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

2011

MSc THESIS

A New Test Paradigm for Semiconductor Memories in the Nano-Era

Venkataraman Krishnaswami

Abstract

Due to rapid and continuous technology scaling, faults in semiconductor memories (and ICs in general) are becoming pervasive and weak of nature; weak faults are faults that pass the test program (because they do not lead to erroneous behavior of the system). Nevertheless, they may cause a system failure during the application. This causes the number of escapes to increase while it becomes increasingly difficult to determine the nature of the failures. Components with weak faults which fail at board and system level are sent to suppliers, only to have them returned back as No Trouble Found (NTF). The conventional memory test approach assumes the presence of a single defect a time causing a strong fault (which leads to an error in the system), and therefore is unable to deal with weak faults.

This thesis presents a new memory test approach able to detect weak faults; it is based on assuming the presence of multiple weak faults at a time in a memory system rather a single strong fault at a time. Being able to detect weak faults reduces the number of escapes, hence also the number of NTFs. The experimental analysis done using SPICE simulation for a case of study show e.g., that when assuming two simultaneous weak faults, the missing (defect) coverage can be reduced with up to 10% as compared with the conventional approach.

CE-MS-2011-14

A New Test Paradigm for Semiconductor Memories in the Nano-Era

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

by

Venkataraman Krishnaswami
born in Coimbatore, India

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

A New Test Paradigm for Semiconductor Memories in the Nano-Era

by Venkataraman Krishnaswami

Abstract

Due to rapid and continuous technology scaling, faults in semiconductor memories (and ICs in general) are becoming pervasive and weak of nature; weak faults are faults that pass the test program (because they do not lead to erroneous behavior of the system). Nevertheless, they may cause a system failure during the application. This causes the number of escapes to increase while it becomes increasingly difficult to determine the nature of the failures. Components with weak faults which fail at board and system level are sent to suppliers, only to have them returned back as No Trouble Found (NTF). The conventional memory test approach assumes the presence of a single defect a time causing a strong fault (which leads to an error in the system), and therefore is unable to deal with weak faults.

This thesis presents a new memory test approach able to detect weak faults; it is based on assuming the presence of multiple weak faults at a time in a memory system rather a single strong fault at a time. Being able to detect weak faults reduces the number of escapes, hence also the number of NTFs. The experimental analysis done using SPICE simulation for a case of study show e.g., that when assuming two simultaneous weak faults, the missing (defect) coverage can be reduced with up to 10% as compared with the conventional approach.

Laboratory : Computer Engineering
Codenumber : CE-MS-2011-14

Committee Members :

Advisor:	Dr.ir. Said Hamdioui, CE, TU Delft
Chairperson:	Dr.ir. Koen Bertels, CE, TU Delft
Member:	Prof. Dr.ir. A.J.van de Goor, CE, TU Delft
Member:	Dr.ir. R. Ishihara, ECTM, TU Delft
Member:	ir. A.C.de.Graaf, CAS, TU Delft
Member:	Dr.ir. Z.Alars, CE, TU Delft

Contents

List of Figures

List of Tables

Acknowledgements

First of all, I would like to thank my advisor Dr. Said Hamdioui, for his guidance, support, patience and for encouraging me, throughout the course my thesis. I would further like to thank Sandra Irobi for her suggestions and feedback. I owe my deepest gratitude to Vishwas Raj Jain for his support, and the time he dedicated for fruitful and thought provoking discussions at every phase of my thesis. I am also grateful to Nor. Zaidi Hazron, Seyab Khan, Mottaqiallah Taouil and Nimitt Chaithanya Bhatt for spending their valuable time in giving me feedback for the documentation. I wish to convey my regards to all my friends and colleagues who have made my stay here a very happy and memorable one. I am very thankful to Mr. Pragash Chinappan, my high school teacher for all his efforts and encouragement over the years. I would like to specially thank my cousin Venkat Iyer, for all his support and encouragement he gave me over the years. Many thanks to my friends, Venkatasubramanian and Venkatesh Seshan for always keeping me motivated. Finally, I would like to record my heartfelt gratitude to all my teachers, parents, grandparents and family members for all their blessings, and unconditional support I have received from them over the years, without which this would not have been possible.

Venkataraman Krishnaswami
Delft, The Netherlands
September 12, 2011

1 Introduction

The downscaling of CMOS devices has resulted in the development of smaller and faster circuits, with better performance. However, it comes with the price of reduced robustness, thereby making it more vulnerable to defects. High number of defects reduces the quality and yield of manufactured chips. Therefore, it is very important to study these defects, model them and develop appropriate test algorithms to ensure the quality requirements. The yield of most system on chip today is determined by the memory yield; this is because memory occupies a major share of the area. Hence, it becomes very important to study new unmodeled faults, which could be dominant in semiconductor memories in the nano-era. This thesis describes one such study.

This chapter provides some basics about test technology and semiconductor memories; it also briefly describes the state of the art in memory testing and gives the main contribution of this thesis work. The chapter is organized as follows. Section 1.1 introduces the concept of testing and its importance. Section 1.2 covers briefly the semiconductor memories. Section 1.3 gives the state of the art in memory testing. Section 1.4 presents today's and future challenges in this field. Section 1.5 describes the contribution of this thesis work. Section 1.6 presents the outline of this thesis.

1.1 Testing and its importance

The semiconductor industry has been witnessing tremendous growth over the past few years. This growth has been driven by Moores law, which states that the number of transistors on a given chip doubles every two years [25]. This has led to the development of Ultra Large Scale Integrated (ULSI) circuits. ULSI circuits can be categorized into combinational and sequential circuits. Combinational circuits are purely based on computational logic, without involving memories. Sequential circuits are based on combinational logics with memories. The realization of such circuits in silicon as chips, involves a number of complex manufacturing processes. Imperfections in these manufacturing processes means that the functionality of these chips is not always guaranteed. This is because the chips are prone to manufacturing defects. Defects can be caused due to presence of impurities, dislocations, unwanted (extra) components or absence of required components. These defects are generally modeled as opens, shorts or bridges. In the real world, defects are almost unavoidable, but can be minimized [9]. Defect Per Million (DPM) gives the number of defective pieces of a chip; for every million chips, the customer is willing to accept. This is often used as a benchmark for quality specifications. In order to meet the quality specifications provided by the customer, it becomes very important to adopt good test strategies to ensure proper functioning of the chips.

Fig 1.1 shows the testing methodology for a chip under production. In producing a chip, two types of testing are required, namely; verification testing and manufacturing testing. Testing performed to verify the functionality of the design (to be implemented in silicon) immediately after design stage is known as *verification testing*. The testing which is performed after manufacturing stage to detect faulty behavior due to defects is known as *manufacturing testing*. Manufacturing testing is mainly done to meet the quality requirements provided by the customer. There are also two main important aspects related to testing, namely *detection* and *diagnosis*. Detection deals with obtaining the pass/fail information of a chip with regard to its functionality. Diagnosis deals with the localization of the fault in a given faulty chip. The information obtained from diagnosis is used as a feedback to improve the existing designs and manufacturing processes.

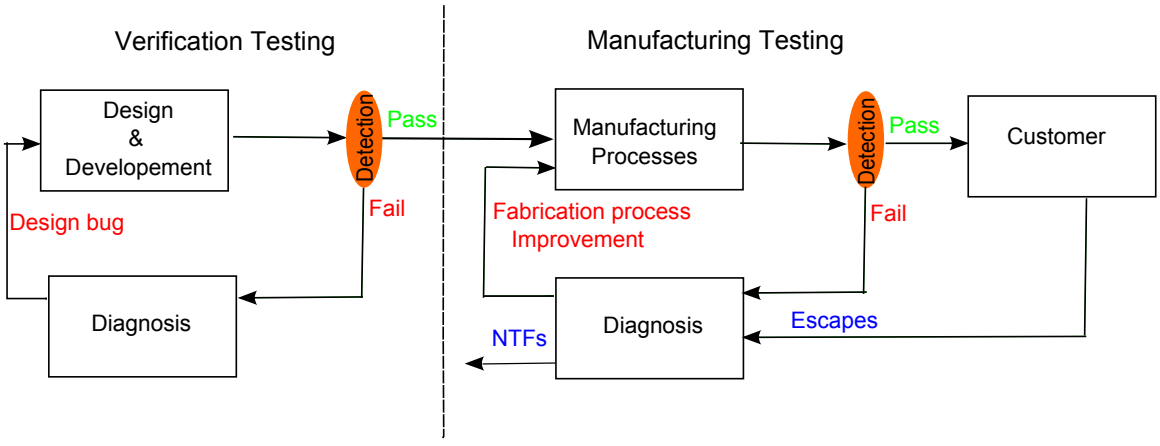


Figure 1.1: Testing and diagnosis principle

The detection of faults is achieved by applying a set of known input patterns, to obtain a certain set of responses, which is then compared with the set of expected responses to obtain pass/fail information. This means that the chip has to be tested for every possible combination of input test vectors to ensure that the chip is fully functional. This type of testing approach is known as *functional testing*. This is more useful for 'verification testing'. But such an approach may not be useful for 'manufacturing testing' purposes owing to large test times, especially when it comes to complex chips with large number of input test vectors. Test strategies in industries are developed to ensure that the testing process is completed within a given time frame. As a result, functional testing approach is not used for manufacturing testing purposes. Test algorithms that are developed based on fault models are used for this purpose. This type of testing approach is known as *structural testing*. In structural testing, the detection of number of faulty chips, is based on the effectiveness of the test algorithms and fault models that are used. It is often seen that even though the chip is faulty, it escapes the test program only to have them returned back by the customer. Hence, testing is a very critical step in the whole design and manufacturing chain; not only because it has to screen out all the defective chips before they are sold, but also because it is the last chance to deliver the required quality and reliability to the end customer. It constitutes a major part of the manufacturing costs of todays products, especially in critical applications such as the automotive, health care, security and aerospace sectors [31].

Progressive technology scaling, as tracked by the International Technology Roadmap for Semiconductors (ITRS) and encapsulated by Moores law [25], has driven the phenomenal success of the semiconductor industry. Silicon technology has now entered the nano-era and the 10nm transistors are expected to be in production by 2018. This will allow for the integration of a wider variety of functions. However, it is widely recognized that variability in device characteristics and its impact on the overall quality and reliability of the system represent major challenges to scaling and integration for present and future nanotechnology generations [2] [3]. What is more, newly emerging complex failure mechanisms in the nano-era (which are not understood yet), are causing the fault mode of the chips to be dominated by transient, intermittent and weak faults rather than hard and permanent faults; hence causing more reliability problems than quality problems [4, 5, 17, 41]. Many companies are reporting not being able to explain all electronic failures with the existing approaches [24] [43, 28]. For instance, AUDI reported from all electronic failures, 35% can be mapped using the existing approaches into well defined semiconductor failures, while 41% can not be understood (NTF: No Trouble Found) [24]. This shift in failure mechanisms is therefore seriously impacting the quality and reliability. This means that the design of future systems fabricated using nanotechnology is a major challenge. In turn, this will demand revolutionary changes in how future systems are designed and tested to meet the increasing quality requirements on such systems.

1.2 Semiconductor Memories

Most applications in the real world require sequential logic based designs. As mentioned earlier sequential logic involves memories. The study of semiconductor memories and their testing methods are crucial to ensure the proper functioning for a given application. A semiconductor memory is a device that can be used to store and retrieve a given data. Semiconductor memories are superior to other classes of memories in terms of performance, versatility, power consump-

tion etc. Semiconductor memories are classified into non-volatile and volatile memories, as shown in Fig 1.2. Nonvolatile memories are capable of storing or holding the information even in the absence of power supply. Memory elements like ROM, MRAM, flash memory etc., belong to this category. On the other hand, volatile memories require power supply to be switched ON to store or hold the information. Memory elements like SRAM, DRAM etc., belong to this category; SRAMs are mainly used as cache elements, whereas DRAMs are used as a main memory in a given computer system.

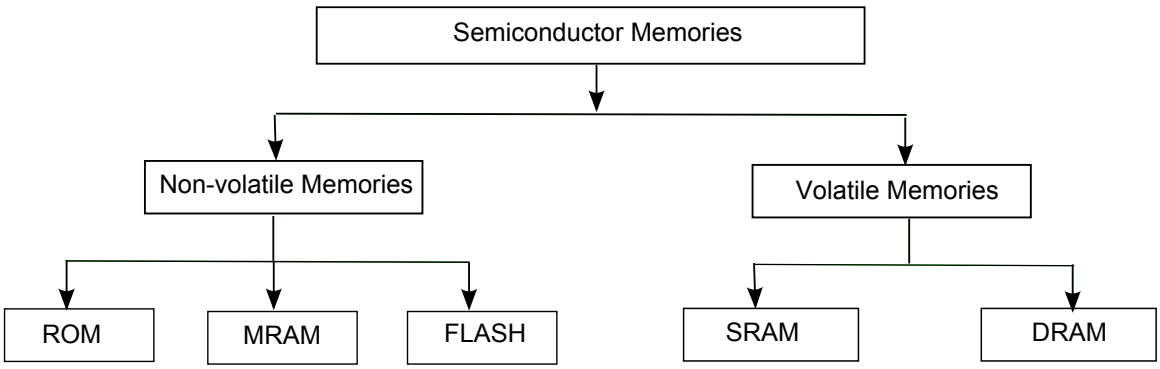


Figure 1.2: Semiconductor memories classification

The Static Random Access Memories (SRAM) are based on latches. Two cross coupled inverter configuration can be realized using MOS technology to form a single SRAM cell. Each cell is capable of storing 1 bit of information. This configuration has two stable states and is capable of storing either logic 1 or logic 0. Hence SRAM cell is referred as a bistable element. SRAM is taken as the memory under consideration in this thesis.

1.3 State of the art in memory testing

The ever growing demands of many applications require the use of more memory than computational logic. Hence, with downscaling memory tends to dominate the computational logic in terms of area on a given System on Chip (SoC). This means that the yield of SoC primarily depends on the memory yield. Thus, testing of memory is of paramount importance.

Tests for semiconductor memories have undergone a long development process. Before 1980, tests for a given fault coverage (FC) were time consuming. FC is defined as the number of detected faults divided by the number of total faults. The time complexity of these tests were of order $O(n^2)$ (where n is the size of memory)[8]. Such tests can be termed as the 'ad hoc' tests because they lack fault models and proofs. Tests like GALPAT, Zero-One test, walking 1/0 tests etc., belong to this class [40].

In order to reduce the test time and improve the FC, test development primarily focused on possible faulty behaviors in the memory at the functional level. For that reason, functional fault models, which are abstract fault models, were introduced during the early 1980s. The

advantage of these models was that the FC could be improved, while the test time was usually linear with the size of the memory [29, 10, 40]. Some of the important fault models introduced during this time were Stuck At Fault (SAF), Coupling Fault (CF), Address decoder Fault (AF) etc. [40]. However, it was later found that the fault models were not sufficient to achieve high fault coverage. This is because the fault models developed were abstract and were not based on real defects leading to faults. To reflect the faulty behavior of real defects in real designs, defect injection and circuit simulation, as well as inductive fault analysis (IFA) were developed and used [34]. This led to the development of newer fault models based on new failure mechanisms that were understood through IFA. This led to the introduction of fault models like the State Coupling Fault (CF_{st}), the Data Retention Fault (DRF), Stuck Open Fault (SOF), etc. March tests were then developed based on these fault models. March tests are a finite sequence of march elements. March elements are basically a series of read or write operations performed over the memory. March tests became very popular and dominant owing to the advantage of providing linear test times with respect to the size of the memory and their fault coverage can be proven mathematically [29, 10, 40].

In the late 1990s, industrial results indicated that many tests detect faults which could not be explained by the existing fault models at the time [13, 32, 38] mainly because of the used technology node. This raised many scientific questions regarding the defect mechanisms, fault models and tests used. This led to the introduction of new concepts, new fault models and new tests schemes [22]. These fault models mainly focused on timing related faults or dynamic faults.

1.4 Today's and future challenges

Nowadays, embedded memories represent the great majority of embedded electronics in Systems on Chip (SoC). It is very common to find SoCs with hundreds of memories representing more than 50% of the overall chips area. According to the ITRS, todays SoCs are moving from logic-dominant to memory-dominant chips in order to deal with application requirements of today and the future. Fig 1.3 shows how the dominant-logic is changing to memory, approaching 94% of the chip area in 2014. Consequently, embedded memory test challenges will significantly impact the overall testability of SoC [21]. Solving such challenges for memories will substantially contribute to the resolution of electronic system test problems in the future; hence, supporting the Fig 1.3. Share of embedded memories in systems on chip [12] continuation of the semiconductor technology revolution and the manufacturability of future highly complex systems (giga-scale) and highly integrated technologies (nano-scale).

Today, as the silicon industry moves towards the end of the CMOS technology roadmap, controlling the fabrication of scaled memory devices is becoming a major challenge. Device-parameter variations (e.g. threshold voltage), high defect density as well as new failure mechanisms in the nano-era are expected to be significantly larger in the future [3, 5, 6, 36]. This leads to major challenges in designing and testing memories in nano-era. Conventional fault models and test approaches are inadequate to realize the required product quality [5, 16, 17, 26, 28, 44], especially for critical applications like automotive, security, health care and aerospace sectors. In the absence of new theories capable of modeling their failure mechanism and developing appropriate test solutions, the production of future electronic systems will become infeasible.

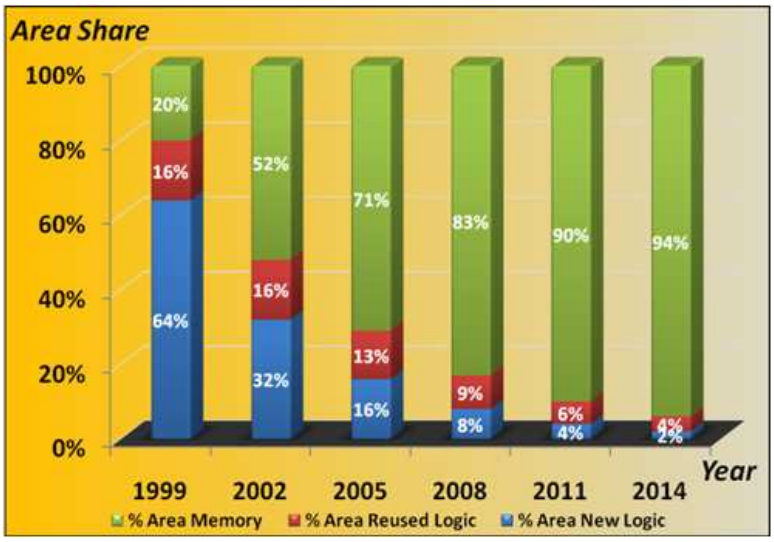


Figure 1.3: Future of embedded memories [12]

Given the state-of-the art in fault modeling and test generation for embedded memory, one can conclude the following:

- The physical defect mechanisms due to process variations and scaling in all components of a memory system have yet to be analyzed and understood; especially for 45nm CMOS and beyond.
- There are, as yet, no fault models to describe the above failure mechanisms. The models have to consider not only the presence of a single defect at a time (as it has been the case so far), but it has to consider also the presence of multiple weak-faults/defects simultaneously (this is particularly important in the nano-era). A weak fault is not able to fully sensitize a fault, but it partially sensitize a fault; e.g., due to a defect that creates a small disturbance of the voltage of the true node of the cell. However, a fault can be fully sensitized (i.e., becomes strong) when two (or more) weak faults are sensitized simultaneously in the memory systems since their fault effects can be additive.
- As new failure mechanisms manifest themselves in a different way than the conventional faults, new methods to be used to develop efficient test algorithms/solutions need to be developed in order to support the manufacturability of future memory technologies.

1.5 Thesis Contribution

This thesis aims to develop a new memory test approach being able to deal with complex faulty behaviors in the nano era. A novel test scheme is developed for the detection of erroneous behavior, due to weak faults. Being able to detect such faulty chips reduces the number of escapes, hence also the number of NTFs.

The contribution of this thesis work can be summarized as follows:

- Provides a theoretical foundation to deal with complex faulty behavior of embedded memories in the nano era.
- Presents a systematic approach to deal with detection of erroneous behavior due to multiple weak faults.
- Explains the failure mechanism caused by multiple weak faults.
- Introduces test approaches that are required for the detection of faults arising due to multiple weak faults.
- Provides a case study to validate the concept experimentally through SPICE simulations.
- Gives SPICE simulations to estimate the defect coverage improvements over the conventional approach.

1.6 Outline of thesis

The thesis is organized as follows. Chapter 1 introduces the concept of testing, along with the importance of memory testing. It also provides the state of the art in memory testing and highlights today's challenges and future needs, which motivates this work.

Chapter 2 describes the memory architecture, and explains the proper functioning of the memory. The basic architecture of SRAM is described in a top down manner, at different levels of abstraction, i.e. from behavioral to electrical models. The electrical model is later used as a base to analyze the behavior of memory with defects which lead to different kinds of faults.

Chapter 3 describes the memory fault space. The concept of fault primitive is introduced, and the classification of whole fault space involving static, dynamic and weak faults is discussed. Weak faults are analyzed in more detail in the chapter relating to new memory test paradigm.

Chapter 4 introduces the new memory test paradigm. The erroneous behavior due to presence of weak faults is discussed, by studying the effects of multiple (weak) defects present in different parts of memory system at the same time.

Chapter 5 gives the details of simulation model, and experimental validations to demonstrate the superiority and the effectiveness of the new test paradigm.

Chapter 6 provides the conclusions and future work.

2 Memory Architecture

As stated in chapter 1, it is very important to develop efficient test algorithms for the detection of faults that could arise in the memory system. This requires a good understanding of the structure of memories, along with their way of working. In order to achieve this, memory models, have been developed over the years, at different levels of abstractions i.e. from functional level to electrical level. Based on such an understanding, we can develop fault models and corresponding test algorithms.

This chapter describes the memory models at different levels of abstraction in an hierarchical approach. Section 2.1 introduces the idea behind memory modeling. Section 2.2 describes the behavioral memory model. Section 2.3 describes the functional memory model. Section 2.4 presents the electrical model of the memory system, which includes the memory cell array, address decoders and peripheral circuits. Section 2.5 briefly presents the memory process technology.

2.1 Memory modeling

Complex systems are difficult and hard to understand. Modeling offers simplification and proper structuring of an entity and its surroundings. For example, modeling a given system into a set of functional blocks as subsystems gives a better clarity about the structure and the operation of the main system. This is because subsystems are mostly homogeneous and therefore, are easier to understand. In case of a memory system, the model can be divided into three subsystems: memory cell array, address decoders and peripheral circuits. Memory modeling also holds the key to simplify the development of tests for physical faults occurring in the memory system. The testing for physical faults can be performed in two ways: physical inspection of the system, by examining their internal structure, or by comparing the logical behavior of a system, with that of the behavior of a known good system [40]. Since physical inspection of the system is almost impractical and unreliable, the main stream testing strategy often relies on the latter way, i.e., based on logical comparisons. To adopt such a testing strategy, physical faults must be mapped on to logical faults using fault models. This also makes the test approaches to become more independent to technology and manufacturing processes.

Modeling introduces a new level of abstraction for any given system. Each level of abstraction is called as model of a system. Models describe only the relevant information and hides all the other irrelevant information at any given abstraction level. The higher the level of abstraction, the simpler the models become. On the other hand, the lower the level of abstraction, the easier it becomes for fault localization. This is because lower level of abstraction relates closer to the physical or layout level where the actual defects or faults occur.

Fig 2.1 shows the different levels of abstraction for a given system in a top down manner. There are five main levels of abstraction:

- **Behavioral model**: It forms the highest level of abstraction. Most of the information relating to the implementation of the system and their internal details are hidden. The only information available in this level is the input and output signals, while treating the system as a black box.

- **Functional model**: It distinguishes the function the system needs to fulfill to perform properly. In this level, the system is divided into a number of subsystems, with each of them having its own set of functionalities.

- **Logical model**: It is based on logic gates, which are derived from simple boolean relations, governing the system's (or subsystem's) functionality.

- **Electrical model**: It is the electrical level schematic representation of the system using components like resistors, transistors etc. This model is mainly used for the purpose of fault modeling and test algorithm developments.

- **Physical or layout model**: It represents the lowest level of abstraction. This model describes the geometrical representation of the physical implementation of a given system.

Another important type of modeling is **mixed level** modeling. In this type of modeling, specific focus is given to the low level implementations, only for a given area of interest in a system, while the other areas are maintained at a higher abstraction level.

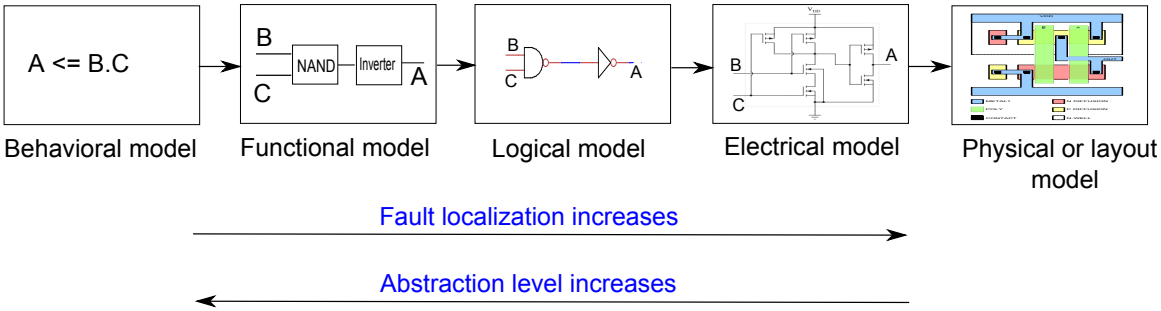


Figure 2.1: System modeling at different abstraction levels

In the following sections, the memory system is discussed at different abstraction levels.

2.2 Behavioral memory model

The behavioral model describes the memory as a black box, with just the input and output pins made visible. All the internal details of the memory system are hidden. This model helps in establishing the relation between the inputs and outputs of the system along with their associated timing information. All the internal components are clocked by black box. This model can be used to perform verification testing, to verify the functional behavior of the system [40]. Fig 2.2 shows the memory system as a black box with its corresponding input and output signals. The input signals mainly comprises; clock, an N-bit address line, M-bit data input line and C-bit control signals. The output signal consists of data output pins. Sufficient pins are also provided for ground and power supply. Clock is used for driving the inner components in the memory system. The N-bit address lines are used for accessing the memory, while the M-bit Data in lines are used to provide M-bit data to be stored. The M-bit Data out lines are often multiplexed with the Data in lines, to save the number of pins. The memory system is controlled by the set of control signals like chip select, write enable, read enable, output select etc. The chip select signal, allows the user to select or de-select the device when desired. The write and read enable signals, are used to enable write and read operations from the SRAM, while the output select signal controls the data output of the memory. The timing specifications for all these signals, are generally provided to the customer by the manufacturer of the chip.

The commercial commodity SRAM chips available in the market can be viewed as a black box, with just the input and output pins made visible. Fig 2.3 [33] shows a commercial SRAM chip. The A0-A18 pins are the address pins, while pins DQ0 to DQ7 are used to provide or access an 8-bit data. The $\overline{CE}$ is the chip enable signal. The chip is accessed when the signal is in active low state. The $\overline{WE}$ and $\overline{OE}$ is the write enable and output enable signals used for writing and reading the data from the chip. The chip is also provided with pins, connected to supply voltage and ground.

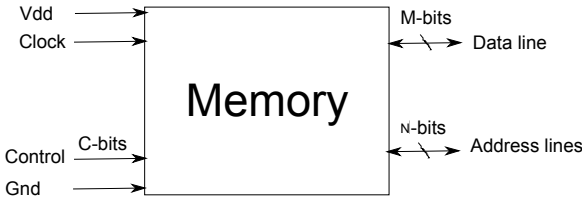


Figure 2.2: Behavioral model for a memory system

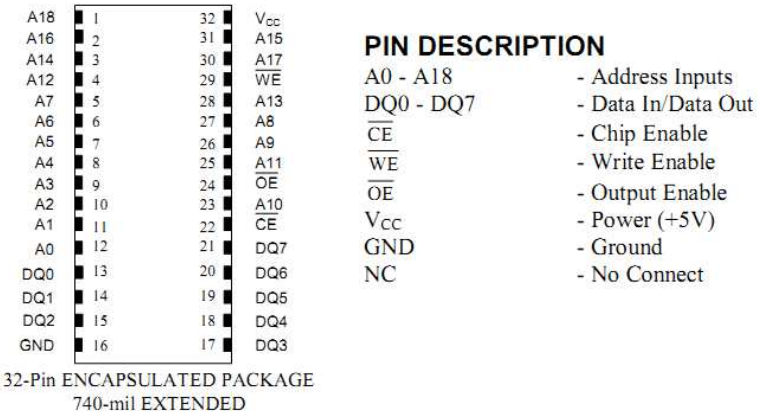


Figure 2.3: Dallas semiconductor 4MB commercial SRAM chip DS1250YAB [33]

2.3 Functional memory model

A functional fault model is based on the functional specifications of a given system [40]. The internals of the structure are made partly visible, and is therefore referred as gray-box model. This model can be used for verification of the functioning of the system by making a set of assumptions about the internal implementations of these functions [40].

As stated earlier, modeling a system into different functional blocks or subsystems leads to easier understanding of the functionality of the system. Fig 2.4 shows the functional block diagram of an SRAM. The memory system is modeled into different functional blocks namely; Memory Cell Array (MCA), Address Decoders (AD), the Peripheral Circuits (PC) and timing generation and control. The MCA forms the heart of the memory system. It consists of a number of memory cells arranged in a number of rows and columns, i.e., in a matrix like structure. If 'R' is the number of rows and 'C' is the number of columns, then the cell array has a capacity of $R \times C$ bits. The number of rows can be any integer, while the number of columns in memory is restricted: there is always an integer number of memory words in one row (i.e., for a 'N' bit memory C mod N =0). The AD is used for selection of a particular row and column to access a cell. The higher order of address bits from the address line are generally used to select the rows, while the lower order of bits are used to select the columns. The PC consists of read, write and pre-charge circuitries. During read operation, the content of the cell influences the states of the bit lines, which allows the sense amplifier to sense the logic state of the cell. This logic state is then stored into the data-out latches, through which the logic value can be read. During write

operation, the bit lines are influenced by the write driver depending upon the value in data-in latches. The logic state is then forced onto the cell through the bit lines. The pre-charge circuits are used to restore equal voltage levels over the bit lines after every read or write operations. The pre-charge phase ensures that successive operations over the same columns do not influence each other, and every single operation is made independent with respect to each other.

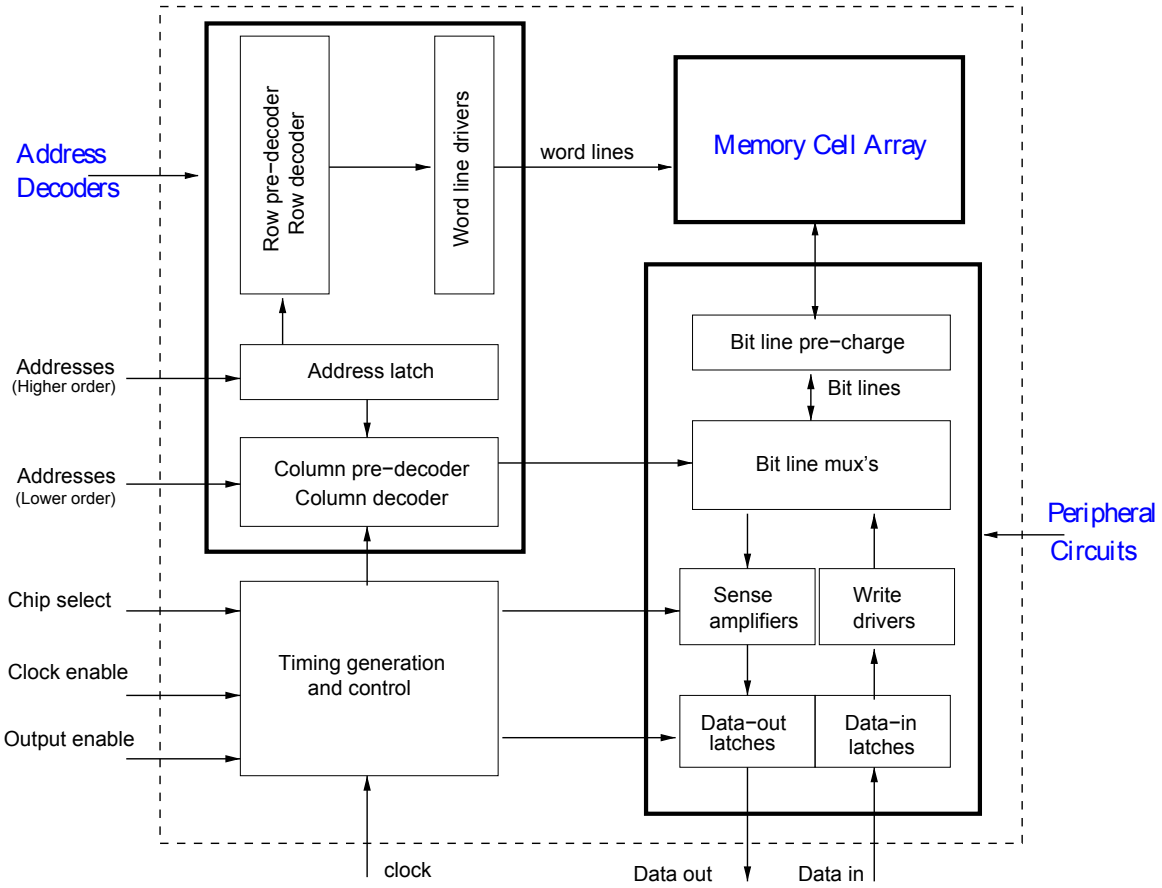


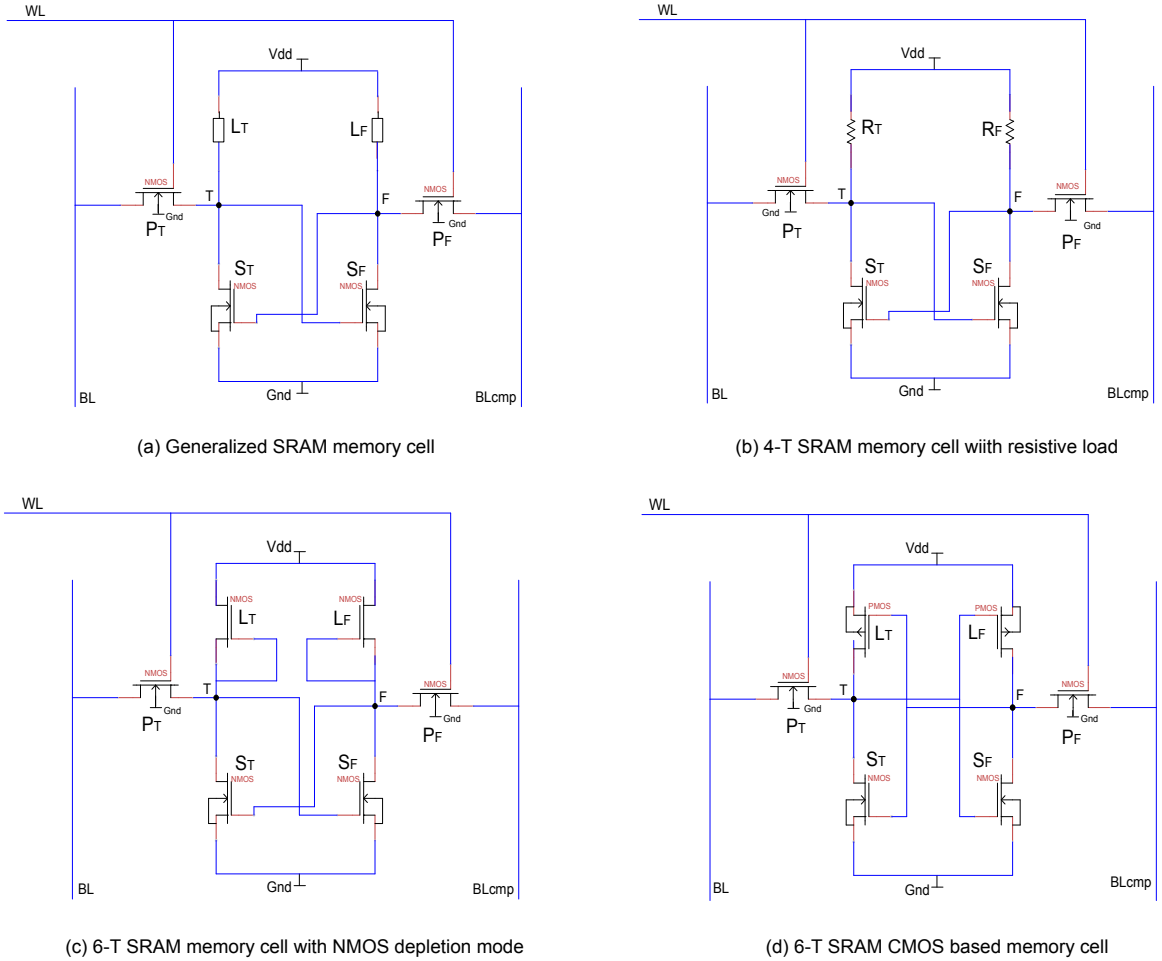
Figure 2.4: Functional block diagram of an SRAM

2.4 Electrical memory model

The electrical model of a system provides the schematic representation of its functional blocks at a circuit level. The schematic representation at a circuit level comprises components like transistors, resistors, capacitors, etc. As discussed earlier, the defects occurring at layout level, can be mapped onto the circuit level by modeling them as open, shorts or bridges. Thus fault modeling is made easier by using an electrical memory model. This section discusses the electrical model of each of the functional blocks in memory; i.e., MCA, AD and PC.

2.4.1 Memory Cell Array

The fundamental component in a MCA is the memory cell. The memory cell has a bistable circuit characteristic. Hence, it can be driven into any one of the two states at a given time; i.e., either logic '1' or logic '0'. The value in the cell is retained even after removing the stimulus, and as long as the power is switched ON. The SRAM cell stores the value in the true node, while the complement of the value is stored in false node. The true node and false node are connected to bit lines 'BL' and 'BLcmp', respectively, through the pass transistors. The pass transistors are controlled by the word line signal, which is driven by the row decoder. On the other hand the bit lines are controlled using the column decoder. When the pass transistors are made ON and the bit lines of a given column are selected at the same time, a cell with that unique address (decoded by row and column decoder) is accessed. The memory cell can have several **configurations**. Fig 2.5 shows different configurations of the memory cell.



(a) Generalized SRAM memory cell

(b) 4-T SRAM memory cell wiith resistive load

(c) 6-T SRAM memory cell with NMOS depletion mode

(d) 6-T SRAM CMOS based memory cell

Figure 2.5: Configurations of SRAM cell

Fig 2.5 (a) shows the generalized memory cell. It consists of two load elements L_T and L_F, and two storage elements S_T and S_F, along with two pass transistors P_T and P_F connected

to bit lines BL and BLcmp respectively. The storage transistor along with the load element forms an inverting configuration. Two such inverters are cross coupled to form a latch. This latch together with the pass transistors forms an SRAM cell. Different configurations of the SRAM cell can be built by changing the load element as shown in Fig 2.5(b), 2.5(c), 2.5(d) The choice of the configuration of the memory cell purely depends on the design and application requirements. Each of the configurations have their own merits and demerits. The 4-T (i.e., 4 transistors are used in this configuration) SRAM memory cell, (see Fig 2.5 (b)) with resistive load (polysilicon devices) occupies a lesser area, but has more power dissipation due to high currents. The depletion mode NMOS can also be used as a load element (see Fig 2.5(c)). The depletion mode NMOS can also be replaced by enhancement mode NMOS. Nevertheless, the depletion mode NMOS is generally preferred owing to its high impedance, better switching performance, and insensitive to power supply variations [7]. The most commonly used configuration is the 6-T cell CMOS based SRAM cell (see Fig 2.5(d)). The CMOS based SRAM cell offers lesser static power consumption. Power is consumed only during the time of switching activity. However, it has higher manufacturing complexity when compared to the NMOS depletion load based SRAM cell.

Fig 2.6 shows the write operations in a cell. Data can be stored in an SRAM using a **write operation**. The write operation is performed by forcing the state of the cell to a certain logic value as required. The data (either logic '0' or logic '1') is fed to the data-in latch, depending on which the write driver influences the state of the (already pre-charged) bit lines (either BL or BLcmp). When the required cell is accessed, the cell node voltages are affected due to the voltage levels in bit lines. Hence, the logic value is then forced on the cell, according to the state of the bit line.

Fig 2.6(a) shows the write '1' operation in a cell. Assume the cell initially has a logic '0'. When a logic '1' has to be written on the cell, the BLcmp bit line is pulled down to logic '0'. The required cell is accessed through the pass transistors. The false node is now at a higher voltage than bit line 'BLcmp', while the true node is at a lower voltage than bit line 'BL'. The potential difference allows the false node to be discharged and the true node to be charged simultaneously. Since SRAM behaves like a latch, the true and false node voltages influence each other. The cell flips when the voltage of false node falls below a certain threshold (or the true node voltage goes above the threshold) and logic '1' is now stored in true node and the complementary is stored in false node. The effects are complementary during a write '0' operation as shown in Fig 2.6(b).

Fig 2.7 shows the read operations in a cell. Data can be retrieved from an SRAM using a **read operation**. The read operation of a cell, involves a sense amplifier, which measures the differential voltages of the two bit lines 'BL' and 'BLcmp', to provide the data output. The bit lines are initially pre-charged to a certain value to an equal level. When the cell is accessed, the bit line voltages are influenced by the cell node voltages, resulting in the discharge of one of the bit lines. The data in the cell determines which of the two bit lines voltages are affected. Hence, the differential sense amplifier sees either a positive difference or negative difference of voltages between bit lines, depending on which the output of logic '0' or logic '1' is stored in the data-out latch.

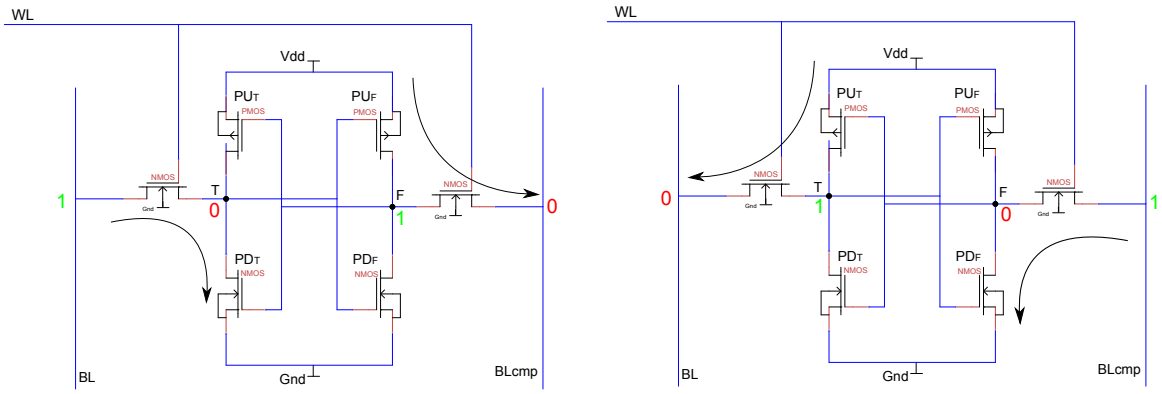


Figure 2.6: Write operations in SRAM cell

Fig 2.6(b) shows the read ‘0’ operation in a cell. If a logic ‘0’ has to be read from a cell, the bit lines are pre-charged initially to a logic ‘1’. When the cell is accessed, the true node voltage has a lower voltage than the bit line (BL) voltage, while the false node and the other bit line (BLcmp) are at same voltage levels. The bit line ‘BL’ is discharged though the true node. This results in a differential voltage being developed between bit lines ‘BL’ and ‘BLcmp’. The differential voltage is of the order of several millivolts. The differential sense amplifier senses the difference between the two bit lines to give a logic ‘0’ output,through the data-out latch. Fig 2.6(a) shows the read ‘1’ operation on a cell. In this case, the ‘BLcmp’ is discharged through the false node. The differential voltage is sensed by the sense amplifier to give a logic ‘1’ at the output, through the data-out latch.

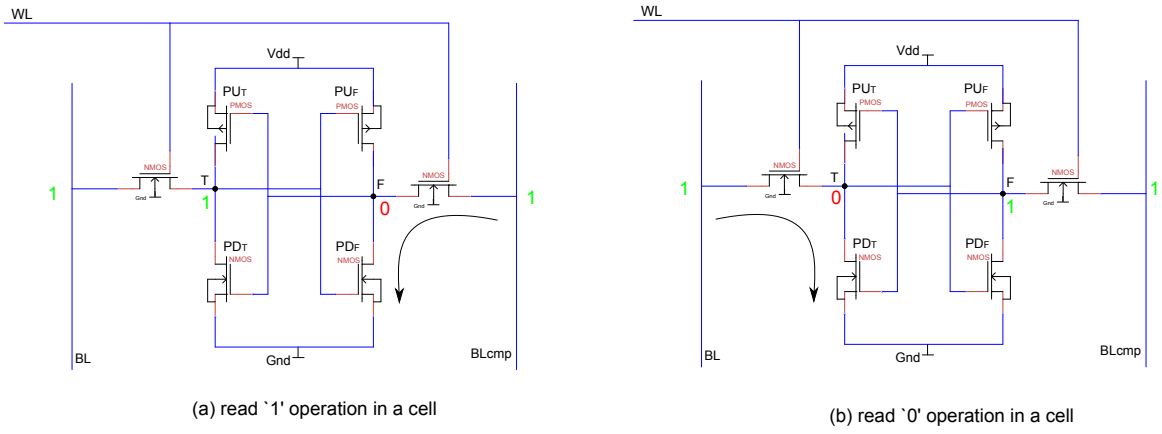


Figure 2.7: Read operations in SRAM cell

During read operation, one of the bit lines, discharges through one of the cell nodes. Thus there is a slight increase in voltage at the cell node. This is known as the ‘read bump’ voltage. Care should be taken that the read bump voltage does not cross the threshold voltage of the cell, which might result in the cell flipping. For this, the design of cell is done in such a way that

the resistance of pull down transistor in cell is at least 1.5 to 2 times greater than the resistance of the pass transistor. The beta ratio is an indicator of stability of the cell. This parameter is commonly used during the cell design. The beta ratio is defined as the ratio of strengths of pull down transistor to the pass transistor [30].

$$\beta = \frac{(W_{eff}/L_{eff})_{pulldowntransistor}}{(W_{eff}/L_{eff})_{passtransistor}}$$

2.4.2 Address Decoders

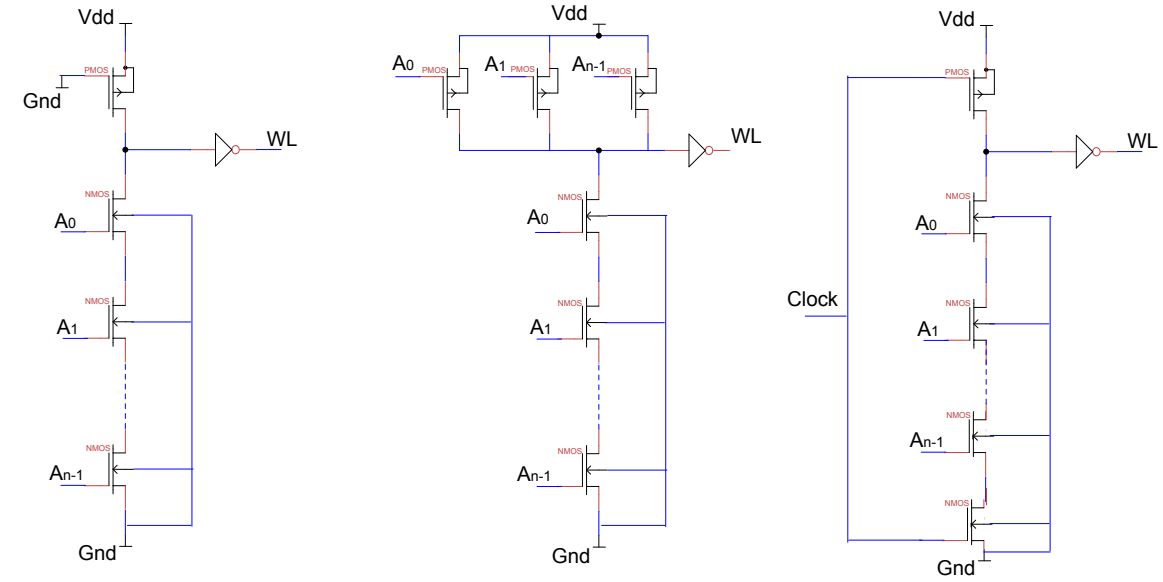
The AD are used to access a given cell from the cell array. One dimensional addressing scheme is mostly not used, owing to large word lines and size of the decoders. Two dimensional scheme of addressing is used within the chip. This demands the use of two decoders, one for row decoder and one for column decoder. The row decoder, is used for selection of word lines connecting a group of cells present in the same row. The column decoder is used for selection of bit lines which run across a given column in the memory.

Row decoders can be either static or dynamic as shown in Fig 2.8. A static PMOS load based row decoder is shown in Fig 2.8(a). The address lines (or their complements) drive only the NMOS transistors, whereas in a CMOS based row decoder (see Fig 2.8(b)) the address lines drive both PMOS and NMOS transistors. This makes the PMOS load based decoder to be much faster, as there is a reduction in the input capacitance when compared to that of CMOS based row decoder. The area occupied by a CMOS based decoder also is more than that of the PMOS load based row decoder. But CMOS row decoder offers very less static power consumption. Power is consumed only during the switching activity. This is the primary reason for wide spread usage for CMOS based static decoders. Fig 2.8(c) shows a dynamic based address decoder, which aims to combine the advantages of the PMOS load based static decoder and the CMOS static decoder. The area occupied by the dynamic decoder is not as much as the CMOS based static decoder as the number of PMOS in the pull transistors is reduced. This also reduces the input capacitance leading to faster decoding. However, it comes with the price of the usage of a clock signal which drives the PMOS transistor and an NMOS transistor.

The column decoder is used to select the bit lines (or a pair of bit lines). A tree based decoder, as shown in Fig 2.9(a), is used for single ended memories which operate with a single bit line for performing read and write operations. The tree based decoder is very simple, but also very slow. The PMOS load based static column decoder shown in Fig 2.9(b) can be used for selection of multiple bit lines. The output from the decoder drives the input gate signal for NMOS transistors which are present between the PC and the MCA. A CMOS based static column decoder can also be used, as it offers advantages over static power consumption in comparison with PMOS load based column decoder.

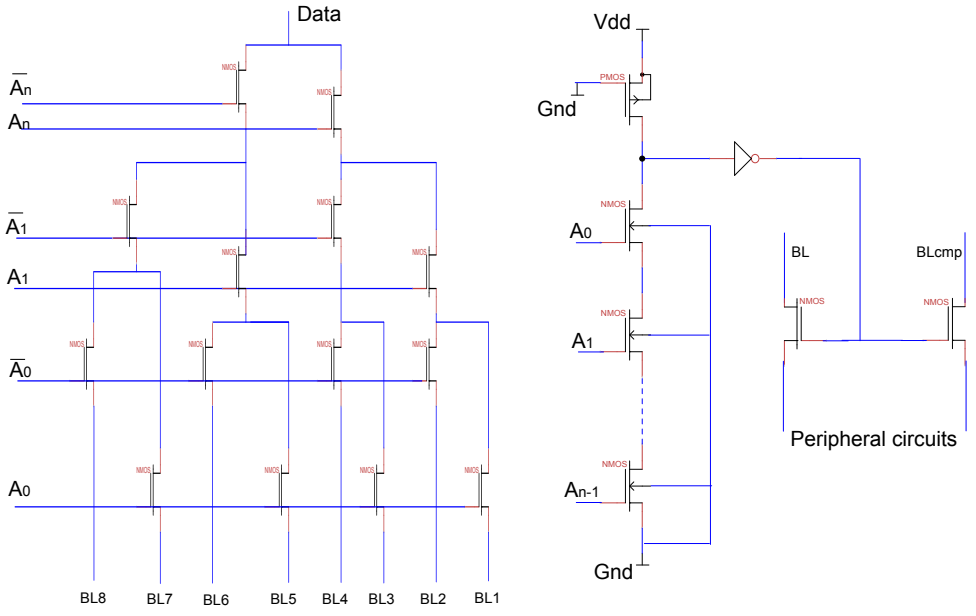
2.4.3 Peripheral Circuits

The peripheral circuit consists of write, read and pre-charge circuitries. The **write driver** circuitry is responsible for influencing the bit lines during write operations. The write driver can have many possible circuit implementations. Fig 2.10 shows the two possible implementations



(a) PMOS-load static row decoder (b) CMOS-load static row decoder (c) Dynamic row decoder

Figure 2.8: Static and Dynamic row decoders

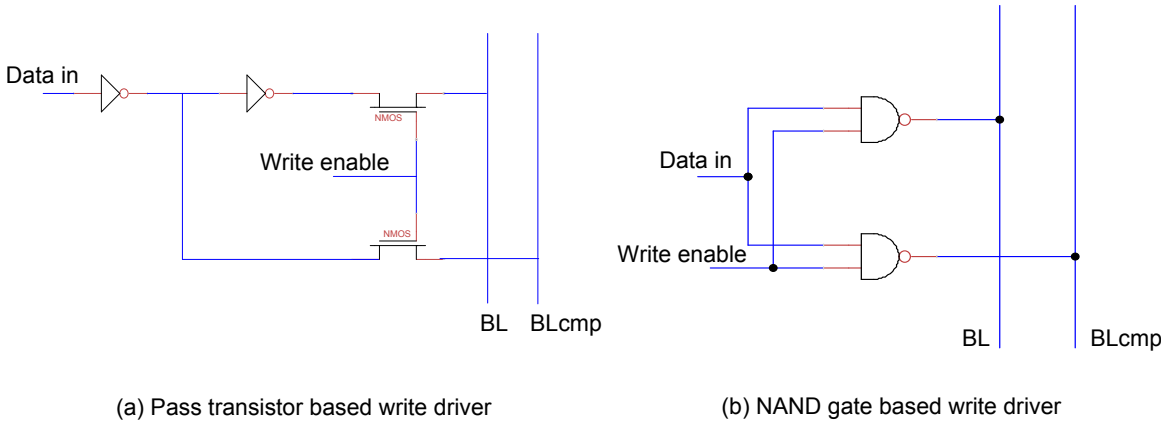


(a) Tree based column decoder (b) PMOS-load static column decoder

Figure 2.9: Static and Dynamic column decoders

of the driver based on pass transistor logic or NAND gate based logic. In case of pass transistor logic, (see Fig 2.10(a)) the data from the data-in latch is buffered though inverters, to the source of the NMOS transistors. These NMOS transistors are controlled by the write enable signal. When the write enable signal is driven to a logic '1', the NMOS transistors are turned ON and

the logic state of the data influences the bit lines, which forces a write operation in the required cell. In case of NAND gate based logic (see Fig 2.10(b)), the output of the NAND gate driven by signals 'data in' and write enable signals directly influences the state of the bit lines.

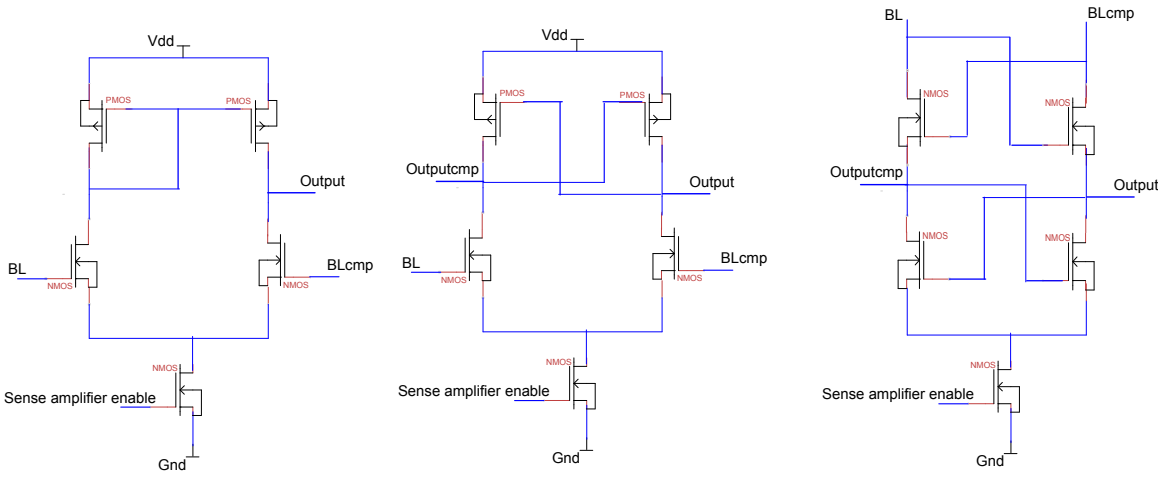


(a) Pass transistor based write driver (b) NAND gate based write driver

Figure 2.10: Write driver circuitries

The read circuity consists of a **sense amplifier**, which can sense the state of the bit line(s) after a read operation. The read circuitry implemented depends on the type of memory cells to be read, i.e., single ended or differential. In addition to , it can also be based on voltage mode or current mode. Fig 2.11(a) and 2.11(b) show the two different types of voltage mode sense amplifiers which can be implemented. The differential voltage mode sense amplifier is most preferred circuit for implementation, owing to their cross coupled structure which makes them faster than its counter parts. The current mode sense amplifier (see Fig 2.11(c)) operates by sensing the current levels of the bit lines. The drawback of this type of current mode sense amplifiers is that the output of the sense amplifier keeps changing as the current value changes. This is in contrary to voltage mode sense amplifiers where the data once stored in output register is locked, and is not affected even when there is a change in voltage levels of bit line. All these sense amplifiers have an access transistor in their implementations. The access transistor is basically an NMOS device, which is controlled by the sense amplifier enable signal. This ensures that the sense amplifier is switched ON only after the read operation takes place.

The **pre-charge circuitry** performs the pre-charge operation over the bit lines and data lines present in the memory system. For the next operation to take place correctly, it becomes necessary to restore the voltage levels (by pre-charging) of the bit lines and data lines before the next operation takes place. Fig 2.12 shows the circuit which is used for pre-charge operation. It consists of three NMOS transistors. Two PMOS transistors, connect the bit lines to the supply voltage source. However, due to mismatch in these two transistors, sometimes the bit lines may not be pre-charged equally, leading to a differential voltage across the two bit lines. This may have a negative impact during read or write operations. In order to avoid such situations, another PMOS transistor known as the equalizing transistor is used. This transistor connects the two bit lines directly, thus ensuring that there is no differential voltage developed across the bit lines. All the three PMOS transistors are controlled by the pre-charge input signal 'Blprechcmp' as



(a) Single ended sense amplifier (b) Differential ended voltage mode sense amplifier (c) Current mode sense amplifier

Figure 2.11: Sense amplifier circuitries

shown in Fig 2.12. When this signal is active high, the two bit lines are decoupled by the equalizing transistor, which allows the bit lines to be equally pre-charged before each read or write operation.

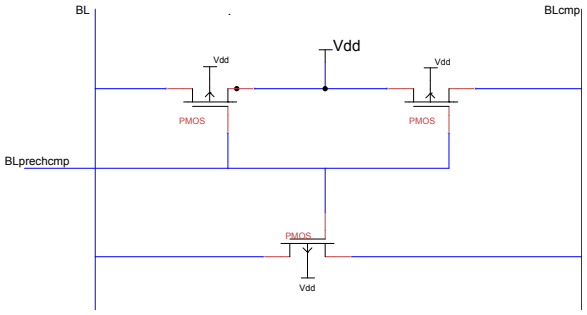


Figure 2.12: Pre-charge circuitry

2.5 Memory process technology

The description of the layout or physical implementation forms the last stage in the design of a memory system. The design of SRAM relies heavily on the process technology of MOS devices. In this section, the fabrication process of MOS devices is discussed. Silicon is used as a base semiconductor material in fabrication of integrated circuits. This is because the resistivity of silicon can be easily changed by introducing special type of impurity atoms in a silicon crystal. The MOS fabrication process begins with the growth of large cylindrical silicon crystals of pure silicon, which are sliced into disks called wafers. The wafers are about 20cm to 30cm thick in diameter. This wafer is used for the fabrication of circuits. The process involves a series of successive steps to form patterns alternated with diffusion and implantations of various dopants

into the opening of the pattern. The patterning utilizes a number of masks, to define a pattern in a process called photo lithography. The size of the mask, and the pattern of the mask determines a significant cost of the fabrication process.

Fig 2.13 [14] shows the steps in a basic MOS process. The process starts with a wafer; the silicon can be doped with suitable materials to form a positively (or negatively) doped semiconductor material. The silicon is first doped with p type (or n type) charge carriers, followed by a thermal oxidation process to form an oxide layer of SiO_2 over the p-type silicon substrate (see Fig 2.13(a)). The oxides are covered with a special material called photoresist (see Fig 2.13(b)). The photoresist is a ploymer which is sensitive to light. The photoresist is soluble under certain solutions, when exposed to UV light (see Fig 2.13(c)). Certain areas of the silicon where the doping has to take place, are exposed to UV light after which the photoresist is removed for doping process (see Fig 2.13(d)). The areas which are exposed to the UV light, can be controlled by using appropriate masks, with specific patterns. The mask pattern hence defines the source, drain channels and diffusion regions. Dopant atoms, are now introduced in those exposed areas, in high temperature gas environments or with an ion beam accelerator for the diffusion to take place (see Fig 2.13(e)). Finally the photoresist can be removed from the SiO_2 surface using special solvents. This process is continued to produce complicated patterns as per requirements (see Fig 2.13(f)).

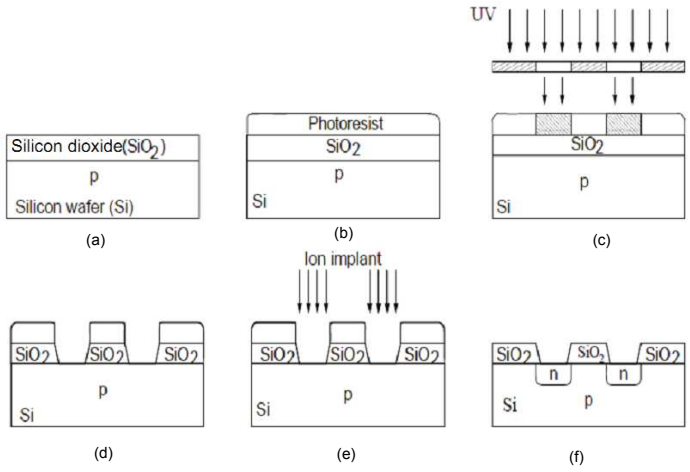


Figure 2.13: Basic MOS process [14]

The scaling of SRAM has been challenged with the threshold voltage mismatch due to variability in transistor performance [23]. To suppress this effect very complex fabrication techniques are used. Recently the segmented bulk MOSFET or the segFET design was proposed to improve the threshold voltage variations [37]. The fabrication process of a segFET device is mostly similar to that of conventional MOSFET, but for the structure of the starting material. A corrugated substrate is used for the fabrication of the segFET device [35]. A full 3D 6-T based SRAM cell can be developed using the segFET technology to exploit the benefits of advanced MOS transistors built at 22nm technology node [35].

3 Memory Fault Space

Precise fault modeling is required for simpler understanding of faulty behavior of memory. Better understanding of fault models and their notations, can lead to development of efficient test algorithms. Proper classification of the fault models, also helps us to approach fault diagnosis in a systematic manner. This chapter introduces the memory functional models and fault primitives which are used for the analysis of functional fault models. This chapter is organized as follows. Section 3.1 describes the reduced memory functional model. Section 3.2 introduces the concept of fault primitives. Section 3.3 and 3.4 describes the complete set of static and dynamic fault space. Section 3.5 briefly introduces strong and weak faults.

3.1 Reduced memory functional model

Memory functional model consists of a number of functional blocks or subsystems which can be implemented. However, the three most widely used functional blocks in different memory functional models are the Address Decoders, Memory Cell Array and the Peripheral Circuits as shown in Fig 3.1. These three functional blocks together comprise the reduced memory functional model. The advantage of using such a model is that the functional fault models developed by considering faults in these three functional blocks will be applicable to most of the memory systems built for various applications. The reduced functional model also results in a straight forward approach towards testing memories; thus making it faster and easier to analyze.

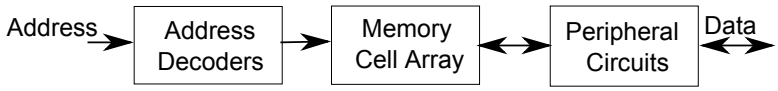


Figure 3.1: Reduced memory functional model

3.2 Concept of fault primitives and fault classification

Functional faults occurring in a memory system can be defined as the deviation of behavior of the memory from that of the expected one. These functional faults are detected by applying certain set of memory operations and comparing their responses with respect to the expected responses. Hence there are two main steps in fault modeling [1]:

1. A list of memory operations to be performed, to sensitize a fault. The sequence of such operations are referred as *Sensitizing Operation Sequence (SOS)* [1].

2. A list of mismatches between observed (faulty) behavior and the expected behavior.

In order to specify a certain memory fault, one has to represent it in the form of a fault primitive (FP) [42], denoted as $<S/F/R>$. S describes the operation sequence that sensitizes the fault, F describes the logic level in the faulty cell ($F \in \{0, 1, \uparrow, \downarrow\}$), and R describes the logic output level of a read operation ($R \in \{0, 1, -\}$). R has a value of 0 or 1 when the fault is sensitized by a read operation, while the '-' is used when a write operation sensitizes the fault. For example, in the FP= $<0w1/0/->$, which is the up-transition fault (TF1), S=$0w1$ means that a w1 operation is written to a cell initialized to 0. The fault effect F=0 indicates that after performing $w1$, the cell remains in state 0. The output of the read operation (R=$-$) indicates there is no expected output for the memory. The classification of FPs is shown in Fig 3.2.

- Depending on the number of ports available in the memory, the FPs are classified either as **single-port faults** and **multiple-port faults**. Single port faults (1PFs) require, at most one port in order sensitize the fault. If #P defines the number of ports, then for a single port fault, #P=1. On the other hand multi-port faults (pPFs) require two or more operations to be performed simultaneously via different ports, i.e. $\#P \geq 2$. Depending on the value of #P, multi-port faults can be further classified into two port faults (2FPs), three port faults (3FPs) and so on.

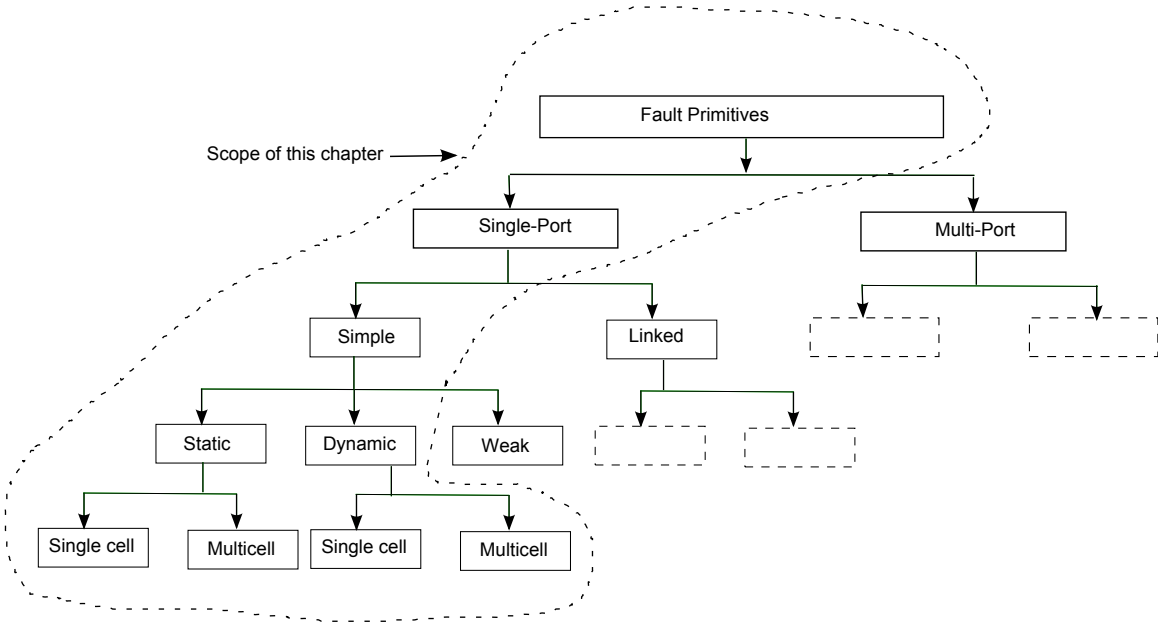


Figure 3.2: Fault Primitive classification

- Depending on how FPs manifest themselves, they are classified into **simple faults** and **linked faults**. Simple faults do not interfere with the other faults occurring in the memory system; i.e., they are independent. Linked faults on the other hand, influences the behavior of other faults present in the system. It can be seen as a combination of two or more simple faults. Linked faults, may lead to fault masking.

- Depending on the number of operations '#O' performed sequentially for fault sensitization, FPs are classified as **static faults** and **dynamic faults**. Static faults are sensitized by at most a single operation, i.e., $\#0 \leq 1$. Dynamic faults requires more than one operations to be successively performed for the fault to be sensitized i.e., $\#O > 1$. Depending on the number of operations required for sensitization they are called two-operation dynamic faults, three operation dynamic faults and so on. Dynamic faults are mostly timing and delay related faults, and is of gaining importance especially with decreasing feature sizes.

- Depending on the number of different cells affected on the application of the SOS, the FPs are classified into **single cell fault** or **multi cell faults**. Let #C be number of different cells accessed during a SOS. Depending on #C, FPs can be divided into single cell faults and multi cell faults. Single cell faults have FPs involving only a single cell; they have the property that the cell used for sensitizing the fault (by applying SOS) is same as where the fault appears. Coupling faults have FPs that involve more than one cell; they have the property that the cell(s) which sensitizes (or contribute for sensitizing) the fault (e.g, by applying the SOS) is different from the cell where the fault appears. Depending on #C, this class can be further divided into two coupling fault primitives whereby #C=2, three coupling fault primitives whereby #C=3, etc.

This chapter mainly focuses on single port, simple, static and dynamic faults involving single cell and two cell as indicated by dotted line in Fig 3.2.

3.3 Static faults

Static faults can be sensitized using a single operation. Since memory operations involve only write and read operations within two logic states, the set of sensitizing operations 'S' is given by $S = \{0w0, 0w1, 1w0, 1w1, r1, r0, 1, 0\}$ whereby, 'w' denotes a write operation and 'r' denotes a read operation, with '1' and '0' denoting the logic states of the cell. Similarly, the set of faults 'F' is given by $F = \{0, 1, \uparrow, \downarrow\}$ whereby, '0' and '1' denotes the logic state of the cell and $\uparrow$ ($\downarrow$) denotes the up(down) transition due to the sensitizing operation. The 'R' component in the $< S/F/R >$ notation denotes the outcome of the read operation where the set of 'R' is given by $R = \{0, 1, -\}$ whereby '?' denotes a random value and '-' denotes that the read operation is not applicable. These $< S/F/R >$ notations are widely used in the following sections to describe the static functional faults in Memory Cell Array.

3.3.1 Static Memory Cell Array Faults

The static MCA faults can be divided into single-cell FPs or multi-cell FPs. Single-cell FPs are FPs involving single cell; while multi-cell FPs are FPs involving more than cells. For multi-cell FPs the discussion is restricted to two cell FPs, because they are considered as an important class in single-port SRAM faults. Fig 3.3[14] shows the classification of (single-port) static faults. Single port faults involving a single cell (1PF1s) have the property that the cell used for sensitizing the fault, is the same cell where the fault appears [14]. On the other hand, the single port two cell faults can be divided into three types, depending on the cell which is used for applying the sensitizing operation:

1. $1PF2_s$: It has the property that the state of the aggressor cell (a-cell c_a), rather than the operation applied to c_a, sensitizes a fault in the victim cell (v-cell c_v). It is to be noted that no operation is required in this case for sensitization. The subscript 's' in the notation $1PF2_s$ stands for 'state'.

2. $1PF2_a$: It has the property that the application of a single-port operation to the a-cell sensitizes a fault in the v-cell.

3. $1PF2_v$: It has the property that the application of a single-port operation to the v-cell, with a-cell in a certain state, sensitizes the fault in the v-cell.

3.3.1.1 Single cell static fault primitives

Table 3.1 gives the single cell static fault primitives. These fault primitives can be compiled into functional fault models, based on which test algorithms can be developed. Given that $S \in \{0, 1, 0w0, 1w1, 0w1, 1w0, r0, r1\}$, $F \in \{0, 1, \uparrow, \downarrow\}$ and $R \in \{0, 1, -\}$; three cases can be distinguished (See table 3.1):

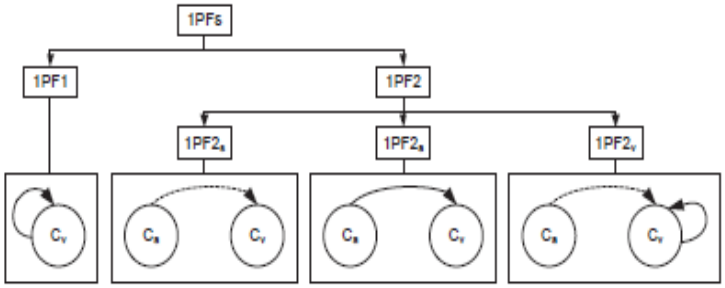


Figure 3.3: Fault Primitive classification [14]

Table 3.1: Single-cell static FPs

#	S	F	R	$< S/F/R >$	FFM	#	S	F	R	$< S/F/R >$	FFM
1	0	1	-	$< 0/1/- >$	SF	7	$0r0$	$\uparrow$	0	$< 0r0/\uparrow/0 >$	DRDF
2	1	0	-	$< 1/0/- >$	SF	8	$1r1$	$\downarrow$	1	$< 1r1/\downarrow/1 >$	DRDF
3	$0w0$	$\uparrow$	-	$< 0w0/\uparrow/- >$	WDF	9	$0r0$	$\uparrow$	1	$< 0r0/\uparrow/1 >$	RDF
4	$1w1$	$\downarrow$	-	$< 1w1/\downarrow/- >$	WDF	10	$1r1$	$\downarrow$	0	$< 1r1/\downarrow/0 >$	RDF
5	$0w1$	0	-	$< 0w1/0/- >$	TF	11	$0r0$	0	1	$< 0r0/0/1 >$	IRF
6	$1w0$	1	-	$< 1w0/1/- >$	TF	12	$1r1$	1	0	$< 1r1/1/0 >$	IRF

- $S \in \{0, 1\}$
  if $S = 0$, $F = 1$ and $R = -$; this results in FP1 of Table 3.1.
  if $S = 1$, $F = 0$ and $R = -$; this results in FP2.

- $S \in \{0w0, 1w1, 0w1, 1w0\}$
  if $S = 0w0$, $F = \uparrow$ and $R = -$; this results in FP3.
  if $S = 1w1$, $F = \downarrow$ and $R = -$; this results in FP4.
  if $S = 0w1$, $F = 0$ and $R = -$; this results in FP5.
  if $S = 1w0$, $F = 1$ and $R = -$; this results in FP6.

- $S \in \{r0, r1\}$
  if $S = r1$, $F = \downarrow$ and $R = 0$; this results in FP7.
  if $S = r1$, $F = \downarrow$ and $R = 1$; this results in FP8.
  if $S = r0$, $F = \uparrow$ and $R = 1$; this results in FP9.
  if $S = r0$, $F = \uparrow$ and $R = 0$; this results in FP10.
  if $S = r0$, $F = 0$ and $R = 1$; this results in FP11.
  if $S = r1$, $F = 1$ and $R = 1$; this results in FP12.

It can be observed that the set of fault primitives (FP1,FP2), (FP3,FP4), (FP5,FP6), (FP7,FP8), (FP9,FP10), (FP11,FP12) represent similar behavior with complementary values. Hence these fault primitives are grouped together to form functional fault models. the functional fault models can be used to devlop test algorithms. The functional fault models formed by grouping these primitives are describes below [14].

1. **State Fault (SF)**: A cell is said to have a state fault, when the logic value of a cell changes on its own, even when no operation is performed on it. No special operation is required to sensitize it and therefore it only depends on the initial value in the cell. The SF consists of two Fps: $< 0/1/- >$ and $< 1/0/- >$.

2. **Transition Fault (TF)**: A cell is said to have a transition fault, when it fails to undergo a transition from one logic state to the other. The TF is sensitized by a write operation, with a complementary data value to that of the value stored in the cell. The TF consists of two Fps: $< 0w1/0/- >$ and $< 1w0/1/- >$.

3. **Write Destructive Fault (WDF)**: A non transition operation in a cell leading to a transition operation ,results in write destructive fault.. WDF consists of 2 FPs: $< 0w0/\uparrow/- >$ and $< 1w1/\downarrow/- >$.

4. **Read Destructive Fault (RDF)**: A read operation, leading to the destruction of the value stored in the cell, and resulting in return of an incorrect value in the output is RDF. It consists of 2 FPs: $< 0r0/\uparrow/1 >$ and $< 1r1/\downarrow/0 >$.

5. **Deceptive Read Destructive Fault (DRDF)**: A read operation causes a transition in the cell, but returning the value of the cell before transition. DRDF consists of 2 FPs: $< 0r0/\uparrow/0 >$ and $< 1r1/\downarrow/1 >$.

6. **Incorrect Read Fault (IRF)**: A read operation returns an incorrect value at the output, while the cell has the correct value. IRF consists of 2 FPs: $< 0r0/0/1 >$ and $< 1r1/1/0 >$.

3.3.1.2 Two cell fault primitives

As discussed earlier, the two cell faults involves an a-cell and a v-cell. For two-cell coupling faults, the FPs are described as $< S_a; S_v/F/R >_{a,v}$ where S_a (S_v) is the sensitizing operation applied to aggressor (victim) cell: $S_a, S_v \in \{0, 1, 0w0, 1w1, 0w1, 1w0, r0, r1\}$. Here a (v) denotes the address of the aggressor (victim) cell for a coupling fault. It is to be noted that, if S_a is an operation, than S_v can be only a state (0,1). If S_a is a state, S_v can be either a state or an operation. 'F' denotes the faulty behavior in the affected cell and 'R' denotes the outcome of the read operation. The table 3.2 [14], lists all possible combinations of th values in $< S_a; S_v/F/R >$ notation that results in FPs. Given that $S_a, S_v \in \{0, 1, 0w0, 1w1, 0w1, 1w0, r0, r1\}$, $F \in \{0, 1, \uparrow, \downarrow\}$ and $R \in \{0, 1, -\}$; two cases can be distinguished while maintaining restriction of #O$\leq$1 :

- $S_a \in \{0, 1\}$: The notation $< x; S_v/F/R >$ with $x \in \{0, 1\}$ is a superset of the notation $< S_v/F/R >$. Hence based on the sensitizing operation or the state of S_v the FPs are further classified into two classes:
  - $S_v \in \{0, 1\}$: The state of the aggressor cell, sensitizes a fault in the victim cell. This results in 4 FPs, FP1 to FP4 as shown in 3.2.
  - $S_v \in \{0w0, 1w0, 1w1, 0w1, r0, r1\}$: The state of the aggressor, and a particular sensitizing operation in the victim cell, contributes to the sensitization of the fault in the victim cell. This results in 20 FPs FP5 to FP24 listed in 3.2.

Table 3.2: Two-cell static FPs [14]

#	S_a	S_v	F	R	$< S_a; S_v/F/R >$	FFM	#	S_a	S_v	F	R	$< S_a; S_v/F/R >$	FFM
1	0	0	1	-	$< 0; 0/1/- >$	CFst	19	1	$r0$	$\uparrow$	0	$< 1; r0/\uparrow/0 >$	CFdrd
2	0	1	0	-	$< 0; 1/0/- >$	CFst	20	1	$r1$	$\downarrow$	1	$< 1; r1/\downarrow/1 >$	CFdrd
3	1	0	1	-	$< 1; 0/1/- >$	CFst	21	0	$r0$	0	1	$< 0; r0/0/1 >$	CFir
4	1	1	0	-	$< 1; 1/0/- >$	CFst	22	0	$r1$	1	0	$< 0; r1/1/0 >$	CFir
5	0	$0w1$	0	-	$< 0; 0w1/0/- >$	CFtr	23	1	$r0$	0	1	$< 1; r0/0/1 >$	CFir
6	0	$1w0$	1	-	$< 0; 1w0/1/- >$	CFtr	24	1	$r1$	1	0	$< 1; r1/1/0 >$	CFir
7	1	$0w1$	0	-	$< 1; 0w1/0/- >$	CFtr	25	$r0$	0	$\uparrow$	-	$< r0; 0/\uparrow/- >$	CFds_{rx}
8	1	$1w0$	1	-	$< 1; 1w0/1/- >$	CFtr	26	$r1$	0	$\uparrow$	-	$< r1; 0/\uparrow/- >$	CFds_{rx}
9	0	$0w0$	$\uparrow$	-	$< 0; 0w0/\uparrow/- >$	CFwd	27	$0w1$	0	$\uparrow$	-	$< 0w1; 0/\uparrow/- >$	$\text{CFds}_{xw!x}$
10	0	$1w1$	$\downarrow$	-	$< 0; 1w1/\downarrow/- >$	CFwd	28	$1w0$	0	$\uparrow$	-	$< 1w0; 0/\uparrow/- >$	$\text{CFds}_{xw!x}$
11	1	$0w0$	$\uparrow$	-	$< 1; 0w0/\uparrow/- >$	CFwd	29	$0w0$	0	$\uparrow$	-	$< 0w0; 0/\uparrow/- >$	CFds_{xwx}
12	1	$1w1$	$\downarrow$	-	$< 1; 1w1/\downarrow/- >$	CFwd	30	$1w1$	0	$\uparrow$	-	$< 1w1; 0/\uparrow/- >$	CFds_{xwx}
13	0	$r0$	$\uparrow$	1	$< 0; r0/\uparrow/1 >$	CFrd	31	$r0$	1	$\downarrow$	-	$< r0; 1/\downarrow/- >$	CFds_{rx}
14	0	$r1$	$\downarrow$	0	$< 0; r1/\downarrow/0 >$	CFrd	32	$r1$	1	$\downarrow$	-	$< r1; 1/\downarrow/- >$	CFds_{rx}
15	1	$r0$	$\uparrow$	1	$< 1; r0/\uparrow/1 >$	CFrd	33	$0w1$	1	$\downarrow$	-	$< 0w1; 1/\downarrow/- >$	$\text{CFds}_{xw!x}$
16	1	$r1$	$\downarrow$	0	$< 1; r1/\downarrow/0 >$	CFrd	34	$1w0$	1	$\downarrow$	-	$< 1w0; 1/\downarrow/- >$	$\text{CFds}_{xw!x}$
17	0	$r0$	$\uparrow$	0	$< 0; r0/\uparrow/0 >$	CFdrd	35	$0w0$	1	$\downarrow$	-	$< 0w0; 1/\downarrow/- >$	CFds_{xwx}
18	0	$r1$	$\downarrow$	1	$< 0; r1/\downarrow/1 >$	CFdrd	36	$1w1$	1	$\downarrow$	-	$< 1w1; 1/\downarrow/- >$	CFds_{xwx}

- $S_a \in \{0w0, 1w0, 1w1, 0w1, r0, r1\}$: then $S_v \in \{0, 1\}$and $R = -$. In this case, the application of an operation in the aggressor cell sensitizes the fault in the victim cell.
  - $S_v = 0$: then $F \in \{\uparrow, ?\}$. This results in fault primitives FP25 to FP30 in table 3.2, for the 6 different sensitizing operations in the aggressor cell.
  - $S_v = 0$: then $F \in \{\downarrow, ?\}$. Similarly the FPs from FP30 to FP36 in table 3.2 can be obtained from six different sensitizing operating sequences in aggressor cell.

As stated earlier, the single port fault primitives are divided into three types, based on the sensitizing criteria i.e., 1PF2s, 1PF2a, and 1PF2v.

The 1PF2s functional fault models:

1. **State coupling fault (CFst):** In this type of coupling fault, depending on the state of the aggressor cell, the victim cell is sensitized. No special sensitizing operations are hence required. Hence the initial state of the cell, determines the sensitization of the fault. It has 4 FPs $< 0; 0/1/- >$, $< 0; 1/0/- >$,$< 1; 0/1/- >$ and $< 1; 1/0/- >$.

The 1PF2a functional fault models:

1. **Disturb coupling fault (CFds):** In this type of coupling fault, the operation performed in the aggressor cell enables the flipping of the victim cell.It has 12 FPs $< r0; 0/\uparrow/- >$, $< r0; 1/\downarrow/- >$, $< r1; 0/\uparrow/- >$,$< 0w1; 0/\uparrow/- >$, $< 0w1; 1/\downarrow/- >$, $< 1w0; 0/\uparrow/- >$,$< 1w0; 1/\downarrow/- >$,$< 0w0; 0/\uparrow/- >$, $< 0w0; 1/\downarrow/- >$, $< 1w1; 0/\uparrow/- >$ and $< 1w1; 1/\downarrow/- >$.

2. **Idempotent coupling fault (CFid):** In this type of fault, a transition write operation in the aggressor cell, causes the victim cell to flip.It has 4 FPs, $< 0w1; 1/\uparrow/- >$,$< 0w1; 1/\uparrow/- >$,$< 1w0; 0/\uparrow/- >$ and $< 1w0; 1/\uparrow/- >$

3. **Inversion coupling fault (CFin):** If the logic value of the victim cell is inverted in case of a transition write operation, performed on the aggressor cell. It contains 2 pairs of FPs in $< 0w1; 0/\uparrow/- >$,$< 0w1; 1/\downarrow/- >$,$< 1w0; 0/\uparrow/- >$ and $< 1w0; 1/\downarrow/- >$. This is because the state of the victim cell can either be at logic '0' or at logic '1', and transition has to take place accordingly in victim cell, when the fault is sensitized in the aggressor cell.

The 1PF2v functional fault models:

1. **Transition coupling fault (CFtr):** In this type of coupling fault, the transition operation performed in the victim cell fails for a given state of the aggressor cell. The CFtr consists of following FPs $< 0; 0w1/0/- >$, $< 1; 0w1/0/- >$, $< 0; 1w0/1/- >$, $< 1; 1w0/1/- >$.

2. **Write Destructive coupling fault (CFwd):** In this type of coupling fault, a non-transition operation performed in the aggressor cell enables the victim cell to undergo a transition for a given state of the aggressor cell. It has 4 FPs $< 0; 0w0/\uparrow/- >$, $< 1; 0w0/\uparrow/- >$, $< 0; 1w1/\downarrow/- >$ and $< 1; 0w0/\downarrow/- >$.

3. **Read Destructive coupling fault (CFrd):** In this type of coupling fault, a read operation performed on the victim cell, changes the state of the victim cell and returns an incorrect value at the output for a given state of the aggressor cell. It has 4 FPs $< 0; r0/\uparrow/1 >$, $< 1; r0/\uparrow/1 >$, $< 0; r1/\downarrow/0 >$, and $< 1; r1/\downarrow/0 >$.

4. **Deceptive Read Destructive coupling fault (CFdrd):** In this type of coupling fault, a read operation performed on the victim cell, changes the state of the victim cell and returns an a correct value at the output for a given state of the aggressor cell. It has 4 FPs $< 0; r0/\uparrow/0 >$, $< 1; r0/\uparrow/0 >$, $< 0; r1/\downarrow/1 >$, and $< 1; r1/\downarrow/1 >$.

5. **Incorrect Read coupling fault (CFir):** In this type of coupling fault, a read operation performed on the victim cell, returns an incorrect value at the output for a given state of the aggressor cell. It has 4 FPs $< 0; r0/0/1 >$, $< 1; r0/0/1 >$, $< 0; r1/1/0 >$, and $< 1; r1/1/? >$.

3.3.2 Static Address Decoder Faults

The static address decoder faults, are faults occurring in row decoders and column decoders, which can be sensitized by performing at most one operation. The static faults in the address decoder are assumed to demonstrate similar behavior during the read and the write operation. Fig 3.4 shows 4 main categories of functional address decoder faults defined for bit oriented memories:

1. **Fault 1** refers to the situation where a particular address cannot access its cell.
2. **Fault 2** refers to the situation where a particular memory cell is not accessed by any address.
3. **Fault 3** refers to the situation where a particular address access multiple cells at the same time.
4. **Fault 4** refers to the situation where a particular memory cell is accessed by many addresses at the same time.

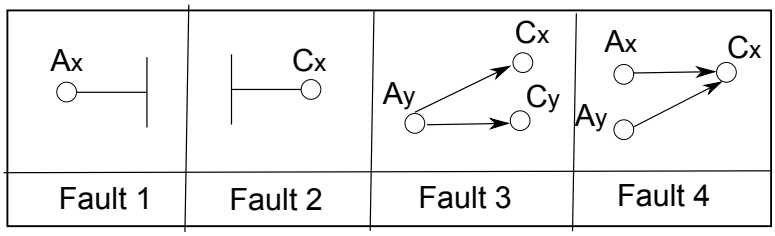


Figure 3.4: Static address decoder faults

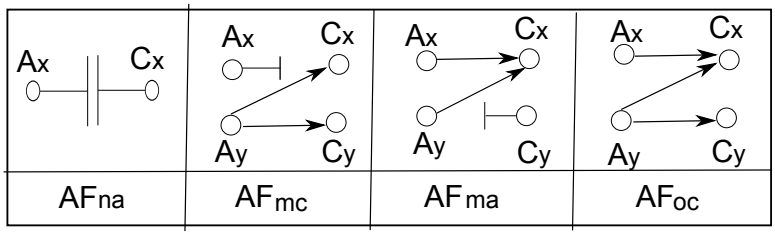


Figure 3.5: Fault combinations in static address decoder faults

Static faults occurring in the address decoder are not completely independent with respect to each other. This is because there are as many cells as addresses. Hence the fault always takes place in pairs. Fig 3.5shows the faults occurring from combinations of these 4 faults.

- Fault AF_{nca}: It is a combination of Fault 1 and Fault 2, and is called a no cell and no address fault.
- Fault AF_{nmc}: It is a combination of Fault 1 and Fault 3, and is called a no cell and multiple cell fault.
- Fault AF_{nma}: It is a combination of Fault 2 and Fault 4, and is called a no address and multiple address fault.
- Fault AF_{mca}: It is a combination of Fault 3 and Fault 4, and is called a multiple cell and multiple address fault.

3.3.3 Static Peripheral Circuit Faults

Static Peripheral Circuit faults occur in read or write circuitries such as sense amplifiers, precharge circuits, write drivers, etc. These faults can be caused by spot defects involving opens, shorts and bridges. Functionally, these faults are mapped onto memory cell array faults [40]. It is assumed that any static fault in the peripheral circuitry (such as stuck at fault) can be detected with any test for memory cell array as any operation applied to the memory has to go through the peripheral circuitry.

3.4 Dynamic Faults

Dynamic based faults, are faults which require more than one operation to be performed sequentially for their sensitization. Thus, the number of operations $\#O \geq 2$. Depending on the number of operations required for sensitization, dynamic faults are classified into 2-operation dynamic fault, 3-operation dynamic faults and so on. This section deals with the discussion of 2-operation dynamic faults occurring in different parts of memory system.

3.4.1 Dynamic Memory Cell Array Faults

Dynamic faults occurring in Memory Cell Array can be classified into single cell dynamic faults and multi cell dynamic faults. In the following sections the two operation single-cell and two-cells dynamic faults are discussed.

3.4.1.1 Single cell fault primitives

Single-cell dynamic faults are sensitized by applying more than one operation sequentially to the same cell. As mentioned earlier, a single-cell fault primitive is denoted as $< S/F/R >$. S is the sensitizing operation sequence. Since, we are considering two operations, S can be of the form xO_1yO_2z where $x, y, z \in \{0, 1\}$ and O_1A,O_2 can be read/write operations. Combining all permutations, 18 different sensitizing sequences are possible [15].

- 8 S have the form of write after write operation: $xwywz$. For example, $1w1w0$ denotes a w1 operation to a memory cell whose initial state is '1', immediately followed by a $w0$ operation.
- 2 S have the form of read after read operation: $xrxrx$. For example, $1r1r1$ denotes a r1 operation to a memory cell whose initial state is '1', immediately followed by a $r1$ operation.
- 4 S have the form of write after read operation: $xrxwz$. For example, $1r1w0$ denotes a r1 operation to a memory cell whose initial state is '1', immediately followed by a $w0$ operation.
- 4 S have the form of read after write operation: $xwyry$. For example, $1w1r1$ denotes w1 operation to a memory cell whose initial state is '1', immediately followed by a $r1$ operation.

Table 3.3: Single-cell dynamic FPs and FFMs [19]

#	FFM	Fault Primitives
1	dRDF	$< 0r0r0/1/1 >, < 1r1r1/0/0 >,$ $< 0w0r0/1/1 >, < 1w1r1/0/0 >,$ $< 0w1r1/0/0 >, < 1w0r0/1/1 >$
2	dDRDF	$< 0r0r0/1/0 >, < 1r1r1/0/1 >,$ $< 0w0r0/1/0 >, < 1w1r1/0/1 >,$ $< 0w1r1/0/1 >, < 1w0r0/1/0 >$
3	dIRF	$< 0r0r0/0/1 >, < 1r1r1/1/0 >,$ $< 0w0r0/0/1 >, < 1w1r1/1/0 >,$ $< 0w1r1/1/0 >, < 1w0r0/0/1 >$
4	dTF	$< 0w0w1/0/- >, < 1w1w0/1/- >,$ $< 0w1w0/1/- >, < 1w0w1/0/- >,$ $< 0r0w1/0/- >, < 1r1w0/1/- >$
5	dWDF	$< 0w0w0/1/- >, < 1w1w1/0/- >,$ $< 0w1w1/0/- >, < 1w0w0/1/- >,$ $< 0r0w0/1/- >, < 1r1w1/0/- >$

In the FP notation, F denotes the behavior/state of the faulty cell: $F \in \{0, 1\}$. R denotes the logical value at the output of the memory: $R \in \{0, 1, -\}$. $R = -$ signifies that the last operation was a write operation and output data is not available. Based on the values of S, F and R, 30 dynamic single-cell FPs are determined. All these FPs can be compiled in 5 FFMs as shown in Table 3.3 [15]; they are explained next.

1. **Dynamic Read Destructive Fault (dRDF)**: An operation (i.e., read or write) followed immediately by a read operation, performed on a single cell changes the logical state of the cell, and returns an incorrect value on the output [15]. The 2-operation dRDF consists of six Fps:$< 0r0r0/1/1 >$, $< 1r1r1/0/0 >$, $< 0w0r0/1/1 >$, $< 1w1r1/0/0 >$, $< 0w1r1/0/0 >$ and $< 1w0r0/1/1 >$.

2. **Dynamic Deceptive Read Destructive Fault (dDRDF)**: An operation (i.e., read or write) followed immediately by a read operation, performed on a single cell changes the logical state of the cell, and returns a correct value on the output [15]. The 2-operation dDRDF consists of six Fps: $< 0r0r0/1/0 >$, $< 1r1r1/0/1 >$, $< 0w0r0/1/0 >$, $< 1w1r1/0/1 >$, $< 0w1r1/0/1 >$ and $< 1w0r0/1/0 >$.

3. **Dynamic Incorrect Read Fault (dIRF)**: A read operation performed immediately after an operation (i.e., read, transition, or non-transition write) on a single cell returns an incorrect value on the output, while the cell remains in its correct state [15]. The 2-operation dIRF consists of 6 FPs: $< 0r0r0/0/1 >$, $< 1r1r1/1/0 >$, $< 0w0r0/0/1 >$, $< 1w1r1/1/0 >$, $< 0w1r1/1/0 >$ and $< 1w0r0/0/1 >$.

4. **Dynamic Transition Fault (dTF)**: A transition write operation performed immediately after an operation (i.e., read, transition or non-transition write) fails [15]. The 2-operation

dTF consists of 6 FPs: $< 0w0w1/0/- >$, $< 1w1w0/1/- >$, $< 0w1w0/1/- >$, $< 1w0w1/0/- >$, $< 0r0w1/0/- >$ and $< 1r1w0/1/- >$.

5. **Dynamic Write Destructive Fault (dWDF)**: A non transition write operation, performed on a cell immediately after an operation (i.e., read, transition, non-transition write), causes the cell to flip. The 2-operation dWDF consists of 6 FPs: $< 0w0w0/1/- >$, $< 1w1w1/0/- >$, $< 0w1w1/0/- >$, $< 1w0w0/1/- >$, $< 0r0w0/1/- >$ and $< 1r1w1/0/- >$.

3.4.1.2 Two cell fault primitives

The two cell fault primitives, as seen in case of static faults, involves two cells: a-cell and a v-cell. In case of 2-operation dynamic faults, the sensitizing operations are classified into four categories, depending on the order in which sensitizing sequences are applied to the a-cell and to the v-cell. The four types of sensitizing operations are:

1. $\mathbf{S_{aa}}$: where both the sensitizing operations are applied to the a-cell, with the v-cell taking a certain state.
2. $\mathbf{S_{vv}}$: where both the sensitizing operations are applied to the v-cell, with the a-cell taking a certain state.
3. $\mathbf{S_{av}}$: where the first sensitizing operation is applied to the a-cell and the second operation to the v-cell.
4. $\mathbf{S_{va}}$: where the first sensitizing operation is applied to the v-cell and the second operation to the a-cell.

Faults caused due to S_{aa}

S_{aa}: These faults have the property that the application of two successive operation to the a-cell will cause the v-cell to flip. This can be represented using the fault primitive notation $< S_a; S_v/F/R >$. The sensitizing operation in the aggressor cell is of the form yO_1zO_2t, where O_1, O_2 are write or read operations and $y, z, t \in \{0, 1\}$ and the initial state of the victim cell an be 'x' where $x \in \{0, 1\}$. The flipping state of the cell can be represented by $\overline{x}$, while $R = -$ as no operation is performed on the victim cell. The Fault primitive is hence of the form $< yO_1zO_2t; x/\overline{x}/- >$. Fault primitives for different combinations of x,y,z,t,O_1, O_2 etc can be formed and grouped to form a functional fault model as shown in Table 3.4 [19].

- **Dynamic Disturb Coupling Fault (dCFds)**: In dCFds a write operation followed immediately by a read operation performed on the a-cell causes the v-cell to flip its logic state [22]. Depending on the operations performed on the aggressor cell dCFds can be divided in four sub-parts.
  - dCFds$_{ww}$: Operations applied in a write after write sequence. dCFds$_{ww}$ consists of 16 FPs: $< 0w0w0; x/\overline{x}/- >$, $< 1w1w1; x/\overline{x}/- >$, $< 0w0w1; x/\overline{x}/- >$, $< 1w1w0; x/\overline{x}/- >$, $< 0w1w0; x/\overline{x}/- >$, $< 1w0w1; x/\overline{x}/- >$, $< 0w1w1; x/\overline{x}/- >$ and $< 1w0w0; x/\overline{x}/- >$.

Table 3.4: Two-cell dynamic FPs and FFMs caused by S_{aa} [19]

#	FFM	Fault Primitives
1	dCFds$_{ww}$	$< 0w0w0; x/\overline{x}/- >, < 1w1w1; x/\overline{x}/- >,$ $< 0w0w1; x/\overline{x}/- >, < 1w1w0; x/\overline{x}/- >,$ $< 0w1w0; x/\overline{x}/- >, < 1w0w1; x/\overline{x}/- >,$ $< 0w1w1; x/\overline{x}/- >, < 1w0w0; x/\overline{x}/- >$
2	dCFds$_{wr}$	$< 0w0r0; x/\overline{x}/- >, < 1w1r1; x/\overline{x}/- >,$ $< 0w0r1; x/\overline{x}/- >, < 1w0r0; x/\overline{x}/- >$
3	dCFds$_{rw}$	$< 0r0w0; x/\overline{x}/- >, < 1r1w1; x/\overline{x}/- >,$ $< 0r0w1; x/\overline{x}/- >, < 1r1w0; x/\overline{x}/- >$
4	dCFds$_{rr}$	$< 0r0r0; x/\overline{x}/- >, < 1r1r1; x/\overline{x}/- >$

- dCFds$_{wr}$: Operations applied in a read after write sequence. dCFds$_{wr}$ consists of 8 FPs: $< 0w0r0; x/\overline{x}/- >$, $< 1w1r1; x/\overline{x}/- >$, $< 0w0r1; x/\overline{x}/- >$ and $< 1w0r0; x/\overline{x}/- >$.
- dCFds$_{rw}$: Operations applied in a write after read sequence. dCFds$_{rw}$ consists of 8 FPs: $< 0r0w0; x/\overline{x}/- >$, $< 1r1w1; x/\overline{x}/- >$, $< 0r0w1; x/\overline{x}/- >$ and $< 1r1w0; x/\overline{x}/- >$.
- dCFds$_{rr}$: Operations applied in a read after read sequence. dCFds$_{rr}$ consists of 4 FPs: $< 0r0r0; x/\overline{x}/- >$ and $< 1r1r1; x/\overline{x}/- >$.

Faults caused due to S_{vv}

S_{vv}: These faults have the property that the application of two successive operation to the v-cell for sensitization. This can be represented using the fault primitive notation $< S_a; S_v/F/R >$. The sensitizing operation in the victim cell is of the form yO_1zO_2t, where O_1, O_2 are write or read operations and $y, z, t \in \{0, 1\}$. 'F' is the resulting fault in the victim cells, while 'R' will be the logical output of the last read operation performed in a given sensitizing sequence i.e., data read out after a read operation in O_2. Fault primitives can be developed by taking different types of operations for O_1, O_2, along with different logic states for y, z, and t. These fault primitives can be grouped into different functional models as shown in table 3.5 [20].
S_{vv}: These faults have the property that the application of two successive operation to the v-cell for sensitization. This can be represented using the fault primitive notation $< S_a; S_v/F/R >$. The sensitizing operation in the victim cell is of the form yO_1zO_2t, where O_1, O_2 are write or read operations and $y, z, t \in \{0, 1\}$. 'F' is the resulting fault in the victim cells, while 'R' will be the logical output of the last read operation performed in a given sensitizing sequence i.e., data read out after a read operation in O_2. Fault primitives can be developed by taking different types of operations for O_1, O_2, along with different logic states for y, z, and t. These fault primitives can be grouped into different functional models as shown in table 3.5 [20].

1. **Dynamic Read Destructive Coupling Fault (dCFrd)**: A write operation followed immediately by a read operation performed on the v-cell changes the logical state of the v-cell and returns an incorrect value on the output, iff the a-cell is in a certain specific state [19]. The 2-operation dCFrd consists of 12 FPs as shown in table 3.5 [19], where $x \in \{0, 1\}$.

Table 3.5: Two-cell dynamic FPs and FFMs caused by S_{vv} [20]

#	FFM	Fault Primitives
1	dCFrd	$< x; 0r0r0/1/1 >, < x; 1r1r1/0/0 >,$ $< x; 0w0r0/1/1 >, < x; 1w1r1/0/0 >,$ $< x; 0w1r1/0/0 >, < x; 1w0r0/1/1 >$
2	dCFdrd	$< x; 0r0r0/1/0 >, < x; 1r1r1/0/1 >,$ $< x; 0w0r0/1/0 >, < x; 1w1r1/0/1 >,$ $< x; 0w1r1/0/1 >, < x; 1w0r0/1/0 >$
3	dCFir	$< x; 0r0r0/0/1 >, < x; 1r1r1/1/0 >,$ $< x; 0w0r0/0/1 >, < x; 1w1r1/1/0 >,$ $< x; 0w1r1/1/0 >, < x; 1w0r0/0/1 >$
4	dCFtr	$< x; 0w0w1/0/- >, < x; 1w1w0/1/- >,$ $< x; 0w1w0/1/- >, < x; 1w0w1/0/- >,$ $< x; 0r0w1/0/- >, < x; 1r1w0/1/- >$
5	dCFwd	$< x; 0w0w0/1/- >, < x; 1w1w1/0/- >,$ $< x; 0w1w1/0/- >, < x; 1w0w0/1/- >,$ $< x; 0r0w0/1/- >, < x; 1r1w1/0/- >$

2. **Dynamic Deceptive Read Destructive Coupling Fault (dCFdrd)**: A write followed immediately by a read operation on the v-cell changes the data in the v-cell and returns a correct value on the output, iff the a-cell is in a certain specific state [19]. The 2-operation dCFdrd consists of 12 FPs as shown in table 3.5 [19], where $x \in \{0, 1\}$.

3. **Dynamic Incorrect Read Coupling Fault (dCFir)**: A write or read operation followed immediately by a read operation performed on the v-cell returns an incorrect value on the ouput, while the c-cell remains in its correct state, iff the a-cell is in a certain specific logic state [20]. The 2-operation dCFir consists of 12 FPs as shown in table 3.5 [19], where $x \in \{0, 1\}$.

4. **Dynamic Transition Coupling Fault (dCFtr)**: A write or a read operation followed immediately by a transition write operation performed on the v-cell results in a failing write operation iff the a-cell is in a certain specific state. The 2-operation dCFtr consists of 12 FPs as shown in table 3.5 [19], where $x \in \{0, 1\}$.

5. **Dynamic Write Destructive Coupling Fault (dCFwd)**:A read or write operation followed immediately by a non-transition write operation performed on the v-cell cause that cell to flip, iff the a-cell is in a certain specific logic state [20]. The 2-operation dCFwd consists of 12 FPs as shown in table 3.5 [19], where $x \in \{0, 1\}$.

Faults caused due to S_{av}

S_{av}: These faults have the property that the application of of an operation to the a-cell, followed immediately by an operation to the v-cell, sensitizes a fault in the v-cell. The fault primitive notation is given by $< S_a; S_v/F/R >$. $< S_a >$ will be shown as xO_1y and $< S_v >$

Table 3.6: Two-cell dynamic FPs and FFMs caused by S_{av} [19]

#	FFM	Fault Primitives
1	dCFrd	$< xOy; 0r0/1/1 >, < xOy; 1r1/0/0 >$
2	dCFdrd	$< xOy; 0r0/1/0 >, < xOy; 1r1/0/1 >$
3	dCFir	$< xOy; 0r0/0/1 >, < xOy; 1r1/1/0 >$
4	dCFtr	$< xOy; 0w1/0/- >, < xOy; 1w0/1/- >$
5	dCFwd	$< xOy; 0w0/1/- >, < xOy; 1w1/0/- >$

will be shown as zO_2t, where O_1 and O_2 denote the first and second operation receptively, which can be either a write or a read and $y, z, t \in 0, 1$. 'F' is the fault that appears and 'R' is the output after the read operation, in O_2. Fault primitives are developed based on several combinations of the sensitizing operations, and fault. These fault primitives are grouped and categorized into functional models as show in table 3.6 [19].

1. **Dynamic Read Destructive Coupling Fault (dCFrd)**: A read operation performed on the v-cell performed immediately after a read or a write operation in an a-cell, causes a transition in the v-cell and returns the new incorrect value on the output. The 2-operation dCFrd consists of 12 FPs as shown in table 3.6 [19], where $x, y \in \{0, 1\}$.

2. **Dynamic Deceptive Read Destructive Coupling Fault (dCFdrd)**: A read operation performed on the v-cell performed immediately after a read or a write operation in an a-cell, causes a transition in the v-cell and returns the old correct value on the output. The 2-operation dCFdrd consists of 12 Fps as shown in table 3.6 [19], where $x, y \in \{0, 1\}$.

3. **Dynamic Incorrect Read Coupling Fault (dCFir)**: A read operation performed on the v-cell, performed immediately after a read or a write operation in an a-cell, returns an incorrect value of output while the cell retains the correct logic state. The 2-operation dCFir consists of 12 FPs as shown in table 3.6 [19], where $x, y \in \{0, 1\}$.

4. **Dynamic Transition Coupling Fault (dCFtr)**: A transition write operation performed on the v-cell immediately after a read or a write operation in an a-cell, results in transition of the logic state in v-cell. The 2-operation dCFtr consists of 12 FPs as shown in table 3.6 [19], where $x, y \in \{0, 1\}$.

5. **Dynamic Write Destructive Coupling Fault (dCFwd)**: A non transition write operation performed on the v-cell after a read or a write operation in an a-cell, results in transition of the logic state in v-cell. The 2-operation dCFwd consists of 12 FPs as shown in table 3.6 [19], where $x, y \in \{0, 1\}$.

Faults caused due to S_{va}

S_{va}: In this type of two cell fault, first the operation is performed over the v-cell, and then followed by the operation in the a-cell leading to a fault being sensitized in the v-cell. The fault primitive is of the form $< S_a; S_v/F/R >$. $< S_a >$ will be shown as xO_2y and $< S_v >$ will be

Table 3.7: Two-cell dynamic FPs and FFMs caused by S_{va} [19]

#	FFM	Fault Primitives
1	dCFrd	$< 0r0/1/1; xOy >, < 1r1/0/0; xOy; >$
2	dCFdrd	$< 0r0/1/0; xOy >, < 1r1/0/1; xOy >$
3	dCFir	$< 0r0/0/1; xOy >, < 1r1/1/0; xOy >$
4	dCFtr	$< 0w1/0/-; xOy >, < 1w0/1/-; xOy >$
5	dCFwd	$< 0w0/1/-; xOy >, < 1w1/0/-; xOy >$

shown as zO_1t, where O_1 and O_2 denote the first and second operation receptively, which can be either a write or a read and $y, z, t \in 0, 1$. 'F' is the fault that appears and 'R' is the output after the read operation, in O_2. Fault primitives are developed based on several combinations of the sensitizing operations, and fault. These fault primitives are grouped and categorized into functional models as show in table 3.7 [19].

1. **Dynamic Read Destructive Coupling Fault (dCFrd)**: A read operation performed on the a-cell after a read or a write operation in an v-cell, causes a transition in the v-cell and returns the new incorrect value at the output. The 2-operation dCFrd consists of 12 FPs as shown in table 3.7 [19], where $x, y \in \{0, 1\}$.

2. **Dynamic Deceptive Read Destructive Coupling Fault (dCFdrd)**: A read operation performed on the a-cell after a read or a write operation in an v-cell, causes a transition in the v-cell and returns the old correct value at the output. The 2-operation dCFdrd consists of 12 Fps as shown in table 3.7 [19], where $x, y \in \{0, 1\}$.

3. **Dynamic Incorrect Read Coupling Fault (dCFir)**: A read operation performed on the a-cell after a read or a write operation in a v-cell, returns an incorrect value on the output while the cell contains the correct logic value. The 2-operation dCFir consists of 12 FPs as shown in table 3.7 [19], where $x, y \in \{0, 1\}$.

4. **Dynamic Transition Coupling Fault (dCFtr)**: A transition write operation performed on the a-cell after a read or a write operation in an v-cell, results in transition of the logic state in v-cell. The 2-operation dCFtr consists of 12 FPs as shown in table 3.7 [19], where $x, y \in \{0, 1\}$.

5. **Dynamic Write Destructive Coupling Fault (dCFwd)**:A non transition write operation performed on the a-cell after a read or a write operation in the v-cell, results in transition of the logic state in v-cell. The 2-operation dCFwd consists of 12 FPs as shown in table 3.7 [19], where $x, y \in \{0, 1\}$.

3.4.2 Dynamic Address Decoder faults

The address decoders consists of row decoders and column decoders. Dynamic faults in address decoder are also refereed as delay faults. These delay faults may be caused due to the presence of resistive open defects present within the gates or in between the gates in an row or column decoder. Fig 3.6 [18] shows the two main types of delay faults in address decoder:

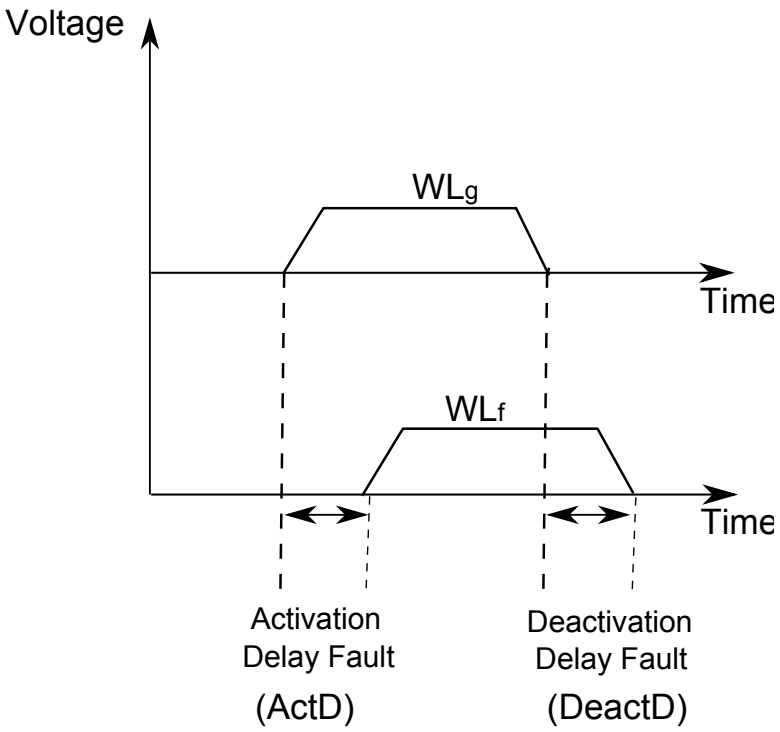


Figure 3.6: Dynamic address decoder faults [18]

1. **Activation delay fault**: The activation delay fault refers to the slow activation of the word line signal, when it is selected by appropriate transition of the address line selecting it. This results in improper operations being performed over the cell, leading to faulty behavior of the memory system [18].

2. **Deactivation delay fault**: The deactivation delay fault refers to the slow deactivation of the word line signal, when a given word line is deselected by appropriate transitions in the address lines. This may result in more than one cell being active for a small period of time, resulting in undesired operations being performed over the good cells [18].

3.4.3 Dynamic faults in Peripheral Circuits

The dynamic faults in peripheral circuits are mostly speed related faults from write driver, sense amplifier etc. or faults due to excessive leakage of pass transistors [39]. There are 4 main types of dynamic faults which are seen in the peripheral circuits:

1. **Slow write driver fault**: The slow write driver fault may be caused due to resistive open defects in the write driver. Presence of this defect can lead to improper discharging of one of the two bit lines, during a write operation. This leads to a lesser differential voltage between the two bit lines, which may lead to the failure in write operation. The worst case scenario, for the bit lines is to perform complementary write operations. Complementary write operations, when performed back to back using the same write driver forms the worst case scenario for detection of slow write driver fault.

2. **Slow sense amplifier fault**: The slow sense amplifier fault may be caused due to the presence of resistive opens in the read path between the cell and the sense amplifier. The defect may cause an offset voltage, which will lead the sense amplifier to produce a wrong output. The worst case condition for detecting this fault is to perform a read operation, after performing a write operation with a complementary value. For e.g., a write '1' operation may be performed a given cell, then a read '0' operation in other cell sharing the same sense amplifier should be performed for strong sensitization of slow sense amplifier fault.

3. **Slow pre-charge circuit fault**: The slow pre-charge circuit fault, may arise due to improper pre-charging of bit lines. The pre-charging of bit lines is very important, to ensure that the operations are done completely independently with respect to each other, i.e., to avoid the influence of one operation over the other. But in the presence of a defect the bit lines may not be pre-charged fully and the state of the bit line is as influenced by the previous operation, rather than that of the current operation to be performed. This might lead to a faulty behavior of the system. The strong sensitization condition for this fault is similar to that of slow sense amplifier fault.

4. **Bit line imbalance fault**: The bit line imbalance fault is mainly caused due to the leaky pass transistors. This is becoming more common especially with down scaling of technology. The effects of leakage current might influence a read operation. The differential voltage developed between the bit lines during a read operation, may be neutralized by the pass transistor, thereby leading to a faulty read operation. This effect is strongly sensitized when a read operation of a particular data, is performed on a cell, which is present in a column where all the other cell store the complementary data values.

3.5 Strong vs Weak faults

As stated earlier, defects at fabrication level production of chips are inevitable. The size of the defect determines the functional behavior of the system. If the size of the defect is very large, it leads to a *strong fault*. Strong faults, severely affect functionality of the system by causing an error, which leads to the failure of the system. Such faults are easily fully sensitized and detected by using appropriate detection conditions. However, if the defects are not large enough to cause an erroneous behavior of the system, it leads to *weak faults*. Weak faults causes small disturbances within the system. These disturbances are within the tolerable limits, thereby leading to proper functioning of the system. However, presence of multiple weak faults can cause an error in the system as their fault effects can be additive in nature. Detection of such multiple weak faults require simultaneous sensitization of all the multiple weak faults. This approach has been successfully verified in case of multi port memories, which led to the development of number of functional fault models based on weak faults [14].

4 New memory test paradigm

Rapid and continuous technology scaling, in semiconductor memories have led the faults to be more pervasive and weak rather than strong. Weak faults are faults, that do not cause an error. However, multiple weak faults can have additive effects that do not cause the memory to fail(i.e., do not cause an error). The proposed new memory test paradigm aims at detection of such faults.

This chapter begins with a brief description of notation for test algorithms and stresses which will be used through out this thesis; it then describes the traditional testing approach that is used for memory testing and highlights its missing link. It then discusses the proposed new memory test paradigm. Section 4.1 introduces the notation of test algorithms. Section 4.2 introduces the traditional approach of memory testing which assumes a single (strong) defect at a time in the memory system. Section 4.3 describes the new memory test approach which considers the presence of two weak faults at a time in different blocks of the memory system. Section 4.4 describes the new memory test approach when considering three weak faults at a time in different blocks of the memory system. Section 4.5 generalizes the new test approach.

4.1 Notation of test algorithms and stresses

A test consists of a Base Test 'BT', applied using a particular Stress Combination 'SC'. A BT forms the main test algorithm, whereas an SC consists of a combination of values for the different stresses; e.g., VDD = 1.8V, Temp = 70C, etc. Most BTs have a march test format. March test algorithms are the most common algorithms used for testing memories [40]. A *march test* consists of a finite sequence of march elements. A *March Element (ME)* is a finite sequence of operations applied to every cell in the memory before proceeding to the next cell. A complete march test is delimited by the '$\{...\}$' bracket pair; while a march element is delimited by the '$(...)$' bracket pair. The march elements are separated by semicolons, and the operations within a march element are separated by commas. The way one proceeds to the next cell is determined by the *Address Orders (AOs)* which can be an increasing address order (e.g., increasing AO from the cell 0 to the cell $n-1$), denoted by $\Uparrow$ symbol, or a decreasing AO, denoted by $\Downarrow$ symbol, and which is the exact inverse of the $\Uparrow$ AO. When the AO is irrelevant, the symbol $\Updownarrow$ (i.e., $\Uparrow$ or $\Downarrow$) will be used. An example of a march algorithm is MATS+ [27], defined as:

MATS+: $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$

MATS+ consists of three MEs, which are separated by the ';' symbol. The ME '$\Uparrow (r0, w1)$' specifies the $\Uparrow$ AO, while to each address a read operation with expected logic value '0' will be applied, after which a logic '1' will be written.

A BT can be applied with different stresses. The stresses can be divided into two types i.e., *algorithmic* and *non algorithmic* stresses [39].

- A *non-algorithmic* stresses is also referred as an environmental stress, as it specifies the environmental values, such as the supply voltage, the temperature, the timing (the clock frequency), etc.; they are effective during the application of the test [39].
- An *algorithmic* stress specifies the way the test is performed, and therefore it influences the sequence and/or the type of the memory operations performed. The most known algorithm stresses are the *address direction*, *counting method* and *data-background* [39] [18].

Next, the three different algorithmic stresses will be discussed.

Address direction is the addressing extension of the one-dimensional 'AO' to the two dimensional space of the memory cell array [39]. A real memory consists of a number or rows and columns (and thus also of a number of diagonals). The address directions specifies the direction (i.e., rows, columns, or diagonals) in which the address sequence has to be performed. The commonly used address directions in the industry are:

- **Fast row (fr)**: In fr addressing, each address increment or decrement operation causes an adjacent physical row to be accessed.
- **Fast column (fc)**: In fc addressing, each address increment or decrement operation causes an adjacent physical column to be accessed.

- **Fast diagonal (fd)**: In fD addressing, each address increment or decrement operation causes an adjacent physical diagonal to be accessed. Fast D is used less frequently in industry [39].

Counting method determines the address sequence. It has been shown that the counting method is important for detecting AD delay faults. Some of the counting methods used are linear, Address Complement (Ac), 2^i, H1 addressing etc [18].

- **Linear counting method**: This is the most common counting method used. It is denoted by the superscript 'L of the AO (e.g., $^{L}\Uparrow$), where 'L' specifies the address sequence 0,1,2,3, etc. Because it is the default counting method, the superscript 'L is usually deleted.
- **Address Complement counting method**: It specifies an address sequence: 000, **111**, 001, **110**, 010, **101**, 011, and **100**; each bold address is the 1s complement of the preceding address.
- **2^i counting method**: This counting method is typically used by the MOVI algorithm It repeats the PMOVI algorithm 'N' times (N = is the number of memory address bits) with an address increment/decrement value of 2^i; with $0 \leq i \leq N-1$.
- **H1 counting method**: It specifies an address sequence: 000, 001, 000, 010, 000, 100, 000; each address has a hamming distance of '1' with respect to the preceding address.

Data Background 'DB' is the pattern of zeros and ones as seen in an array of memory cells [39]. The most common types of DBs are:

- **Solid (sDB)**: (0000.../0000...) or (1111.../1111...)
- **Checkerboard (cDB)**: (0101.../1010.../0101.../1010...)
- **Column Stripe (cDB)**: (0101.../0101.../0101.../0101..)
- **Row Stripe (rDB)**: (0000.../1111.../0000.../1111..)

4.2 Traditional memory test approach

Traditionally fault modeling, is done by injecting a *single defect at a time* -such as a resistor- either in the MCA, the AD or in the PC. Electrical simulation (e.g., SPICE) is then performed and the faulty behavior is analyzed using either static analysis or dynamic analysis [42, 11, 22]. Depending on how severe the defect is (e.g., how big is the value of the resistor representing the defect), the fault may or may not cause an error and therefore cause the memory to fail. If the fault causes an error, then it is mapped into a functional fault model. An appropriate detection condition is then developed and compiled thereafter in a test algorithm being able to detect the observed fault model. In the rest of the section the traditional approach using circuit simulation for three main blocks of a memory system; namely the MCA, AD, and the PCs will be discussed. For circuit simulation, a defect will be modeled as a resistor (e.g., opens, ridges etc) with variable value.

4.2.1 Detecting faults in Memory Cell Array

Fig 4.1 shows some of the possible locations of resistive opens that could occur within the cell. Due to symmetrical structure of the SRAM cell, many defects will cause complementary fault behavior. For example, the defects OC1s and OC1c in Figure 4.1 cause complementary fault behavior. By simulating one defect, one can derive the fault behavior that can be caused by the other defect.

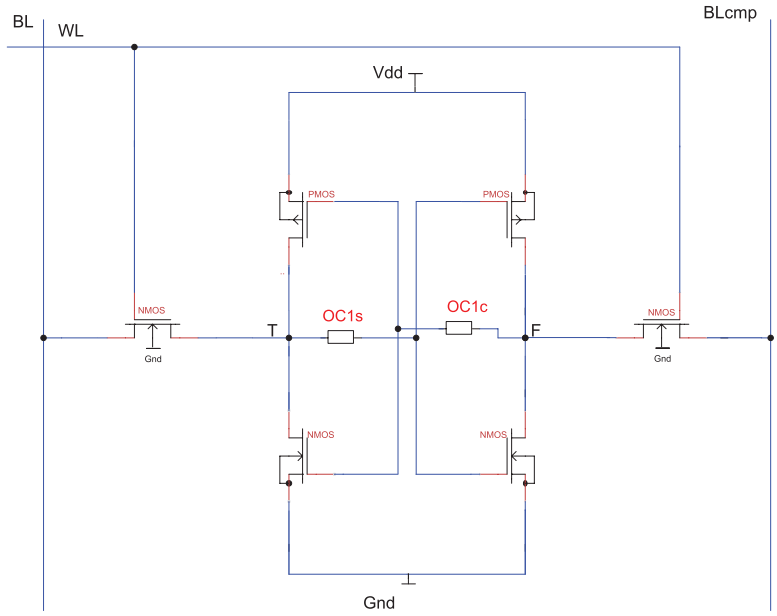


Figure 4.1: Opens within a cell

Consider the case of a single resistive open defect between the two cross coupled inverters (e.g., OC1s or OC1c in Fig 4.1). The behavior of the SRAM cell in the presence of such a defect will strongly depend on the size of the defect 'R_{def}' (i.e., magnitude of the resistance representing the defect). Therefore the circuit simulation will be performed for three cases:

1. Defect free case
2. R_{def} has a large value
3. R_{def} has a small value

For all three cases the sequence "S=1w0, r0, w1, r1" is performed. This sequence consists of (1) Initialize the cell to '1' and perform write '0' operation, (2) perform read '0' operation, (3) perform write '1' operation, and (4) perform read '1' operation. Fig 4.2 shows the voltage of the true node of the cell for the simulated sequence for the three cases.

1. Defect free case: The top graph in Figure 4.2 shows the voltage of the true node when simulating the sequence 'S'. As the graph shows, all the operations pass correctly and the voltage

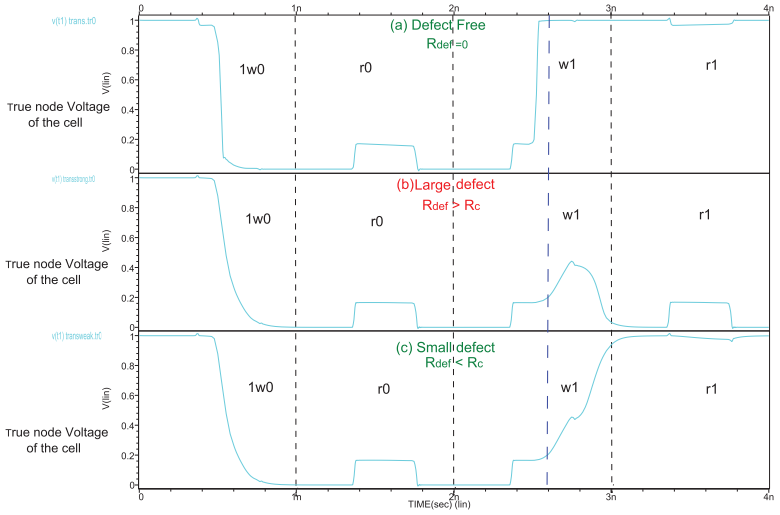


Figure 4.2: True node voltage of a cell without and with defects

of the true node switches form logic state '1' to logic state '0' during the '1w0' operation, and correctly switches from logic '0' state to logic '1' state during '0w1' operation. During the read operations, the cell keeps its state and there is a small increase in voltage of the true node (i.e., read bump voltage) due to the discharge of the bit line voltage through the pull down transistor of the cell (see Fig 4.2).

2. Defect with large value: The middle graph in Figure 4.2 shows the voltage of the true node when simulating the sequence 'S' for large value of the defect. The large value of this defect, induces a large delay (by combining with the circuit capacitance) to the input signal driving the opposite inverter. Consequently, it becomes very difficult for the cell to flip from logic state '1' to logic state '0'. It can be seen from the graph, the write '1' operation fails (i.e., the third operation) and the cell remains in logic state '0'. This faulty behavior of the cell will cause an error in the memory resulting into a memory failure. This faulty behavior is known as transition fault. The FP for the transition fault is given by $< 0w1/0/- >$ (see chapter 3). Once the fault behavior is identified and mapped to a fault model, a detection condition can be developed in order to compile it in a test algorithm. In order to detect this transition fault observed in this case, a transition write operation, followed by a read operation should be performed. The detection condition can be compiled to a march test as follows:

$$\{\Updownarrow (w0); \Updownarrow (w1); \Updownarrow (r1)\}$$

3. Defect with small value: The bottom graph in Figure 4.2 shows the voltage of the true node when simulating the sequence 'S' for small value of the defect. The defect induces a delay over the input signal driving the opposite inverter. However, the delay is not large enough for the write operation to fail. It is clear from the graph that the operation passes correctly. Nevertheless, careful inspection of the graph reveals that 'w1' operation does not pass smoothly as in defect free case (see top graph in Fig 4.2). The true node is pulled up to vdd voltage about 0.4ns later as

compared with the defect free case; hence, the switching of the cell is delayed due to the defect. This delay does not cause an 'error', it rather causes a 'weak fault'.

Figure 4.3 illustrates the difference between the fail (strong fault) and pass (weak faults) regions depending on the value of the defect as we externally experience it. If the defect is beyond a certain value, say R_c, then a strong fault will be sensitized, leading to an error in the memory system which can be detected when appropriate tests are used. However, below this value, the fault will be not detected. Because when performing memory fault modeling and test generation we assume the presence of a single fault at a time, the defects below say R_c(causing *weak faults*) will never be detected.

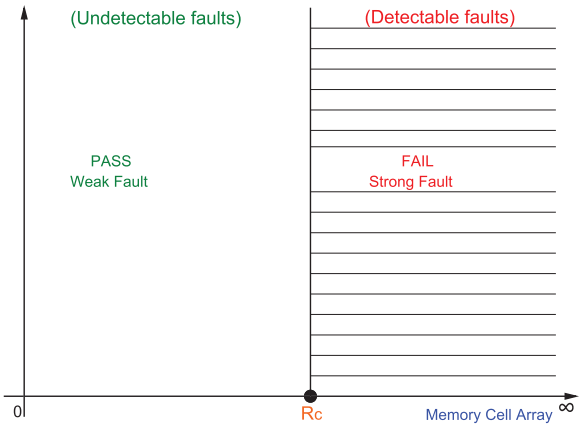


Figure 4.3: Detected vs non-detected defects in MCA

4.2.2 Detecting faults in Address Decoders

The AD consist of row decoder and column decoder. Fig 4.4(a) shows a 2-input static CMOS NAND based row decoder. The two address lines 'AD0' and 'AD1' are used for selection of the word lines. The resistive opens may be present in AD and can lead to faulty behavior of the memory system. Depending on the location of the defect the opens present in AD may be classified as *intergate opens* or *intragate opens* [18]. If defects are present in between the gates or in between the address line and input line of any given gate then they are referred to intergate opens. On the other hand, if defects are present within the gates as shown in Fig 4.4(b) (between transistors), then they are referred to intragate opens.

In the rest of this section the traditional approach to develop test algorithms for faults in AD will be discussed. Consider the case of a intergate open defect as shown in Fig 4.4(a). The behavior of the word line signal in the presence of such a defect will strongly depend on the size of the defect 'R_{def}' (i.e., magnitude of the resistance representing the defect). Therefore the circuit simulation will be performed for four cases:

1. Defect free case
2. R_{def} has a large value

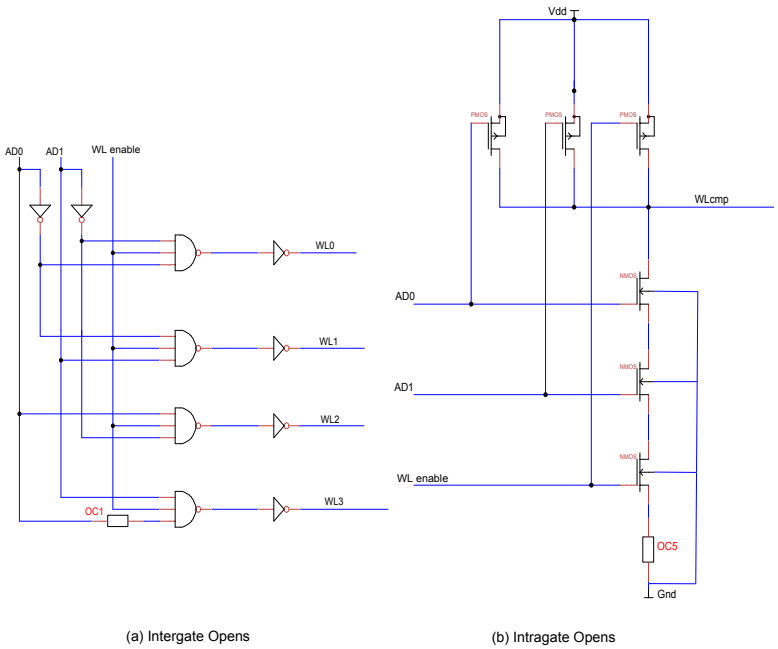


Figure 4.4: Defects in address decoder

3. R_{def} has a intermediate value

4. R_{def} has a small value

For all the four cases, the inputs 'AD0' and 'AD1' (see Fig 4.4) are set in order to select the word line 'wl3'. The simulation results are shown in Fig 4.5 and will be explained next.

1. Defect free case: The word line'wl3' signal for a defect free case is indicated in Fig 4.5. It can be observed that the 'wl3' signal is activated at the right time and is switched 'ON' for the required duration. Therefore, any operations performed on cells accessed by word line'wl3' passes.

2. Defect with large value: The 'wl3' signal for a defect with large values (e.g., $R_{def} = 200k$) is indicated in Fig 4.5. It can be observed that the 'wl3' signal is stuck at logic '0' i.e., the 'wl3' is never activated. Consequently, any operations performed on cells accessed by 'wl3' fails. This is a faulty behavior which will cause error in the memory resulting into a memory failure. This faulty behavior is known as stuck at fault and can be denoted by $< \forall/0 >$. Once the fault behavior is identified and mapped to a fault model, a detection condition can be developed in order to compile it in a test algorithm. In order to detect this stuck at fault observed in this case, two conditions should be satisfied:

1. $\Uparrow (rx...w\overline{x})$

2. $\Downarrow (r\overline{x}wx)$ where, $x \in \{0, 1\}$; '...' means any number of read and writes can be performed

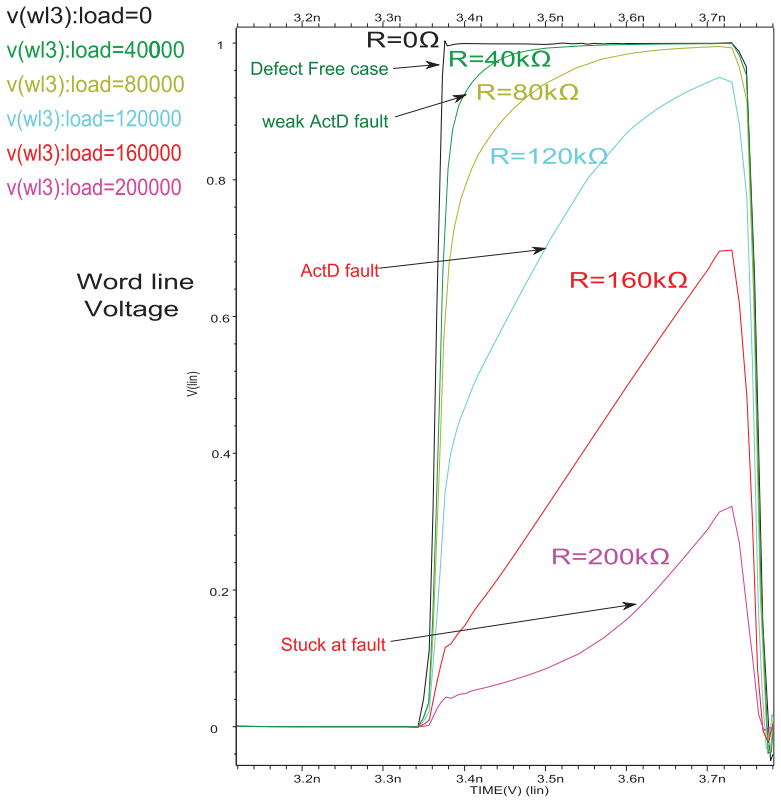


Figure 4.5: Simulation results for row decoder

These two detection conditions can be compiled in the form of a test algorithm. For e.g., MATS+ algorithm uses this detection condition to detect the static ADFs [27].

MATS+: $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$

3. Defect with intermediate value: The word line 'wl3' signal for a defect with intermediate values (e.g., $R_{def} = 120k$) is indicated in Fig 4.5. The defect induces a delay in the input signal driving the NAND gate. When the input line undergoes a transition from logic '0' to logic '1', the word line signal 'wl3' is activated (or pulled up) only after a certain delay. Hence, operations performed in the memory may not be fully accomplished as the cell is not made accessible to the bit lines for sufficient duration. This faulty behavior which will cause an error in the memory, resulting into a memory failure. This faulty behavior is known as *Activation delay fault* (ActD) [18] (see chapter 3.4.2). Once the fault behavior is identified and mapped to a fault model, a detection condition can be developed in order to compile it in a test algorithm. In order to sensitize the activation delay fault observed in this case, two conditions must be satisfied [18]:

1. Sensitizing address transitions
2. Sensitizing operation sequences

Sensitizing address transitions Sensitizing address transition(s) can be caused by an address pair or an address triplet. A Sensitizing Address Pair (SAP) consists of a sequence of two addresses $A_g; A_f$ or $A_f; A_g$ (where A_f and A_g are addresses of faulty and good cell respectively), which have to be applied in sequence because ADFs are sensitized by address transitions. When the two SAPs, $A_g; A_f$ and $A_f; A_g$, are applied in sequence, the Sensitizing Address Triplet (SAT) $A_g; A_f; A_g$ can be applied instead. The addresses of the SAPs/SATs, required for sensitizing ADFs, are generated using an Counting Method (CM) (see section 4.1). To each address of a SAP or a SAT at least one operation has to be applied, resulting in a Sensitizing Operation Sequence (SOS) consisting, respectively, of two operations for a SAP and three operations for a SAT since a least one operation has to be applied to each address of a SAP (SAT) [18].

These two detection conditions can be compiled in the form of a test algorithm. For e.g., An SAT using AC complement counting method with RAR SOS can be combined to form a march test as shown [18]:

$$\{\Updownarrow (w0);^{AC} \Uparrow (r0, w1, r1);^{AC} \Uparrow (r1, w0, r0);^{AC} \Downarrow (r0, w1, r1);^{AC} \Downarrow (r1, w0, r0);\}$$

3. Defect with low value: The word line 'wl3' signal for a defect with low values (e.g., $R_{def} = 40k$) is indicated in Fig 4.5. The defect induces a delay in the input signal driving the NAND gate. When the input line undergoes a transition from logic '0' to logic '1', the word line signal 'wl3' is activated (or pulled up) only after a certain delay. However, the delay in the word line signal does not lead to a failure of any operation because the bit lines are connected to the cell for a sufficient duration. Hence, the delay in word line signal does not cause an 'error', it rather causes a 'weak fault'.

Figure 4.6 illustrates the difference between the fail (strong fault) and pass (weak faults) regions depending on the value of the defect as we externally experience it. If the defect is beyond a certain value, say R_{c1}, then a strong static fault will be sensitized, leading to an error in the memory system which can be detected when appropriate tests are used. If the defect is beyond a certain value, say R_{c2}, then a strong dynamic fault (ActD) will be sensitized, leading to an error in the memory system which can be detected when appropriate tests are used. However, below this value, the fault will be not detected. Because when performing memory fault modeling and test generation we assume the presence of a single fault at a time, the defects below say R_{c2}(causing *weak faults*) will never be detected.

4.2.3 Detecting faults in Peripheral Circuits

The PC consist of write circuitry and read circuitry. In this section the traditional approach to develop test algorithms for faults in write driver of PC will be discussed. Consider the case of a defective write driver as shown in Fig 4.7. The behavior of the write driver in the presence of such a defect will strongly depend on the size of the defect 'R_{def}' (i.e., magnitude of the resistance representing the defect). Therefore the circuit simulation will be performed for four cases:

1. Defect free case

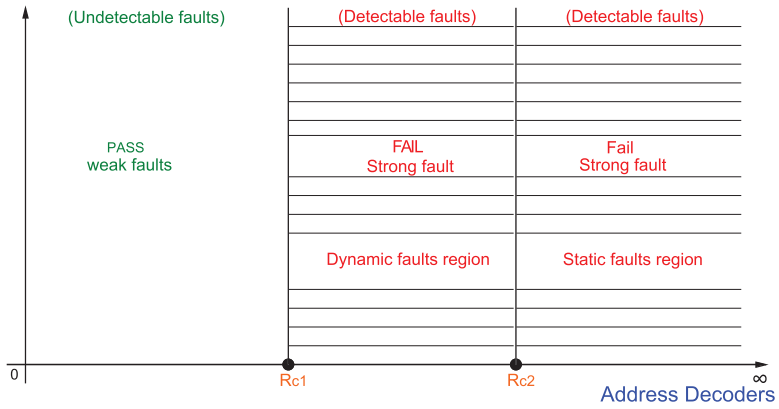


Figure 4.6: Detected vs not detected defects in AD

2. R_{def} has a large value
3. R_{def} has a intermediate value
4. R_{def} has a small value

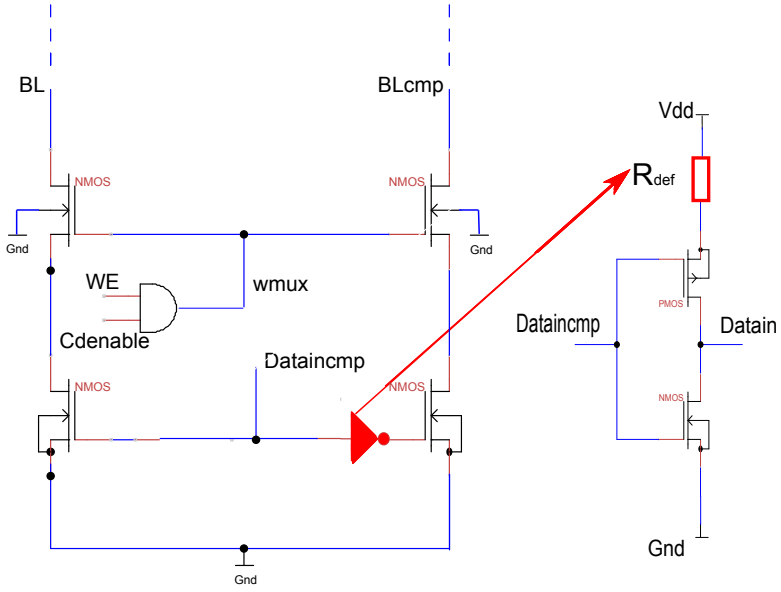


Figure 4.7: Defective write driver in peripheral circuitry

The simulation results for all four cases are shown in Fig 4.8 and will be explained next.

1. Defect free case: The bit line'BL2cmp' signal for a defect free case ($R_{def} = 0k$) is indicated in Fig 4.8. It can be observed that the 'BL2cmp' signal is completely pulled down

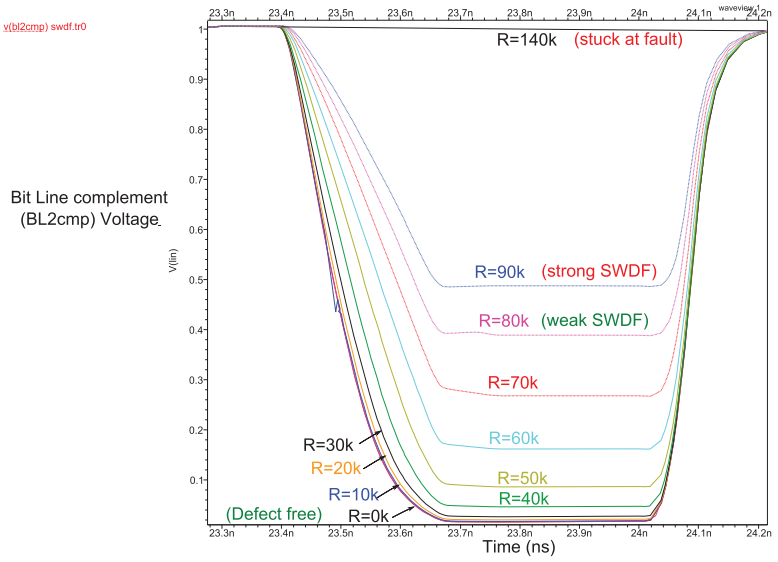


Figure 4.8: Simulation results for write driver

when a write one operation is performed on a cell. The operation passes smoothly and hence no error is observed.

2. Defect with large value: The 'BL2cmp' signal for a defect with large values (e.g., $R_{def} = 140k$) is indicated in Fig 4.8. It can be observed that the 'BL2cmp' signal is stuck at logic '1' i.e., the 'BL2cmp' is never pulled down during a write one operation. Consequently, write one operation performed on a given cell along the bit line 'BL2cmp' fails. This is a faulty behavior which will cause error in the memory resulting into a memory failure. This faulty behavior is known as stuck at zero fault and can be denoted by $< \forall/0 >$. Once the fault behavior is identified and mapped to a fault model, a detection condition can be developed in order to compile it in a test algorithm. The MATS+ test algorithm can be used to detect stuck at faults (see section 4.2.2).

3. Defect with intermediate value: The word line 'BL2cmp' signal for a defect with intermediate values (e.g., $R_{def} = 90k$) is indicated in Fig 4.8. The defect induces a delay in pulling down the bit line signal, thus making the write driver slow. The result will be that the differential voltage on the bit lines (BLs) during the write operation is reduced. This may cause the cell not to be written. This faulty behavior which will cause an error in the memory, resulting into a memory failure. This faulty behavior is known as *Slow Write Driver Fault* (SWDF) [39] (see chapter 3.4.3). Once the fault behavior is identified and mapped to a fault model, a detection condition can be developed in order to compile it in a test algorithm. In order to provide maximum stress for sensitization of the SWDF two back to back complementary write operations can be performed using the same write driver i.e., using fast row address direction. The following march test can be used to detect SWDF when performed in fast row with a solid data background.

$\{\Updownarrow (wD); \Uparrow_r (w\overline{D}, r\overline{D}, wD); \Updownarrow (w\overline{D}); \Uparrow_r (wD, rD, w\overline{D})$; where, $D \in \{sDB \; or \; cDB\}$

3. Defect with low value: The bit line 'BL2cmp' signal for a defect with low values (e.g., $R_{def} = 80k$) is indicated in Fig 4.8. The defect induces a delay in pulling down the bit line signal, thus making the write driver slow. The result will be that the differential voltage on the bit lines (BLs) during the write operation is reduced. However, the differential voltages on the bit lines are sufficient for the write operations to complete successfully. Hence, the delay in pulling down the bit line signal does not cause an 'error', it rather causes a 'weak fault'.

Figure 4.8 illustrates the difference between the fail (strong fault) and pass (weak faults) regions depending on the value of the defect as we externally experience it. If the defect is beyond a certain value, say R_{c1}, then a strong static fault will be sensitized, leading to an error in the memory system which can be detected when appropriate tests are used. If the defect is beyond a certain value, say R_{c2}, then a strong dynamic fault (SWDF) will be sensitized, leading to an error in the memory system which can be detected when appropriate tests are used. However, below this value, the fault will be not detected. Because when performing memory fault modeling and test generation we assume the presence of a single fault at a time, the defects below say R_{c2}(causing *weak faults*) will never be detected.

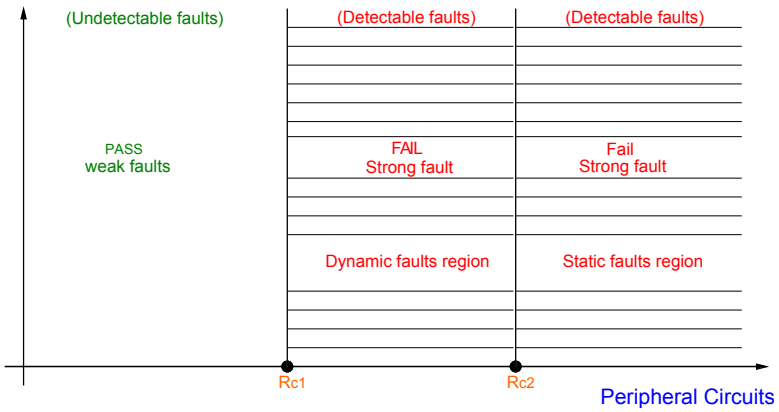


Figure 4.9: Detected vs not detected defects in PC

4.2.4 Limitation of the traditional approach

It is clear from the previous subsections that the traditional memory test approach guarantees the detection of defects only if the value of the defect is larger than a certain critical value. A defect may be present in the memory and it escapes the memory test if it only causes a weak fault. The presence of multiple weak faults in the memory system may result in an error during the application if their fault effects are additive and the application simultaneously sensitizes the weak faults. Newly emerging complex failure mechanisms in the nano-era (which are not understood yet), are causing the fault mode of the chips to be dominated by transient, intermittent and weak faults rather than hard and permanent faults; hence causing more reliability problems than quality problems [4, 5, 17, 41]. Many companies are reporting not being able to explain all electronic failures with the existing approaches [24, 28, 43]. For instance, AUDI reported from all electronic failures, 35% can be mapped using the existing approaches into well defined

semiconductor failures, while 41% cannot be understood (NTF: No Trouble Found) [24]. This shift in failure mechanisms is therefore seriously impacting the quality and reliability, which means that the design of future systems fabricated using nanotechnology is a major challenge. In turn, this will demand *revolutionary changes* in how future systems are designed and *tested* to meet the increasing quality requirements on such systems.

4.3 Two weak faults test approach

As already mentioned, technology scaling is causing more complex failure mechanisms having small disturbances in different parts of memory system such as address decoders and read/write circuitry [16, 26, 28]. Therefore, taking the fault effects of such disturbances together into consideration while developing fault models and designing test algorithms is required not only to increase the outgoing product quality and reduce the number of escapes, but also to contribute to the reduction of NTFs as well. In the rest of this section, the new approach will be explained and illustrated. As already mentioned, the basic idea is to consider the combined fault effects of *multiple weak* faults. Multiple weak faults, leading to a strong fault and therefore to an error, may escape the test program due to lack of simultaneous sensitization of the concerned weak faults. Hence, test algorithm development should be adapted in such a way that the sensitization conditions of multiple weak faults have to be considered *simultaneously*; i.e., the sensitization conditions have to combined into a *single* condition. In order to illustrate the scenario of multiple weak faults, the reduced memory system consisting of three main blocks i.e., MCA, AD and PC as shown in Fig 4.10 is used. It is further assumed that each of the three blocks may suffer from a weak fault. Fig 4.10 shows the location of weak defects in three blocks of memory system that will be considered for the rest of this section.

4.3.1 Detecting errors due to weak faults in MCA and AD

Consider the presence of defects R_{def1} and R_{def2} (as shown in Fig 4.10) in row decoder of AD and MCA respectively, at the same time. From the previous section we can conclude the following

Defect R_{def1} is present in the memory cell array between the two cross coupled inverters. The value of the defect is lesser than its critical resistance. Hence a transition write operation is not very smooth, resulting in a weak transition fault. In short:

- Caused fault: weak Transition Fault
- Sensitizing condition: Transition write operation

Defect R_{def2} is present along the input address line of the row decoder. The value of the defect is lesser than its critical resistance. The activation of the addressed word line is delayed due to this open defect; this results into an weak Activation Delay (ActD) fault. In short:

- Caused fault: weak Activation Delay Fault
- Sensitizing condition: Address Transitions and Sensitizing Operating Sequence

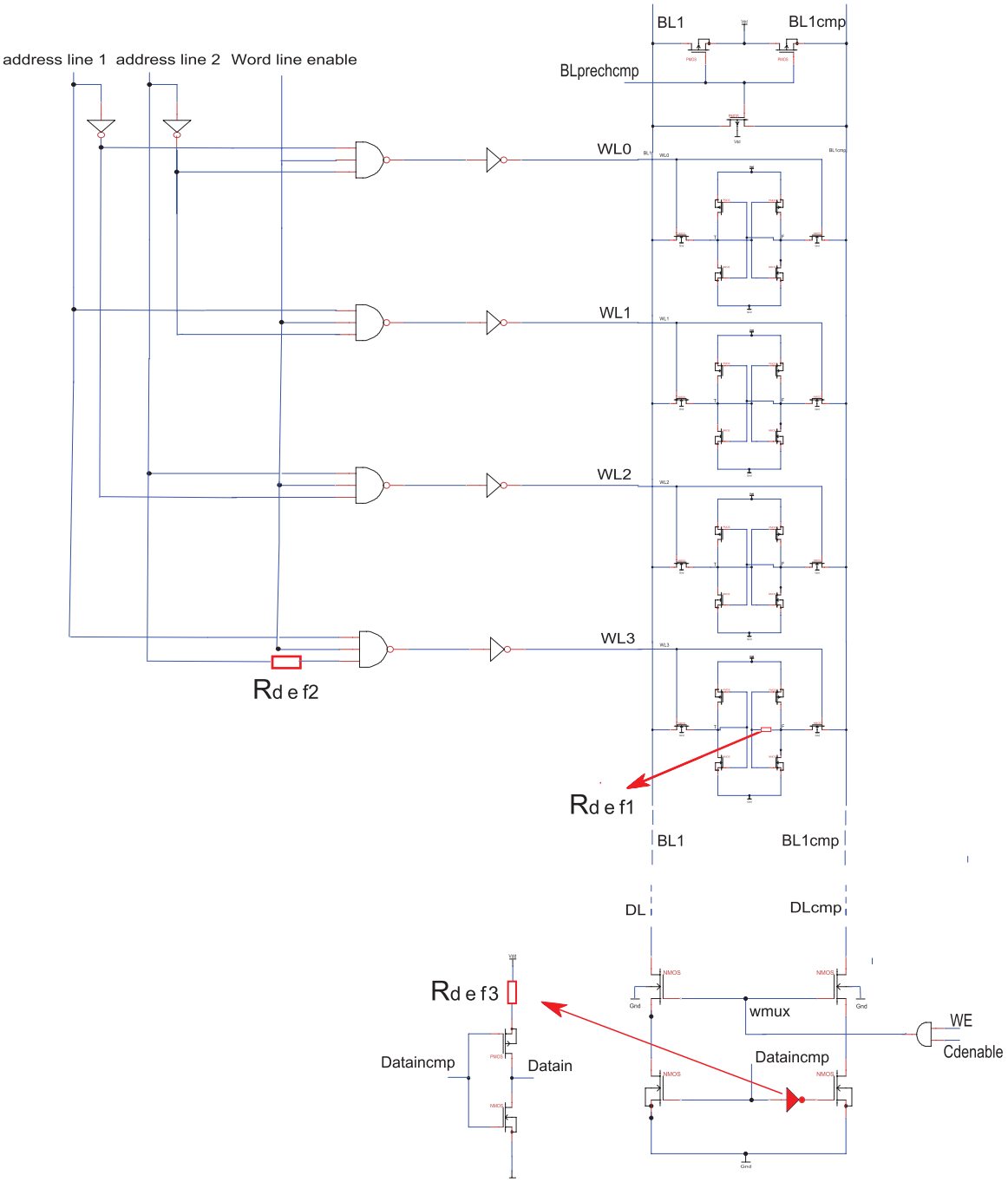


Figure 4.10: Multiple weak defects in memory system

When both the defects R_{def1} and R_{def2} are simultaneously sensitized the weak faults arising due to each of these defects have an additive effect. The defect R_{def2} leads to lesser cell access time. This is because the pass transistors are switched ON for a lesser duration by the word line signal. On the other hand, defect R_{def1} combines with the circuit capacitance (thereby increasing

the RC time constant) to cause a delay during the write operation. These two weak faults have an additive effect which leads to the failure of transition write operation, thereby resulting in an error during the application of the system. In order to detect such errors, it is necessary to sensitize each of these weak faults *simultaneously*. This can be achieved by combining the detection conditions of each of these weak faults. Hence, a transition write operation is combined performed along with the required address transitions and sensitizing operation sequences. The detection condition formed can be compiled into a new test algorithm as shown:

$\{\Updownarrow (wD);_r^{AC} \Uparrow (w\overline{D});_r^{AC} \Uparrow (r\overline{D});_r^{AC} \Uparrow (wD);_r^{AC} \Uparrow (rD);_r^{AC} \Downarrow (w\overline{D});_r^{AC} \Downarrow (r\overline{D});_r^{AC} \Downarrow (wD);_r^{AC} \Downarrow (rD);\}$ where, $D \in \{bDB \; or \; dDB\}$

It is worth noting that in addition to the detecting the errors due to weak faults, the new test algorithm will also be able to detect the errors that may arise due to strong faults that may be caused due to these defects. Hence the missing fault coverage can be improved due to the detection of the errors arising due to weak faults. The defect/fault coverage can be illustrated in a two dimensional graph as shown in Figure 4.11. The hashed areas in the graph represent the coverage that can be achieved based on the traditional approach. Obviously, this coverage will be also realized with the approach based on two defects at a time. Moreover, the new approach is able to improve the defect/fault coverage, which is represented by the region marked with '*' in the graph. Therefore, some weak faults that may escape the traditional test approach will be detected when assuming two defects at a time.

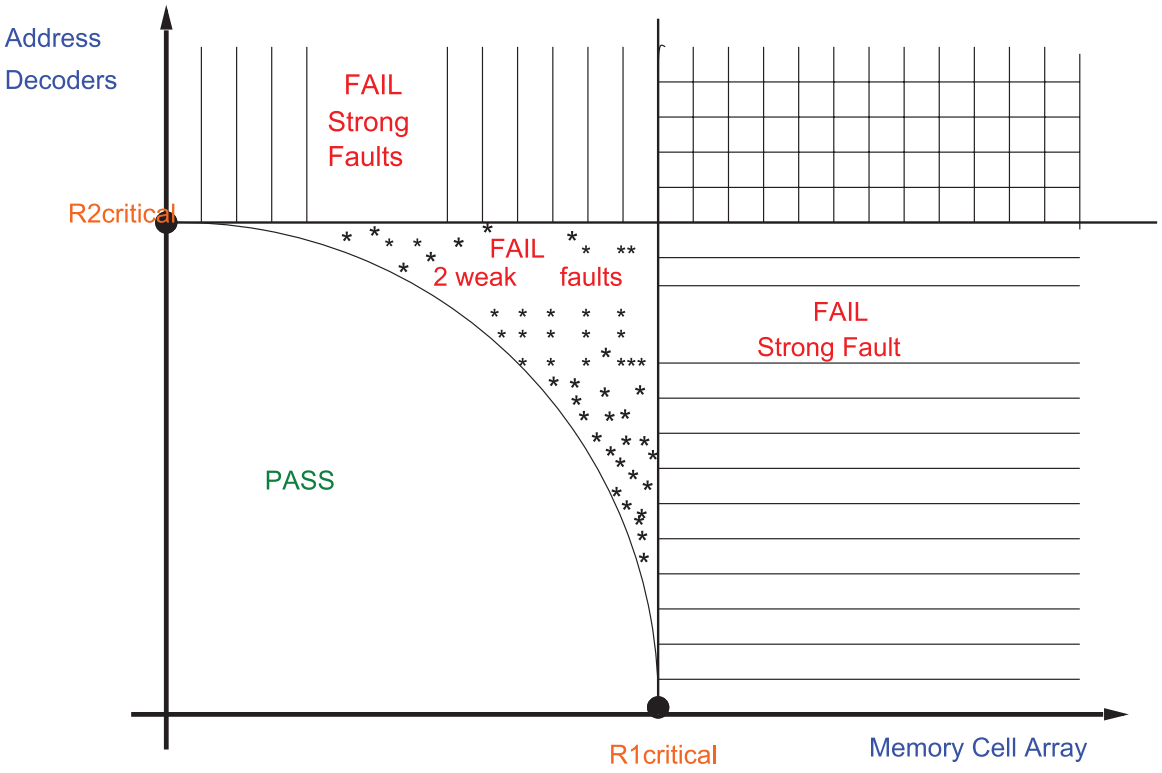


Figure 4.11: Fault coverage with two weak defects at a time

It is worth noting that the effects due to weak faults are not always additive in nature. On the contrary they can also have masking effects with respect to each other. Assume that a defect in address decoder is leading to a weak deactivation fault. This will result in longer access time for the cell. Now consider a defect in the memory cell which leads to a weak transition fault,(obviously due to lack of rise (or fall) time limits to be within the specified cell access time). In the presence of both the faults, the deactivation delay fault provides larger cell access time, which can result in a correct transition operation. Hence the weak transition fault is masked.

4.3.2 Detecting errors due to weak faults in AD and PC

Consider the presence of defects R_{def2} and R_{def3} (as shown in Fig 4.10) in row decoder of AD and PC respectively, at the same time. From the previous section we can conclude the following

Defect R_{def2} is present along the input address line of the row decoder. The value of the defect is lesser than its critical resistance. The activation of the addressed word line is delayed due to this open defect; this results into an weak Activation Delay (ActD) fault. In short:

- Caused fault: weak Activation Delay Fault
- Sensitizing condition: Address Transitions and Sensitizing Operating Sequence

Defect R_{def3} is present in a defective inverter in the write driver. The value of the defect is lesser than its critical resistance. The write driver becomes slow due to the open defect. The result will be that the differential voltage in the bit lines during the write operation is reduced, causing a cell to be weakly written. This fault is referred as a weak Slow Write Driver Fault. In short:

- Caused fault: weak Slow Write Driver Fault
- Sensitizing condition: Back to back complementary write operations in fast row direction i.e., using the same write driver

When both the defects R_{def2} and R_{def3} are simultaneously sensitized the weak faults arising due to each of these defects have an additive effect. The defect R_{def2} leads to lesser cell access time. This is because the pass transistors are switched ON for a lesser duration by the word line signal. On the other hand, defect R_{def3} results in a differential bit line voltage thereby causing a delay during the write operation. These two weak faults have an additive effect which leads to the failure of transition write operation, thereby resulting in an error during the application of the system. In order to detect such errors, it is necessary to sensitize each of these weak faults *simultaneously*. This can be achieved by combining the detection conditions of each of these weak faults. Hence, back to back complementary write operations in fast row direction should be performed using specific address transitions. The detection condition formed can be compiled into a new test algorithm as shown:

$$\{\Updownarrow (w0);_r^{AC} \Updownarrow (w1, r1, w0); \Updownarrow (w1);_r^{AC} \Updownarrow (w0, r0, w1);\}$$

The missing defect/fault coverage can be illustrated in a two dimensional graph in a similar manner as in Fig 4.11. The above test is similar test as that for slow write driver fault which is applied along fast row using the same write driver with proper address transitions rather than linear addressing mode. It is worth noting that the new test developed will hence be able to detect errors due to both the activation delay fault, and slow write driver faults, as the required detection conditions for both the faults are satisfied. Also since both the activation delay faults and slow write driver faults are sensitized at the same time, two weak faults present may results in sensitization of the strong fault,leading to an error which can be detected.

4.3.3 Detecting errors due to weak faults in MCA and PC

Consider the presence of defects R_{def1} and R_{def3} (as shown in Fig 4.10) in row decoder of MCA and PC respectively, at the same time. From the previous section we can conclude the following

Defect R_{def1} is present in the memory cell array between the two cross coupled inverters. The value of the defect is lesser than its critical resistance. Hence a transition write operation is not very smooth, resulting in a weak transition fault. In short:

- Caused fault: weak Transition Fault
- Sensitizing condition: Transition write operation

Defect R_{def3} is present in a defective inverter in the write driver. The value of the defect is lesser than its critical resistance. The write driver becomes slow due to the open defect. The result will be that the differential voltage in the bit lines during the write operation is reduced, causing a cell to be weakly written. This fault is referred as a weak Slow Write Driver Fault. In short:

- Caused fault: weak Slow Write Driver Fault
- Sensitizing condition: Back to back complementary write operations in fast row direction i.e., using the same write driver

When both the defects R_{def1} and R_{def3} are simultaneously sensitized the weak faults arising due to each of these defects have an additive effect. The defect R_{def1} combines with the circuit capacitance (thereby increasing the RC time constant) to cause a delay during the write operation. On the other hand, defect R_{def3} results in a differential bit line voltage thereby causing a delay during the write operation. These two weak faults have an additive effect which leads to the failure of transition write operation, thereby resulting in an error during the application of the system. In order to detect such errors, it is necessary to sensitize each of these weak faults *simultaneously*. This can be achieved by combining the detection conditions of each of these weak faults. Hence, back to back complementary write operations in fast row direction should be performed. The detection condition formed can be compiled into a new test algorithm as shown:

$\{\Updownarrow (wD); \Uparrow_x (w\overline{D}, r\overline{D}, wD); \Updownarrow (w\overline{D}); \Uparrow_x (wD, rD, w\overline{D})$; where, $D \in \{sDB \; or \; cDB\}$

The detection conditions of faults in peripheral circuits can be easily combined with the faults in memory cell array. It is worth noting that the detection condition formed in this case results in the same detection condition as that of the slow write driver fault. This is because of the fact that the detection condition for transition fault is already embedded in the detection condition of the slow write driver fault. However it is to be noted that it is difficult to combine detection conditions for dynamic peripheral circuit faults and dynamic memory cell array faults at the same time. This is because memory cell array requires two successive operations to be sensitized and the dynamic faults in peripheral circuits require only one operation.

4.3.4 Three weak faults based test approach

The missing defect/fault coverage can even be improved if more than two weak faults at a time are considered. Figure 4.12 illustrates the additional coverage that can be realized when assuming three simultaneous weak faults: defect R_{def1} in MCA causing weak fault, defect R_{def2} in the AD causing weak fault and R_{def3} in the write driver of PC causing weak fault. Again each of the defects cause a weak fault with a specific sensitizing conditions:

Defect R_{def1} is present in the memory cell array between the two cross coupled inverters. The value of the defect is lesser than its critical resistance. Hence a transition write operation is not very smooth, resulting in a weak transition fault. In short:

- Caused fault: weak Transition Fault
- Sensitizing condition: Transition write operation

Defect R_{def2} is present along the input address line of the row decoder. The value of the defect is lesser than its critical resistance. The activation of the addressed word line is delayed due to this open defect; this results into an weak Activation Delay (ActD) fault. In short:

- Caused fault: weak Activation Delay Fault
- Sensitizing condition: Address Transitions and Sensitizing Operating Sequence

Defect R_{def3} is present in a defective inverter in the write driver. The value of the defect is lesser than its critical resistance. The write driver becomes slow due to the open defect. The result will be that the differential voltage in the bit lines during the write operation is reduced, causing a cell to be weakly written. This fault is referred as a weak Slow Write Driver Fault. In short:

- Caused fault: weak Slow Write Driver Fault
- Sensitizing condition: Back to back complementary write operations in fast row direction i.e., using the same write driver

As discussed in the previous sections, it is clear that effects due to each of the single weak faults are additive with respect to the other two weak faults which are present in the memory system. Hence, three weak faults also have an additive effect leading to the failure of the write operation. In order to detect the error arising due to all three weak faults, all three weak faults

have to be sensitized simultaneously. Hence,the detection condition formed by combining all three sensitizing conditions results in the following:

$\{\Updownarrow\ (wD); \Uparrow^{AC}\ (w\overline{D}); \Uparrow^{AC}\ (r\overline{D}); \Uparrow^{AC}\ (wD); \Uparrow^{AC}\ (rD); \Downarrow^{AC}\ (w\overline{D}); \Downarrow^{AC}\ (r\overline{D}); \Downarrow^{AC}\ (wD); \Downarrow^{AC}\ (rD);\}$ where, $D \in \{bDB\ or\ dDB\}$

Figure 4.12 shows the missing defect/fault coverage will be further improved as compared with Figure 4.11; this improvement is given by the regions marked with '+' in the figure. Note that the regions marked with '*' indicate the improvement when considering two simultaneous weak faults as compared with the traditional approach. Again, the improvement can be realized only when the detection condition used to develop the test algorithms combines the three sensitization/detection of all the three weak faults.

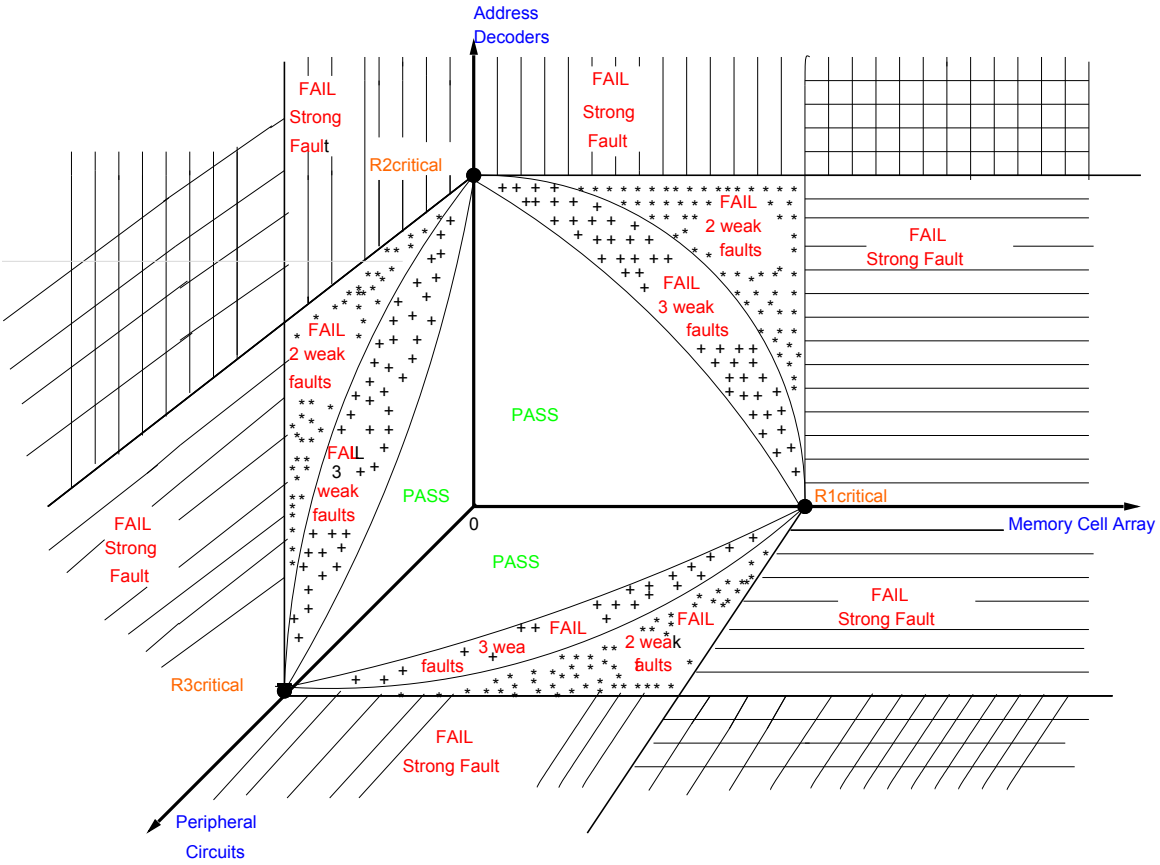


Figure 4.12: Missing defect coverage improvement with three faults based approach

5 Experimental results

The IFA is a commonly used technique to establish fault models and develop test algorithms. The IFA can be experimentally verified through simulations. Electrical models developed, for a memory system can be used for generation of a SPICE based net list. Defects can be inducted into the circuit, at various locations and the fault behavior of the circuit can be simulated. Appropriate test algorithms developed for a given fault model can also be verified. This chapter provides SPICE based, experimental results for the new test paradigm developed for a given case of study. Section 5.1 describes the SRAM simulation model, used for the study. Section 5.2 gives the experimental results of the conventional test approach. Section 5.3 gives the experimental results for new memory test approach in 2D. Section 5.4 gives the experimental results for the 3D approach. Section 5.5 provides the FC analysis.

5.1 SRAM simulation

As stated earlier, an electrical model of the system is the representation of its functional blocks at circuit level. Defects occurring in physical layout level can be mapped on to electrical model in the form of opens, bridges and shorts. The SRAM has three functional blocks in AD, MCA, and PC. The electrical model of these three functional blocks, as shown in Fig 5.1 is considered for simulation. These models can be simulated in a SPICE like tool, by building (or generating) the net list of the circuit under consideration. The net list describes the connectivity of components, at circuit level for any given design. Defect injection, can be done by introducing additional components between any two nodes in the circuit.

Please note that the dotted lines of same colour means that there is a physical connection between the two points. The entire physical connections have been avoided due to lack of space.

5.1.1 Description of SRAM simulation model

For the purpose of simulations, an appropriate memory simulation model has been used; it consists of a 4x4 cell array, address decoder and each column has its own set of peripheral circuits such as sense amplifies, write drivers, pre-charge circuits, etc. Figure 5.2 presents the gate and transistor level description of the memory model used for simulation. It is to be noted that the dotted lines of same color means that there is a physical connection between the two points. The entire physical connections have been avoided due to lack of space.

As mentioned in chapter 2, all the memory components of a memory system can be categorized in three major functional blocks, namely, the memory cell array, the address decoder and, the peripheral circuitry. The three functional blocks are highlighted in the figure and they consist of following components:

- Memory cell array: A 4x4 cell array, with cells numbered from 1 to 16 along vertical direction.
- Address decoder
  - Row decoder
  - Column decoder
- Peripheral circuitry
  - Precharge circuits for bit lines and sense amplifiers.
  - Differential mode voltage sense amplifiers
  - Write drivers

In the figure, only one set of peripheral circuitry has been shown but as it has been mentioned above every column in the memory cell array has its own set of peripheral circuitry. In the labels for peripheral circuitry, x $\in \{1, 2, 3, 4\}$ signify that the peripheral circuitry is connected to different bit-lines. For the ease of presenting the simulation results, depending on which memory cell column is read, the outputs of four sense amplifiers (soutx and soutcmpx) will be multiplexed to give outputs sout and soutcmp; thus in the simulation waveforms the results of

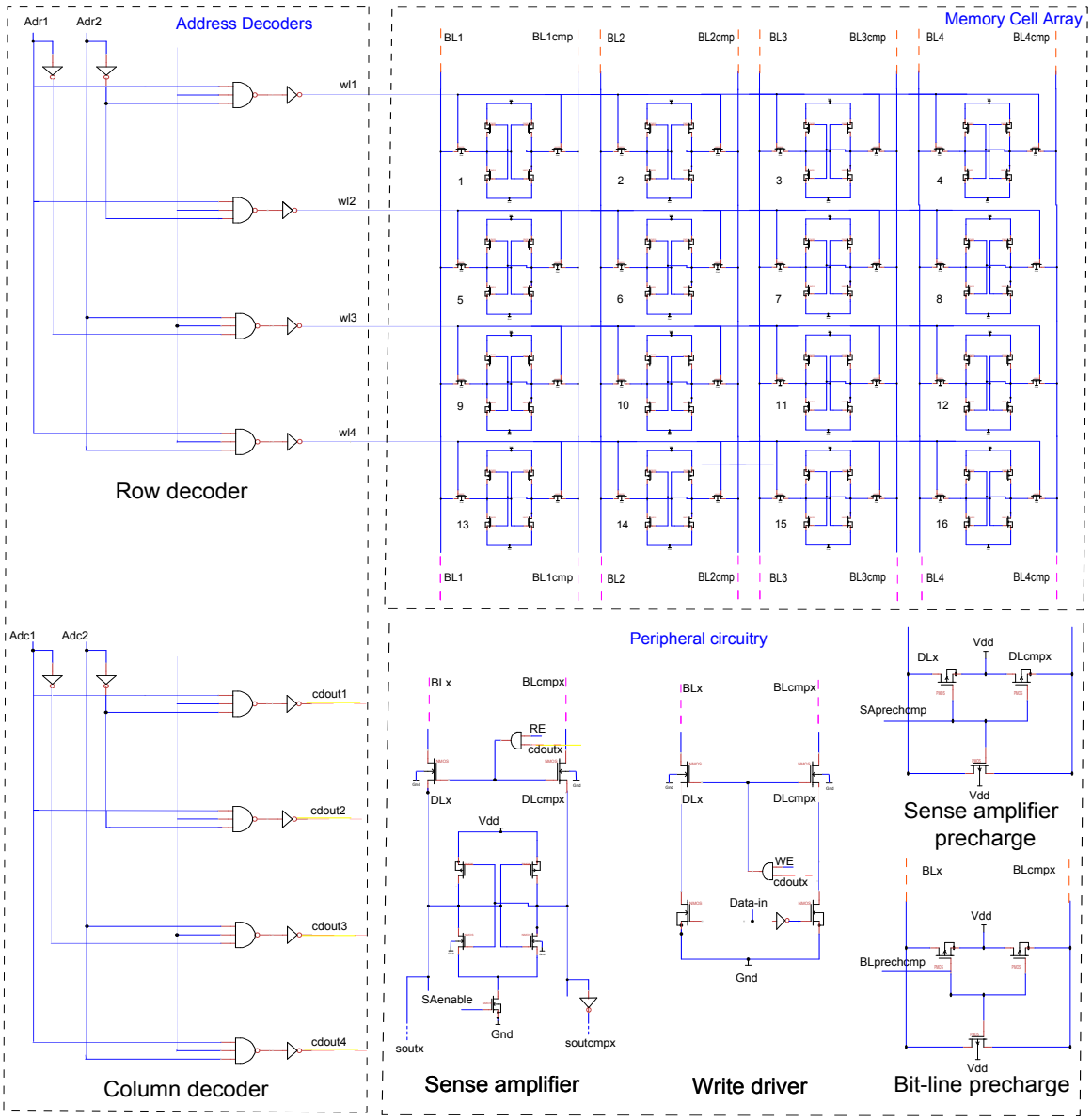


Figure 5.1: Electrical level schematic of SRAM

the read operations performed on memory will be presented using one common signal sout and soutcmp.

The timing generation and control circuitry has not been shown, as for simulations signals were defined manually using Piece Wise Linear (PWL) function in HSPICE. The voltage levels for different signals are defined for different instances of time and the simulations are performed through transient analysis. The inputs to the memory model consists of signals:

- Bit-line precharge enable (BLprechcmp in figure)
- Data input (Data-in)

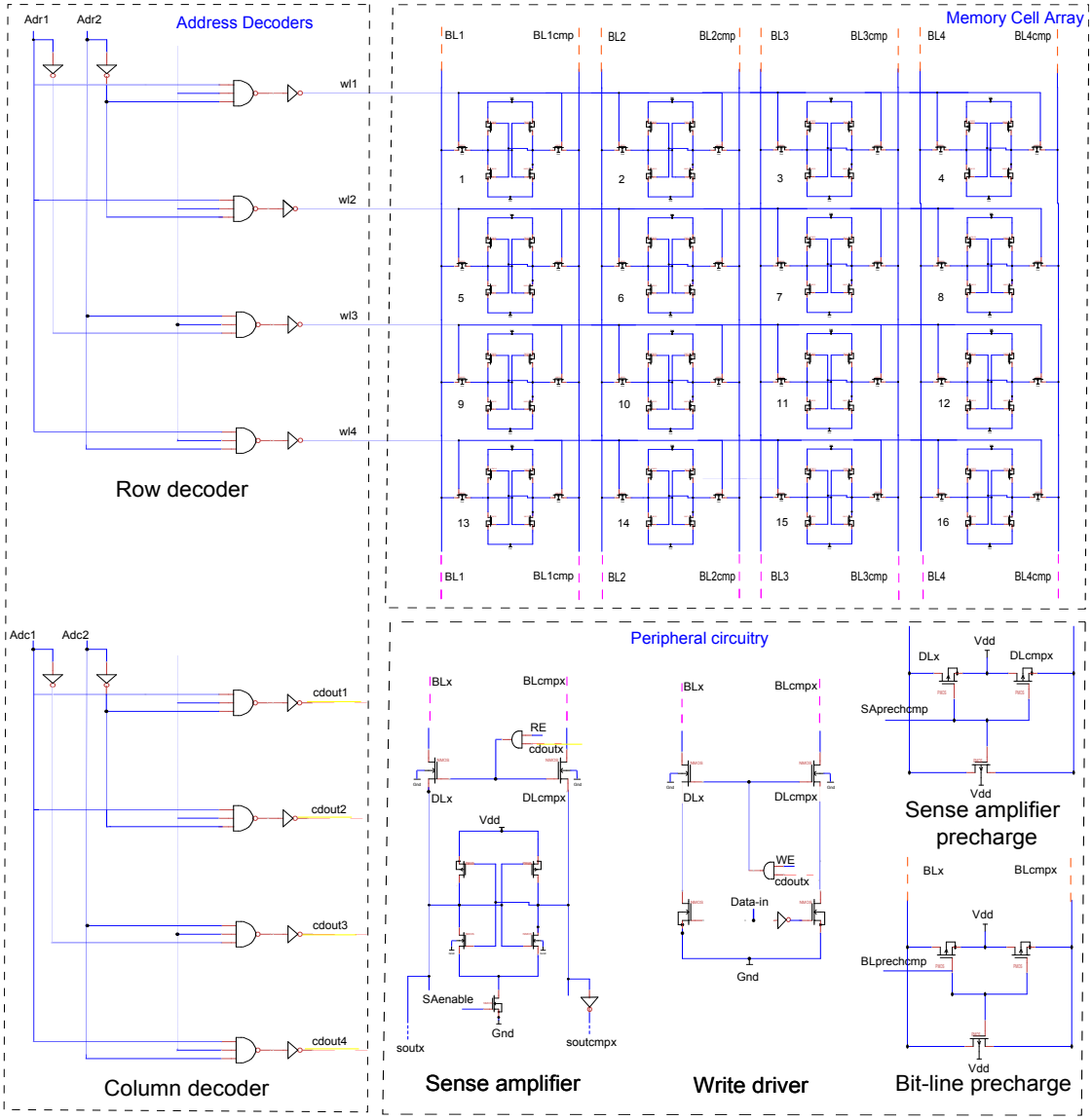


Figure 5.2: Electrical level schematic of SRAM

- Write enable signal (WE)
- Read enable (RE)
- Sense amplifier precharge enable (SAprechcmp)
- Sense amplifier enable (SAenable)
- Row decoder enable (rdenable)
- Column decoder enable (cdenable)
- Row and column address lines (Adr1, Adr2, Adc1 and, Adc2)

The behavior of simple components like resistors, capacitors, etc. can be easily simulated by HSPICE. However, to simulate the behavior of a transistor, a number of parameters must be taken into account. The MOS model cards provides the specification for the parameters, which describe the working of a transistor. The transistor parameters used for design and implementation of the simulation model are as described for 45nm PTM based models. Several other parameters describing the simulation circuit and simulation conditions are listed below:

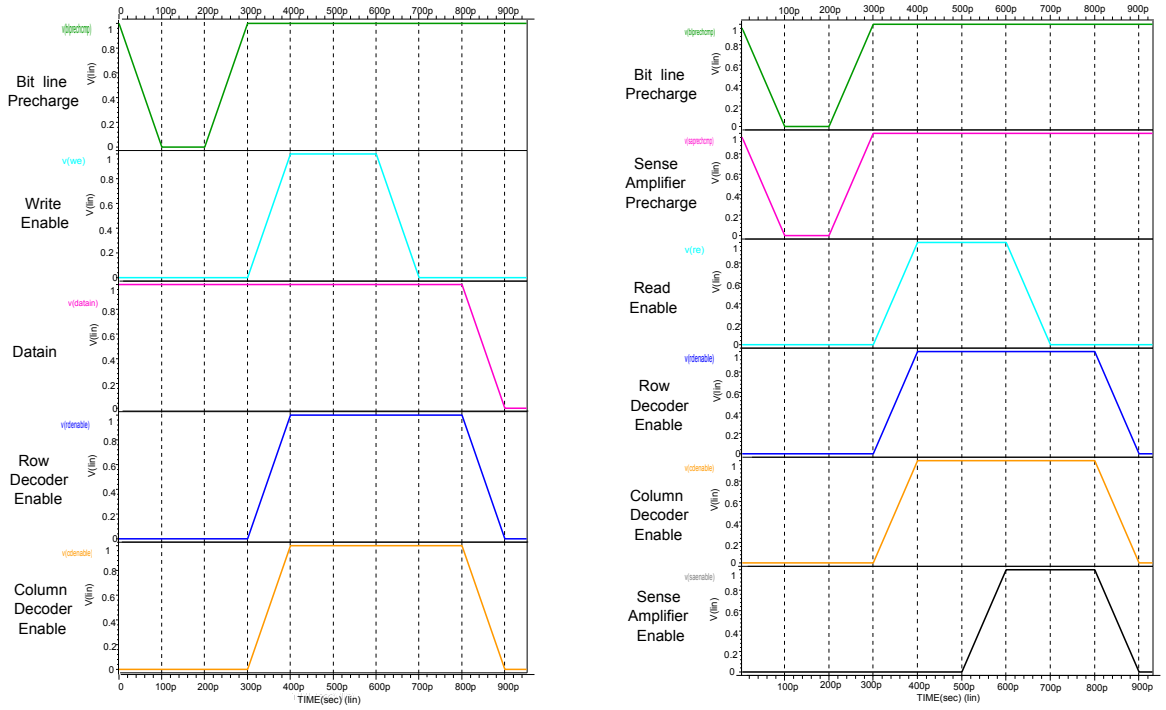
- The supply voltage (vdd) is 1v. Voltage levels between 0v to vdd/2 (0.5v) are refereed as logic 0, while voltage levels between vdd/2 to vdd are refereed as logic 1.
- The bit lines, word lines and data lines have a capacitance to ground of 0.1pF.
- The coupling capacitances between bit lines and word lines are neglected.
- All simulations are performed at 300K temperature conditions.

5.1.2 Timing of input signals

To analyze the behavior of SRAM, we have to perform write and read operations. Both the write and read operations should be completed within one clock cycle. The clock frequency determines the speed of the SRAM, which is again dependent on the technology node used. The cycle time chosen for the simulation model used in this thesis is about 1ns. Some of the signals used for the implementation of the model are row decoder enable (rdenable), column decoder enable (cdenable), word line (wlx), column line (cdoutcmpx), sense amplifier enable (SAenable),bit line pre-charge (Blprechmp), write enable (WE), read enable (RE),sense amplifier pre-charge (SAprecharge), Datain etc. The timing specifications of some of these signals, during write and read operations are shown in Fig 5.3(a) and 5.3(b) respectively. The write and read operations, consists of two main phases i.e., the pre-charge phase and the execution phase. The pre-charging phase for both write and read operations are completed within 0.3ns. Similarly before the execution phase of every read operation, the sense amplifier is pre-charged within a period of 0.3ns. The execution phase for both write and read operations are 0.7ns.

5.1.3 Simulation of defect free SRAM circuit

An HSPICE based simulation of the defect free memory model is done to check the behavior and functionality of the memory under defect free conditions. The two basic operations in memory are write and read operations. The write and read operations are performed over a given cell for both logic states i.e., logic ‘0’ and logic ‘1’. Fig 5.4 shows the simulation results for w1,r1,w0,r0 operations in a defect free memory cell. The cell is initially at logic ‘0’. It can be observed that the true node voltage undergoes transitions from logic ‘0’ to logic ‘1’ during ‘w1’ operation. The bit line voltage, ‘Bl1cmp’ is pulled down during write operation and is pre-charged only before the next operation begins. During ‘r1’ operation the sense amplifier reads the value of true node voltage of cell as logic ‘1’ as expected. Similar effects are observed during the ‘w0’ and ‘r0’ operations.



(a) Timing diagram for write operation

(b) Timing diagram for read operation

Figure 5.3: Timing diagrams for write and read operations in SRAM

5.2 Experimental results for conventional testing approach

The conventional testing approach, assumes the presence of single (strong) defect at a given time in the whole memory system. As a case of study, the following three defects were considered (see Figure 4.10):

- Defect R_{def1}: The connection between the cross-coupled inverters of the memory cell array suffer from a resistive open defect.
- Defect R_{def2}: A resistive open in one of the address line inputs of an NAND gate of the row decoder.
- Defect R_{def3}: A resistive open in the pull-up transistor of the inverter in the write driver circuit.

Each of the following defects were analyzed, for wide range of values and the corresponding fault models were identified in case a faulty behavior is observed. The tests developed for these functional fault models were simulated, to verify and check their detection capabilities. This section presents the simulation results for the case of study, by assuming single defect at a time in each of the memory functional blocks.

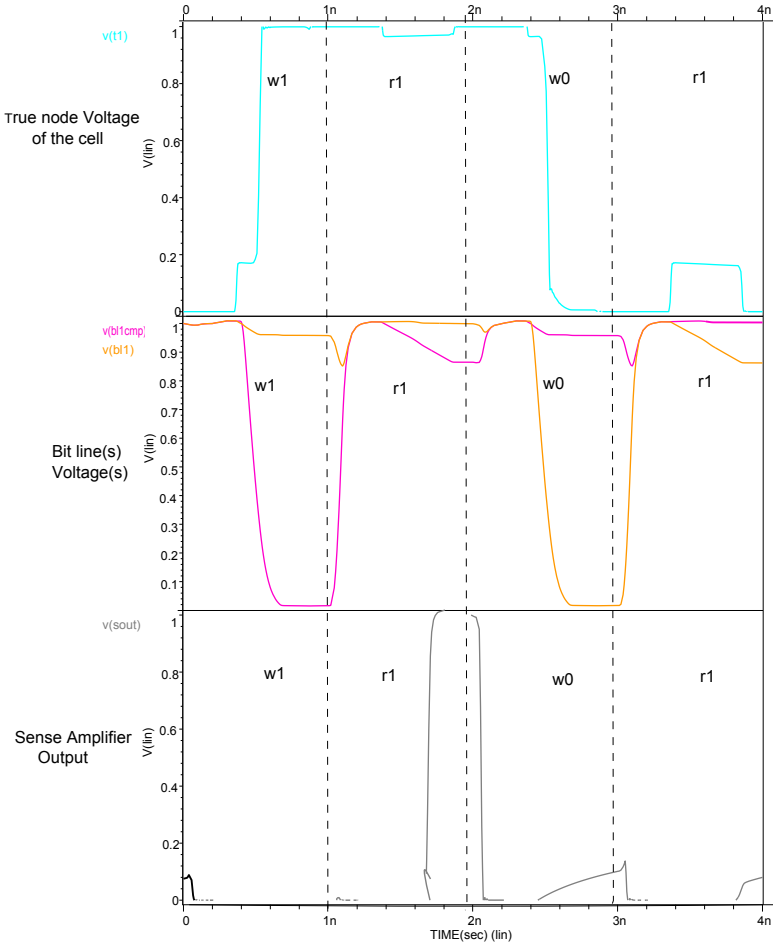


Figure 5.4: Simulation results for a '0w1,r1,1w0,r0' operations in a defect free SRAM cell

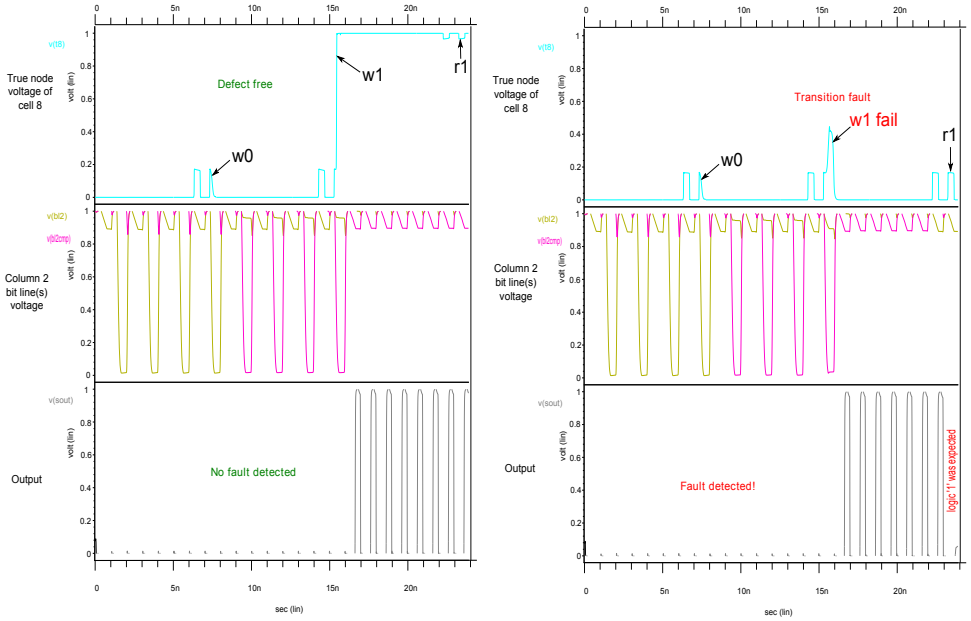
5.2.1 Single defect in Memory Cell Array

Consider the presence of only defect, R_{def2} in a particular memory cell in the cell array (see Fig 4.10). As discussed in chapter 4, depending on the value of defect, different types of faults can be modeled. Simulation results have shown that the transition write from logic '0' to logic '1' fails beyond a certain value of resistance. The value of resistance is found to be $R_{def2} = 488K\Omega$.This value of the resistance is called the critical resistance. This type of faulty behavior, can be detected by using the test formed by detection condition for a transition fault as shown.

Test TF: $\{\Uparrow_{fy} (w0); \Uparrow_{fy} (w1); \Uparrow_{fy} (r1); \}$

The test simulation is performed for all the cells, present in the first two columns in the memory cell array. Cell number 8, present in the last row of the second column, suffers from defect R_{def2}. Fig 5.5(a) shows the simulation result for the above detection condition, under defect free conditions. Fig 5.5(b) shows the simulation results of the memory suffering from

defect $R_{def2} = 489K\Omega$. It can be observed that the true node voltage of cell 8, fails to undergo transition from logic '0' to logic '1' in the presence of defect. The output response in Fig 5.5(b), clearly is not same as that of the expected response, which is the response obtained under defect free conditions in Fig 5.5(a). Hence the test TF is able to detect transition faults for defect values greater than its critical resistance.



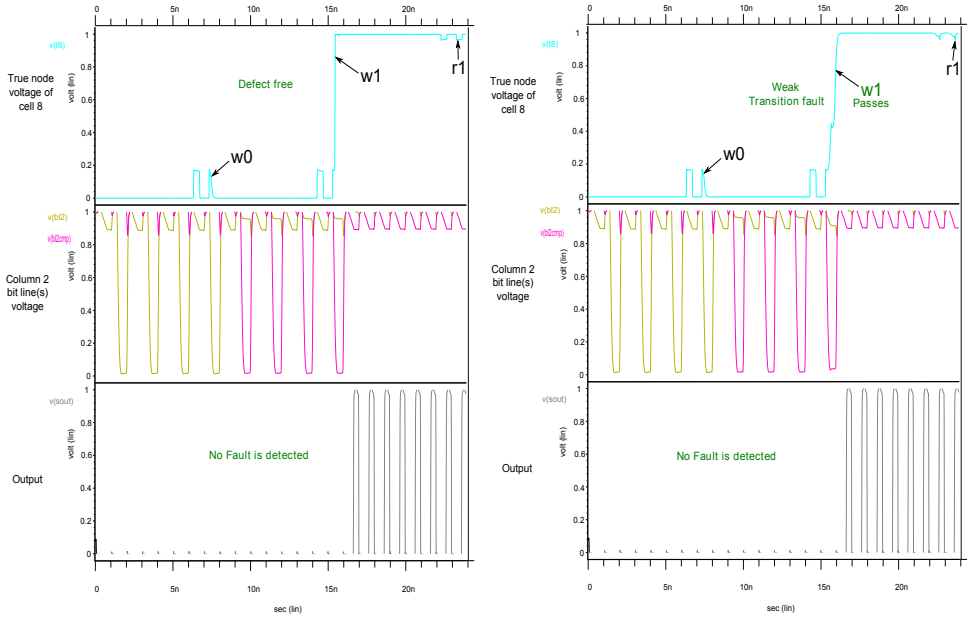
(a) Defect free memory cell (R_{def2}= 0) (b) Defective memory cell (R_{def2}= 489kΩ); R_{def2}>R_c

Figure 5.5: Simulation of transition fault using Test TF

The simulation results for, defect values lesser than the critical resistance i.e., say $R_{def2} = 487K\Omega$ (see Fig 5.5(b)) shows that the transition write operation passes successfully. The output response of Test TF under both defect free (see Fig 5.5(a)) and defective conditions, (with defect lesser than its critical resistance) is also same. Hence no fault is detected. However, careful inspection of the true node voltage of cell 8 reveals, that there is a delay in the transition from logic '0' to logic '1' when compared to the transition time in the defect free case. This type of fault alone, may not affect the functionality of the system, because it is not fully sensitized by this test algorithm. Hence the region below the value of critical resistance becomes the region for weak faults (see Fig 4.3).

5.2.2 Single defect in Address decoders

Consider the presence of only defect, R_{def1} in the row decoder(see Fig 4.10). The behavior of the memory, suffering from this defect can be studied under three different regions i.e., static, dynamic and the weak fault regions. When the value of defect is very high, it leads to static address decoders faults. Simulation results have shown that for values of $R_{def1} \geq 190k\Omega$ static address decoder fault of 'AFoc' is observed. This type of functional faults can be detected by test:



(a) Defect free memory cell (R_{def2}= 0)

(b) Defective memory cell (R_{def2}= 488KΩ); R_{def2}< R_c

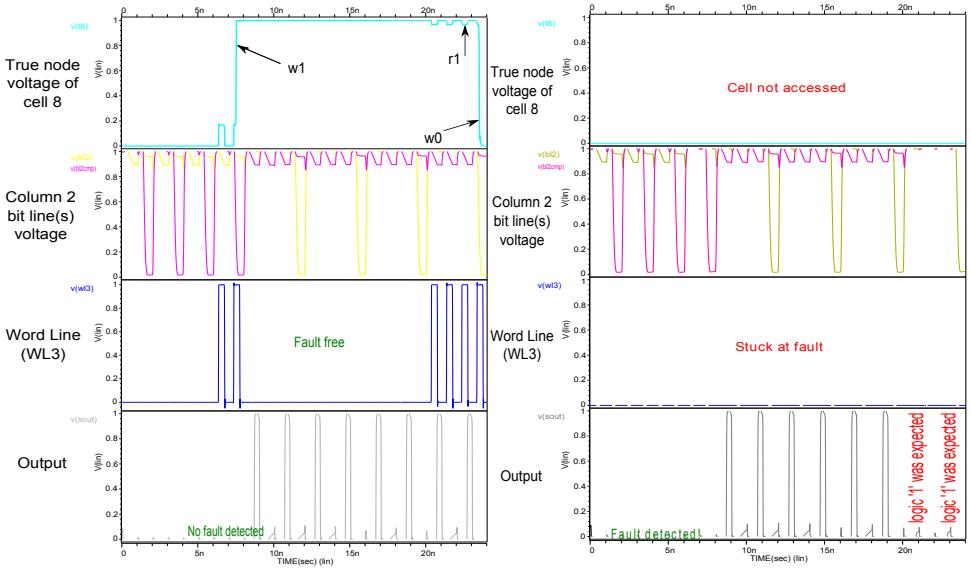
Figure 5.6: Simulation of weak transition fault using Test TF

Test sADF: $\{\Updownarrow (w1); \Uparrow (r1, w0); \}$

Fig 5.7(a) provides simulation results for defect free case. It can be observed that all operations in test sADF, pass correctly. In case of presence of high value of defects, say $R_{def1} = 2M\Omega$ the word line (wl3) as shown in Fig 5.7(a) is entirely cut off. Thus the cells present along row 4, (enabled by word line (wl3)), are completely not accessible. The inspection of true node voltage of cell '8' which is present in the affected row, also reveals the failure of all operations in test sADF. When the output of this test is compared with that of the expected response, we see that the read operations fails for all the cells present in the corresponding affected row (i.e., cell number 4 and cell number 8). This type of fault is observed for all values of $R_{def1} \geq 190k\Omega$. This region is referred as static fault region.

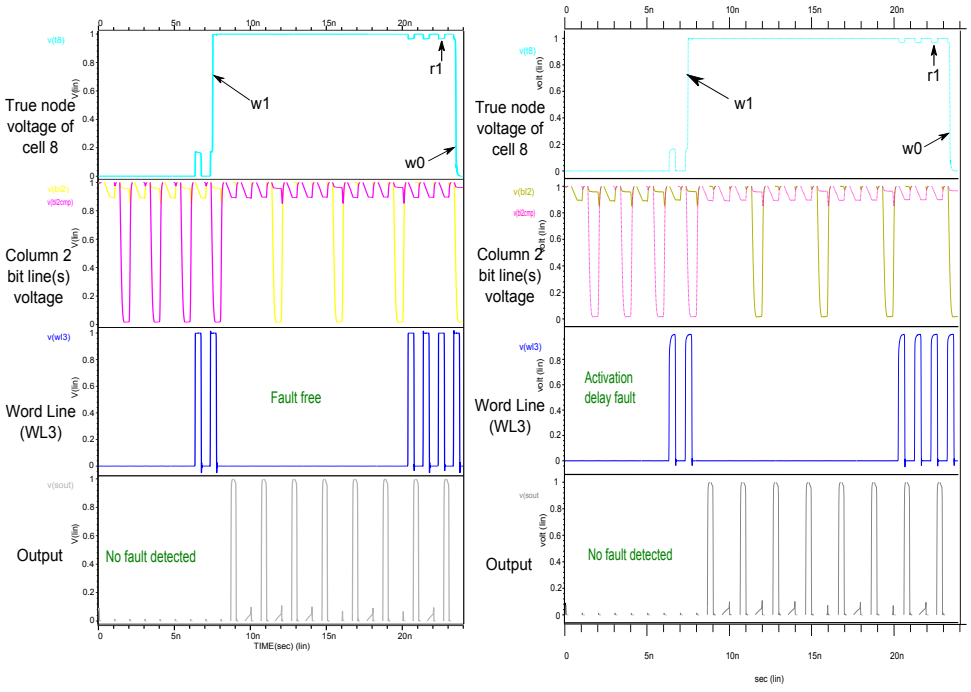
For values of $R_{def1} \leq 190k\Omega$ the test sADF passes and no there is no deviation from the expected output. This can be illustrated in Fig 5.8(a) and Fig 5.8(b). However, it can be observed that there is a slight difference in the activation time of the word line signal (wl3). Such delay related faults are called dynamic faults. But the effect is not significant enough to cause a fault. This is because the test sADF uses a linear addressing mode, which does not fully sensitize the fault. This fault can be fully sensitized only when proper address transitions and sensitizing operating sequences are used.

As discussed in chapter 4, in order to detect the delay faults, the following detection condition compiled as test ActD can be used:



(a) Defect free address decoder (R_{def1}= 0) (b) Defective address decoder (R_{def1}= 2MΩ) $R_{def1} > R_c$

Figure 5.7: Simulation of stuck at fault using Test sADF



(a) Defect free (linear addressing) (R_{def1}= 0)

(b) Dynamic activation delay fault (linear addressing) (R_{def1}= 70KΩ)

Figure 5.8: Simulation of activation delay fault using Test sADF

Test ActD: $\{\Updownarrow (w0); \Uparrow^{AC} (r0, w1, r1); \}$

This detection condition has special address transitions and operating sequence. Hence the activation delay faults which occur below the range of $R_{def1} \leq 190k\Omega$ are detected using this test. Fig 5.8(a) and Fig 5.8(b) shows the required simulation results, for defect free and strong activation delay fault conditions. There is a deviation between expected response and the actual response. It can be seen that both cell number 4 and cell number 8 present in the same row are affected. the true node voltage of cell 8 also indicated the failure of a write one operation. Hence simulation results have shown that the ActD test, is able to detect activation delay faults in region of $55k\Omega \leq R_{def1} \leq 190k\Omega$. This region is referred as dynamic fault region.

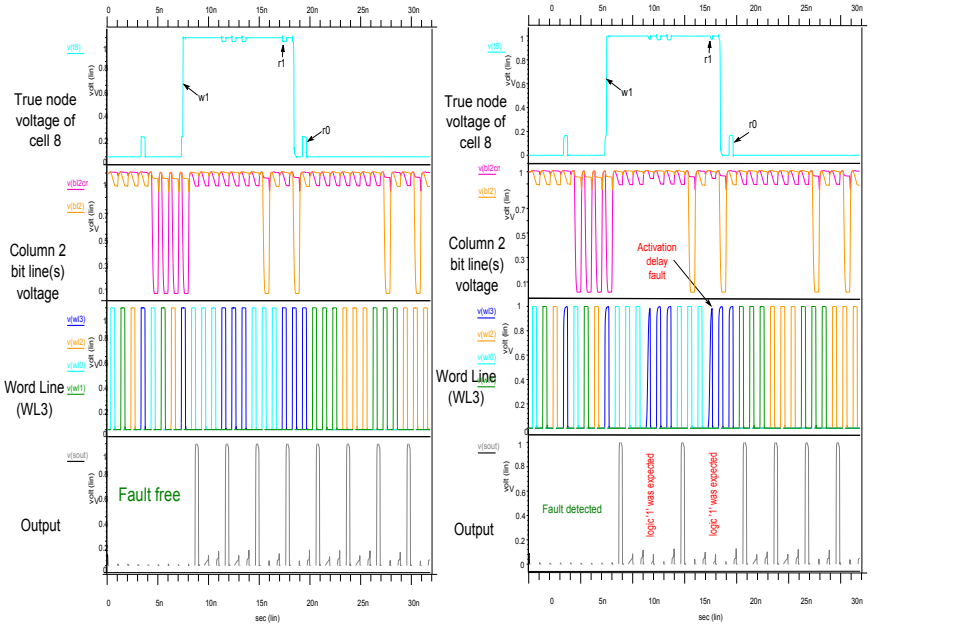
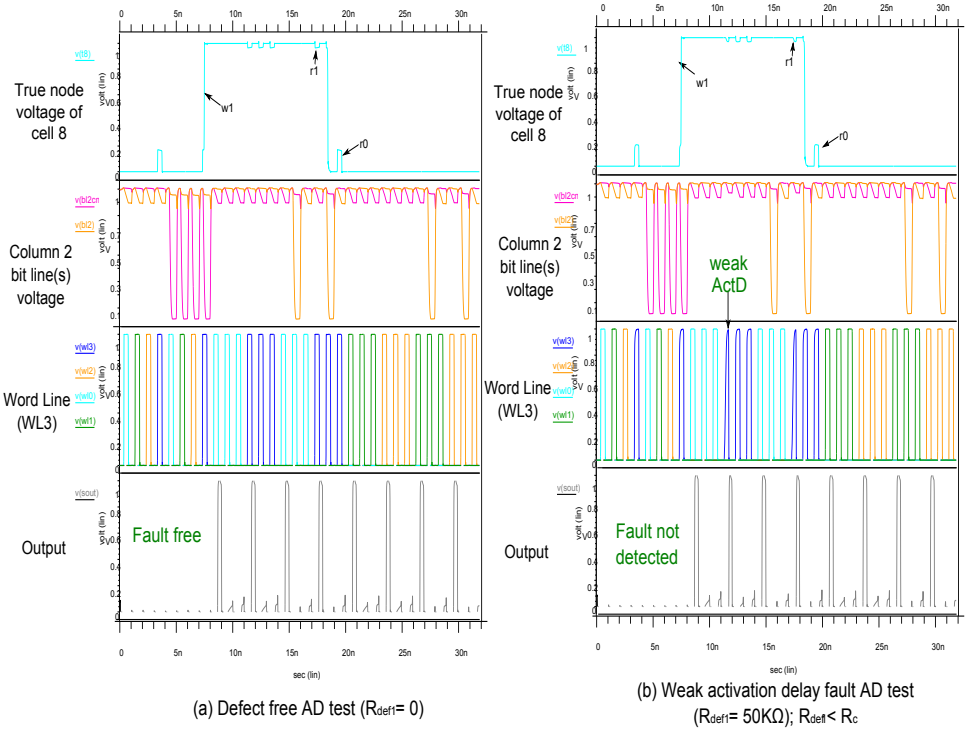


Figure 5.9: Simulation of activation delay fault using Test ActD

Below the critical resistance, of $R_{def1} \leq 55k\Omega$. the ActD test is not able to detect the delay faults. However, it can be observed that the presence of the defect, still induces a delay in activation of the word line signal, but the effect is not significant enough to generate a fault. This region is the weak fault region. The simulation results for the defect free and weak fault regions are provided in Fig 5.10(a) and Fig 5.10(b).

5.2.3 Single defect in Peripheral Circuit

Consider the presence of only defect, R_{def3} in the pull up path of inverter in write driver (see Fig 4.10). The behavior of the memory, suffering from this defect can be studied under three different regions i.e., static, dynamic and the weak fault regions. When the value of defect is very high, it leads to static stuck at faults. his type of faulty behavior can be detected by the following test:



(a) Defect free AD test (R_{def1}= 0)

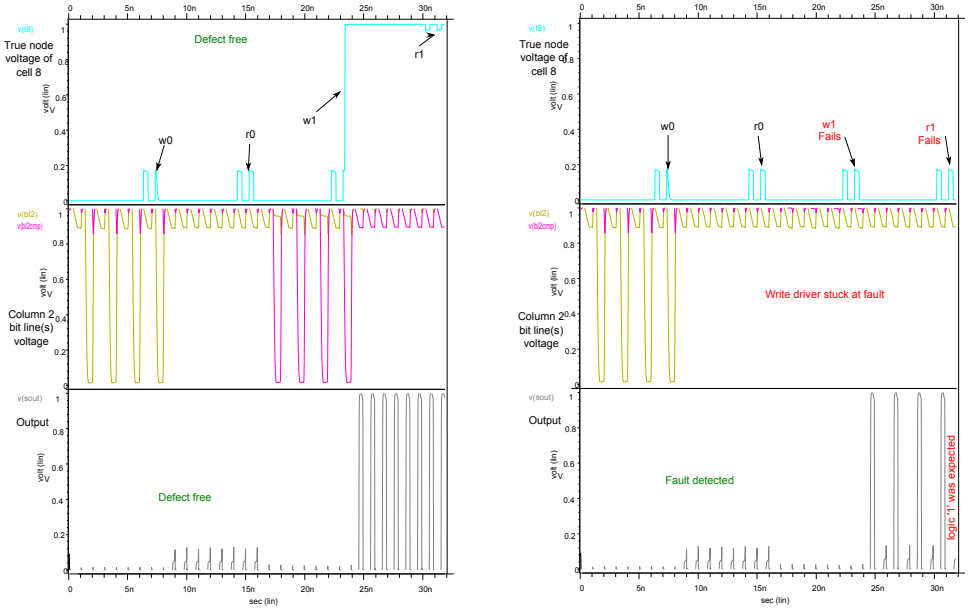
(b) Weak activation delay fault AD test (R_{def1}= 50KΩ); R_{def}< R_c

Figure 5.10: Simulation of weak activation delay fault using Test ActD

Test stuckat: $\{\Uparrow_{fy}(w0); \Uparrow_{fy}(w1); \Uparrow_{fy}(r1);\}$

Fig 5.11(a) and 5.11(b) shows the simulation results of test stuckat for defect free write driver and defective write driver, with very high value of R_{def3}. A write '1' operation for all the cells in the column sharing the same write driver will fail. Inspection of the true node voltage of cell 8 also reveals that the write one operation of the cell fails in case of defective write driver. The bit line voltage indicates that the 'Bl2cmp' voltage signal is never pulled down to zero. The output hence, differs from that of the expected response, with the failure of read one operation. This type of fault behavior is observed in the region $R_{def2} \geq 140k\Omega$.This region in defect space is known as the static fault region.

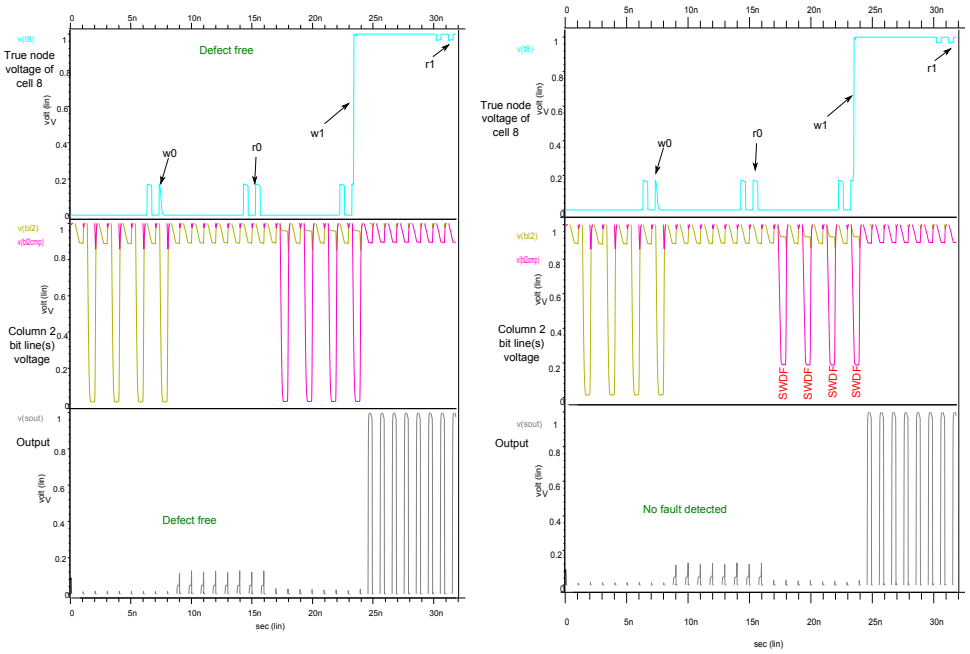
For values of $R_{def2} \leq 190k\Omega$ the test stuckat passes and no there is no deviation from the expected output. The simulation results for test stuckat in this region with defect free write driver and defective write driver are as shown in Fig 5.12(a) and 5.12(b) respectively. It can be seen that the bit line voltages of the write driver in the affected column is not fully pulled down, due to the presence of the defect. This may cause the write operation to fail. But test stuckat passes, because the bit lines are not brought into worst case scenario, for the fault to be sensitized and detected.



(a) Defect free write driver (Rdef=0)

(b) Defective write driver (Rdef2= 2MΩ);

Figure 5.11: Simulation of stuck at fault in write driver using Test stuckat



(a) Defect free write driver (Rdef=0)

(b) Defective write driver (Rdef3=140kΩ); $R_{def3}>R_c$

Figure 5.12: Simulation of Slow Write Driver Fault using Test stuckat

As discussed in chapter 4, in order to detect the slow write driver faults, the following detection condition compiled as test SWDF can be used:

Test SWDF: $\{\Updownarrow (w0); \Uparrow_{fx} (w1, r1, w0); \}$

This detection condition, involving back to back complementary write operations has to be performed along fast x direction , i.e., using same write driver. Hence the slow write driver fault which occur below the range of $R_{def3} \leq 140k\Omega$ are detected using this test. Fig 5.13(a) and Fig 5.13(b) shows the required simulation results, for defect free and strong slow write driver fault conditions. It an be observed that the operation in cell number 5 (1st cell in column 2) passes, as it is not strongly sensitized i.e., no back to back complementary write operation. However the write '1' operations in cell number 6,7 and 8 fail (see. true node voltage of cell 8 in 5.13(b)). This is because a write '1' is performed in all these cells after a write '0' is performed in the preceding cell leading to strong sensitization of the slow write driver fault. Hence read '1' operations for cell 6,7 and 8 fails, while that of cell 5 passes. Simulation results have hence, shown that the test SWDF, is able to detect slow write driver faults in region of $91k\Omega \leq R_{def3} \leq 140k\Omega$. This region is referred as dynamic fault region.

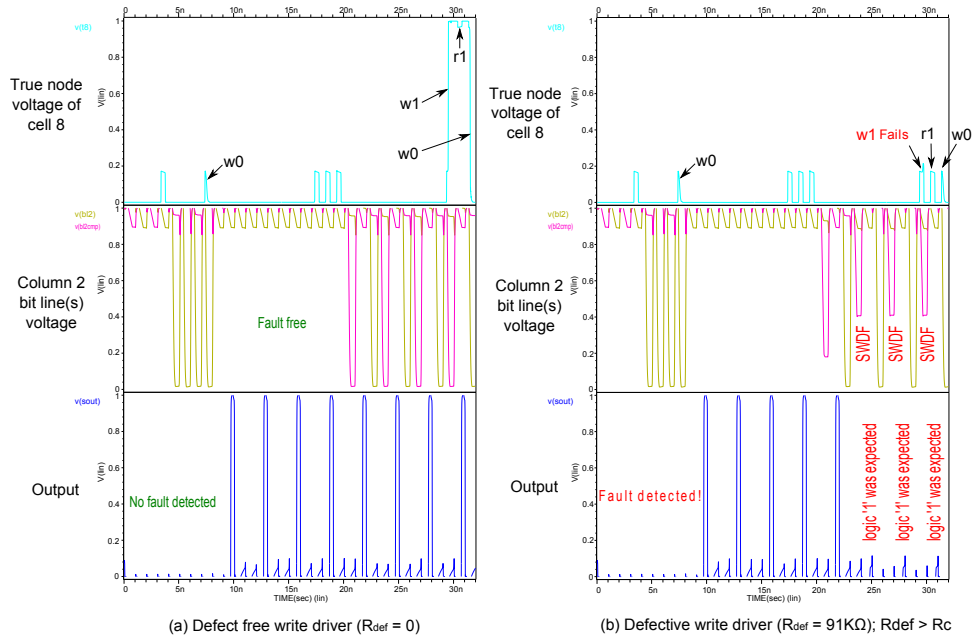


Figure 5.13: Simulation of Slow Write Driver Fault using Test SWDF

Below the critical resistance, of $R_{def3} \leq 91k\Omega$. the test SWDF, is not able to detect the slow write driver faults. However, it can be observed that the presence of the defect, still induces an improper discharge of the bit lines, but the effect is not significant enough to generate a fault. This region is the weak fault region. The simulation results for the defect free and weak fault regions are provided in Fig 5.14(a) and Fig 5.14(b).

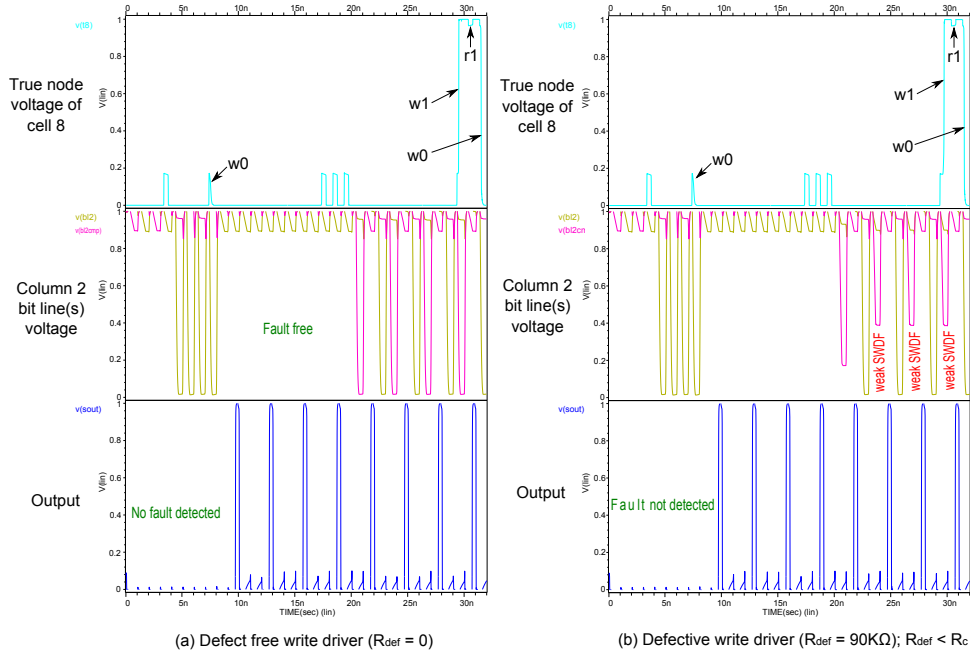


Figure 5.14: Simulation of weak Slow Write Driver Fault using Test SWDF

5.3 Experimental results for new memory test approach

The conventional testing approach considers only single (strong) defect at a time. As discussed in chapter 4, combination of weak faults from different parts of the memory can have additive effects, leading to the faulty behavior of the system. The new test paradigm developed is able to detect such faulty behavior. This section presents the simulation results for the new memory test developed by considering multiple (weak) defects at a time in different parts of the memory system.

5.3.1 Multiple defects approach in Memory Cell Array and the Address Decoders

Consider the scenario where both weak defects $R_{def1} < R_{c1}$,$R_{def2} < R_{c2}$ present in address decoders and memory cell respectively at the same time (see Fig 4.10). Each of the defects R_{def1} and R_{def2} alone causes weak faults. However when both are present at the same time, then it leads to a strong fault, as their effects are additive in nature. Simulations are now performed for three different tests: 1)Test ActD 2)Test TF AND 3) Test TF & ActD

For simulation, the values of defects were chosen as $R_{def1} = 50k\Omega$,$R_{def2} = 480k\Omega$. The simulation results for test ActD are shown in Fig 5.15(a) and 5.15(b) for defect free and multiple weak defect conditions respectively. The output response for the simulated weak defects case is identical with that of the defect free case. This is because the activation delay fault and the transition fault are not sensitized simultaneously at the same time by this test. When there is a weak transition operation, the weak activation fault is not strongly sensitized at the same time, because of no proper address transitions. Similarly when the activation fault is strongly

sensitized there is no transition write operation taking place. Hence the fault goes undetected due to lack of strong sensitization of both weak faults at the same time.

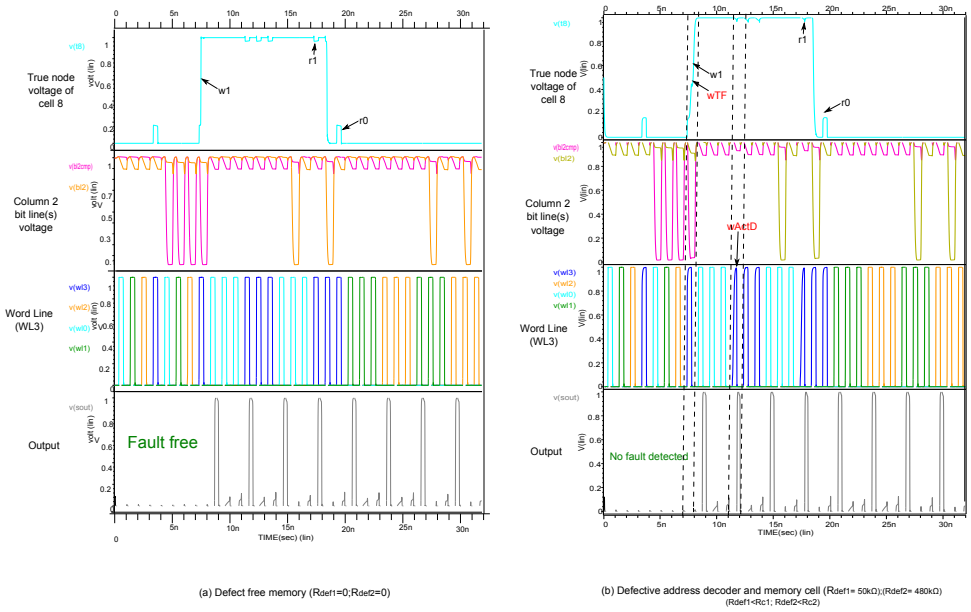


Figure 5.15: Simulation of weak transition fault and weak activation delay fault using test ActD

Now consider the simulation results for test TF as shown in Fig 5.16(a) and 5.16(b) for defect free and multiple weak fault conditions respectively. The output response of the test under defective conditions matches with the output response of the defect free conditions. This indicates that the test TF is not sufficient to strongly sensitize both the weak faults at the same time to sensitize a strong fault. Careful inspection of the word line signal, shows that the weak activation delay fault is almost absent. This is because the test TF is performed in fast y direction. Hence there are no proper transitions produced for the defect in row decoder to be sensitized at all. Hence test TF also fails to detect the weak faults.

From the simulation results presented for tests ActD and test TF alone, it is clear that they are not sufficient to detect the strong faults, that may arise due to weak faults. Hence a new test 'TF & Actd' is developed to sensitize both the weak faults at the same time and detect the faulty behavior due to weak faults. As discussed in chapter 4, the new test 'TF & Actd' is formed by combining the two detection conditions of TF and ActD.

test TF& ActD: $\{\Uparrow_{fx}\ (wD); \Uparrow_{fx}^{AC}\ (w\overline{D}); \Uparrow_{fx}^{AC}\ (r\overline{D}); \Uparrow_{fx}^{AC}\ (wD); \Uparrow_{fx}^{AC}\ (rD)\}$ where $D \in \{bDB, cDB\}$

Fig 5.17(a) and 5.17(b) shows the simulation results for test TF & ActD under defect free and presence of multiple weak defects conditions respectively. The response obtained when multiple weak defects are present is different from the expected response. This is because both the weak activation delay fault and weak transition fault are detected at the same time, leading to a strong

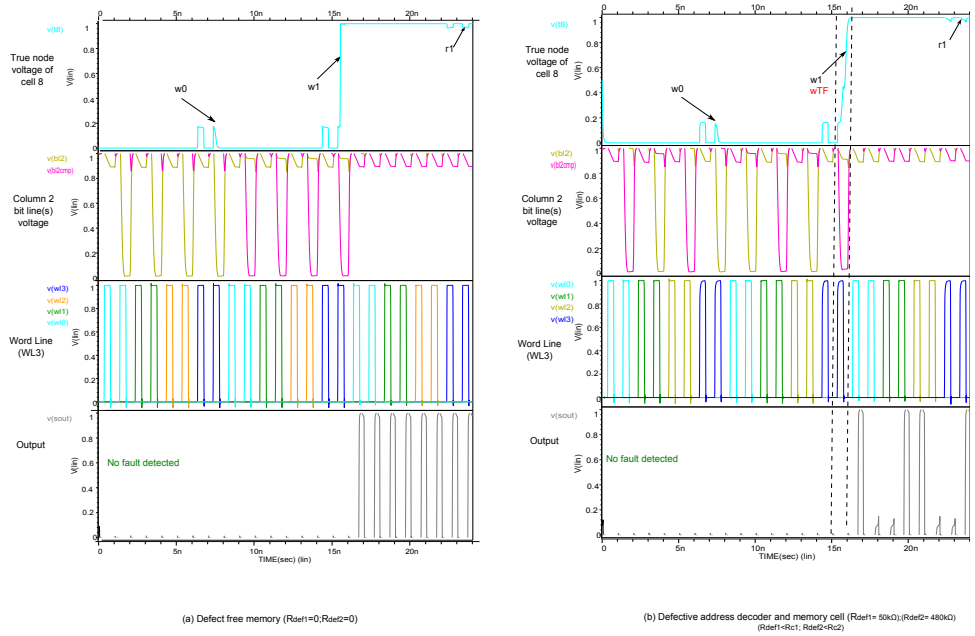


Figure 5.16: Simulation of weak transition fault and weak activation delay fault using test TF

sensitization of both the weak faults, thereby resulting in sensitizing a strong fault. The write one operation fails, in cell number 8, as it is affected by both weak activation delay fault and weak transition fault. However the result of read operation from cell number 4, present in same row affected by weak activation delay fault is correct. This is because it is purely suffering from weak activation delay faults, whereas cell number 8, suffers from both weak transition fault and weak activation delay fault, being sensitized at the same time. As explained in chapter 4, this test can also be used to detect, (strong) transition faults and (strong) address decoder faults.

5.3.2 Multiple defects approach in Address Decoders and the Peripheral Circuits

Consider the scenario where both weak defects $R_{def1} < R_{c1}$,$R_{def3} < R_{c3}$ present in address decoders and the write driver respectively at the same time (see Fig 4.10). Each of the defects R_{def1} and R_{def3} alone causes weak faults. However when both are weak faults are sensitized at the same time, then it leads to a strong fault, as their effects are additive in nature. Simulations are now performed for three different tests: 1)Test ActD 2)Test SWDF AND 3) Test ActD & SWDF

For simulation, the values of defects were chosen as $R_{def1} = 50k\Omega$,$R_{def3} = 88k\Omega$. The simulation results for test ActD are shown in Fig 5.18(a) and 5.18(b) for defect free and multiple weak defect conditions respectively. The output response for the case with defect is identical to that of the simulated response, for a defect free case. This is because the activation delay fault and the slow write driver fault are not sensitized simultaneously at the same time by this test. When the sensitizing write operation (the write after complementary write using the same write driver)takes place activation fault is not strongly sensitized at the same time, because of no

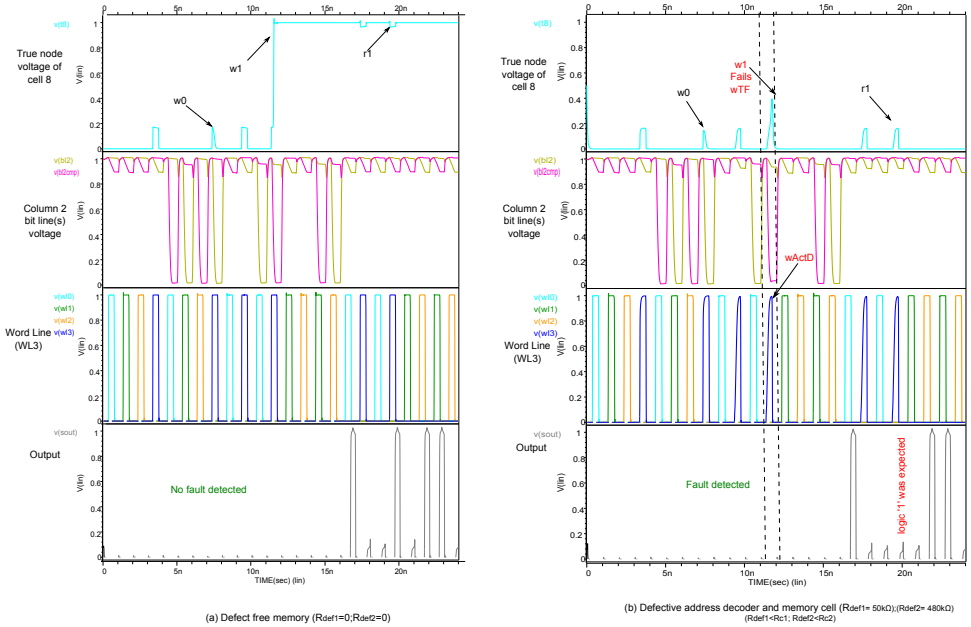


Figure 5.17: Simulation of weak transition fault and weak activation delay fault using Test ActD & Test TF

proper address transition. Similarly when the activation fault is strongly sensitized there is no sensitizing write operation is taking place. Hence the fault goes undetected due to lack of strong sensitization of both weak faults at the same time.

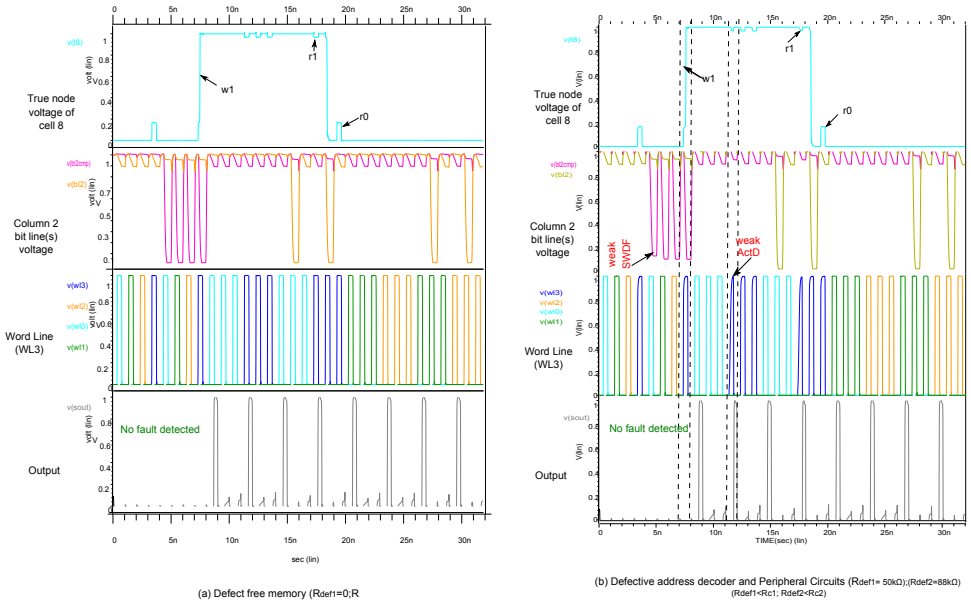


Figure 5.18: Simulation of weak slow write driver fault and weak activation delay fault using test ActD

Now consider the simulation results for test SWDF as shown in Fig 5.19(a) and 5.19(b) for defect free and multiple weak fault conditions respectively. The output response of the test under defective conditions matches with the output response of the defect free conditions. This indicates that the test SWDF is not sufficient to strongly sensitize both the weak faults at the same time to sensitize a strong fault. Careful inspection of the word line signal, shows that the weak activation delay fault is almost absent. This is because the test SWDF is having linear address transitions. Hence there are no proper transitions produced for the defect in row decoder to be sensitized at all. Hence test SWDF also fails to detect the weak faults.

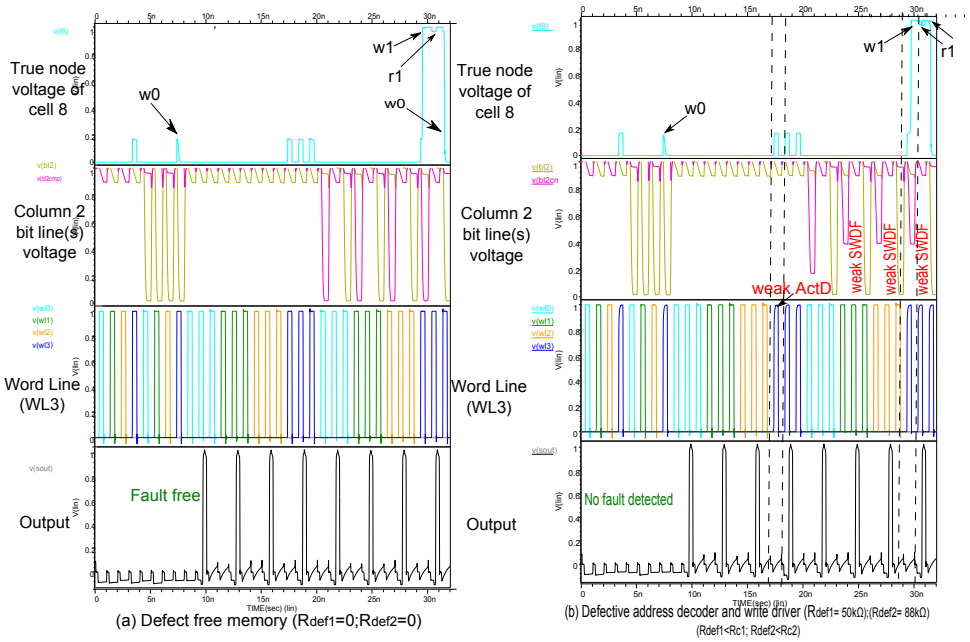


Figure 5.19: Simulation of weak slow write driver fault and weak activation delay fault using test SWDF

From the simulation results presented for tests ActD and test SWDF alone, it is clear that they are not sufficient to detect the strong faults, that may arise due to weak faults. Hence a new test 'Actd & SWDF' is developed to sensitize both the weak faults at the same time and detect the faulty behavior due to weak faults. As discussed in chapter 4, the new test 'Actd & SWDF' is formed by combining the two detection conditions of SWDF and ActD.

Test Actd & SWDF: $\{\Updownarrow (w0); \Updownarrow_x^{AC} (w1, r1, w0); \Updownarrow (w1); \Updownarrow_x^{AC} (w0, r0, w1);\}$

Fig 5.20(a) and 5.20(b) shows the simulation results for test SWDF & ActD under defect free and presence of multiple weak defects conditions respectively. The response obtained when multiple weak defects are present is different from when the expected response. This is because both the weak activation delay fault and weak slow write driver fault are detected at the same time, leading to a strong sensitization of both the weak faults, thereby resulting in sensitizing

a strong fault. The write one operation fails, in cell number 8, as it is affected by both weak activation delay fault and weak slow write driver fault. However the result of read operation from cell number 4 present in same row affected by weak activation delay fault is correct. This is because it is purely suffering only from weak activation delay faults. Similarly the cells 5,6,7 in column 2 suffer purely only from weak slow write driver fault, and hence the operations passes successfully and no fault is detected. Only cell number 8, which suffers from both weak slow write driver fault and weak activation delay fault, fails when both weak faults are sensitized at the same time. As explained in chapter 4, this test can also be used to detect, (strong) transition faults and (strong) address decoder faults.

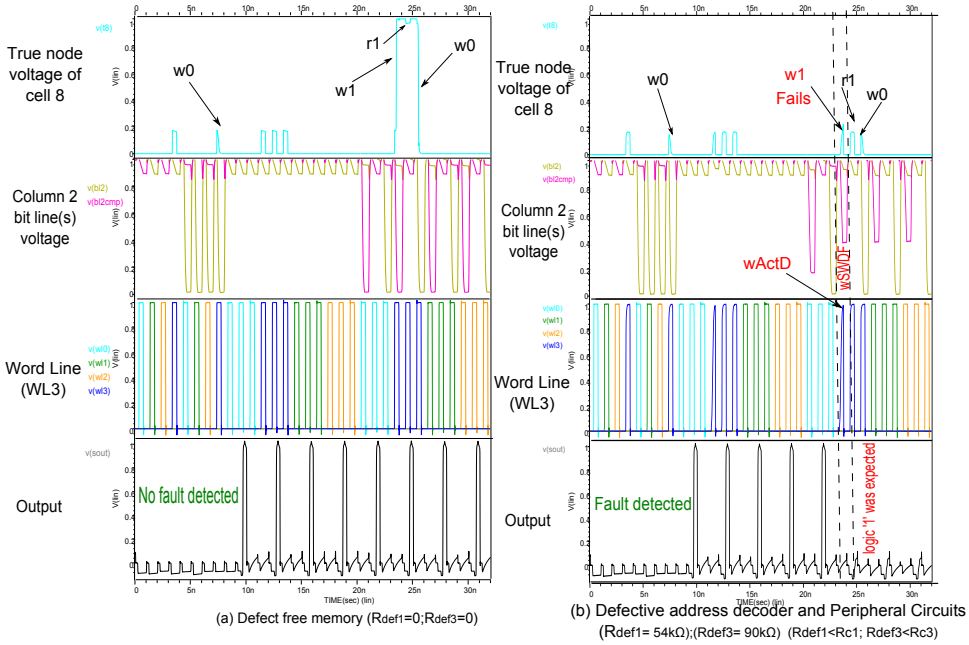


Figure 5.20: Simulation of weak slow write driver fault and weak activation delay fault using Test ActD& SWDF

5.3.3 Multiple weak defects approach in Memory Cell Array and Peripheral Circuits

Consider the scenario where both weak defects $R_{def2} < R_{c2}$,$R_{def3} < R_{c3}$ present in memory cell array and the write driver respectively at the same time (see Fig 4.10). Each of the defects R_{def2} and R_{def3} alone causes weak faults. However when both weak faults are sensitized at the same time, then it leads to a strong fault, as their effects are additive in nature. Simulations are now performed for two different tests: 1)Test TF 2)Test SWDF

For simulation, the values of defects were chosen as $R_{def2} = 480k\Omega$,$R_{def3} = 88k\Omega$. The simulation results for test TF are shown in Fig 5.21(a) and 5.21(b) for defect free and multiple weak defect conditions respectively. The output response for the simulated weak defects case is identical with that of the defect free case. This is because the detection condition in test TF

do not present a worst case scenario for the bit lines, which are affected by a slow write driver fault. The test TF does not have successive back to back complementary operations do not take place in the same write driver. Besides the operations in test TF are done in fast y direction, whereas the requirement for sensitization of slow write driver fault is that the operations should be performed on same write driver, i.e., fast x direction. Hence the fault goes undetected due to lack of strong sensitization of both weak faults at the same time.

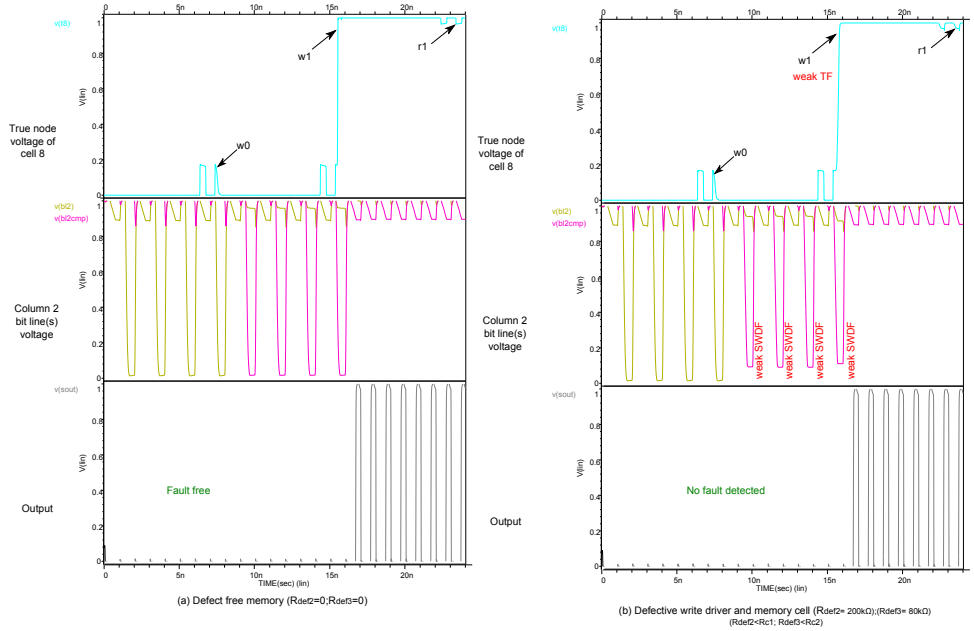
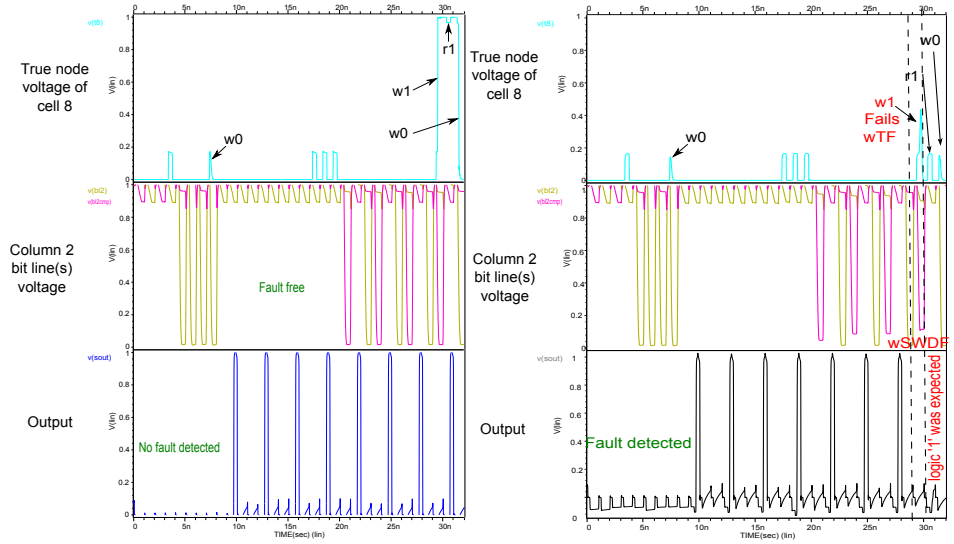


Figure 5.21: Simulation of weak slow write driver fault and weak transition fault using Test TF

Now consider the simulation results for test SWDF as shown in Fig 5.22(a) and 5.22(b) for defect free and multiple weak fault conditions respectively. The output response of the test under defective conditions does not match with the output response of the defect free conditions. This indicates that the test SWDF is sufficient to strongly sensitize both the weak faults at the same time to sensitize a strong fault. When back to back complementary write operations are applied, along fast x, direction both the slow write driver fault and transition fault is is sensitized at the same time. Hence test SWDF is sufficient.

5.3.4 Multiple weak defects approach in Memory Cell Array, Address Decoders and Peripheral Circuits

Consider the scenario where three weak defects, $R_{def1} < R_{c1}$, $R_{def2} < R_{c2}$,$R_{def3} < R_{c3}$ present in row decoders, memory cell array and the write driver respectively at the same time (see Fig 4.10). Each of the defects R_{def1}, R_{def2} and R_{def3} alone causes weak faults. However when all the weak faults are sensitized at the same time, then it leads to a strong fault, as their effects are additive in nature. Simulations are now performed for 4 different tests: 1)Test TF 2)Test SWDF 3)Test ActD 4) Test ActD & Test TF 5) Test ActD& Test SWDF



(a) Defect free memory (R_{def2}=0;R_{def3}=0)

(b) Defective memory cell and write driver (R_{def2}=200kΩ);(R_{def3}= 80kΩ) (R_{def2}<R_{c2}; R_{def3}<R_{c3})

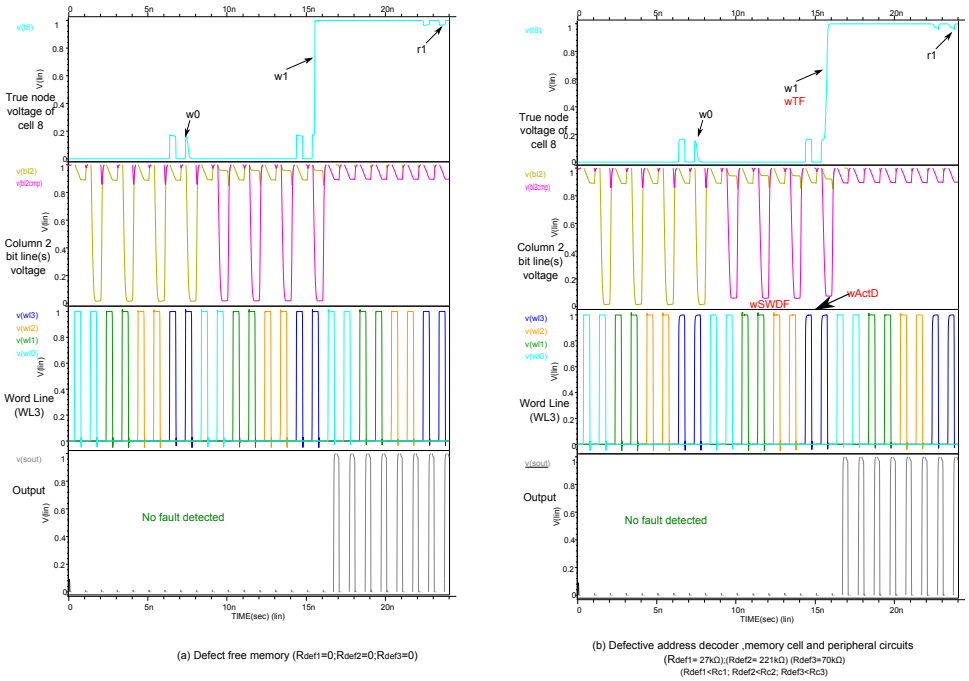
Figure 5.22: Simulation of weak slow write driver fault and weak transition fault using Test SWDF

For simulation, the values of defects were chosen as $R_{def1} = 27k\Omega$, $R_{def2} = 220k\Omega$,$R_{def3} = 70k\Omega$. Simulation results for test TF is shown in Fig 5.23(a) and 5.23(b) for defect free and multiple weak defect conditions. It can be seen that the test passes, without any problems. This is because both the other sensitizing conditions i.e., for activation delay fault and slow write driver fault are absent in this test, since there are no address transitions and test is done in fast y direction.

Simulation results for test SWDF is shown in Fig 5.24(a) and 5.24(b) for defect free and multiple weak defect conditions. It can be seen that the test passes, without any problems. This is because the weak fault due to activation delay is not fully sensitized, due to linear addressing mode used in test SWDF.

Simulation results for test ActD is shown in Fig 5.25(a) and 5.25(b) for defect free and multiple weak defect conditions. It can be seen that the test passes, without any problems. This is because the weak faults at write driver and the weak fault in cell is not sensitized at the same time.

Simulation results for test ActD & Test TF is shown in Fig 5.26(a) and 5.26(b) for defect free and multiple weak defect conditions. It can be seen that the test is able to sensitize and detect the fault. This is because the detection condition, sensitizes all the three weak faults at the same time. The test is applied under checkerboard data background. The test is able to sensitize slow writ driver fault, because it has back to back complementary write operations on same write



(a) Defect free memory (R_{def1}=0;R_{def2}=0;R_{def3}=0)

(b) Defective address decoder ,memory cell and peripheral circuits
(R_{def1}= 27kΩ);(R_{def2}= 221kΩ) (R_{def3}=70kΩ)
(R_{def1}<R_{c1}; R_{def2}<R_{c2}; R_{def3}<R_{c3})

Figure 5.23: Simulation of weak slow write driver fault, weak transition fault and weak activation delay fault using Test TF

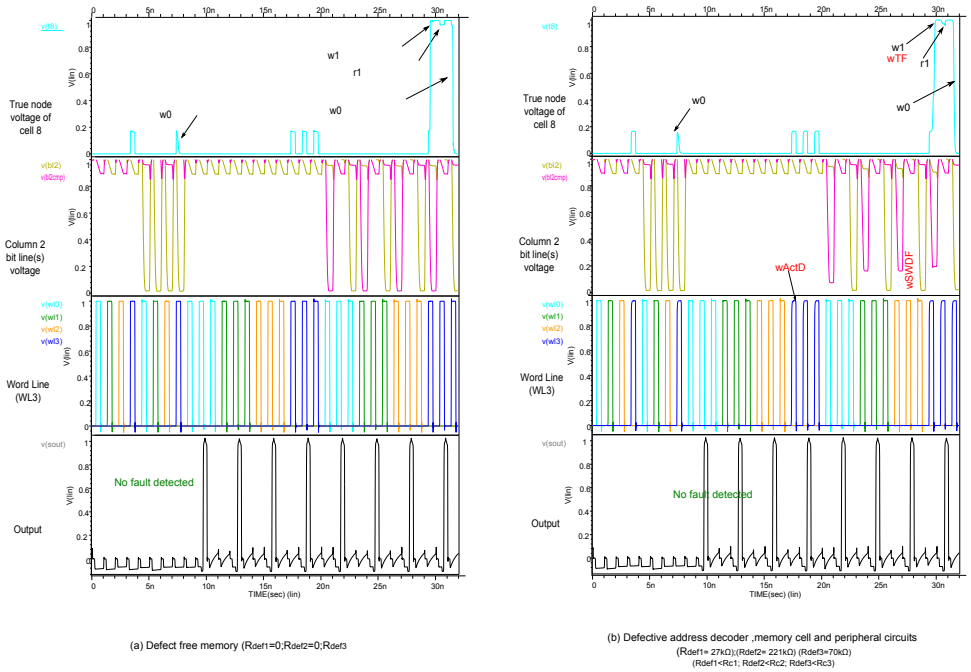
driver, i.e., in fast x direction. Also it sensitizes activation delay and transition fault at the same time, leading to detection.

Simulation results for test ActD & Test SWDF is shown in Fig 5.27(a) and 5.27(b) for defect free and multiple weak defect conditions. It can be seen that the test is able to sensitize and detect the fault. This is because the detection condition, sensitizes all the three weak faults at the same time. The test is applied under solid data background. The test sensitizes, activation delay fault and slow write driver fault. As discussed in previous section, the detection condition transition fault is also covered in the detection condition for slow write driver fault. Hence all three weak faults can be considered at the same time, leading to faulty behavior of the memory being sensitized and detected.

5.4 Fault Coverage analysis

The fault/defect coverage analysis, can be performed over the defect space region for all three components of the memory subsystem. Fig 5.28 shows the fault coverage area along with their critical resistances, for 1-dimensional analysis i.e., single defect at a time approach for memory cell array, address decoders, and peripheral circuits.

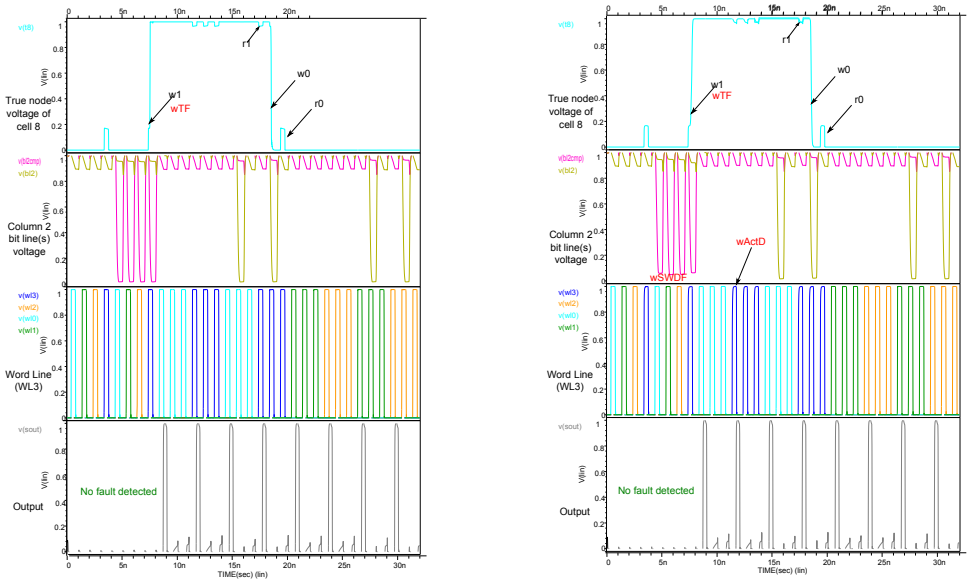
The new test approach adopts multi defect approach at a time, considering defects present in different parts of the memory system. When two defects are considered at a time, the defect



(a) Defect free memory (R_{def1}=0;R_{def2}=0;R_{def3}

(b) Defective address decoder ,memory cell and peripheral circuits
(R_{def1}= 27kΩ);(R_{def2}= 221kΩ) (R_{def3}=70kΩ)
(R_{def1}<R_{c1}; R_{def2}<R_{c2}; R_{def3}<R_{c3})

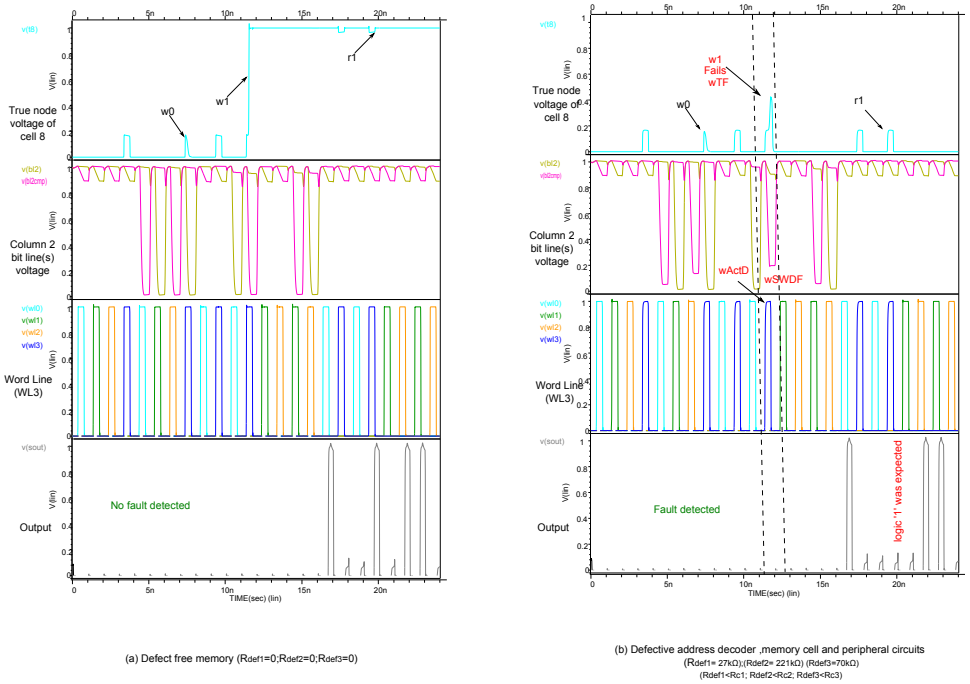
Figure 5.24: Simulation of weak slow write driver fault, weak transition fault and weak activation delay fault using Test SWDF



(a) Defect free memory (R_{def1}=0;R_{def2}=0;R_{def3}=0)

(b) Defective address decoder ,memory cell and peripheral circuits
(R_{def1}= 27kΩ);(R_{def2}= 221kΩ) (R_{def3}=70kΩ)
(R_{def1}<R_{c1}; R_{def2}<R_{c2}; R_{def3}<R_{c3})

Figure 5.25: Simulation of weak slow write driver fault, weak transition fault and weak activation delay fault using Test ActD



(a) Defect free memory (R_{def1}=0;R_{def2}=0;R_{def3}=0)

(b) Defective address decoder ,memory cell and peripheral circuits (R_{def1}= 27kΩ);(R_{def2}= 221kΩ) (R_{def3}=70kΩ) (R_{def1}<Rc1; R_{def2}<Rc2; R_{def3}<Rc3)

Figure 5.26: Simulation of weak slow write driver fault, weak transition fault and weak activation delay fault using Test ActD & test TF

space can be viewed in 2 dimensions, with each of the functional blocks in memory taking one dimension. FIG 5.29(a) shows the defect coverage obtained from the simulations which has been performed for different values of R_{def1} and R_{def2} (i.e., for address decoders and memory cell array defects)in order to identify the pass and fail regions. An estimation of the realized defect coverage can be calculated by calculating the size of the area marked with '*' and dividing it by the area defined by $R1_c$ and $R2_c$; this resulted in a defect coverage improvement of 10%.

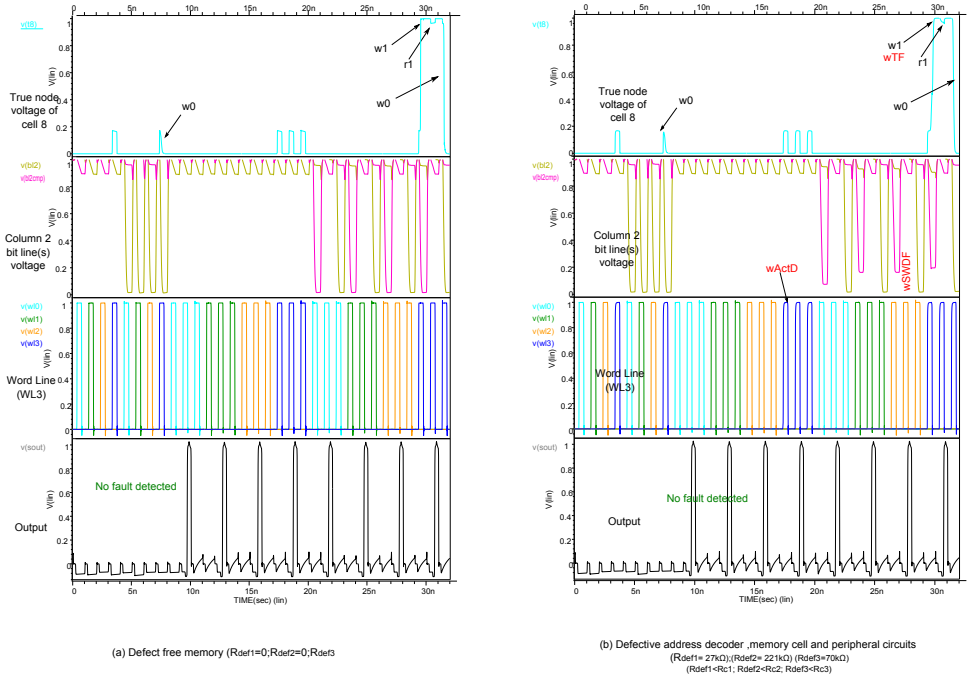


Figure 5.27: Simulation of weak slow write driver fault, weak transition fault and weak activation delay fault using Test ActD & test SWDF

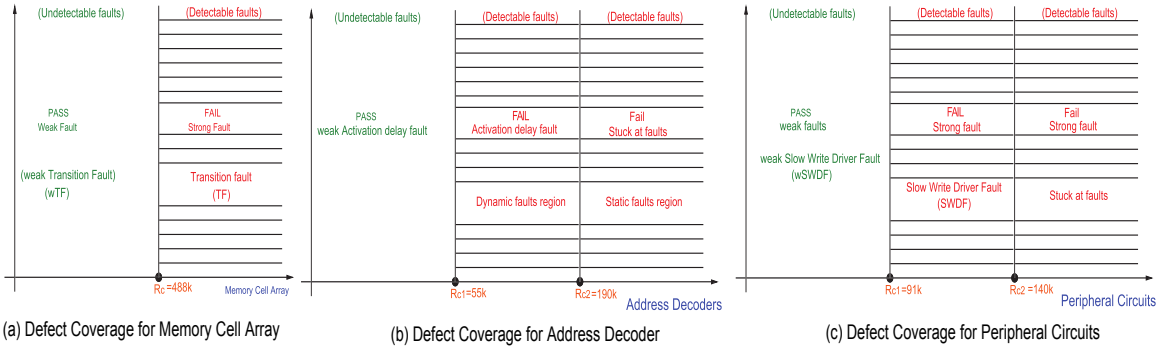
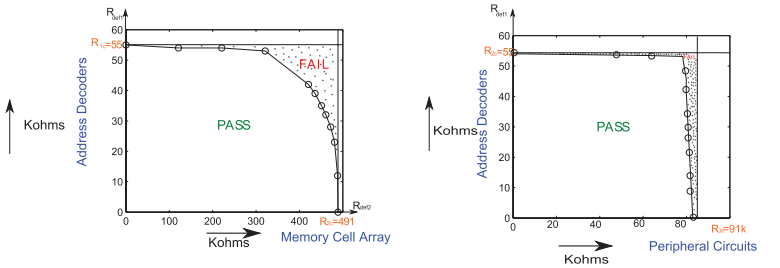


Figure 5.28: Defect coverage for 1 Dimensional approaches in memory

Similar approach and simulation have been preformed while considering simultaneously weak R_{def1} and weak R_{def3} (i.e., address decoder and peripheral circuits). The results obtained is as shown in FIG 5.29(b). An improvement of about 7% defect coverage, can be realized in this particular case of multiple weak faults. On the other hand, considering R_{def2} and R_{def3} simultaneously will not realize any defect coverage improvement as the detection condition for R_{def3} also covers that of R_{def2}.



(a) Coverage improvement based on 2-weak faults in Memory Cell Array and Address Decoders

((b) Coverage improvement based on 2-weak faults in Peripheral Circuits and Address Decoders

Figure 5.29: Fault coverage for 2 Dimensional approach in memory

6 Conclusions and future work

This chapter presents the conclusion of this thesis and provides suggestions for future works.

6.1 Conclusions

This thesis presents the following conclusions:

1. Weak faults, which are becoming more and more important with downscaling of technology, can be held responsible for the some of the faults which are not very well understood. This information can be used, for understanding some of the complex faulty behavior of the embedded memories, especially in the nano era.

2. The presence of two or more weak faults in the memory system, can have an additive impact, to cause a strong fault, leading to faulty behavior of the system. However it is also possible that the combination of multiple weak faults may have a canceling effect.

3. The detection of weak faults cannot be always established using the traditional testing approach, as it takes into account only single defect at a time i.e., there may not be simultaneous sensitization of two or more weak faults at a given time.

4. A new testing approach is developed for detecting weak based faults. Detection of such faults requires deep understanding of each of the weak faults, development of detection conditions for each fault, and thereafter combining all the detection conditions into a single test able to sensitize all the weak faults at once. This will help in the reduction of test escapes and hence the NTFs.

5. The SPICE simulations done on memory model based on 45nm PTM transistor parameters reveal that by just considering two weak faults at a time, the defect coverage can be increased with 10%. More the number of weak faults are combined, higher is the defect coverage.

6.2 Future Work

This thesis provides a new foundation which can be used to systematically develop new fault models and test algorithms for failure mechanisms of memory systems in the nano-era. Some of the suggestions for future work includes:

1. An extensive study of weak faults, to understand its complex behavior and influences of one weak fault over the other. A complete set of fault primitives can be provided.

2. Establishing fault models based on these fault primitives, developed for weak faults.

3. Test approaches can be adapted, and optimized to enhance their detection capabilities, to detect weak based faults. This might result in achieving lower DPM levels without going for sophisticated testing techniques, which are often costly.

4. Effects of multiple weak faults within the same functional block, can also be studied.

Bibliography

[1] Z. Al-Ars, *Dram fault analysis and test generation*, Ph.D. thesis, Delft University of Technology, June 2005.

[2] B. Becker, S. Hellebrand, I. Polian, B. Straube, W. Vermeiren, and H.-J. Wunderlich, *Massive statistical process variations: A grand challenge for testing nanoelectronic circuits*, Dependable Systems and Networks Workshops (DSN-W), 2010 International Conference on, 28 2010-july 1 2010, pp. 95 –100.

[3] Azeez Bhavnagarwala, Shekhar Borkar, Takayasu Sakurai, and Siva Narendra, *Ep2: The semiconductor industry in 2025*, Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International, feb. 2010, pp. 534 –535.

[4] S. Borkar, *Designing reliable systems from unreliable components: the challenges of transistor variability and degradation*, Micro, IEEE **25** (2005), no. 6, 10 – 16.

[5] ———, *Design perspectives on 22nm cmos and beyond*, Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE, july 2009, pp. 93 –94.

[6] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, *Parameter variations and impact on circuits and microarchitecture*, Design Automation Conference, 2003. Proceedings, june 2003, pp. 338 – 342.

[7] B.Prince, *Semiconductor memories, a handbook of design and manufacturing and application*, John Wiley and sons, West Sussex, 1991.

[8] M.A. Breuer and A.D. Friedmzan, *Diagnosis and reliable design of digital systems*, Computer Science Press, Woodland Hills, CA,USA, 1976.

[9] D. S. Cleverley, *Product quality level monitoring and control for logic chips and modules*, IBM Journal of Research and Development **27** (1983), no. 1, 4 –10.

[10] R. Dekker, F. Beenker, and L. Thijssen, *A realistic fault model and test algorithms for static random access memories*, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on **9** (1990), no. 6, 567 –572.

[11] Luigi Dilillo, Patrick Girard, Serge Pravossoudovitch, Arnaud Virazel, and Magali Bastian, *Analysis and test of resistive-open defects in sram pre-charge circuits*, J. Electron. Test. **23** (2007), 435–444.

[12] Allen et.al, *The international technology roadmap for semiconductors*, 2000.

[13] H. Goto, S. Nakamura, and K. Iwasaki, *Experimental fault analysis of 1 mb sram chips*, VLSI Test Symposium, 1997., 15th IEEE, apr-1 may 1997, pp. 31 –36.

[14] S. Hamdioui, *Testing multi-port memories: Theory and practice*, Ph.D. thesis, Delft University of Technology, The Netherlands, January 2001.

[15] S. Hamdioui, Z. Al-Ars, G. N. Gaydadjiev, and J.D Reyes, *Investigation of single-cell dynamic faults in deep-submicron memory technologies*, IEEE Proc. European Test Symposium Digest of Papers, May 2006.

[16] S. Hamdioui, Z. Al-Ars, J. Jimenez, and J. Calero, *Ppm reduction on embedded memories in system on chip*, Test Symposium, 2007. ETS '07. 12th IEEE European, may 2007, pp. 85 –90.

[17] S. Hamdioui, Z. Al-Ars, L. Mhamdi, G. Gaydadjiev, and S. Vassiliadis, *Trends in tests and failure mechanisms in deep sub-micron technologies*, Design and Test of Integrated Systems in Nanoscale Technology, 2006. DTIS 2006. International Conference on, sept. 2006, pp. 216 –221.

[18] S. Hamdioui, Z. Al-Ars, and A.J. van de Goor, *Opens and delay faults in cmos ram address decoders*, Computers, IEEE Transactions on **55** (2006), no. 12, 1630 –1639.

[19] S. Hamdioui, G. N. Gaydadjiev, and A.J. van de Goor, *A fault primitive based analysis of dynamic memory faults*, proceedings of PRORISC'03, November 2003, pp. 84–89.

[20] _____, *A fault primitive based analysis of dynamic memory faults*, proceedings of PRORISC'03, November 2003, pp. 84–89.

[21] S. Hamdioui and A.J. van de Goor, *Advanced embedded memory testing: Reducing the defect per million level at lower test cost*, Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010 IEEE 13th International Symposium on, april 2010, p. 7.

[22] Said Hamdioui, Zaid Al-ars, Ad, J. Van, De Goor, Mike Rodgers, and A. Ivanov, *Dynamic faults in random-access-memories: Concept, fault models and tests*, Journal of Electronic Testing: Theory and Applications **19** (2003), 2003.

[23] E. Josse, S. Parihar, O. Callen, P. Ferreira, C. Monget, A. Farcy, M. Zaleski, D. Villanueva, R. Ranica, M. Bidaud, D. Barge, C. Laviron, N. Auriac, C. Le Cam, S. Harrison, S. Warrick, F. Leverd, P. Gouraud, S. Zoll, F. Guyader, E. Perrin, E. Baylac, J. Belledent, B. Icard, B. Minghetti, S. Manakli, L. Pain, V. Huard, G. Ribes, K. Rochereau, S. Bordez, C. Blanc, A. Margain, D. Delille, R. Pantel, K. Barla, N. Cave, and M. Haond, *A cost-effective low power platform for the 45-nm technology node*, Electron Devices Meeting, 2006. IEDM '06. International, dec. 2006, pp. 1 –4.

[24] O. Mende, *Halbleiterbauelemente in der automobilelektronik*, 2008.

[25] G.E. Moore, *Cramming more components onto integrated circuits*, Electronics magazine **V.38** (1965), no. 1, N. 8.

[26] N. Mukherjee, A. Pogiel, J. Rajski, and J. Tyszer, *High volume diagnosis in memory bist based on compressed failure data*, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on **29** (2010), no. 3, 441 –453.

[27] R. Nair, *An optimal algorithm for testing stuck-at faults random access memories*, IEEE transactions on Computers, Vol. C-28, No. 3, pp. 258-261, 1979.

[28] T. Powell, A. Kumar, J. Rayhawk, and N. Mukherjee, *Chasing subtle embedded ram defects for nanometer technologies*, Test Conference, 2005. Proceedings. ITC 2005. IEEE International, nov. 2005, pp. 9 pp. –850.

[29] E.S. Cooley R.D. Adams, *False write through and un-restored write electrical level fault models for srams*, Memory Technology, Design and Testing, 1997. Proceedings., International Workshop on, aug 1997, pp. 27 –32.

[30] R.D.Adams, *High performance memory testing: Design principles, fault mod-eling and selft-test*, Kluwer Academic Pub., Dordrecht, The Netherlands, 2003.

[31] Walden C. Rhines, *Keynote speech at ieee workshop*, 2007.

[32] I. Schanstra and A.J. Van De Goor, *Industrial evaluation of stress combinations for march tests applied to srams*, Test Conference, 1999. Proceedings. International, 1999, pp. 983 –992.

[33] Dallas Semiconductor, *Maxim ds1250y/ab 4mb sram*, 2008.

[34] J.P. Shen, W. Maly, and F.J. Ferguson, *Inductive fault analysis of mos integrated circuits*, Design Test of Computers, IEEE **2** (1985), no. 6, 13 –26.

[35] Changhwan Shin, Y. Tsukamoto, Xin Sun, and Tsu-Jae King Liu, *Full 3d simulation of 6t-sram cells for the 22nm node*, Simulation of Semiconductor Processes and Devices, 2009. SISPAD '09. International Conference on, sept. 2009, pp. 1 –4.

[36] J. Srinivasan, S.V. Adve, P. Bose, and J.A. Rivers, *The impact of technology scaling on lifetime reliability*, Dependable Systems and Networks, 2004 International Conference on, june-1 july 2004, pp. 177 – 186.

[37] Xin Sun, Qiang Lu, V. Moroz, H. Takeuchi, G. Gebara, J. Wetzel, Shuji Ikeda, Changhwan Shin, and Tsu-Jae King Liu, *Tri-gate bulk mosfet design for cmos scaling to the end of the roadmap*, Electron Device Letters, IEEE **29** (2008), no. 5, 491 –493.

[38] A.J. van de Goer and J. de Neef, *Industrial evaluation of dram tests*, Design, Automation and Test in Europe Conference and Exhibition 1999. Proceedings, 1999, pp. 623 –630.

[39] Ad.J. van de Goor, S. Hamdioui, and R. Wadsworth, *Detecting faults in the peripheral circuits and an evaluation of sram tests*, Test Conference, 2004. Proceedings. ITC 2004. International, oct. 2004, pp. 114 – 123.

[40] A.J. van de Goor, *Testing semiconductor memories, theory and practice*, 2, Publishing, Gouda, The Netherlands, 1998.

[41] B. Vermeulen, C. Hora, B. Kruseman, E.J. Marinissen, and R. van Rijsinge, *Trends in testing integrated circuits*, Test Conference, 2004. Proceedings. ITC 2004. International, oct. 2004, pp. 688 – 697.

[42] Z.Al-Ars and A.J.van de Goor, *Static and dynamic behavior of memory cell array spot defects in embedded drams*, Computers, IEEE Transactions on **52** (2003), no. 3, 293 – 309.

[43] Jun Zhou and H.-J. Wunderlich, *Software-based self-test of processors under power constraints*, Design, Automation and Test in Europe, 2006. DATE '06. Proceedings, vol. 1, march 2006, pp. 1 –6.

[44] _____, *Software-based self-test of processors under power constraints*, Design, Automation and Test in Europe, 2006. DATE '06. Proceedings, vol. 1, march 2006, pp. 1 –6.

A New Test Paradigm for Semiconductor Memories in the Nano-Era

Said Hamdioui Venkataraman Krishnaswami Ijeoma Sandra Irobi Zaid Al-Ars
Computer Engineering Laboratory, Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands
{S.Hamdioui, I.S.Irobi, Z.Alars}@tudelft.nl; V.Krishnaswami-1@student.tudelft.nl

Abstract

Due to rapid and continuous technology scaling, faults in semiconductor memories (and ICs in general) are becoming pervasive and weak rather than strong; weak faults are faults that escape the test program (because they don't fully sensitize the fault) and may cause faults during the application. Components with weak faults which fail at board and system level are sent to suppliers, but only to have them returned back as No Trouble Found (NTF). This is because the conventional memory test approach assumes the presence of a single defect at a time causing a strong fault (i.e. fault fully sensitized), and is therefore unable to deal with weak faults. This paper presents a new memory test approach able to detect weak faults; it is based on assuming the presence of multiple weak faults at a time in a memory system rather than a single strong fault at a time. Being able to detect weak faults reduces the number of escapes, hence also the number of NTFs. The experimental analysis done using SPICE simulation for a case of study show e.g., that when assuming two simultaneous weak faults, the defect coverage can be increased with about 10% as compared with the conventional approach.

I. Introduction

Progressive technology scaling, as tracked by the International Technology Roadmap for Semiconductors (ITRS) and encapsulated by Moore's law [1], has driven the phenomenal success of the semiconductor industry. Silicon technology has now entered the nano-era and the 10nm transistors are expected to be in production by 2018. This will allow for the integration of a wider variety of functions. However, it is widely recognised that variability in device characteristics and its impact on the overall quality and reliability of the system represent major challenges to scaling and integration for present and future nanotechnology generations [2], [3]. What is more, newly emerging complex failure mechanisms in the nano-era (which are not understood yet), are causing the fault mode of the chips to be dominated by transient, intermittent and weak faults rather than hard and permanent faults; hence causing more reliability problems than quality problems [4], [5], [6], [7]. Many companies are reporting not being able to explain all electronic failures with the existing approaches [8], [9], [10], [11]. For instance, AUDI reported from all electronic failures, 35% can be mapped using the existing approaches into well defined semiconductor failures, while 41% cannot be understood (NTF: No Trouble Found) [9]. This shift in failure mechanisms is therefore seriously impacting the quality and reliability, which means that the design of future systems fabricated using nanotechnology is a major challenge. In turn, this will demand *revolutionary changes* in how future systems are designed and *tested* to meet the increasing quality requirements on such systems.

Nowadays, embedded memories represent the great majority of embedded electronics in Systems on Chip (SoC). It is very common to find SoCs with hundreds of memories representing more than 50% of the overall chip area. According to the ITRS, today's SoCs are moving from logic-dominant to memory-dominant chips in order to deal with the requirements of todays and future applications. Consequently, embedded memory test challenges will significantly impact the overall testability of SoC. Solving such challenges for memories will substantially contribute to the resolution of electronic system test problems in the future; hence, supporting the continuation of the semiconductor technology revolution and the manufacturability of future highly complex systems ('giga-scale') and highly integrated technologies ('nano-scale').

The purpose of this paper is the development of a new foundation which can be used to systematically develop new fault models and test algorithms for failure mechanisms of memory systems in the nano-era. The new foundation is able to deal with complex faulty behaviors of memory systems, increase the defect coverage and enable the reduction of NTFs. It is based on simultaneously considering the presence of *weak* faults in the different parts of the memory system (e.g., memory array and address decoder); this significantly impact the test detection

and test development approach. The approach is validated using a SPICE simulation. The simulation results show that the defect coverage can be increased with up to 10% for the considered case-of-study.

The paper is organized as follows. Section 2 presents an brief overview of the state-of-the art of embedded memory testing. Section 3 briefly gives the notation used to describe memory fault models and test algorithms as these will be used in the rest of the paper. Section 4 discusses the traditional approach in embedded memory testing. Section 5 presents the new memory test approach. Section 6 gives the simulation results to validate the approaches. Section 7 concludes the paper.

II. State-of-the art in memory testing

Tests for semiconductor memories have undergone a long development process. Before 1980, tests for a given fault coverage (FC) were time consuming (FC is defined as the number of detected faults divided by the number of total faults) [12]. Such tests can be termed as the 'ad hoc' tests because they were not based on fault models and proofs. To reduce the test time and improve the FC, test development has been focused on possible faulty behaviors in the memory at the functional level. For that reason, functional fault models, which are abstract fault models, were introduced during the early 1980s. After that, 'March' tests have become dominant because their test times are in linear proportion with the size of the memory, and their FC can be proven mathematically [13], [14], [15]. In the late 1990s, industrial results indicated that many tests detect faults which could not be explained by the existing fault models at the time [16], [17], [18] mainly because of the used technology node. This led to the introduction of new concepts, new fault models and test schemes [19]-[22]. Today, as the silicon industry moves towards the end of the CMOS technology roadmap, controlling the fabrication of scaled memory devices is becoming a major challenge. Device-parameter variations (e.g. threshold voltage), high defect density as well as new failure mechanisms in the nano-era are expected to be significantly larger in the future [3], [5], [23], [24], and this will create major challenges in designing and testing memories in nanotechnology. Conventional fault models and test approaches are inadequate to realize the required product quality [5], [6], [10], [11], [25], [9], [26]. In the absence of new theories capable of modeling their failures mechanism and developing appropriate test solutions, the production of future electronics systems will become infeasible.

Given the state-of-the art in fault modeling and test generation for embedded memory, one can conclude there are, as yet, no fault models to describe the above failure mechanisms. The models have to consider not only the presence of a single defect at a time (as it has been the case so far), but it also has to consider the presence of multiple weak-faults/defects simultaneously (this is particularly important in the nano-era). A weak fault is not able to fully sensitize a fault, but it partially sensitizes a fault; e.g., due to a defect that creates a small disturbance of the voltage at the true node of the cell. However, a fault can be fully sensitized (i.e., becomes strong) when two (or more) weak faults are sensitized simultaneously in the memory systems since their fault effects can be additive.

This paper will advance the knowledge in the field of memory fault modeling and test generation by introducing a new theoretical foundation being able to deal with the complex faulty behavior of embedded memories in the nano-era. A systematic approach will be developed in order to accurately model the memory faulty behavior, and develop appropriate test algorithms and solutions. The approach fundamentally differs from the conventional approach in two aspects: (a) it considers not only strong faults/defects, but also weak faults and parametric deviations as they may impact the memory reliability as well, (b) it considers not only single fault/defect at a time, but also multiple-faults/defects simultaneously in the different parts of the memory system (e.g., memory array and address decoder). This, in turn, will allow the explanation of some of today's not understood faults, and facilitate the understanding and modeling of future electronic failures in 32nm technology and beyond. The approach will increase the defect coverage, reduce the number of escapes and enable the reduction of NTFs.

III. Fault models and algorithms notation

This section describes the fault model and algorithms notation that will be used in this paper.

In order to specify a certain memory fault, one has to represent it in the form of a fault primitive (FP) [19], denoted as $<S/F/R>$. S describes the operation sequence that sensitizes the fault, F describes the logic level in the faulty cell ($F \in \{0, 1\}$), and R describes the logic output level of a read operation ($R \in \{0, 1, -\}$). R has a value of 0 or 1 when the fault is sensitized by a read operation, while the '-' is used when a write operation sensitizes the fault. For example, in the FP= $<0w1/0/->$, which is the up-transition fault (TF1), $S=0w1$ means that a w1 operation is written to a cell initialized to 0. The fault effect $F=0$ indicates that after performing $w1$, the cell remains in state 0. The output of the read operation ($R=-$) indicates there is no expected output for the memory.

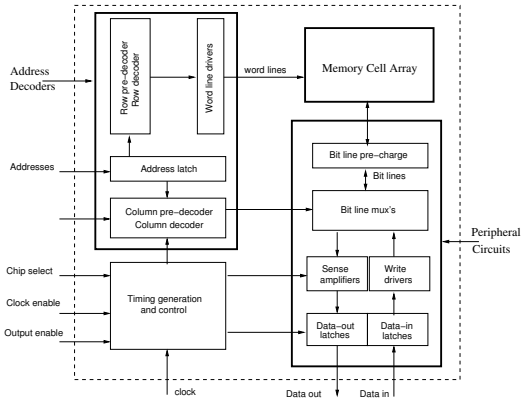


Fig. 1. Simplified memory block diagram

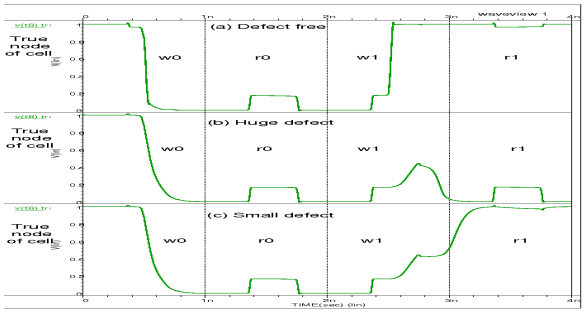


Fig. 2. Simulation results for a defective cell

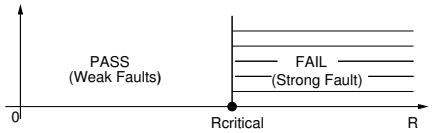


Fig. 3. Detected versus non-detected defects

The algorithms in this paper are described with an extended notation for march algorithms. March algorithms are the most common algorithms used for testing memories [15]. A *march test* consists of a finite sequence of march elements. A *March Element (ME)* is a finite sequence of operations applied to every cell in the memory before proceeding to the next cell. The way one proceeds to the next cell is determined by the *Address Orders (AOs)* which can be an increasing address order (e.g., increasing AO from the cell 0 to the cell $n-1$), denoted by $\Uparrow$ symbol, or a decreasing AO, denoted by $\Downarrow$ symbol, and which is the exact inverse of the $\Uparrow$ AO. When the AO is irrelevant, the symbol $\Updownarrow$ (i.e., $\Uparrow$ or $\Downarrow$) will be used. A complete march test is delimited by the '$\{...\}$' bracket pair; while a march element is delimited by the '$(...)$' bracket pair. The march elements are separated by semicolons, and the operations within a march element are separated by commas. An example of a march algorithm is MATS+ [29], defined as: $\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0)\}$. MATS+ consists of three MEs, which are separated by the ';' symbol. The ME '$\Uparrow\ (r0, w1)$' specifies the $\Uparrow$ AO, while to each address a read operation with expected value '0' will be applied, after which a '1' will be written.

IV. Traditional memory test approach

Figure 1 shows a simplified block diagram of a semiconductor memory. The figure distinguishes three main blocks which are usually considered for fault modeling and test purposes: memory cell array, the address decoders and the peripheral circuits (PCs). The address decoders consist of address latches, row decoders, column decoders, etc. while the PCs consist of all read and write circuitry such as write drivers and sense amplifiers.

Traditionally fault modeling is done by injecting a *single defect at a time* -such as a resistor- either in the array, the address decoder or in the peripheral circuit. Electrical simulation (e.g., SPICE) is then performed and the faulty behavior is analyzed using either static analysis or dynamic analysis [27], [19], [20]. Depending on how severe the defect is (e.g., how big is the value of the resistor representing the defect), the fault may or may not be sensitized. If the fault is sensitized, then it is mapped into a functional fault model. An appropriate detection condition is then developed and compiled thereafter in a test algorithm being able to detect the observed fault model.

Lets consider the case where a single defect will be injected in the memory cell array. Assume that the connection between the cross-coupled invertors of the memory cell array suffer from a resistive open defect. An appropriate memory memory has been built for the simulation; it includes 4x4 memory array, sense amplifies, write drivers, pre-charge circuits, address decoders, etc. The simulation has been done for the sequence $S = w0, r0, w1, r1$. Figure 2 shows the voltage at the true node of the memory cell for three cases: (a) defect free case (top graph), (b) high value of the defect (middle graph), and (c) small value of the defect (bottom graph). The middle graph shows that the $w1$ operation fails and that the state of the cell will remain in 0, causing "*Transition Fault*" ($< 0w1/0/- >$). Inspection of the bit line voltages and the output of the sense amplifier (not included in the figure) in this case reveals that the $r1$ operation returns 0 instead of an expected 1. Hence, the sequence $S = w0, r0, w1, r1$ will detect the fault. In case the defect is not high enough, all the operations will pass and no fault will be observed at the output of the sense amplifier. Nevertheless, careful inspection of the memory cell (see bottom figure) shows that the transition from 0 to 1 does not pass smoothly as it is the case for defect-free (see also the top graph). Hence, even externally it seems that there is no trouble, in reality the cell suffers from small disturbance which is not strong enough to cause a

strong fault; we call this fault a *weak fault*.

Figure 3 illustrates the difference between the Fail (strong fault) and Pass (weak faults) regions depending on the value of the defect as we externally experience it. If the defect is beyond a certain value, say $R_{critical}$, then a strong fault will be sensitized and detected. However, below this value, the fault will be not detected. Because when performing memory fault modeling and test generation we assume the presence of a single fault at a time, the defects below say $R_{critical}$ (causing *weak faults*) will never be detected. If we now assume the presence of another weak fault in another part of the memory cell (e.g. address decoder), then the fault effects of the two faults may be additive and *fully* sensitize a fault. That is something that may occur during the application and cause the memory system to fail. A manufacturing test algorithm can detect such faults only if it *combines* the detection conditions of the two (or more) weak faults; that is the missing link in the traditional approach.

V. New memory test approach

As already mentioned, technology scaling is causing more complex failure mechanisms; having small disturbances in different parts of memory system such as address decoders and read/write circuitry is a reality [10], [25], [26]. Therefore, taking the fault effects of such disturbances together into consideration while developing fault models and designing test algorithms is required not only to increase the outgoing product quality and reduce the number of escapes, but also to contribute to the reduction of NTFs as well. In the rest of this section, the new approach will be explained and illustrated.

As already mentioned, the basic idea is to consider the combined fault effects of *multiple weak* faults. In our case, we will use the reduced memory system consisting of three main blocks (see Figure 1): memory cell array, address decoders and peripheral circuits (i.e., read/write circuitry). We further assume that each of the three blocks may suffer from a weak fault. In case of a single-defect at a time approach (i.e. traditional approach), the faulty behavior of the memory can be graphically illustrated in a similar way as shown in Figure 3. If now, we consider the presence of two defects at a time, say defect R1 in the memory cell array and defect R2 in the address decoders, then the defect/fault coverage can be illustrated in a two dimensional graph as shown in Figure 4(a). The hashed areas in the graph represent the coverage that can be achieved based on the traditional approach. Obviously, this coverage will be also realized with the approach based on two defects at a time. Moreover, the new approach is able to improve the defect/fault coverage, which is represented by the region marked with '*' in the graph. Therefore, some weak faults that may escape the traditional test approach will be detected when assuming two defects at a time. It is worth noting that the way test algorithms are developed has to be adapted in such way that the sensitization conditions of both faults have to be considered *simultaneously*; i.e., the sensitization conditions have to combined into a *single* condition. For example, assume that R1 in the memory cell array causes a transition fault TF=<0w1/0/-> and R2 in the address decoder causes an *activation delay (ActD)* fault in the row decoder [30], [31]. The sensitization condition of TF requires the application of the transition w1 operation to a memory cell, while ActD fault requires special address transitions (such as *Address Complement*) and the use of *fast row* address direction. Combining these two condition will result in a transition write operations through fast row using e.g., address complement.

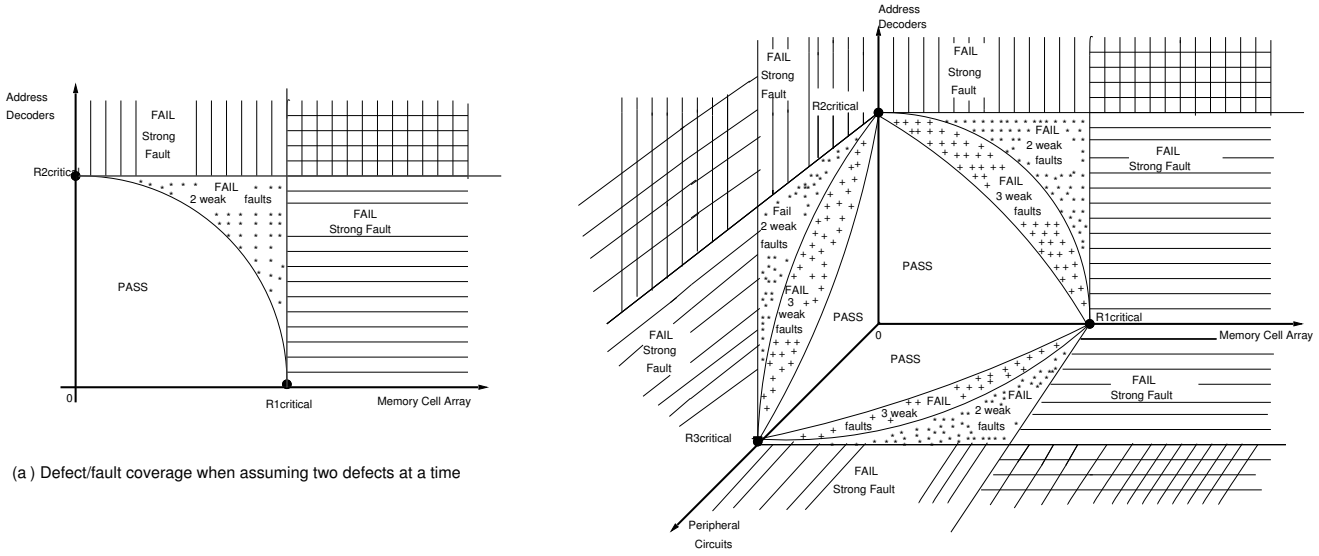
The defect/fault coverage can even be improved if more than two defects at a time are considered. Figure 4(b) illustrates the additional coverage that can be realized when assuming three simultaneous defects: defect R1 in the memory cell array, defect R2 in the address decoders and defect R3 in the peripheral circuit. As the figure shows, the coverage will be further improved as compared with Figure 4(a); this improvement is given by the regions marked with '+' in the figure. Note that the regions marked with '*' indicate the improvement when considering two simultaneous defects as compared with the traditional approach. Again, the improvement can be realized only when the detection condition used to develop the test algorithms combines the three sensitization/detection of all the three defects.

VI. Simulation results

An appropriate memory simulation model has been used; it consists of a 4x4 cell array, address decoders and each column has its own set of peripheral circuits such as sense amplifies, write drivers, pre-charge circuits, etc. (see Figure 1). The model transistor parameters are as described for 45nm PTM models.

As a case of study, we considered the following three defects (see Figure 1):

- Defect R1: The connection between the cross-coupled inverters of the memory cell array suffer from a resistive open defect.
- Defect R2: A resistive open in one of the address line inputs of an NAND gate of the row decoder.
- Defect R3: A resistive open in the pull-up transistor of the inverter in the write driver circuit.



(a) Defect/fault coverage when assuming two defects at a time

(b) Defect/fault coverage when assuming three defects at a time

Fig. 4. Defect/Fault coverage improvement

TABLE I. Single defects simulation results

Df	R_c (Ω)	Fault model	Detection condition
R1	489K	$<0w1/0/->$	A transition w1 followed by a read
R2	55K	ActD	Two back-to-back operations with opposite data values and using special address transitions with fast row
R3	91K	SWDF	Write-after-write back-to-back operations using opposite data values and fast row.

TABLE II. Test for single defects

Df	Fault	Test
R1	TF	$\{\Updownarrow (w0); \Updownarrow (w1, r1)\}$
R2	ActD	$\{\Updownarrow (w1);\ ^{AC}\Uparrow (r1, w0, r0)\}$ with fast row.
R3	SWDF	$\{\Updownarrow (w0); \Uparrow (w1, r1, w0)\}$ with fast row.

In the rest of this section the simulation results and analysis will be shown first for the traditional approach (i.e. assuming a single defect at a time), and thereafter for the new approach by assuming two or three weak faults simultaneously. We will also show how test algorithms can be developed in a systematical way in order to improve the defect/fault coverage.

A. Single defect approach

Each of the defects R1, R2 and R3 has been simulated; again only a single defect at a time is simulated. The fault behavior of the memory is analyzed and compiled into fault models; the results are summarized in Table I. The first column gives the defect, the second column the critical resistance (R_c) of each defect, and the third column the fault model observed. Note that for defects with resistance below R_c, no strong fault was observed, instead a *weak fault* was sensitized; see Figure 3. Hence, the fault will be not detected with tests used for strong faults. The observed strong faults consist of:

- Defect R1: A write 1 fails to change the state of the cell resulting into TF=$< 0w1/0/- >$.
- Defect R2: The activation of the addressed word line is delayed due the open defect; this results into an *Activation Delay (ActD)* fault [30], [31].
- Defect R3: The write driver becomes too slow due to the open defect in the driver. The result will be that the differential voltage on the bit lines during the write operation is reduced, causing the cell not to be written [32], [14]. This fault is called *Slow Write Driver Fault (SWDF)*.

The last column in the table lists the requirements a test has to satisfy in order to detect the corresponding fault. To detect TF, the test has to perform a transition write 1 operation to sensitize the fault, followed with a read operation to detect the fault. For ActD, two back-to-back operations, say opx and $op\overline{x}$ where $op \in \{w0, w1, r0, r1\}$ and $x \in \{0, 1\}$ have to performed using special address transitions (such as address complement or hamming) with fast row [30], [31]. The SWDF, on the other hand, requires the application of two back-to-back write operations with opposite data values with fast row. Table II lists examples of minimal tests for each fault of Table I. Test for TF can use a *linear* address sequence (i.e., 1, 2, 3, .., N where N is the number of addresses) with either fast column or fast row; the test for ActD fault has to use *address complement* sequence (denoted with $^{AC}\Uparrow$ in the test) with fast row, while the test for SWDF can use the linear addressing with fast row.

The tests were simulated using our memory SPICE model for the three defects with values larger than R_c in order to evaluate their coverage. The results are summarized in Table III; a '+' in the table means that the test

TABLE III. Test results

Test	Faults		
	TF	ActD	SWDF
Test TF	+	-	-
Test ActD	+	+	-
Test SWDF	+	-	+

detects the corresponding fault; e.g., Test for ActD detects both TF and ActD but not SWDF. It is worth noting that the development of each of the tests is based on compiling a *single* detection condition into a test algorithm and that the detected faults are *strong* ones.

B. Two weak faults simulation

Next we will assume the presence of two faults at a time; the two defects R1 and R2 are injected in the memory system for the simulation where the values of the defects are smaller than their critical values R_c (R1=420$K\Omega$ and R2= 52$K\Omega$); R1 and R2 can each cause a weak fault. The simulation is performed for three sequences (see also Table II): (a) Test TF, (b) Test ActD, and (c) new sequence generated by combining the detection conditions of TF and ActD faults (see Table I); this test is 'Test TF&ActD' with the following description:

$\{\Uparrow(wd);\ ^{AC}\Uparrow(w\bar{d});\ ^{AC}\Uparrow(r\bar{d});\ ^{AC}\Uparrow(wd);\ ^{AC}\Uparrow(rd)\}$

where d denotes the *checkerboard* data-background [15]; the test is applied in *fast row* address direction. Note that this test satisfies the detection conditions of *both* TF and ActD fault.

Figure 5 gives the voltage at the true node of the memory cell suffering from the defect R1(i.e., cell #8 in the 4x4 array under consideration) for the three simulated tests. The top graph in the figure shows the result for Test TF; it clearly shows the write operations pass correctly even in the presence of the defects. Inspection of the bit lines voltage and the output of the sense amplifier (not included in the figure) reveals that the read operation returns the correct value, hence the presence of two weak faults cannot be detected. The middle graph show the results for Test ActD; also here the operations pass correctly and no fault has been externally observed. The bottom graph show the results for the first three march elements of Test TF&ActD; the cell fail to undergo an up write transition, hence the fault is sensitized. In addition, externally the fault was detected by read operation producing 0 instead of 1. Note that for this test checkerboard data-background d is used, and that d=0 for cell #8 in the 4x4 memory array under consideration.

The simulation has been performed for different values of R1 and R2 in order to identify the pass and fail regions. The results are plotted in Figure 6(a). An estimation of the realized defect coverage can be estimated by calculating the size of the area marked with '*' and dividing it by

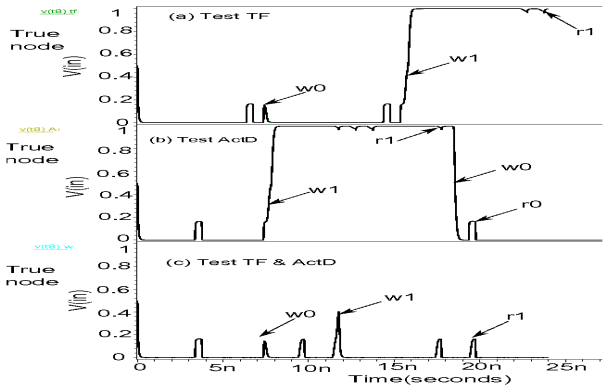


Fig. 5. simulation for two weak faults

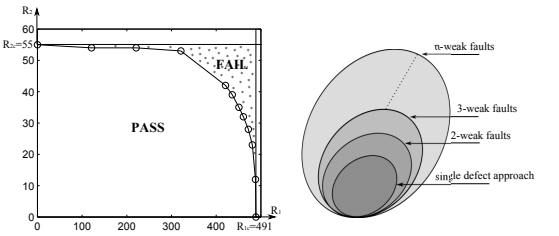


Fig. 6. Difference in defect coverage

the area defined by $R1_c$ and $R2_c$; this resulted in a defect coverage improvement of 10%.

Similar approach and simulation have been preformed while considering simultaneously weak R2 and weak R3. The results show that an improvement of about 7% defect coverage can be realized. On the other hand, considering R1 and R3 simultaneously did not realize any defect coverage improvemebnt as the detection condition for R3 also covers that of R1.

C. Three weak fault simulation

In this case, the presence of three weak defects at a time has to be considered. Therefore, the three defects R1, R2 and R3 have to be injected in the different parts of the memory; the values of defects are smaller than their critical values. Thereafter, seven test sequences have to be simulated:

(a) The three tests targeting single-cell defect at a time as discussed in Section VI-A and given in Table II.
(b) The three tests targeting two simultaneous weak faults such as Test TF&ActD discussed in Section VI-B.
(c) A new test generated by combining all the three detection conditions into a single condition and compiling it into a single test.

The simulation results show that the sequence under (c) outperforms (in terms of defect coverage) all the sequences under (a) and under (b). Figure 6(b) shows the relationship between the defect/fault coverage for the different approaches. The more simultaneous weak faults

considered when developing a test algorithm, the higher the realized coverage.

VII. Conclusions

In this paper, a new memory test paradigm has been presented. A new systematic approach being able to deal with complex faulty behavior of embedded memories in the nano-era is presented. It is based on assuming the presence of two or more weak faults at a time in the memory system; such weak faults may have an additive fault effect and therefore fully sensitize the fault. Detection of such faults requires deep understanding of each of the weak faults, development of detection conditions for each fault, and thereafter combining all the detection conditions into a single one being able to sensitize all the weak faults at once. The SPICE simulation done on memory model based on 45nm PTM transistor parameters reveals that by just considering two weak faults at a time, the defect coverage can be increased with 10%. The more weak faults are combined, the higher the defect coverage.

References

[1] G. E. Moore,"Cramming More Components onto Integrated Circuits", Electronics magazine, V. 38, N. 8, April 19, 1965.

[2] B. Becker, et. al, 'Massive statistical process variations: A grand challenge for testing nanoelectronic circuits', International Conference on Dependable Systems and Networks Workshops, pp. 95 - 100, 2010.

[3] A. Bhavnagarwala, S. Borkar, T. Sakurai, and S Narendra, ' The semiconductor industru in 2025', IEEE international Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pp. 534-535, 2010.

[4] S. Borkar, "Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation", IEEE Micro, November-December 2005.

[5] S. Borkar, 'Design and Test Challenges for 32 nm and Beyond', keynote speech at IEEE International Test Conference, p. 13, 2009.

[6] S. Hamdioui, Z. Al-Ars, L. Mhamdi, G. N. Gaydadjiev, S. Vassiliadis, Trends in Tests and Failure Mechanisms in Deep Sub-micron Technologies,, IEEE proceesings of Int. Conference on Design and Test of Integrated Systems in Nanoscale Technology, pp. 216-221, September 2006.

[7] B. Vermeulen, C. Hora, B. Kruseman, E.J. Marinissen and R. van Rijsinge, "Trends in Testing Integrated Circuits," in Proc. IEEE Int'l Test Conf., 2004, pp. 688-697.

[8] Z. Conroy, G. Richmond, G. Xinli, B. Eklow, A practical perspective on reducing ASIC NTFs, *Proc. of Int. Test Conference*, pp. 349, 2005

[9] O. Mende. Halbleiterbauelemente in der Automobilelektronik. Presentation, May 2008.

[10] T. Powell, A. Kumar, J. Rayhawk and N.Mukherjee, 'Chasing Subtle Embedded RAM Defects for Nanometer Technologies', In Proc. of the IEEE Int. Test Conf., paper 33.4, 2005

[11] J. Zhou and H.-J. Wunderlich. Software-based self-test of processors under power constraints. In Proceedings of the Conference on Design Automation and Test in Europe DATE 2006 Munich Germany March 6 10 2006

[12] M.A. Breuer and A.D. Friedmzan, "Diagnosis and reliable design of digital systems", Computer Science Press, Woodland Hills, CA, USA, 1976

[13] R. Dekker, F. Beenker and L. Thijssen, "A realistic fault models and test algorithms for static random access memories", IEEE Trans. Computers, C9, (6), pp. 567-572, 1990.

[14] R.D. Adams and E.S. Cooley, "False write through and un-restored write electrical level fault models for SRAMs", Proc. IEEE Int. Workshop on Memory Technology, Design, and Testing, pp. 27-32, 1997.

[15] A.J. van de Goor, "Testing semiconductor memories, theory and practice", ComTex Publishing, Gouda, The Netherlands, Second version, 1998.

[16] S.N.H. Goto and K. Iwasaki, "Experiment fault analysis of 2 MB SRAM chips", Proc. Design and Test Europe Conference, pp. 623-630, 1999.

[17] I. Schanstra and A.J. van de Goor, "Industrial evaluation of stress combinations for march tests applied to SRAMs", Proc. IEEE Int. Test Conference, pp. 983-992, 1999.

[18] A.J. van de Goor and J. de Neef, "Industrial evaluation of DRAMs tests", Proc. Design Automation and Test in Europe, pp. 623-630, 1999.

[19] Z. Al-ars and A. J. van de Goor, Static and dynamic behavior of memory cell array spot defects in embedded DRAMs, IEEE Trans. Comput., Vol. 52, (3), pp. 293309, 2003.

[20] S. Hamdioui, Z. Al-ars, A.J. van de Goor and M. rodgers, "Dynamic Faults in Random Access Memories: Concept, fault Models and Tests", Journal of Electronic Testing: Theory and Applications, pp. 195-205, April 2003

[21] S. Hamdioui, Z. Al-ars, A.J van de Goor and M. Rodgers, "Linked faults in random-access-memories: concept, fault models, test algorithms and industrial results", IEEE Trans. CAD Integrated Circuits Syst., 23, (5), pp. 737-757, 2004.

[22] Jen-Chieh Yeh, Chao-Hsun Chen, Cheng-Wen Wu, Shuo-Fen Kuo, 'A Systematic Approach to Memory Test Time Reduction', IEEE Design & Test of Computers 25(6), pp. 560-570, 2008.

[23] S. Borkar, et. Al, "Parameter Variations and Impact on Circuits and Microarchitectures", Proc of IEEE Design Automation Conference DAC, pp. 338-342, 2003

[24] J. Srinivasn, et al. The Impact of Technology Scaling on Lifetime Reliability. International Conference on Dependable Systems and Netweoks, June 2004.

[25] S. Hamdioui, Z. Al-Ars, J Jimenez, J Calero, "PPM Reduction on Embedded Memories in System on Chip", IEEE proceedings of European Test Symposium, pp. 85-90, Freiburg, Germany, May 2007

[26] N. Mukherjee, A. Pogiel, J. Rajski, and J. Tyszer, 'High Volume Diagnosis in Memory BIST Based on Compressed Failure Data', IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume: 29 , pp. 441-453, 2010.

[27] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, M. Bastian Analysis and Test of Resistive-Open Defects in SRAM Pre-Charge Circuits, Journal of Electronic Testing: Theory and Applications, Vol. 23, Issue 5, October 2007

[28] Z. Al-Ars, S. Hamdioui, A.J. van de Goor, G. Mueller, Defect Oriented Testing of the Strap Problem Under Process Variations in DRAMs, Proc. IEEE International Test Conf. (ITC 08), California, US, October 2008

[29] R. Nair, 'An Optimal Algorithm for Testing Stuck-at Faults Random Access Memories', IEEE transactions on Computers, Vol. C-28, No. 3, pp. 258-261, 1979.

[30] S. Hamdioui, Z. Al-ars, A.J. van de Goor, "Opens and Delay Faults in CMOS RAM Address Decoders", IEEE Trans. Computers, Vol. 55, No. 12, pp. 1630-1639, Nov., 2006.

[31] M. Sachdev, "Open Defects in CMOS RAM Address Decoders", IEEE Design and Test of Computers, pp. 26-33, Apr.-June 1997.

[32] A.J. van de Goor, S. Hamdioui, and R. Wadsworth, "Detecting Faults in the Peripheral Circuits and Evaluation on SRAM Tests", Proc. IEEE Int'l Test Conf., pp. 114-123, 2004.

[33] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, S. Borri, M Hage-Hassan, "Dynamic read destructive fault in embedded-SRAMs: analysis and march test solution", In proc. Of IEEE European Test Symposium, pp. 140 - 145, 2004.