which is used in the activation determines which cells are targeted. That is for an open defects on the left side, $\Phi 1$ is activated while $\Phi 2$ activates the NMOS device when the defect is on the right side. The sizes of these transistors must be in a ratio of 4:1 (ratio of width to length).

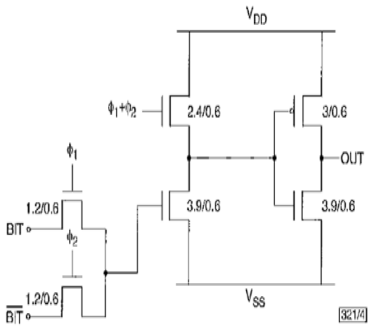


Figure 2.10: DFT Reading Circuit

The 2 NMOS transistors connected to the bitlines and controlled by Φ1 and Φ2 respectively are used to transmit the respective voltages to the bitlines. The sizes of these transistors are chosen such that they can transmit efficiently the voltage on the bitlines. To minimise areas and reduce the load capacitance of the bitlines sizes of width to length ratio of 2:1 can be used. When for example Φ1 signal is activated the voltage at *BIT* is transmitted to the NMOS device connected in series to the NMOS which activates the DFT circuit. This voltage is then transmitted through the inverter to the output.

For a faulty PMOS device at the true node of the cell the output of the inverter will have a logic value of 0 while a fault free cell will have a logic 1. This is because the voltage of *BIT* 0 . This voltage turns on the PMOS device of the inverter which is connected to V_{dd} thus giving us an output of 1. If the device is faulty the voltage transmitted will be 1 thus turning on the NMOS device which is connected to ground, hence the output of the inverter is 0. Sizes of 6:1 for the NMOS transistor device and 5:1 for the PMOS device of the inverter are appropriate when designing this DFT reading circuit.

This DFT technique has no impact on the performance of the memory.The amount of capacitance added by the DFT to the bitlines is not significant thus the normal read and write cycles are not affected. This method has a drawback in area overhead since additional pins are required.

For DRAMs memories DFRs are not yet over emphasised on since the cells are refreshed regularly after a given period of time in the range of 100μs to 100ms [3]. However sleeping sickness which manifests the same as data retention faults in SRAMs occurs in DRAMs. This fault occurs when a memory cell has not been accessed for some time. Where the leakage current becomes high. The charge contained reaches a high level and causes the read path of the refresh operation to produce the wrong value. This results in a wrong value being written back thus the content of the cell changes state. The checker board test is used for detecting this type of faults. These faults might however take a while to develop, since sometimes the cell might start leaking just before a refresh cycle, which thus brings back the state of the memory cell. Most researchers nowadays focus on power consumption reduction during data retention in DRAMs. One of these techniques is proposed in [14].

Method	**Hardware Modification**	**Area**	**Performance**
WWTM	Additional Weak Ram Write (WRW) to every block of memory	Slightly larger row as compared to a normal SRAM cell	No impact on performance
FVT	Minor design modification on the six transistor memory cell	No area penalty	No impact on performance
WLPT	requires modification of the wordline	Negligible	Over killing of good cells if stress is high
PDWTM	Negligible	Negligible	No impact on performance
NWRTM	Negligible since only the global logic is modified	Negligible	No performance penalty
BLS	Negligible	Negligible	Insignificant impact on performance

Table 2.1: Comparison of DFT presented techniques

2.4 Comparison of DFTs Techniques

In this section a comparison of the various DFT techniques presented in section four is given. The basis of comparison will be in terms of hardware modification, performance, area overhead and times overhead. From the comparison in table 2.1, we can say that all DFT techniques presented give a relative reduction in test time. These techniques also add negligible area to the size of the memory and have no impact on the memory performance. However, some hardware modifications are involved with some of the presented techniques, such as in the case of the WWTM, FVT, WLPT and NWRTM. Moreover, these modification do not require much design effort.

Summary

In this chapter a review of DFT techniques for detection of DRFs has been discussed. These techniques present no significant impact on the performance of the memory although most involves some design effort. Most of these techniques have not yet been simulated for silicon validation [21]. From the review presented it is not still clear as to which of type of opens in the cell will result in DRFs. It is also not mentioned the average pause time needed for the detection of DRFs functionally is. These techniques were based on the memory cell array. The peripheral circuits which could manifest faulty behaviors that results in a DRF have not been investigated.

Semiconductor Memory Architecture

3

Designing a simulation model of a circuit entails understanding the architecture of that circuit. For better understanding of the electrical model of an SRAM cell, Section 3.1 of this chapter will be dedicated to a brief description of the basic SRAM functional model and external parameters. In Sections 3.2, 3.3, 3.4 and 3.5, the subsystems of an SRAM circuit will be presented. These subsystems are composed of the following:

- *Memory Cell Array; this is the heart of an SRAM where the data is stored.*
- *Read Path Circuitry; the route to transport the data from the cell array to the data output latch.*
- *Write Path Circuitry; the route to transport data from the data input latch to the cell array.*
- *Address Decoder Circuitry; made up of a column and row decoder and it is used to activate the appropriate wordline and bitline pair during a read or write operation.*

It should be noted that the read and write paths will be treated separately; this is because these operations drive the memory to two different states. The write operation involves pulling down the respective bitline to low voltage while the read involves sensing the voltage difference between two previously precharged bitlines.

All operations in SRAMs are controlled by clock signals. In addition to the above-mentioned subsystems the control and timing circuitry will be presented in Section 3.6.

3.1 SRAM Functional Model

The black-box model of an SRAM cell is based on the specification of the system with no assumptions made on the internal structure of the system. Figure 3.1 presents the black-box representation of an SRAM circuit with its input and output pins. There are three inputs pins; namely,

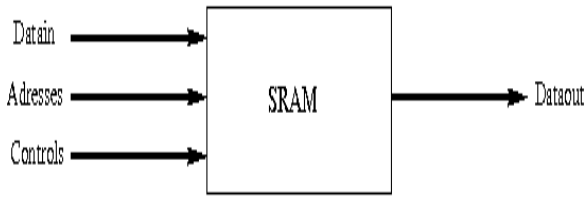


Figure 3.1: Black-Box Representation of SRAM

Datain, Address and Controls. While Dataout is the output pin. This indicates that for a memory to carry out one of its two basic operations i.e read and write, it needs the following:

- Control signals enabling the start of an operation
- Address indicating where data is stored in the case of the read operation or where the data will be written in the case of a write operation

- Datain pin used to feed in data to the memory
- Dataout pin used to feed out data from memory.

Figure 3.1 can be further divided into functional blocks consisting of subsystems of an SRAM. As

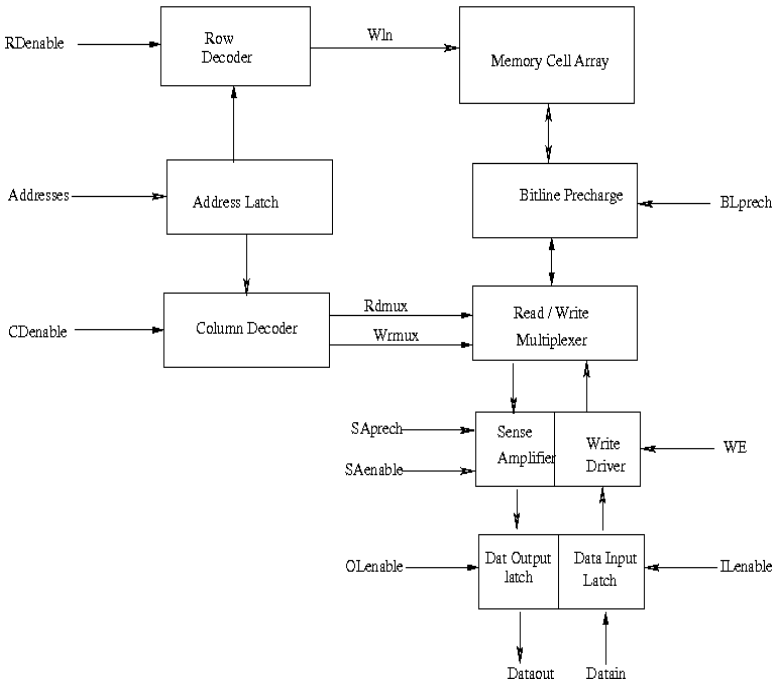


Figure 3.2: Functional Model of SRAM

indicated from Figure 3.2, the heart of an SRAM is the *cell array* where the data is stored. The memory cell array usually consists of thousand of memory cells depending on the number of bits it can store (word size). These memory cells are accessed using n-bit addresses, where n is the sum of the rows address (r) and columns address (c) which are stored in the address latch. During accessing of the memory the row decoder uses a number r addresses to select the appropriate row while the column decoder uses the remaining c addresses to select the appropriate column. The read multiplexer is used to connect the bitlines to the datalines during the read operation. The write multiplexer are used to connect the write driver to the bitlines thus placing the data on the bitlines during the write operation. The sense amplifier is used to sense and amplifier voltage difference between the two bitlines during the read operation The output/input latches are used to feed out out read data and in write data during the read and writ operation respectively.

By using Figures 3.1 and 3.2, the write and read operations of an SRAM circuit can be described. During a write operation, data is supplied from the data input latch. This data is fed into the write driver. The row and column decoders select the desired path for the data. The write driver now forces the desired bitlines to the appropriate voltage level. Consequently forcing the cell connected to that bitline to flip and change its value (state).

Read operation is carried out in a similar manner. During reading, the appropriate address to be read is set by the address decoder (column and row decoders). The data in the cell is put on the bitline and fed through the bitline multiplexer to the sense amplifier. The sense amplifier amplifies the data level to the correct voltage and the right data value is sent to the data output latch.

External Parameters

The external parameters of the SRAM are the specifications setup by manufacturers for customers to conform when using the SRAM. These parameters are divided into three categories. They are:

- **DC Parameters (regarding direct current value for input/output signals)**: These are parameters which are time-independent, and are used to specify the input and output signals of the memory pins.
- **AC Parameters**: These parameters are used for timing of the input and output signals of the memory. Timing is necessary because there must be adequate synchronization between the time data is present and the time data is needed at the dataout pin of the SRAM circuit. The SRAM circuit is made up of transistors which have some capacitive effects. These capacitive effects might result in some delay in the operation of the circuit. AC parameters have to be satisfied for a particular memory operation. They can be further divided into four types known as [24]:
  - Clock Parameters: These parameter, gives requirements for the general clock signal such as the minimal clock frequency.
  - Output Parameters: These parameters are used to setup the minimal and maximal time needed to setup an input signal until the correct output signal is reached.
  - Setup Parameters: These parameters are used to specify the minimal setup time of the different input signals with respect to the general clock pulse.
  - Hold Time Parameters: Specifies the minimal hold time of all input signals when the clock pulse has been activated.

In the following sections the subsystems of the SRAM circuit will be described

3.2 Memory Cell Array

There are several types of SRAM cells. The most commonly used SRAM cell is the CMOS six-transistor SRAM because of its good stability properties and its low power dissipation at steady state. This cell structure, which is illustrated in Figure 3.3 consist of two pull-up PMOS transistors, two NMOS pull-down transistors and two NMOS pass transistors. The true and false nodes of the cell are connected to the bitlines using the NMOS pass transistors. This connection is established by setting the *Wln* to a high level. The symmetrical structure of this device needs particular care during the sizing of the transistors so as to minimise area.

The memory cell array is a bistable circuit which is capable of being driven into one of two states, either 1 and 0. A normal functioning memory cell should be able to retain its state after the driven stimuli is removed.

3.3 Read Path Circuitry

The block diagram of the circuits that involved during read operation is given in Figure 3.4. The control signals driving each of these circuits is represented with arrows direction indicates the flow of the control signal. Each memory cell is connected to a pair of bitlines BL and $\overline{BL}$ and a wln . The read cycle is divided into three phases. These are:

- **Precharge Phase:** As indicated on Figure 3.5 this phase commences the read cycle . During this phase the bitline precharge signal, *BLprech*, is set high, followed by setting *SAprech* to

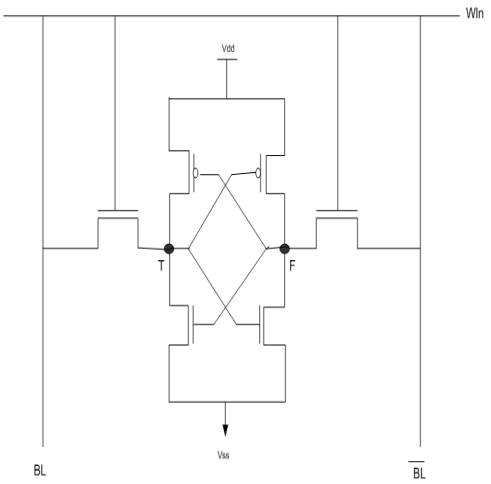


Figure 3.3: The Six-Transistor Memory Cell array

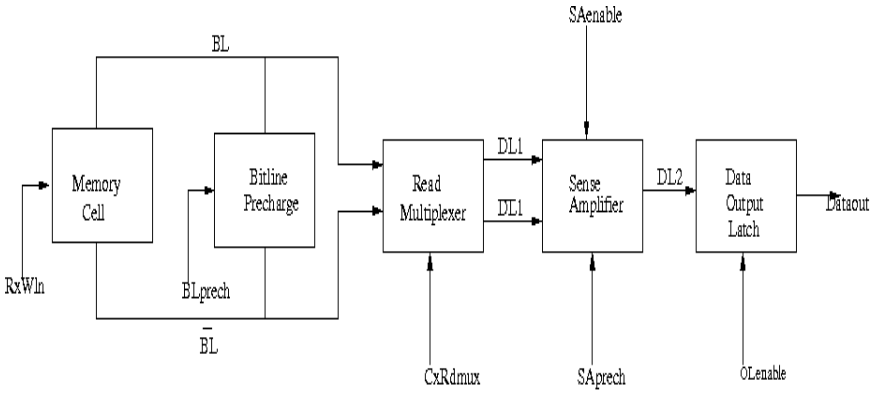


Figure 3.4: The Read Path Circuitry

high. $\overline{BLprech}$ initiates the precharging of the bitline pairs BL and $\overline{BL}$ to the same stable voltage V_{dd}. While $\overline{SAprech}$ precharges the datalines DL and $\overline{DL}$ to a stable voltage to prevent any undesired effect on the datalines. The reason for this precharge is to get both bitlines to the same voltage so as to facilitate the sensing of any differential voltage by the sense amplifier.

- **Sense:** This phase begins with setting of the read multiplexer signal *Rdmux* signal to high. This signal is used to connect the bitlines the the datalines DL and $\overline{DL}$. This is followed by setting the wordline to high using the signal *Wln*. The Wln now triggers the connection of the cell to the bitlines. The data to be read is thus placed on the bitlines. During this phase, the *BLprech* and *SAprech* signals have to go low. This is to maintain the stable voltage level on both bitlines and datalines during the read operation. The sense amplifier is turned on by setting *SAenable* to high just before turning the *RDmux* signal to low. Data on both datalines are fed to the sense amplifier. Since the sense Amplifier needs to be turned on just for a short interval of time.

- **Read Out:** During this phase the sense amplifier amplifies the voltage levels of the data on DL and $\overline{DL}$ and places it on *DL2*. This data is fed into the data output latch by setting the *Olenable* to high. Data from the output latch is now available on the *Dataout* pin of the SRAM circuit. The timing diagram for a single read operation is presented in Figure 3.5.

The various phases are distinguished by dotted lines. The rising and falling edge of each signal indicates the periods which the signals are turned high and low.

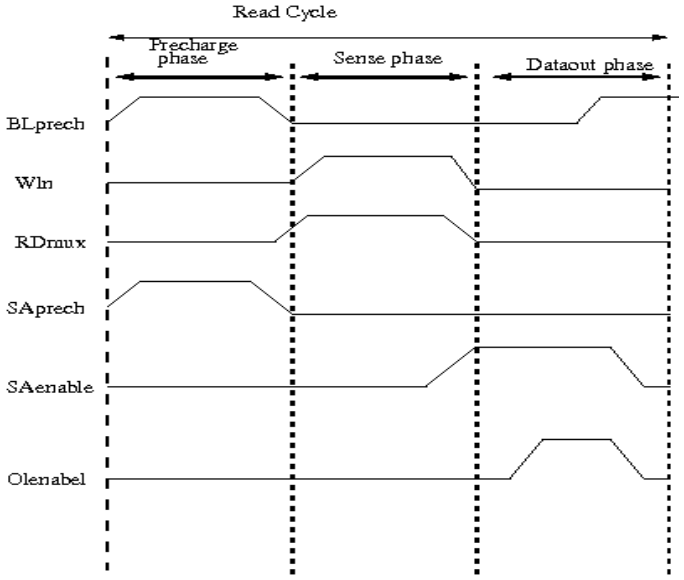


Figure 3.5: Timing Diagram For the Read Cycle

The electrical level representation of each of the circuits in the block diagram will be presented in the succeeding subsection.

3.3.1 Precharge Circuit

The precharge circuit is presented in Figure 3.6. This circuit consist of two PMOS devices, each

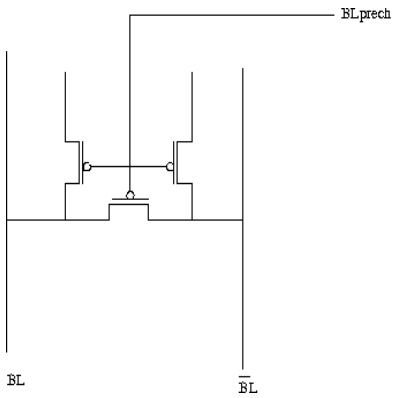


Figure 3.6: Precharge Circuit

connected to bitline/and its complement. Another PMOS transistor acts as equalizer to remove any differential voltage that might occur during precharge.

The precharge circuit in Figure 3.6 is known as V_{dd} precharge. The precharge circuit is made up of PMOS transistors device since they are most preferably in transmitting the V_{dd} level to the bitlines. Other precharge circuits exist such as the V_{dd} - V_T (where V_T the threshold voltage of the transistor device). This type of precharge circuit is not commonly used since it is very slow and depends on a process parameter threshold voltage (V_T) which usually varies from one technology to the other.

For correct value to be transmitted during a read operation, the memory cell nodes must force the bitlines to the sufficient voltage level when the word lines are turned on. The bitlines have a very high load capacitance value therefore charging them usually takes a long time. For this reason, it is better to precharge both bitlines to a particular stable voltage usually V_{dd}. By doing that, the voltage difference between both bitlines is easily determined since the voltage on both bitlines is known and stable.

3.3.2 Read Multiplexer

The read multiplexer circuit is used to connect the bitlines to the sense amplifier during a read operation. This circuit as illustrated in Figure 3.7 is designed in such a way that it should be able to transmit the voltage level on the bitlines to the datalines and then to the sense amplifier. As we can see from the figure, PMOS devices are used for the read multiplexer instead of NMOS transistors. This is because reading involves sensing of a voltage difference between two previously charged bitlines. Using a PMOS device ensures that the high voltage level transmitted is not degraded, since PMOS devices are better transmitters of high voltages than NMOS devices. The PMOS device is controlled by a signal *RDmux* which is set to low when the bitlines needs to be connected to the datalines and high to disconnect the bitlines from datalines and sense amplifier.

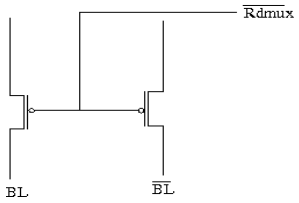


Figure 3.7: Read Multiplexer

3.3.3 Sense Amplifier

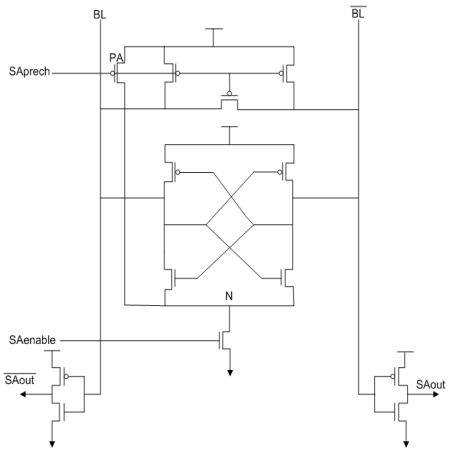


Figure 3.8: The Sense Amplifier Circuit

The sense amplifier circuit as presented in Figure 3.8 consists of two crossed coupled inverters, designed in a similar way as the memory cell, connected to the datlines.

The sense amplifier has a precharge circuit which precharges both bitlines and controlling N node (of NMOS pull-down transistor of the sense amplifier) to a stable voltage V_{dd}. When the *SAenable* signal is turned on the controlling NMOS transistor, will start conducting. Depending on the voltage difference on both bitlines the sense amplifier will flip to one of the its bistable states.

During read operation the wordline of a cell is set to a high. Depending on the value in the cell, one of the bitlines will be pulled down. The bitlines will now have different voltage levels. This voltage difference has to be amplified and sent to the output latch for the correct data to be present on the *dataout* pin.

In order to read the correct voltage, a sense amplifier is placed between the bitline and the output latch. The sense amplifier reduces the wait time for the complete discharge of the bitline thus increases the speed of the read operation.

3.3.4 Data Output Latch

The data output latch is implemented with D-type latch. A clock signal is connected to the transmission gates (TGs). The data output latch is represented in Figure 3.9. Data output latch

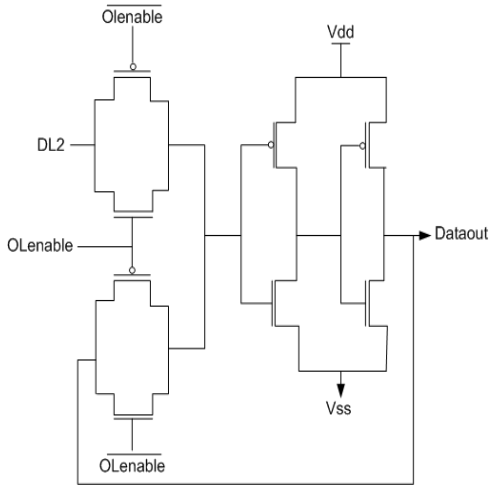


Figure 3.9: Data Output Latch [3]

consists of two parallel TGs which are connected in series with two cascaded inverters. The upper TG has input DL2 and is controlled by *OLenable*. Whilst the lower TG has feedback from *Dataout* and is controlled by *OLenable*.

As can be seen in Figure 3.9 if the *OLenable* is set high the upper TG connected to input DL2 conducts, and the lower TG that is connected to output *Dataout* is disconnected. Input *DL2* is now connected to Dataout through the two inverters. *Dataout* will have the same value as *DL2* as long as *OLenable* is high. On the contrary if *OLenable* goes low the upper TG disconnects. The lower TG connects the two inverters forming a loop, therefore maintaining the output data value at *DL2*. After data has been read out of the memory cell and amplified by the sense amplifier, it is placed in the output latch which is fed to the SRAM *Dataout* pin

3.4 Write Path Circuitry

The write path of the memory is defined as the route from the data input pin through the write driver to the memory cell array. The write path as derived from Figure 3.2 is presented in Figure

3.10. The block diagram in this Figure represents the circuits that are involved during the write

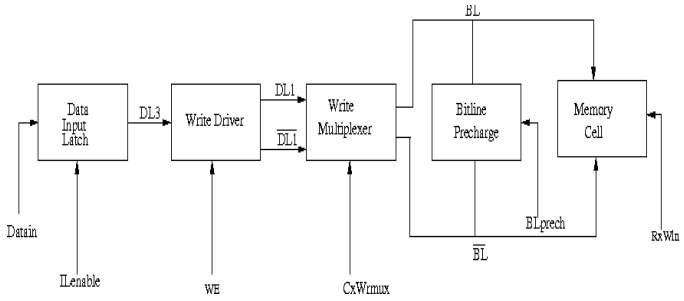


Figure 3.10: Write Path Circuitry

cycle. The inputs to every circuit are control signals represented by arrowed lines.

As shown on the timing diagram of Figure 3.11, the write cycle is made up of two phases, the precharge phase and the write phase.

- **Precharge Phase:** During the precharge phase both bitline BL and $\overline{BL}$ are precharged to high a voltage V_{dd} by setting the *BLprech* signal to high. During this phase the *ILenble* signal of the Data input latch is turned high to place the data on the *Datain* pin.

- **Write Phase:** Data is placed on *DL3* is fed to the write driver. The write driver writes the data on the datalines DL or $\overline{DL}$ depending on the value at *datain*. During this operation the write multiplexer has to be enabled by setting the *Wrmux* signal to high. This enables the bitlines to be connected to the datalines. The bitlines at this moment, contains data from the data input latch so data can be written to the cell by setting the wordlines to high.

  The *Wrmux* signal must be turned high before the wordlines are turned on (for the connection of the datalines to the bitlines). This prevents the bitlines from being discharged by the cell before the data is written. If this happen then the bitlines will need to be precharged again which increases the write cycle time.

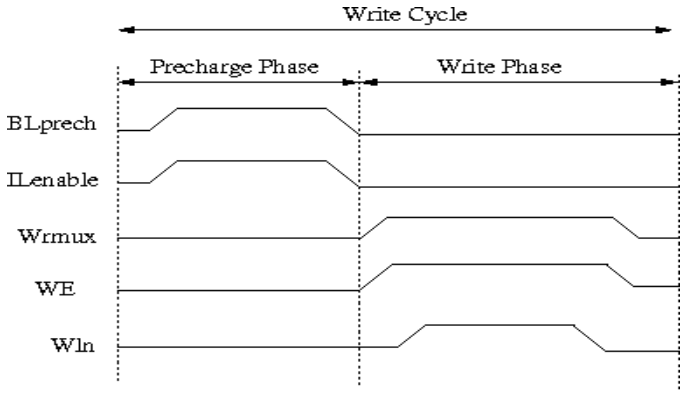


Figure 3.11: Write Cycle Timing Diagram

In the following subsections, the electrical level representation of each subcircuit included in the write path circuitry will be described. These circuits include the write driver and the write multiplexer. The precharge circuit and data input latch for the write operation is the same as that used for the read operation thus will not be described again.

3.4.1 Write Driver

The write driver has as input *Datain* and a control signal *WE* which is activated to enable the write driver during the write phase as shown on Figure 3.11. A simple write driver as depicted in Figure 3.12. is implemented with two inverters and two NMOS transistor devices. The *WE*

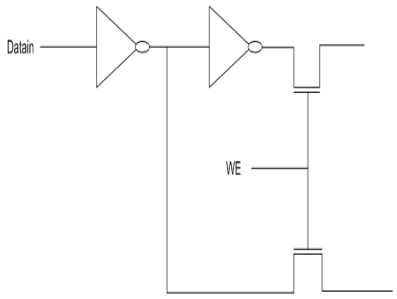


Figure 3.12: Write Driver [2]

is used to place the correct input data from the data input latch to the datalines BL and $\overline{BL}$ during the write cycle. Data to be written is presented on the *Datain* pin. The write signal is turned high to connect the two NMOS transistor devices to the respective bitlines. It is necessary to precharge the bitlines first before the write driver is turned on. This increases the speed of the write operation.

3.4.2 Write Multiplexer

The write multiplexer as shown in Figure 3.13 functions as the interface between the write driver and the memory cell array. The write drivers are required to pull-down one of the bitlines to a low value (ground). Thus the write multiplexer uses NMOS transistor device connected to both bitlines. The write multiplexer is controlled by a signal *Wrmux* which will be enabled during a write operation, after the bitlines have been precharged. When *Wrmux* is enabled the bitlines are connected to the write driver. The use of an NMOS device for the write multiplexer is to ensure that during the pull-down of the bitlines, the low level voltage is not degraded as would have been the case if a PMOS device was used.

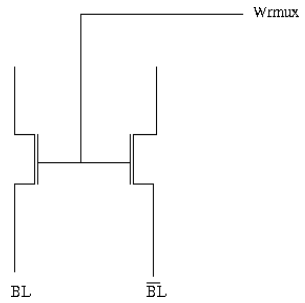


Figure 3.13: Write Multiplexer

3.5 Address Decoder Circuitry

Decoders are used to access a cell or group of cells in a memory cell array. Since for example a 1Mbit chip requires 1million wordlines (i.e 1million addresses of 1-bit words), the chip area required for these wordlines will be too high. For this reason the two dimensional addressing schemes made up

of row decoders with wordlines and column decoders with bitlines is implemented on chip. This reduces the area required for the number of lines to $\sqrt{n}$ as oppose to n, where n is the number of bits in the chip. Figure 3.14 depicts the address decoder circuitry. This circuitry is made up of an

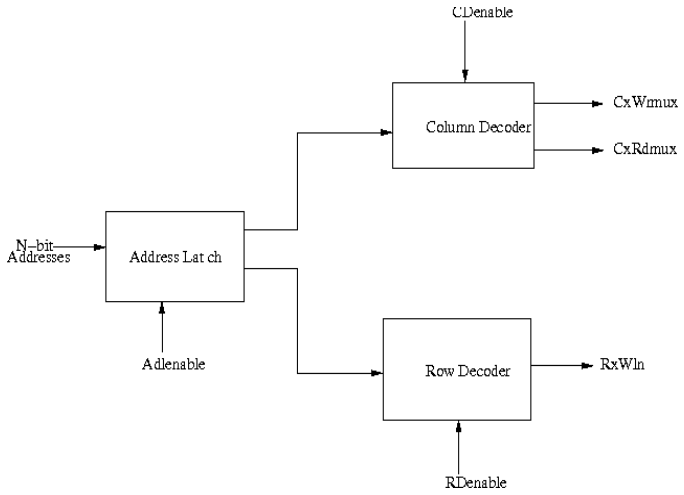


Figure 3.14: Address Decoder Circuit

Address Latch where the n address bits are stored, a row decoder and a column decoder circuits. Each of these circuits will be discussed in the succeeding subsection.

3.5.1 Address latch

This is a D-type latch which stores the n-bits addresses of the memory cell array. These n-bits addresses are divided into $\log_2(c)$ column addresses and $\log_2(r)$ of Row addresses. The implementation of the address latch is the same as the data input/output latch.

3.5.2 Column Decoder

The column decoder is used to select the appropriate column of the memory cell array by activating the bitlines. Figure 3.15 shows the diagram of a column decoder. In order to reduce the size and

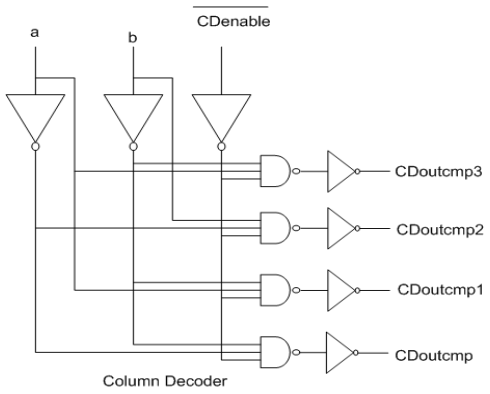


Figure 3.15: Structure of a Column Decoder [3]

increase the speed of the column decoder a predecoder circuit can be added to the column decoder. In this case the bits of the column decoder is split into two parts, a bits predecoder fields and c

bits column decoder fields. The a bits of the predecoder is predecoded to 2^a lines and fed into the column decoder.

A simple column decoder without pre-decoding logic as shown in Figure 3.15 is implemented with seven inverters and four NAND gates. The decoder also contains an input signal *CDenable* on each NAND gate. This signal is set low to disable the decoder thus connecting and disconnecting the bitlines from the sense amplifier and write driver using the read and write multiplexers.

3.5.3 Row Decoder

The row decoder is used to select the appropriate row of a memory cell array by selecting the desired wordline out of the R wordlines. Figure 3.16 below illustrates the implementation 2-to-4 line row decoder. The size of the row decoder can be reduced as the case with the column decoder by implementing a row decoder with predecoding logic. This small size is able to increase

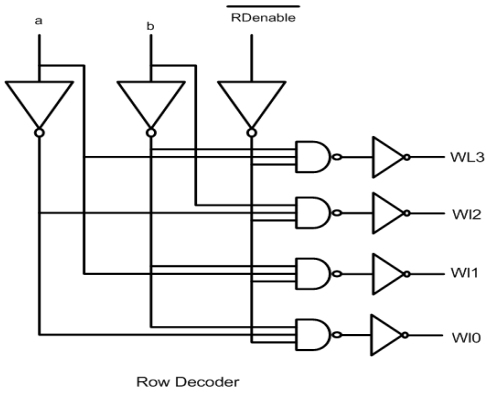


Figure 3.16: Row Decoder

the speed of the row decoder. However in a simple SRAM circuit as the one used in this thesis which is meant for simulation purpose, only the row decoder will be implemented without the predecoder logic. The inputs of the NAND gate of the row decoder is connected to a clock signal $\overline{RDenable}$ which controls the timing for the selection of the wordline. The row decoder is enabled when $\overline{RDenable}$ equals 0. This is the reason why the input of $\overline{RDenable}$ is inverted to match the outputs of the NAND gates.

3.6 Control Circuit and Timing generation

The interconnection of all the circuits described above is needed for the memory to function properly. This interconnection of circuits must be controlled by a timing generation circuit. The timing generation diagram is used to group all the events occurring on every clock switching. In order to implement this circuit we will divide the signals into read and write operations. This gives a better insight of the timing requirements for each operation. When defining the timing requirements it is necessary to limit the time between actions, thus making them as small as possible. The purpose is to increase the speed of the SRAM. The timing requirements as depicted in Figure 3.17 are grouped together thus synchronizing them as much as possible. That is grouping events that occur in the same clock switching gives us the following flow diagram for the read and write cycle as shown in Figure 3.18. Having defined the order of events, a timing generation circuit can be designed. The circuit is presented in Figure 3.19. It should be noted that some periodic signals have to be generated. In the case of an SRAMs circuit, which has just one clock signal a delay element is implemented that shifts the clock signal. The clock signal has to be

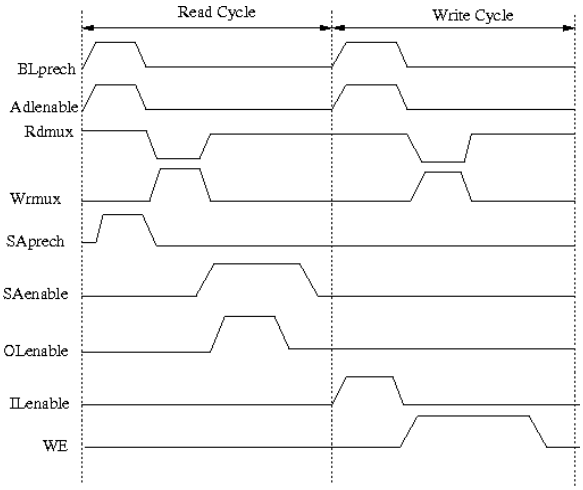


Figure 3.17: Timing Diagram for the Read and Write Cycle

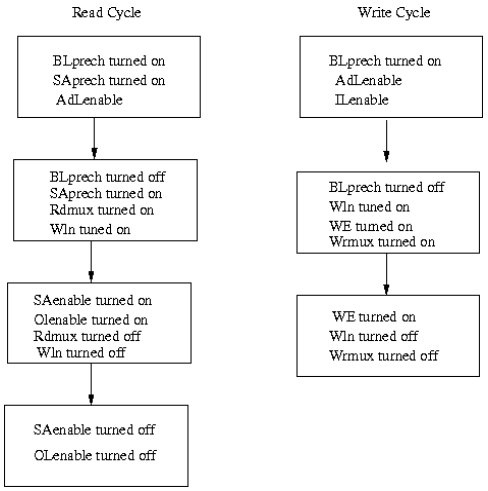


Figure 3.18: Events Timing Flow Chart

high every time there is the generation of a periodic signal. This will require the use of two delay elements with a delay of T/3 (T represents each phase of an operation) since the longest periodic switching is that of the read cycle and it take three periodic switching (considering the speed of the SRAM is 1.5ns). The delay elements are implemented with a series of inverters.

Summary

In this chapter a detailed description of the of the architecture of the SRAM has been presented. The timing control signal and timing generation diagrams have been presented. It is important to note that the read and write operations are carried out separately using different peripheral circuits. the peripheral circuits presented are not standard. Other types of these peripheral circuits exist. Th choice of which circuit to use is sorely dependent on the designers decision. These peripheral circuits will be combined together in the next chapter to design an SRAM simulation model.

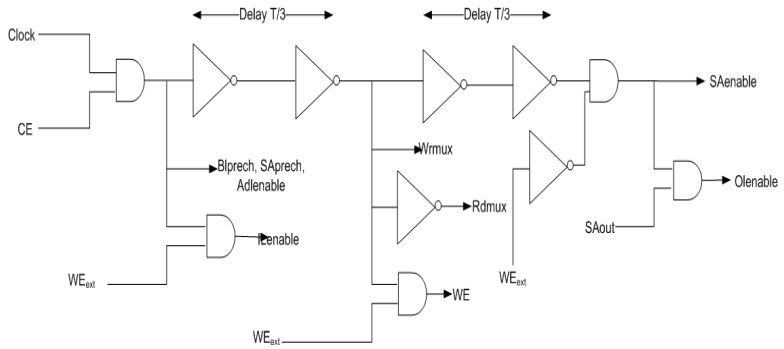


Figure 3.19: Time Generation Circuit

4 SRAM Simulation Model

To be able to evaluate and implement any DFT technique for the detection of DRFs in SRAMs an SRAM electrical model is designed and implemented using HSPICE. In this Chapter the simulation model used including design decisions made during the implementation are described in Section 4.1. The behavior of an SRAM cell is modeled using transient analysis which requires some input signals. In Section 4.2 The timing control diagram of the proposed model is presented. In Section 4.3 graphical experimental results obtained from AvanWaves of the HSPICE simulator for a fault free SRAM is presented . The results includes the validation of the SRAM for its normal read and write operation and for Data retention.

4.1 SRAM Electrical Model

The SRAM electrical model is a cross section of the SRAM circuit. The model includes all the SRAM subcircuits presented in Chapter 3 except the data input/output latches and the timing generation circuits which were omitted. The PieceWise linear (PWL) function in HSPICE was used to replace the timing generation circuit. The PWL function decomposes a set period of time to a small number of intervals. The data input latch is replaced with the *Datain* signal as we assumed data is already present at the *Datain*. The sense amplifier's output is used to represent the data output latch as their values represents the data value which is read from the cell.

The simulation model involves implementation of an SRAM electrical model on an appropriate circuit simulator like HSPICE. The behavior of the SRAM circuit can be investigated by carrying out transient analysis (time derivative of the bias voltages and output terminal current at the particular instant in time). For the design and implementation of the simulation model, an appropriate MOSFET model must be carefully chosen so that the transistor parameters for that model can be used. In our case the SPICE parameters used are 3rd generation parameters. The choice of the model is based on its simplicity. This is because with BSIM4 [1] a single equation is used to describe device characteristics in various operating regions. The model parameters were derived using Predictive Technology Model (PTM) developed at the University of Berkeley Carlifonia.[26] The implementation of the model, an input file is specified. In this input file, a definition of all the subcircuits of the SRAM electrical model and the CMOS model parameters are specified. The HSPICE input file used is presented in Appendix A. The results of the HSPICE simulation is interpreted graphically.

The model as presented in Figure 4.1 consist of the following subcircuits.

- 3 x 3 x 3 memory cell array, cell 1,2,3,4,5,6,7,8 and 9.
- Three Precharge Circuits.
- One 2-to-4-bit Row Decoder.
- One 2-to-4-bit Column Decoder.
- One Sense Amplifier circuit.

[1] New version of Berkely Short channel IGFET Model (BSIM). This is a level 54 MOSFET model used in the simulation of integrated circuits.

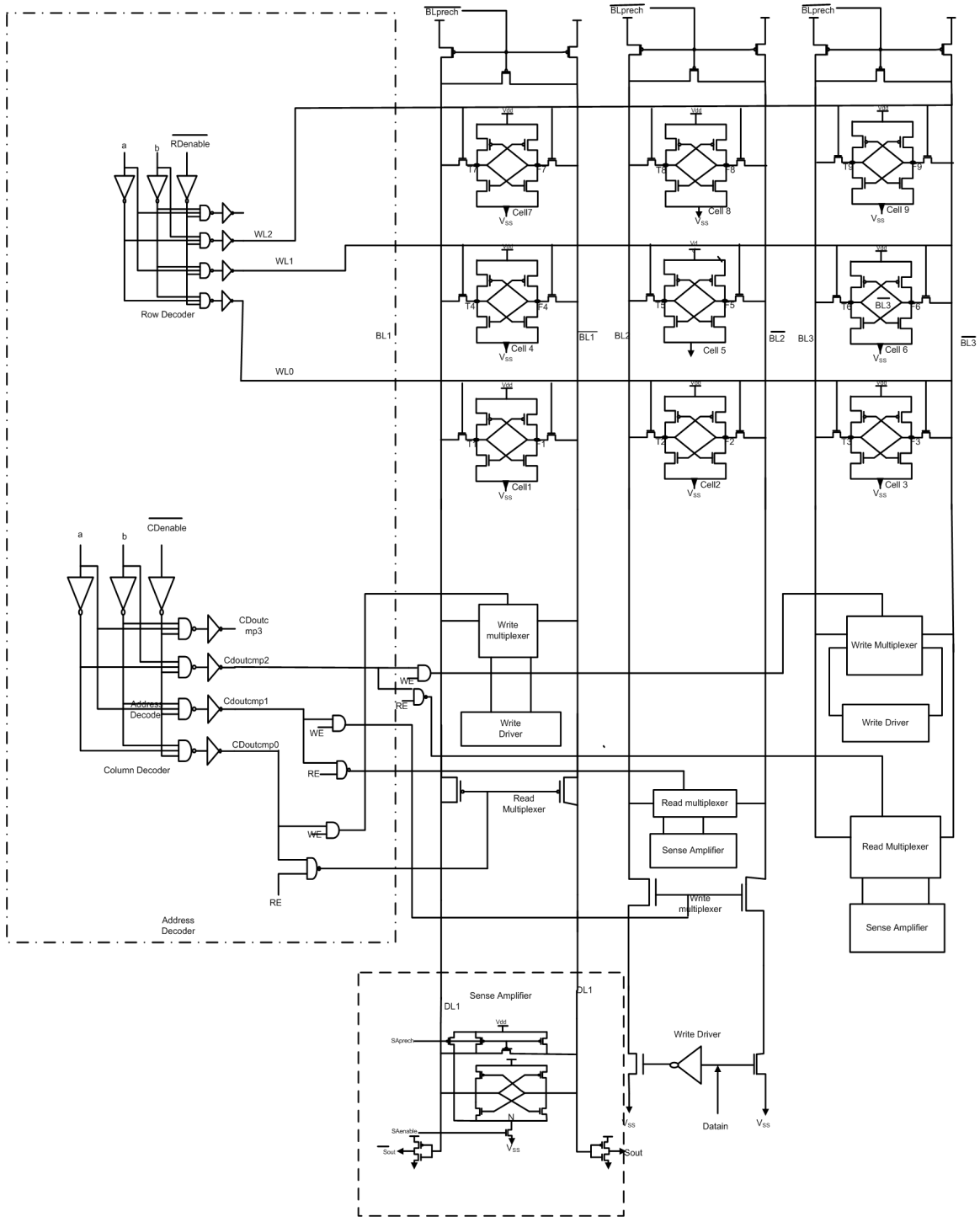


Figure 4.1: Electrical Level Schematic of SRAM Model

- Three read multiplexers.
- Three write multiplexers.
- One write driver.
- Two Inverters used to monitor the output of the sense amplifier.

The memory cell array are connected to the sense amplifier, via the datalines. The datalines are connected to the bitlines using the read multiplexers. The design of the sense amplifiers is similar to the cell as both of them bistable elements. In order to maintain cell stability, transistor sizes must be carefully considered during the design of the cell. Using the six-transistor SRAM cell as presented in Figure 3.3 the following should be taken into consideration.

- The PMOS pull-up transistors must have a ratio of at least 3:2 [28] (Where 4 refers to the width of the transistor and 3 the length). This is because during a write operation, to write a 0 to the cell the 0 on the bitlines ahould be able to create a pull-up effect on the PMOS pull-down device thus causing the cell to flip its value. In our model the sizes of the pull-up device were, W = 0.2μm and L = 0.1μm
- The NMOS pull-down must have the ratio of at least 6:2. This is because the NMOS transistor must be able to slowly pull - down the bitline during a read operation. This leaves the NMOS pull-down transistor with a size equivalent to at least 1.5 to 2$\times$ larger than the size of the pass transistors [28]. The sizes used for our model are; W = 0.5 μm L = 0.1μm
- The NMOS pass transistors are used connect cell to bitlines. For this reason the size of this NMOS transistor is in the ratio of 4:2 which should be at least 1.5$\times$ PMOS pull-up transistors [28]. The size used for our model are; W = 0.3μm and L = 0.1μm

Note that the cell size have to be as small as possible to minimize area and reduce the discharge rate of the bitline capacitance. During the read operation the discharge rate of the very small to ensure the cell does not flip. This implies that the NMOS pass transistors must have a higher resistance than the NMOS pull-down transistors. Thus smaller widths of the NMOS pass device.

Apart from the memory cell array, care has to be taken while designing the other peripheral circuits. The decisions made during the design of the various peripheral circuits will be presented below. These decisions were mostly based on observations made with simulations results obtained using different transistors widths. The transistor length is set to a standard 0.1μm based on the technology used [27].

- **Sense Amplifier:** Although the device is the same in structure as the memory cell array the sizes of the transistors for the NMOS pull-down devices are different than those of the cell array. The bigger the width the faster the device, since only a small voltage difference needs to be sensed and then amplified. The sense amplifier as mentioned earlier is turned on by an NMOS device.

  We are interested in transmitting a pull-down effect on the bitlines, that is the voltage difference between the bitlines one being v_{dd} and the other a little below v_{dd}. The width of the NMOS device should therefore be big enough to enable a fast transition. It was observed during the simulation that reducing the width of this device below 2μ will cause some spikes in the datalines and therefore a more larger pull-down effect on the appropriate bitlines. Increasing the sizes over more that 2μ had no effect on the datalines. Transistor sizes used in our model for the sense amplifier were as follows;

  - NMOS pull - down transistors W = 2μm and L = 0.1μm
  - PMOS pull - up transistors W = 0.1μm and L = 0.1μm
  - NMOS sense amplifier enable device W = 2μm and L = 0.1μm

- **Read Multiplexer:** The read multiplexer as presented earlier is made up of PMOS devices. The width of these transistor devices are chosen in such a way that when connecting the

bitlines to the datalines the high voltage level should not be degraded. Owing to the fact that the smaller the width of a transistor the slower the device, the width of the PMOS devices were set to at least 6μ. The resulting waveforms of the datalines for widths smaller than 6μ of the read multiplexer were different from those above 6μ. The datalines for smaller values of the read multiplexer device presented a lot of spikes meaning the voltage level was being degraded. Transistor sizes used for the PMOS device was W = 6μm and L = 0.1μm.

- **Write Driver:** This device is meant to be very strong to be able to select the required precharged bitline depending on the value at *datain*. Larger width for the NMOS devices used in the write driver is thus advantageous. Transistor sizes used for the NMOS device in this model were as follows W = 12μm and L = 0.1μm.

- **Write Multiplexer:** Connecting the write driver to the bitlines requires high speed. The write multiplexers used are made up of NMOS devices. Their width sizes should just be sufficient enough to initiates this connection. Transistor size used for this model were W = 4μm and L = 0.1μm.

- **Row and Colomn Decoders:** These decoders were made up of CMOS NAND gates. NAND gates were preferred since the PMOS devices of the NAND gates are connected in parallel. This increases the speed of the decoders. To minimize area while maintaining adequate speed, the width of the NAND devices after simulating with different widths was set to 8μm. The output of the column decoder is used to turn on the read and write multiplexers. These two devices are turned on with two different signal. To be able to achieve this the read and write paths were totally separated. This ensured that the datalines are not connected tot he bitlines during the write operation. Tow additional logic was thus added tot he output of the column decoder. These two additional logics were implemented as follows:

  - Two input AND gate used to turn on the write multiplexer: The input of the NAND gate are the WE signal and the output of the column decoder. This was chosen because as indicated in [28], a write operation begins when some external signal selecting the right column and the *WE* signal initiating the write operation thus beginning the write phase of the write cycle.
  - Two input NAND gate used to turn on the read multiplexer: The input of the NAND gate is the column decoder output and an external signal RE. This additional logic enables datalines to be connected to the bitlines only during the read operation.

4.2 Timing Control Diagram for Read and Write Cycles

With the chosen technology the speed of the SRAM is estimated at 1.5ns for each of its operation to be carried out. The timing diagram specified 1n the Figure 4.2 represents the timing used in the simulation. Actions within each operation has been spread over a period of 1.5ns. Note that the precharge circuit is turned on just before the beginning and end of each cycle. This is used to maintain a default value of the bitlines at 1.2v (V_{dd} supply voltage for 90nm technology). The wordlines signal is generated using the row decoder inputs thus it is not represented in the timing diagram. Making use of more floating point numbers will result in more appropriate timing. The Cdenable signal is turned on before the Rdenable signal so as to increase the sped of both the read and write operations.

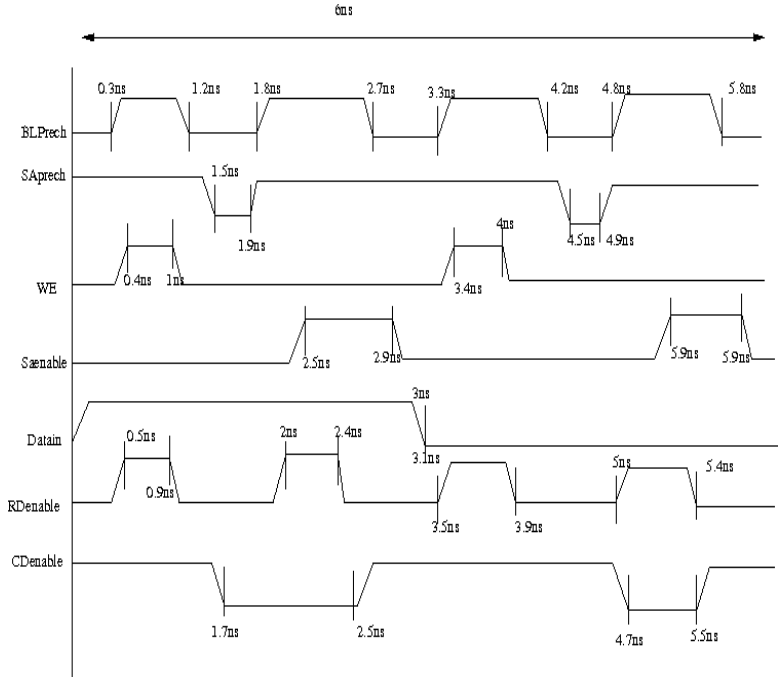


Figure 4.2: Timing Control for *0w1 r1 w0 r0*

4.3 Simulation Results

The graphs produced depend solely on the input data given. The methodology used is that of single cell faults. Single cell faults are faults that occur as a result of a defect in the cell caused by a defect in that particular cell. In an SRAM circuit there are six possible situations which can occur. These situation depends on the type of operation carried out. The six possible situations will be presented on table 4.1. The left column of the table represents the operations carried out

Operation	State of the cell
w0	0 1
w1	0 1
r0	0
r1	1

Table 4.1: Six Possible Situations of an SRAM Cell

and the right column the expected state of the cell. The state of the cell is affected by a write operation. There are two possible types of write operations that can be carried out. These are:

- **A non-transition Write Operation:** In this case the data written to the cell is the same as that present previously.
- **A Transition Write:** In this case the expected state of the cell becomes the value of the operation carried out (writing a 1 when there was a 0 up transition and writing 0 when there was a 1 in the case of a down transition.)

To verify the correct behavior of the memory, two types of operations were carried. Namely:

- **Operation Simulation:** Operation simulations (OPsim) is carried out by performing a sequence of different memory operations to the cell. Four different sequence of memory operations were carried out. Each of these operations lasts 1.5ns as stated by technology (speed

of SRAM). These sequence of the operations are presented on Table 4.2. The validation of

Simulation	Initial State of the cell	Operation Simulated
OPsim 0w1	'0'	w1,r1,w0,r0
OPsim 1w1	'1'	w1,r1,w0,r0
OPsim 1w0	'1'	w0,r0,w1.r1
OPsim 0w0	'0'	w0,r0,w1,r1

Table 4.2: Four Situations to validate SRAM Cell

the write operation is carried out by a subsequent read operation. These four simulations are used to validate the six situations which can occur in the memory. The graphs obtained during the simulations are presented in Figure 4.3.

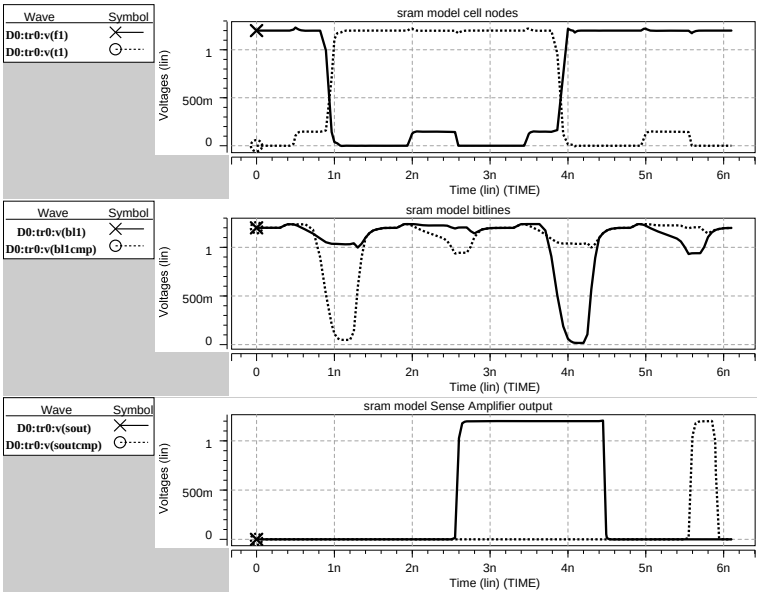


Figure 4.3: Simulation results for $0w1r1w0r0$

This Figure consists of three graphs.

- **True and False nodes:** Since the state of the memory is initialized to 0, the true node is set to 0v and the false node 1.2v. When the wordline is turned on the voltage of the true node is pulled up to high while the false node becomes low. This causes the cell to flip. during the read operation only a small voltage difference is needed between the precharge bitlines. The false node is pulled up just a little over ground when the wordline is turned on high. The voltage of the true node remains at V_{dd}. These sequence of operations are spread between 6ns with each operation lasting for 1.5ns.
- **Bitlines:** The default value for both bitlines is V_{dd}, to avoid long precharge cycles. When the wordline is turned on depending on the value to be written, the appropriate bitline is pulled down. For example to write a 1 in the cell, $\overline{Bl}$ is pulled down to 0. The read operation is followed with the same bitline being pulled down but not to 0 as was the case during the write operation.

– **Sense Amplifier outputs:** This is necessary to verify if the right value is actually read from memory. The output of both inverters connected to both bitlines are presented. As shown on Figure 4.3, during a first read operation the inverter with output $\overline{Sout}$ connected to BL is turned on high. For the subsequent $r0$ operation the inverter connected to $\overline{Bl}$ is turned on high. These two outputs thus validate the correctness of the read data. The timing control signals are presented in Figure 4.4. These signals consist of the following:

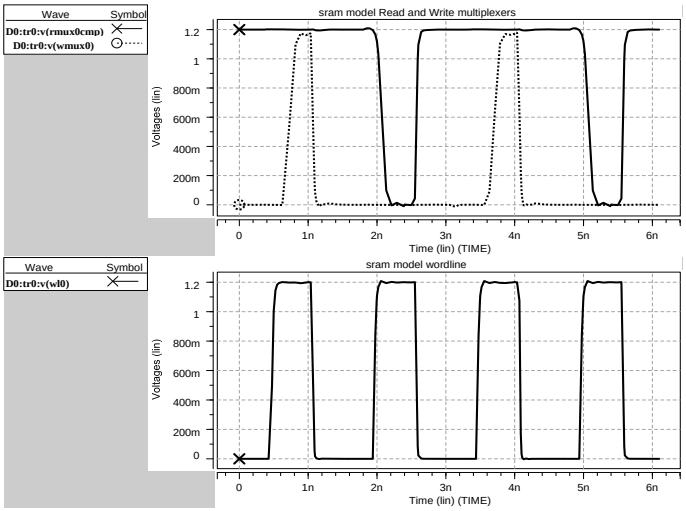


Figure 4.4: Imput signals for simulation *0w1 r1 w0 r0* operation

– **Wordline:** This signal which is generated using the row decoder inputs determines when the cells will be connected to the bitlines.

– **Read/Write Multiplexers:** The read multiplexer controls the connection of the bitlines to the datalines. The read multiplexer signal is active when low since the read multiplexer is a PMOS device. The write multiplexer connects the write driver to the cell and is turned on when the signals is high as seen from the graph. It should be noted from the graphs as explained in Chapter 3.3 that the read and write multiplexer are turned on before the wordline is turned on.

- **Data Retention Simulations:** Data retention simulation (DRsim) to check the ability of the cell to store its logic value(data) after a certain long period. Two types are distinguished as shown on Table 4.3 below.

Simulation	Initial State of the cell
DRsim-0	o
DRsim-1	1

Table 4.3: Two types of Data Retention Simulation

The graphs for the DRsim are presented in Figures 4.5. The waveforms obtained are similar to OPsim with the only difference of the pause period of 400ms induced before the read operation. This wait period was introduced only after the first write operation. For the w1 operation we assumed the cell was not defective.

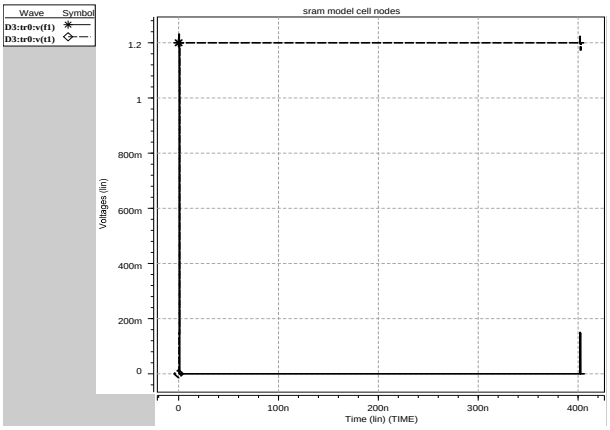


Figure 4.5: Simulation results for Defect free SRAM cell for Data Retention Validation

Summary

An SRAM simulation model has been implemented and simulation results for a defect free memory obtained. During the design it was observed that the accuracy of the simulation model depends to a greater extend on the choice of transistor sizes. This memory has been validated for its normal operation. The ability of the cell to keep its logic value after a certain time period was also verified. This model will be used in the next Chapter to evaluate some DFT techniques in SRAMs.

Implementation and Evaluation of DFT

5

Various DFT techniques have been discussed in Chapter two. Most of these techniques have not been validated. In this Chapter, three of such DFTs (i.e, for the detection of DRFs; WRTM, NWRTM and the PDWTM) will be implemented and evaluated. The efficiency of these techniques in the detection of DRFs will be presented. In Section 5.1, a defective memory will be used to validate the existence of DRF, by evaluating the pause test. In Section 5.2 WWTM, NWRTM and the PDWTM techniques will be implemented and evaluated. Experimental results will be presented to show the efficiency of these techniques in the detection of DRFs. A comparison of the obtained results from the DFT will be presented in Section 5.3.

5.1 Evaluation of the Pause Test

The pause time needed to detect an open that might result to a DRF is usually not known. DRFs as stated earlier occur due to opens on the source and drain of the PMOS pull-up transistors of the SRAM cell array. These opens are classified under two classes of defects symmetrical or asymmetrical. These two classes of defects are illustrated using Figure 5.1. Defect R1 refers to the

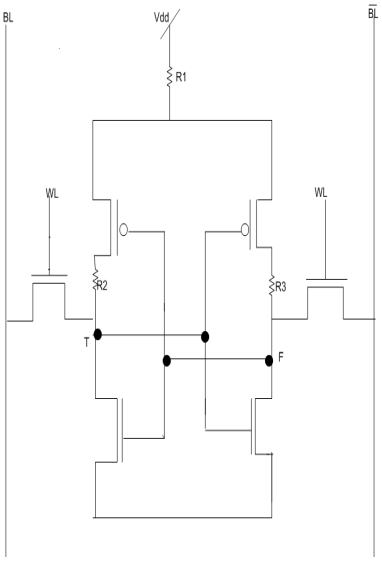
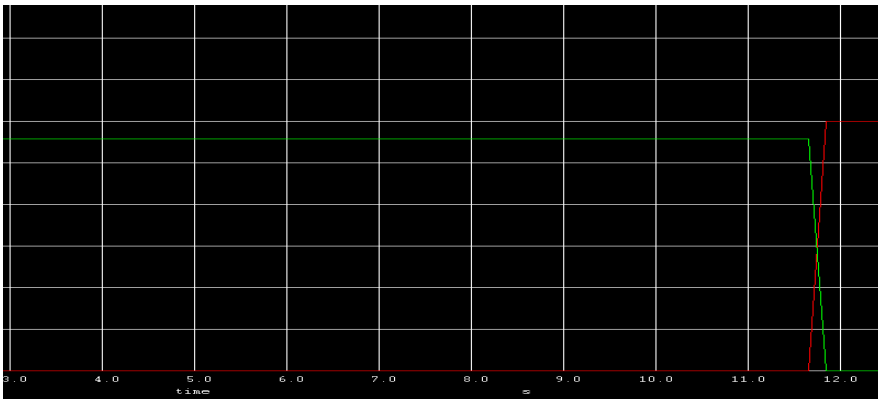


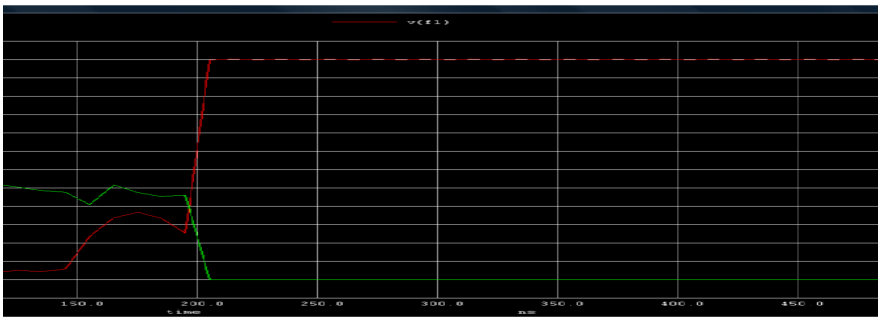
Figure 5.1: SRAM Cell with two classes of Defects

class of defects known as symmetric defects. A defect at R1 will affect the cell no matter what the initial state of the cell is. Defect R2 and R3 are asymmetric defects. these defects affect the cell independently depending on the initial data value in the cell.

The range of the defect considered is from 1kΩ to 10GΩ. The defective memory cell is initialised to 1 using a $w1$ operation. Simulations are carried out separately for the source and drain. Waveforms obtained for two R-values are presented in Figure 5.2. These waveforms represents the behavior of the nodes of the cell in the presence of an open defect. We observe from the waveforms that after a certain period the true node whose voltage level was high is pulled down to 0 due



DRF Validation Using the Pause test R-value = 10M



DRF validation Using Pause Test R-Value = 1G

Figure 5.2: Simulation results of Defective SRAM for DRF validation using Pause Test

to the presence of an open defect. These results illustrates that a cell with an open defect at the drain and source of the PMOS pull-up device will flip after a certain time period after which it is has not been active. These simulations validates the existence of DRFs in SRAM. A summary of the flip time needed for cells to flip for different R-values simulated are presented in Table 5.1 below.

R-value	Flip time Drain Defect	Flip time Source Defect
1kΩ	Not detected	Not detected
10kΩ	Not detected	Not detected
100kΩ	Not detected	*Notdetected*
1MΩ	Not detected	*Notdetected*
1.5MΩ	Not detected	Not detect
10MΩ	12s	12s
1GΩ	550ns	550ns
10GΩ	200ns	200ns

Table 5.1: Defect Coverage For DRF Using Pause Test

From the above results a graph of flip time versus R-value is plotted as shown on Figure 5.3. This graphs as mentioned in chapter two illustrates the importance of testing for DRFs using smarted test methods as the wait time for smaller defects are quite big. The above results will

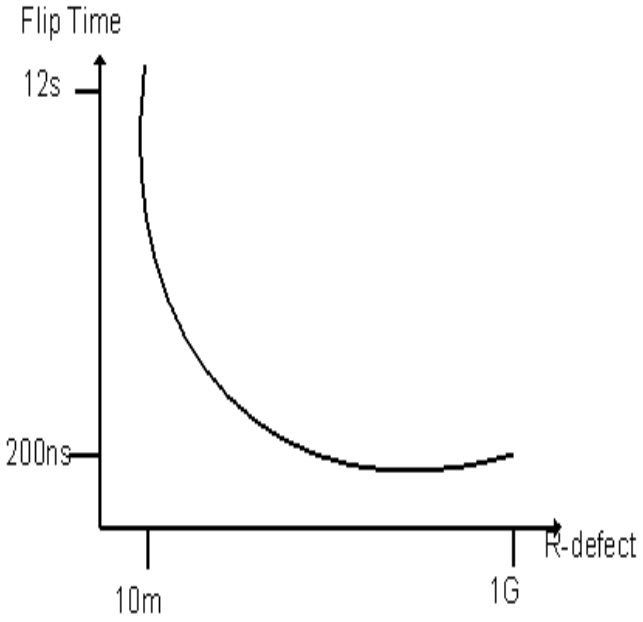


Figure 5.3: Relationship between Flip Time and R-Defect

thus be used to demonstrate the efficiency of DFTs in the detection of DRFS. The implementation and evaluation of these DFTs will be explained in the next section.

5.2 Implementation and Evaluation of DFT Techniques

Three DFT techniques were chosen from the six techniques presented in Chapter two. The choice of these techniques was based on the fact that they were said to detect both symmetric and asymmetric faults in the cell array. The implementation and evaluation results will be presented in the succeeding subsections.

5.2.1 Implementation and Evaluation of WWTM DFT technique

The WWTM technique involves adding six transistors to the bitlines of the SRAM cell. This circuit used is known as Weak Ram Write circuit (WRW) as seen in Figure 2.4. For the implementation of the WRW circuit transistors sizes as explained in Chapter 2.3.1 were used. The algorithm is carried out as follows:

- A background of 1 or 0 is written to the memory using the appropriate write operation.
- The WRW circuit is enabled.
- Depending on the initial value in the cell the appropriate control signal is enabled. For example if the background data value was a 0 the the $WR1$ is enabed to perform a long weak write 1 operation of about 50ns.
- Disable the WRW circuit.
- A read operation is performed to determine if faulty cells were overwritten.

The activation of *WR0, WR1* are solely dependent on the data value in the cell and were modeled using the PWL linear function in HSPICE. This implies that if the data value of the cell was 1 the $WR0$ signal will be activated first. This cycle will be followed by a normal read operation which determines if the defect was detected. Simulations were first performed on a defect free memory cell to ensure that the DFT technique does not overwrite a good cell after a certain period. The results obtained are presented in Figure 5.4.

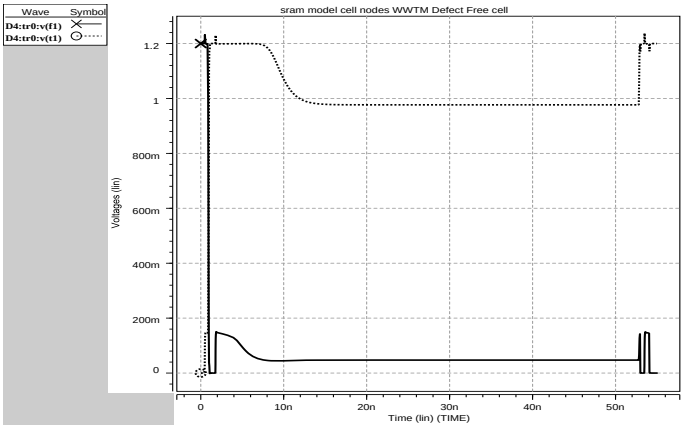


Figure 5.4: WWTM Implementation Defect Free Cell

From the waveforms we observe that during the $WR0$ the true node is pulled a little below V_{dd} while the false node is pulled a little above ground. During the read operation the nodes are pulled back to their respective values and the right value is read from the cell.

R-values ranging between 1kΩ to 1MΩ were added to the defect free cell. Simulations were performed separately for the drain and source. Simulation results obtained for a defective drain are presented on Figure 5.5. The waveforms represent three R-values. As can be observed R-

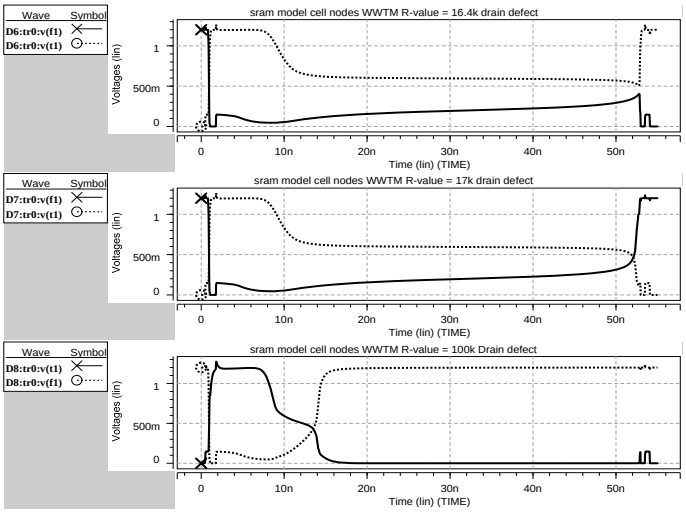


Figure 5.5: WWTM Defective Drain

values of $\leq 16.4k\Omega$ are not detected with WWTM. The true node of the cell as seen from the first waveform with R-value of $16.4k\Omega$ is pulled down to almost $V_{dd}/2$ but the pull down effect is not enough to flip the cell.

Similar results are obtained from a defective source with the only difference that smaller R-values were detected. The results obtained from three defect values are presented in Figure 5.6 From the waveforms obtained we observe that R-values of $\leq 7.5k\Omega$ at the source are not detected

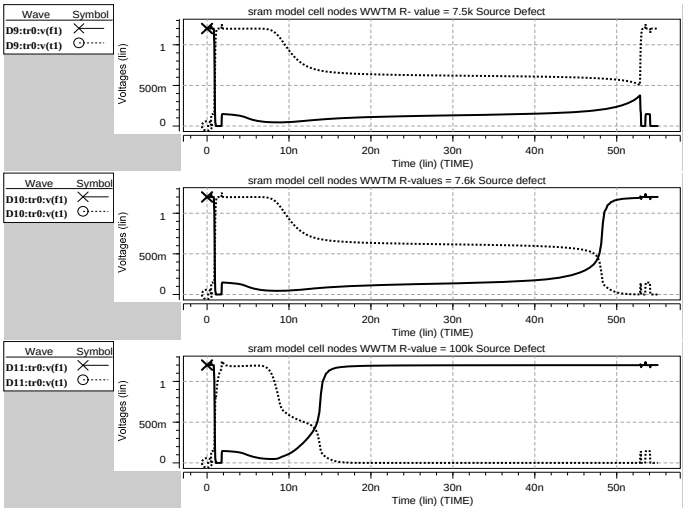


Figure 5.6: WWTM Implemention Defective Source

using WWTM DFT technique. A summary of the defect coverage for all R-values simulated are

presented on Table 5.2.

R-value	Drain Defect	Source Defect
1kΩ	Not Detected	Not Detected
7.5kΩ	Not Detected	Not Detected
8kΩ	Not Detected	Detected
16.4kΩ	Not Detected	Detected
16.5kΩ	Detected	Detected
100kΩ	Detected	Detected
500kΩ	Detected	Detected
1MΩ	Detected	Detected

Table 5.2: Defect Coverage for DRF Using WWTM DFT

5.2.2 Implementation and Evaluation of NWRTM DFT Technique

This DFT technique introduces an additional write cycle known as a No Write Recovery (*NWR0, NWR1*) cycle which is performed before a normal write operation. The purpose of this cycle is to discharge the bitlines to a strong GND and Weak GND respectively. This is achieved by turning on the respective NWR signal while activating the wordline. This operation ensures that the bitlines are discharged before normal write operation begins. For the implementation of this technique two NMOS transistors are connected to one of the bitlines of the cell. These transistors are turned on by their respective signals, (*NWR0, NWR1*) depending on what data value is in the cell. The test as explained in Chapter 2.3.5 is performed as follows

- Write a background 0(1) to the cell.
- Enable NWRTM
- Bitlines BL and BLb are set to a weak GND and strong GND respectively. The weak GND is a voltage level driven to GND but not driven by any force. This is achieved by activation the corresponding NWR signal while the wordline is turned on. At the end of this cycle the bitlines are not precharged back to V_{dd} as is the case with a normal write operation.
- Begin a normal write 1(0) operation.
- Perform a read to detect if faulty cells flipped (Since the voltage of the true node never exceeds that of the false for defective cells).

This DFT techniques was first of all implemented on a defect free cell to illustrate the behavior of the nodes of the cell. The results of the simulation are shown on Figure 5.7.

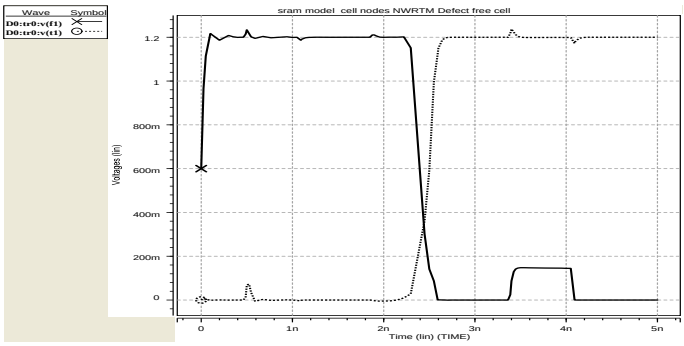


Figure 5.7: Simulation results for NWRTM DFT Implemented on a Defect Free Cell

We can observe from the waveforms that during the $NWR0$ ($NWR0$ cycle since the cell was assumed to value in the cell was 0) the false node of the cell is pulled up to V_{dd} while the true node is pulled down to GND. During the normal write operation the false node is pulled down to GND while the true becomes V_{dd} respectively thus causing the cell to flip. A subsequent read operation performed on the cell produces the write result.

Different R-values ranging from 1kΩ to 10MΩ were added to the defect free SRAM model. Simulations were performed separately for the drain and source. Results obtained for a defective drain are presented on Figure 5.8. The results presented are only for R-values of 1.7MΩ, 1.75MΩ and 10MΩ. As we can observe the NWRTM does not detect R-defects of $\leq 1.7M\Omega$ as the cell fails to flip when the wordline are turned on during a subsequent read operation.

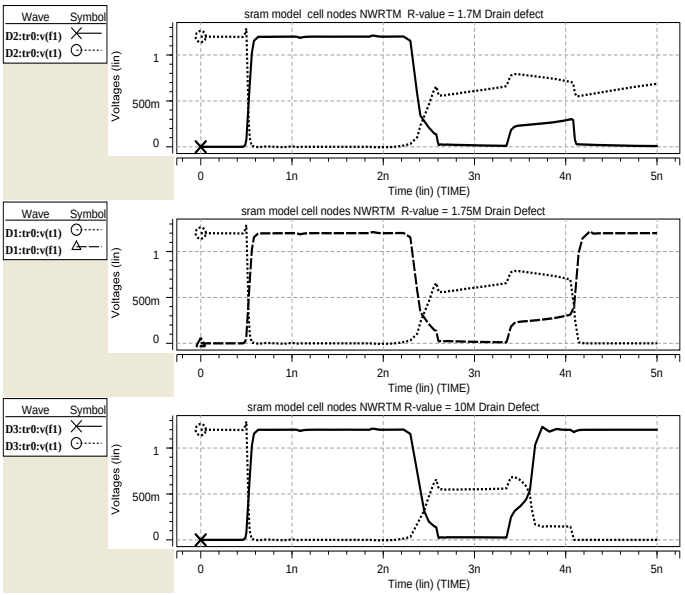


Figure 5.8: Simulation results for NWRTM implemented on defective Drain

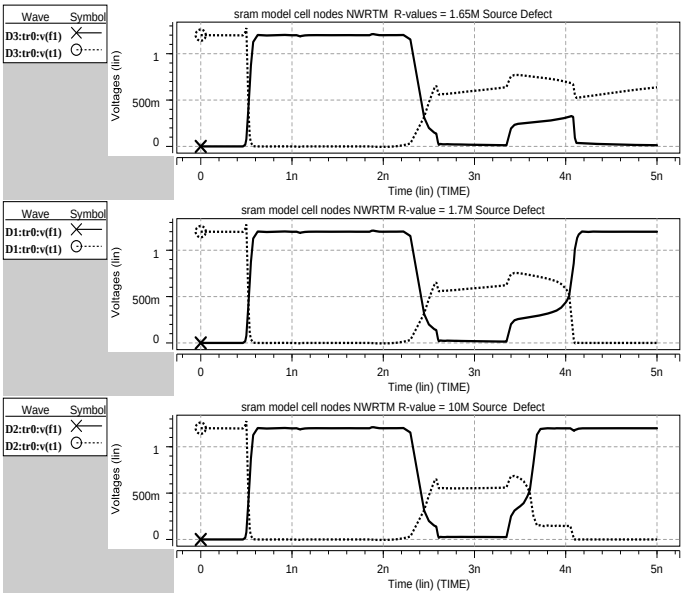


Figure 5.9: Simulation results for NWRTM implemented on defective drain

Similar results are obtained for a defective source with the little difference in that only R-values of $\leq 1.65\Omega$ are not detected. The waveforms obtained are presented in Figure 5.9

A summary of the results obtained for different R-values simulated are presented in Table 5.3. From the results presented in the Table we can conclude that the NWRTM DFT technique is suitable for the detection of large R-defect but unsuitable for smaller R-defects. It is however preferable in terms of test time to the WWTM in the case where only large R-defects are consid-

R-value	Drain Defect	Source Defect
1kΩ	Not Detected	Not Detected
100kΩ	Not Detected	Not Detected
1MΩ	Not Detected	Not Detected
1.65MΩ	Not Detected	Detected
1.7MkΩ	Not Detected	Detected
2MΩ	Detected	Detected
10MΩ	Detected	Detected
1GΩ	Detected	Detected

Table 5.3: Defect Coverage for DRF Using NWRTM DFT

ered detrimental to the chip. It is should be noted that the NWRTM technique can be merged into any march test since the write operation mechanism of this technique is the same as that used in march tests.

5.2.3 Implementation of PDWTM DFT Technique

The PDWTM aims at reducing the extra cycle which is added by other DFT techniques for the detection of DRFs. The main idea of the PDWTM is to perform a pre-discharge write instead of a precharge write as is always the case. This means that the write circle begins with both bitline set to GND. This is achieved using two NMOS devices connected to the bitlines and controlled by two signal *NW0, NW1*. These two signals are not activated independently depending on the initial data value present in the memory cell. The write cycle thus begins with the activation of the corresponding signal and the appropraite bitline being discharge to GND. This operation replaces the normal precharge cycle which is performed before the wordlines are turned on. The test as presented in Chapter 2.3.4. is performed as follows:

- Initialise memory with particular value say 0(1).
- Predischarge bitlines to ground (GND) and perform a write 1(0) operation.
- Precharge bitlines to high and perform a read operation.
- Weak cell are flip while good cell retain their value.

Simulations were performed separately for the drain and source. The results obtained so far indicated that this DFT does not detect opens which might result in a DRF. Some of the results obtained are presented in Figure 5.10

For further verification and validation ot the PDWTM DFt technique simulations still need to be performed .

A comparison of the above implemented DFTs will be presented in the next section.

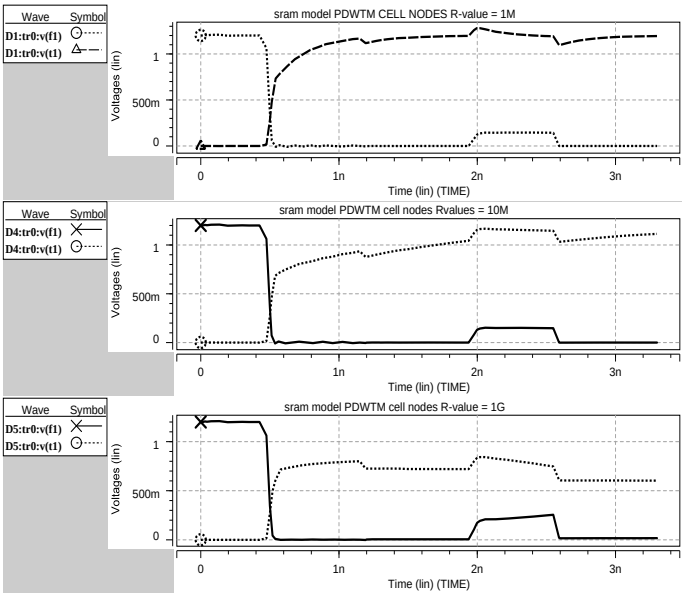


Figure 5.10: Simulation results for PDWTM

5.3 Comparison between Pause Test, WRTM, NWRTM and PDWTM Technique

From the above simulations carried out for detection of DRFs using Pause test, WRTM, NWRTM and PDWTM DFT techniques, a comparison between these techniques is presented. The comparison will be in term of the following

- Test time measure in terms of cell flip time.
- Extra Cycles
- Hardware design efforts measured in terms of how many transistors were added to the cell and design modifications to the memory cell array.
- Effect on the Performance of the memory.
- Power dissipation that is how much power is dissipated by circuit with additional DFT to that of s cell without.

These comparison are presented in Table 5.1 below. We observed from the Table that two of the presented DFT techniques present a mark improvement in test time. This test time reduction is close to 100% as compared to the Pause test in the detection of opens that result in a DRF. The NWRTM DFT technique however presents a better improvement in term of time, since it needs just an extra cycle lasting for approximately 1.5ns atmost to detect the fault. This is however true only for R-values $\geq 1.7N\Omega$. For smaller R-values the WWTM DFT is considered to be most appropriate. The PDWTM DFT technique was proven so far from the simulations performed not to detect opens at the PMOS pull-up device. We can however not conclude so far if these results are valid thus more simulations needs to be performed.

These DFTs add some additional hardware to the cell array. Six transistors attached to the bitlines in the case of WWTM and 2 transistors in the NWRTM and PDWTM. For the WWTM these transistors are designed as small in size as that of the cell thus minimising area and maintaining the stability properties needed by the memory cell. For the NWRTM and PDWTM DFTs the two transistors added are the same in size as the precharge circuit so will require just as much area as the precharge circuit. These three DFTs do not affect the performance of the memory cell since the normal memory operations are performed normally after these DFTs are deactivated.

The amount of power dissipated was negligible. As obtained from the log file, 666.1498nwatts of power was dissipated from a memory cell in normal mode. For a memory cell in test mode with these DFTs, the power dissipated was 672.7930nwatts for WWTM, 87.5017uwatts with the NWRTM DFT, while that of PDWTM is 768.3585nwatts . These values indicate the total amount of voltage source power needed in driving the controlling signals.

Method	**Test Time**	**Number of Cycles**	**Design modification**	**Performance**	**Power Dissipation**
WWTM	50ns long *WR0* cycle	W0, WR0, WR1, R1	Six transistors added to bitlines	No impact on performance	Negligible
NWRTM	1.5ns needed for the *NWR0,NWR1* cycle	NWR0, W1, R1	Two transistors added to Bitlines	No impact on performance	Negligible
PDWTM	No additional cycle needed	W1, R1predischarge 0	Two transistors added to bitlines	No impact on performance	Pwer is dissipated in μ watts, more power dissipated
Pause Test	Long pause time more than 12s for R-values $\leq 10M\Omega$	No area penalty	W1, delay, R1	No impact on performance	No difference with normal functioning memory cell

Table 5.4: Comparison of WWRTM, NWRTM, PDWTM and Pause Test

Comparing WWTM ,NWRTM and PDWTM with the pause test was mainly to indicate the difference in test time. For the other parameters the pause test is not a good standard to use in the comparison. This is why it was necessary to evaluate at least two DFT techniques so as to be able to get a broader view on the importance of DFTs in the detection of DRFs. Also this comparison can be used by designers and test engineers to determine which of the DFTs will be most preferable. Adequate conclusion on which of the DFTs presented in Chapter 2 can not however be drawn since all of these DFTs were not implemented and evaluated.

Summary

In this chapter we have implemented and evaluated three DFT techniques for the detection of DRFs. From simulation results obtained we observed that two of these techniques greatly reduce test time as compared to the traditional functional test. These DFTs do not require much design effort. The amount of power dissipated by the SRAM circuit with these additional hardware is not much although the NWRTM DFT dissipated more power than the WWTM and PDWTM. From these simulations we can conclude that DFTs are better test solutions in the detection of DRFs.

Conclusions and Recommendations

6

6.1 Conclusion

In this thesis the importance of DFTs in the detection of DRFs in embedded SRAMs have been presented. To illustrate their importance an accurate SRAM simulation model has been build. This model was used in the implementation of some existing DFT techniques. The proposed SRAM simulation model includes all peripheral circuits but for the timing generation circuit and the data input and output latch. The advantage of this simulation model is that the model can be used for any simulation purpose concerning memory faults in SRAMs.

The addition of the address decoder logic was due to the fact that, the address decoder is necessary when multiple cells has to be accessed in parallel by selecting the appropriate wordlines. This model thus presents a complete analog behavior of an SRAM circuit. However the additional address decoder logic also increases the delay of the SRAM to a greater extend.

The evaluation of the WWTM, NWRTM and PDWTM DFT techniques have also provided some data as to how efficient these DFT techniques are thereby easing some design decision as to which DFTs can used in the detecting DRFs in the cell array. Since two of these DFT were proven to detect symmetric and asymmetric faults within a reduced test time, they could be used as standards for the evaluation of other exiting and new DFT techniques. The PDWTM DFT was not validated. However we could not based on the obtained results to draw conclusions. Futher simulations should be carried out.

Functional tests most specifically the Pause test was shown to be an inefficient test method in the detection of DFRs since the pause time is very long and not known for small R-values. Using DFTs thus reduces the pause time to a greater extend. DFTs are the most preferred test methods for the detection of DRFs. However the efficiency of any DFT for the detection of DRFs can only be justified if it can detect both symmetric and asymmetric defects, as was the case with two of the DFTs evaluated.

6.2 Recommendations

During the implementation of these DFT techniques it was observed that R-values as small as $15k\Omega$ at the drain and $6k\Omega$ at the source were not detected. Some issues are raised from this observation, which were worth investigated. They are:

- Defects of size $16k\Omega$ and below at the drain of the pull-up PMOS device do not result in DRF or hard to detect defects and needs improve DFT techniques.
- Defects of sizes $\leq 7.5k\Omega$ at the source of the pull-up device might not result in DRFs or are hard to detect faults.
- The strength of a defect that can result to a DRF at the source and drain of the PMOS pull-up device still needs to be determined.

For defects at the source and drain the WWTM DFT was seen to be a better technique as this technique was able to detect R-values of as small as $7k\Omega$.

DFTs should also be considered as test solution to other peripheral faults such as weak sense amplifiers and write drivers. For address decoder delay faults little research has been done in that area. The problem on delay on the address decoder inputs can cause access to wrong cells. Using functional test can be time consuming as the decoder logic consists of many lines. DFTs could be a better solution. During the simulations performed in this thesis it was observed fromt he log file produced by HSPICE that the time interval during which the wordline is turned on by a normal functioning address decoder can be determine. Knowing this time interval it can be possible to put some additional logic like an AND gate to the wordline. This AND gate which will be controlled by an external signal with a set voltage can be monitored to investigate if the wordline is turned on at the appropriate time or not. However to validate such a proposal many parameters had to be defined. This was not done during this research due to time constraints.

Appendix A

HSPICE Input file for SRAM Simulation Model

```
SRAM MODEL
.global n2 n3
VDD vdd 0 1.2V
Xcell1 T1 F1 BL1 BL1cmp WL0 vdd cell
Xcell2 T2 F2 BL2 BL2cmp WL0 vdd cell
Xcell3 T3 F3 BL3 BL3cmp WL0 vdd cell
Xcell4 T4 F4 BL1 BL1cmp WL1 vdd cell
Xcell5 T5 F5 BL2 BL2cmp WL1 vdd cell
Xcell6 T6 F6 BL3 BL3cmp WL1 vdd cell
Xcell7 T7 F7 BL1 BL1cmp WL2 vdd cell
Xcell8 T8 F8 BL2 BL2cmp WL2 vdd cell
Xcell9 T9 F9 BL3 BL3cmp WL2 vdd cell
Xrowdecoder add1 add2 rdenable WL0 wl1 wl2 vdd rowdecoder
Xsa DL1 DL1cmp SAenable vdd senseamp
XSAn0t1 SOUT DL1cmp vdd SAout
XSAn0t2 SOUTcmp DL1 vdd SAout
Xcolondecoder add1 add2 cdenable cdout0cmp cdout1cmp cdout2cmp vdd colondecoder
Xwd datain vdd writedriver
Xwm1 wmux0 BL1 BL1cmp cdout0cmp We vdd writemux
Xwm2 wmux1 BL2 BL2cmp cdout1cmp We vdd writemux
Xwm3 wmux2 BL3 BL3cmp cdout2cmp We vdd writemux
Xrm1 Rmux0cmp BL1 BL1cmp DL1 DL1cmp RE cdout0cmp vdd rmux
Xrm2 Rmux1cmp BL2 BL2cmp DL1 DL1cmp RE cdout1cmp vdd rmux
Xrm3 Rmux2cmp BL3 BL3cmp DL1 DL1cmp RE cdout2cmp vdd rmux
XBLprechBL1 BLprechcmp BL1 BL1cmp vdd precharge
XBLprechBL2 BLprechcmp BL2 BL2cmp vdd precharge
XBLprechBL3 BLprechcmp BL3 BL3cmp vdd precharge
XSAprechDL SAprechcmp DL1 DL1cmp vdd precharge2
************************************************
********** Cell Array **********************
************************************************
.SUBCKT cell T F BL BLcmp WL vdd
MdownT T F 0 0 CMOSN54 L=0.1U W=0.5U
MupT T F vdd vdd CMOSP54 L=0.1u W=0.2u
MdownF F T 0 0 CMOSN54 L=0.1U W=0.5U
MUPF F T vdd vdd CMOSP54 L=0.1U W=0.2U
MpassT BL WL T 0 CMOSN54 L=0.1U W=0.3U
MpassF BLcmp WL F 0 CMOSN54 L=0.1U W=0.3U
.ENDS
************************************************
********** Inverter ************************
************************************************
.SUBCKT inverter in out vdd
M1 out in 0 0 CMOSN54 L=0.1U W=0.4U
```

```
M2 out in vdd vdd CMOSP54 L=0.1U W=0.4U
.ENDS
************************************************
********** NAND GATE**********************
************************************************
.SUBCKT NAND add1 add2 rdenable wl vdd
Mdup1 wl add1 vdd vdd CMOSP54 L=0.1U W=8U
Mdup2 wl add2 vdd vdd CMOSP54 L=0.1U W=8U
Mdup3 wl rdenable vdd vdd CMOSP54 L=0.1U W=8U
Mdown1 wl add1 n7 0 CMOSN54 L=0.1U W=8U
Mdown2 n7 add2 n8 0 CMOSN54 L=0.1U W=8U
Mdown3 n8 rdenable 0 0 CMOSN54 L=0.1U W=8U
.ends
************************************************
********** ROW DECODER **********************
************************************************
.SUBCKT rowdecoder add1 add2 rdenable WL0 wl1 wl2 vdd
Xnand1 add1cmp add2cmp rdenable out1 vdd NAND
Xnand2 add1 add2cmp rdenable out2 vdd NAND
Xnand3 add1cmp add2 rdenable out3 vdd NAND
Xinv1 add1 add1cmp vdd inverter
Xinv2 add2 add2cmp vdd inverter
Xinv4 out1 wl0 vdd inverter
Xinv5 out2 wl1 vdd inverter
Xinv6 out3 wl2 vdd inverter
.ends
************************************************
********** colomn DECODER **********************
************************************************
.SUBCKT colondecoder add1 add2 cdenable cdout0cmp cdout1cmp cdout2cmp vdd
Xnand1 add1cmp add2cmp cdenable cdout0 vdd nand
Xnand2 add1 add2cmp cdenable cdout1 vdd nand
Xnand3 add1cmp add2 cdenable cdout2 vdd nand
Xinv7 add1 add1cmp vdd inverter
Xinv8 add2 add2cmp vdd inverter
Xinv9 cdout0 cdout0cmp vdd inverter
Xinv11 cdout1 cdout1cmp vdd inverter
Xinv12 cdout2 cdout2cmp vdd inverter
.ends
************************************************
********** And **********************
************************************************
.SUBCKT AND rmuxcmp we wmux vdd
MAUP1 WMUXcmp RMUXcmp vdd vdd CMOSP54 L=0.1U W=8U
MAUP2 wmuxcmp we vdd vdd CMOSP54 L=0.1U W=8U
Mdown1 wmuxcmp rmuxcmp n9 0 CMOSN54 L=0.1U W=4U
Mdown2 n9 we 0 0 CMOSN54 L=0.1U W=4U
Xinv16 wmuxcmp wmux vdd inverter
.ENDs
```

```
**************************************************
************ Write Driver **********************
**************************************************
.SUBCKT writedriver Datain vdd
MwdLT N2 Dataincmp 0 0 CMOSN54 L=0.1U W=12U
MwdLF N3 Datain 0 0 CMOSN54 L=0.1U W=12U
Xinv Datain Dataincmp vdd inverter
.ENDS
**************************************************
************ Write Mux **********************
**************************************************
.SUBCKT writemux wmux BL BLcmp cdoutcmp we vdd
Mn1 BL wmux N2 0 CMOSN54 L=0.1U W=4U
Mn2 BLcmp wmux N3 0 CMOSN54 L=0.1U W=4U
XAND cdoutcmp WE wmux vdd AND
.ENDs
**************************************************
************ NAND GATE**********************
**************************************************
.SUBCKT NAND2 nandin1 nandin2 nandout vdd
Mdup1 nandout nandin1 vdd vdd CMOSP54 L=0.1U W=8U
Mdup2 nandout nandin2 vdd vdd CMOSP54 L=0.1U W=8U
Mdown1 nandout nandin1 n4 0 CMOSN54 L=0.1U W=8U
Mdown2 n4 nandin2 0 0 CMOSN54 L=0.1U W=8U
.ends
**************************************************
************ Read Mux **********************
**************************************************
.SUBCKT rmux Rmuxcmp BL BLcmp DL1 DL1cmp RE cdoutcmp vdd
Mp1 BL Rmuxcmp DL1 vdd CMOSP54 L=0.1U W=6U
Mp2 BLcmp Rmuxcmp DL1cmp vdd CMOSP54 L=0.1U W=6U
Xnand2 RE cdoutcmp Rmuxcmp vdd nand2
.ENDS
**************************************************
************ Sense Amplifier **********************
**************************************************
.SUBCKT senseamp DL1 DL1cmp SAenable vdd
MupT DL1 DL1cmp vdd vdd CMOSP54 L=0.1U W=0.5U
MdownT DL1 DL1cmp N1 0 CMOSN54 L=0.1U W=2U
MupF DL1cmp DL1 vdd vdd CMOSP54 L=0.1U W=0.5U
MdownF DL1cmp DL1 N1 0 CMOSN54 L=0.1U W=2U
MSAenable N1 SAenable 0 0 CMOSN54 L=0.1U W=2U
.ENDS
**************************************************
************ NOT GATE **********************
**************************************************
.SUBCKT SAout SOUT DL1cmp vdd
M1 SOUT DL1cmp vdd vdd CMOSP54 L=0.1U W=0.4U
M2 SOUT DL1cmp 0 0 CMOSN54 L=0.1U W=0.4U
```

```
.ENDS
************************************************
********** Sense Amplifier precharge ************
************************************************
.SUBCKT precharge2 SAprechcmp DL1 DL1cmp vdd
Mprech1 N1 SAprechcmp vdd vdd CMOSP54 L=0.1U W=4.8U
Mprech2 DL1 SAprechcmp vdd vdd CMOSP54 L=0.1U W=4.8U
Mprech3 DL1cmp SAprechcmp vdd vdd CMOSP54 L=0.1U W=4.8U
Mprech4 DL1 SAprechcmp DL1cmp vdd CMOSP54 L=0.1U W=4.8U
.ENDS
************************************************
********** Bitline Precharge *********************
************************************************
.SUBCKT precharge BLprechcmp BL BLcmp vdd
Mpr1 BL BLprechcmp vdd vdd CMOSP54 L=0.1U W=12U
Mpr2 BLcmp BLprechcmp vdd vdd CMOSP54 L=0.1U W=12U
Mpr3 BL BLprechcmp BLcmp vdd CMOSP54 L=0.1U W=12U
.ENDS
Cbl1 BL1 0 0.5pF
CBL1cmp BL1cmp 0 0.5pF
Cbl2 BL2 0 0.5pF
CBL2cmp BL2cmp 0 0.5pF
Cbl3 BL3 0 0.5pF
CBL3cmp BL3cmp 0 0.5pF
Cdl1 DL1 0 0.1pF
CDL1cmp DL1cmp 0 0.1pF
VBLprechcmp BLprechcmp 0 PWL(0ns 0v 0.3ns 0v 0.4ns 1.2v 1.2ns 1.2v 1.3ns 0v
+1.8ns 0v 1.9ns 1.2v 2.7ns 1.2v 2.8ns 0v
+3.3ns 0v 3.4ns 1.2v 4.2ns 1.2v 4.3ns 0v
+4.8ns 0v 4.9ns 1.2v 5.7ns 1.2v 5.8ns 0v
+6ns 0v )
VSAprechcmp SAprechcmp 0 pwl(0ns 1.2v 1.4ns 1.2v 1.5ns 0v 1.8ns 0v 1.9ns 1.2v 4.4ns
+1.2v 4.5ns 0v 4.8ns 0v 4.9ns 1.2v 5.8ns 1.2v 5.9ns 0v)
VWE WE 0 PWL(0ns 0v 0.3ns 0v 0.4ns 1.2v 1ns 1.2v 1.1ns
+0v 3ns 0v 3.3ns 0v 3.4ns 1.2v 4ns 1.2v 4.1ns
+0v 6ns 0v)
vSAenable SAenable 0 pwl(0ns 0v 1.5ns 0v 2ns 0v 2.4ns 0v 2.5ns
+1.2v 2.9ns 1.2v 3ns 0v 3.5ns 0v 4ns 0v 5.4ns 0v 5.5ns
+1.2v 5.9ns 1.2v 6ns 0v)
vdatain datain 0 PWL(0ns 1.2v 1.5ns 1.2v 3ns 1.2v 3.1ns 0v)
Vrdenable rdenable 0 PWL(0ns 0v 0.4ns 0v 0.5ns 1.2v 1ns 1.2v 1.1ns 0v 1.9ns 0v 2.0ns 1.2v
+2.5ns 1.2v 2.6ns 0v 3.4ns 0v 3.5ns 1.2v 4ns 1.2v 4.1ns 0v 4.9ns 0v 5.0ns 1.2v
+5.5ns 1.2v 5.6ns 0v 6ns 0v)
Vcdenable cdenable 0 PWL(0ns 0v 0.3ns 0v 0.4ns 1.2v 1ns 1.2v 1.1ns 0v 1.5ns 0v 1.8ns 0v 1.9ns
1.2v 2.5ns 1.2v 2.6ns 0v 3ns 0v
+3.3ns 0v 3.4ns 1.2v 4ns 1.2v 4.1ns 0v 4.5ns 0v 4.8ns 0v 4.9ns 1.2v 5.5ns 1.2v 5.6ns 0v 6ns 0v)
VRE RE 0 PWL(0ns 0v 0.3ns 0v 0.4ns 0v 1ns 0v 1.1ns
+0v 1.8ns 0v 1.9ns 1.2v 2.5ns 1.2v 2.6ns 0v 3ns 0 4.8ns 0v 4.9ns 1.2v 5.5ns 1.2v 5.6ns 0v 6ns 0v)
Vadd1 add1 0 0v
```

```
Vadd2 add2 0 0v
.IC V(T1)=0v
.IC V(F1)=1.2v
.IC V(T2)=0v
.IC V(F2)=1.2v
.IC V(T3)=0v
.IC V(F3)=1.2v
.IC V(T4)=0v
.IC V(F4)=1.2v
.IC V(T5)=0v
.IC V(F5)=1.2v
.IC V(T6)=0v
.IC V(F6)=1.2v
.IC V(T7)=0v
.IC V(F7)=1.2v
.IC V(T8)=0v
.IC V(F8)=1.2v
.IC V(T9)=0v
.IC V(F9)=1.2v
.IC V(BL1)=1.2v
.IC V(BL1cmp)=1.2v
.IC V(BL2)=1.2v
.IC V(BL2cmp)=1.2v
.IC V(BL3)=1.2v
.IC V(BL3cmp)=1.2v
.IC V(DL1)=1.2v
.IC V(DL1cmp)=1.2v
.IC v(add1) = 0v
.IC V(add2) = 0v
.op all
.options fast
.options reltol = 0.005
.option post=2 NOMOD
.TRAN 0.5ns 6.1ns 0 0.1ns
```

The MOSFET parameters which were used are not included in this Appendix since the data was much and were generated from predictive technology [26]

Bibliography

[1] L.Dilillo, P.Girard, S.Pravossoudovitch, A.Virazel, M. B.Hage-Hassan, "Data Retention Fault in SRAM Memories: Analysis and Detection", *Procedures, Proceedings of the 23rd IEEE VLSI Test Symposium (VTS 05)*, May 01-05, 2005, p.183-188

[2] Semiconductor Industry Association (SIA), "International Technology Roadmap for Semiconductors (ITRS)", 2005 Edition.

[3] A. J. van de Goor, *Testing semiconductor memories:* theory and practice, John Wiley and Sons, Inc., New York, NY, 1991

[4] R. Dekker, F. Beenher, and L. Thijssen, "Fault modeling and test algorithm development for static t random access memory, *" in Proc. Int. Test Conf.*, 1988, pp. 343-352.

[5] A. Meixner and J. Banik, "Weak Write Test Mode: An SRAM Cell Stability Design for Test Technique", *International Test Conference*, 1996, pp. 309-318.

[6] K. Zarrineh et al., "Defect Analysis and Realistic Fault Model Extensions for Static Random Access Memories", *Records IEEE Int. Workshop on Memory, Technology, Designand Testing*, 2000, pp. 119-124.

[7] S. Chakravarty and P. J. Thadikaran.*" An introduction to IDDq testing"*, Kluwer Academic Publishers, Boston, MA, 1997

[8] Herold Pilo , R. Dean Adams , Robert E. Busch , Eric A. Nelson , Geoerge E. Rudgers, "Bitline contacts in high density SRAMs: design for testability and stressability", *Proceedings of the IEEE International Test Conference 2001*, October 30-November 01, 2001, p.183-188

[9] J. Castillejos and V.H. Campac," A Force-Volatage Technique to Test Data Retention Faults in CMOS SRAM by IDDQ Testing", in *Proc. Int. Symp. Circuits and Systems* , 1997, pp.433-436.

[10] Pavlov, A. Azimane, M.de Gyvez, J.P. Sachdev, M. "Wordline pulsing technique for stability fault detection in SRAM cell", *Proceedings of IEEE International*, 2005.

[11] Josh Yang; Baosheng Wang; Yuejian Wu; Ivanov, A. "Fast detection of data retention faults and other SRAM cell open defects", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Volume 25, Issue 1, Jan. 2006 pp.167 - 180

[12] Josh Yang , Baosheng Wang , Andr Ivanov, "Open Defects Detection within 6T SRAM Cells using a No Write Recovery Test Mode", *Proceedings of the 17th International Conference on VLSI Design*, January 05-09, 2004, p.493

[13] Champac, V.H., Avendano, V., and Linares, M.: "A bitline sensing strategy to test data retention faults in CMOS SRAMs", *Electron. Lett.*,2000, 36, (14), pp. 1182-1183

[14] Y. Katayama, E. Stuckey, S. Morioka, and Z. Wu, "Fault-tolerant refresh power reduction of DRAMs for quasi nonvolatile data retention", in *Proc. Int. Symp. Defect and Fault Tolerance VLSI Systems*, 1999, pp. 311-318

[15] A. Pavlov, M. Sachdev, and J. P. de Gyvez,"An SRAM weak cell fault model and a DFT techniquewith a programmable detection threshold", in *Proc.IEEE International Test Conference (ITC)*, Oct. 2004, pp. 1106-1115.

[16] Semiconductor Industry Association (SIA), "International Technology Roadmap for Semiconductors (ITRS)", 2003 Edition.

[17] Kuo, C., Toms, T., Neel, B.T., Jelemensky, J., Carter, E., and Smith, P.: "Soft-defect detection (SDD) technique for a high-reliability CMOS SRAM", *IEEE J. Solid-State Circuits*, 1990, 1, pp. 61-66

[18] D. L. Liu and E. J. McCuskey, "CMOS scan-path IC design for stuck-open fault testability", IEEE J. Solid-Stute Circuits, vol. SC-22, no. 5, Oct. 1987, pp. 880-885.

[19] V. H. Champac, J. Castillejos, and J. Figueras, "Iddq Testing of Opens in CMOS SRAMs", *Proc. Of VLSI Test Symposium*, April 1998, pp. 26-30. Gil P Gillingham, *"BIST memory test system"*, US Patent 6,182,257, 2001

[20] S. Hamdioui, "Presentation on Test and PPM Reduction for Embedded Memories for SOCs", Atmel Cooperation, Dec,06.

[21] K. Zarrineh and S. J. Upadhyaya, "On Programmable Memory Built-In Self Test Architectures", *Design, Automation and Test in Europe*, ,1999, pp. 708-7 13.

[22] W.A.M. Van Noije, W.T. Liu, and J. Navarro S. Jr., "Metastability Behavior of Mismatched CMOS Flip-Flops using State diagram Analysis", *IEEE Custom Integrated Circuits Conf.*, 1993, pp. 27.7.127.7.4.

[23] J.E. Simonse, *Circuits structures, design requirements and fault simulations for CMOS SRAMs*, Masters Thesis, Faculty of Information and Systems, Delft University of Technology, August 1998.

[24] M.M.Mano and C. R. Kime, *Logic and Computer Design Fundamentals*, Pince-Hall international limited, London, 1997.

[25] Berkeley Predictive Technology Model, UC Berkeley Device Group. http://www-device.eecs.berkeley.edu/ ptm/.

[26] Etienne Sicard, *"'Introducing 90 nm technoology in Microwind3"*, INSA-Dgei, 135 Av de Rangueil, 31077 Toulouse - France http://www.microwind.org

[27] Prince, *Semiconductor Memories*, 2nd ed. (Wiley, Chichester, UK, 1991).

[28] Joon - Sung Yang and Gahngsoon Moon, *VLSI-I Project Document*, Dec.9, 2005